**Lab 7**

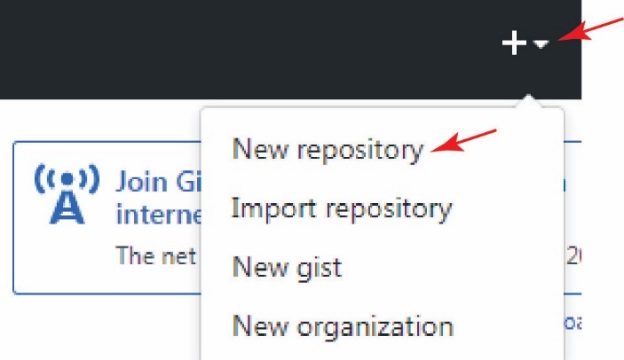
***Pushing to GitHub***

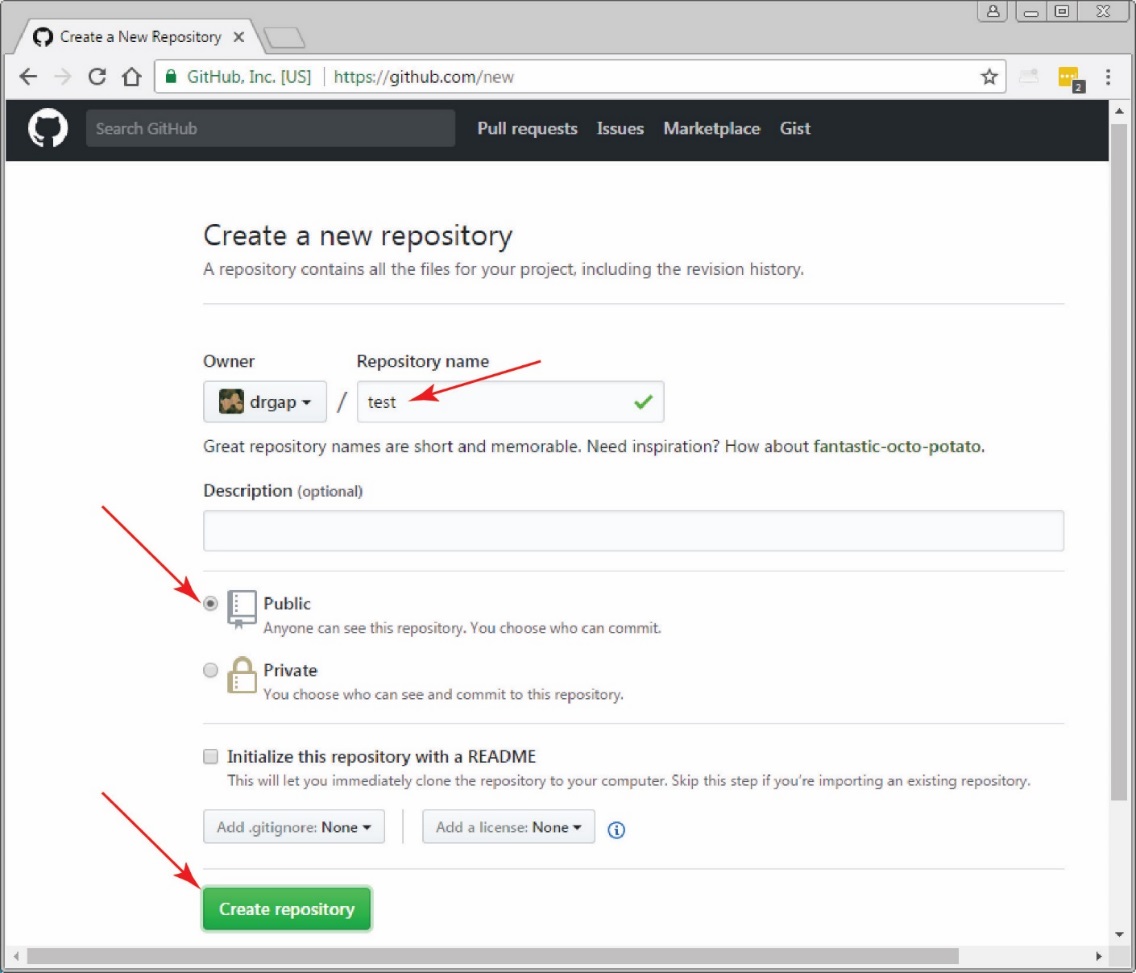
In this lab we will create an empty repo on GitHub and then push our local repository (from Lab 6) to the master branch on the GitHub repo. This lab continues from Lab 6.

**Steps to Complete**

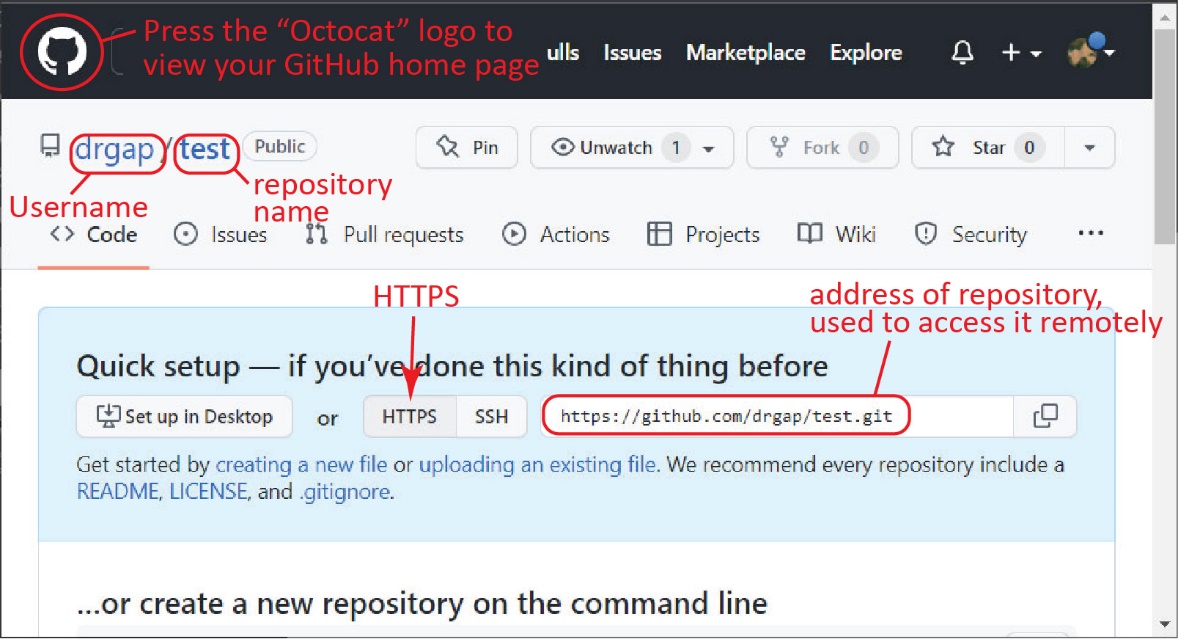
1. Create a *GitHub* account (if you don’t already have one) at: <https://github.com/>. You will also create a username. My user name is “drgap” as you will see in the tutorials.

**You are required to put your full name in the profile during setup. If you didn’t, do it now:**

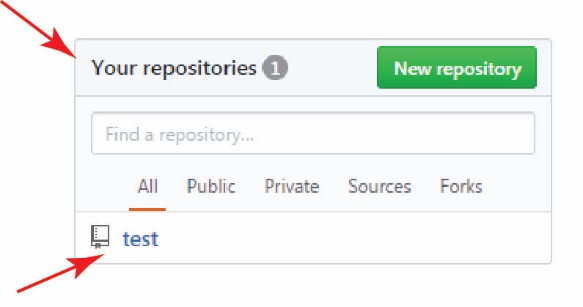
1. Select the dropdown with your profile pic/image in the upper right
2. Select: Your profile
3. On the left, select: Edit profile
4. Type in your name and then choose: Save
5. Follow these steps to create a new repository.
6. Login to GitHub if necessary
7. As shown in the figure on the right, select the drop-down beside the “+” sign in the upper right and choose, “New Repository”.
8. As shown below, name the repository, “test” and then choose “Create repository”



1. The result will look similar to what is shown below. Make sure HTTPS is selected, you’ll need the URL shortly.



1. Press the Octocat logo in the upper-left corner to access your homepage on GitHub. In the upper-left, as shown below (as of summer 2019, github’s display is different than the figure below), you will see a list of your repositories. To access your repository (the page we were on previously), select the name of the repository. Do that now.



1. On the Quick Set Up, choose: HTTPS, and then copy the URL. Mine is shown below. You can use it to paste into the command in Step 5 below, or you can just retype it.

https://github.com/drgap/test.git

1. Open Cmder and navigate to the *gitex* folder where you should have a repository from the previous tutorial.
2. Issue this command substituting your GitHub username (type the first part and then paste the URL):

**λ**  git remote add origin https://github.com/YOUR\_USER\_NAME/test.git

This command is creating an alias (“origin”) for the URL of the remote repository simply so that we don’t have to type the URL out every time we need it. There is nothing special about the name “origin”, it could be anything; however, “origin” is standard. For your information:

|  |  |
| --- | --- |
| Command | Meaning |
| **λ**  git remote -v | Lists all remotes you have defined. Mine is shown below. There is a URL for *pulling (fetching)* and one for *pushing,* which in this case they are the same. |
| **λ**  git remote rm remoteName | Deletes a remote. |
| **λ**  git config -l | Lists Git’s config file. Towards the bottom you will see, *remote.origin.url=…* |

1. Next, we are ready to push our local repository to the *test* repo on GitHub. However, first, we must obtain a *Personal Access Token (PAT)* to use as our password when we push from our local machine to GitHub. Follow all 14 steps (ignore the 10th step) here:

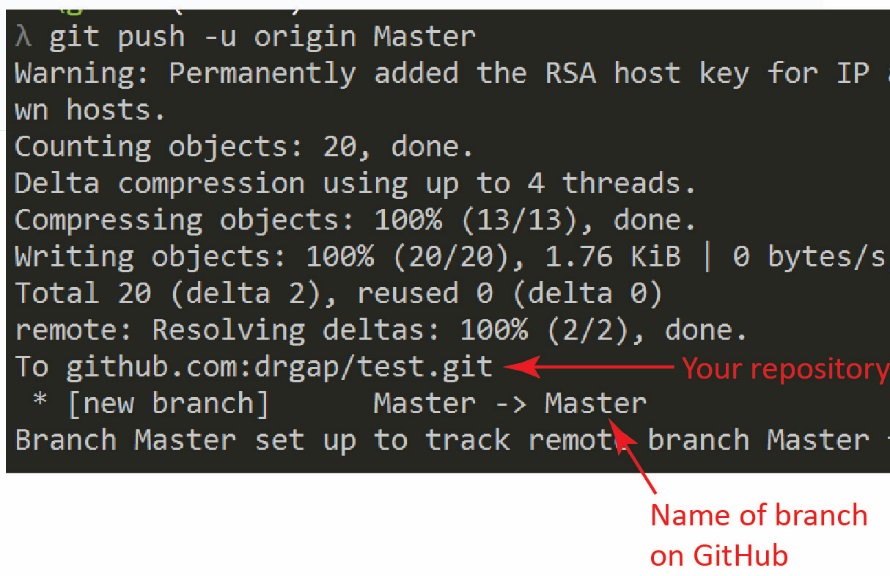
<https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token>

Notes:

* Step 1 – The dialog on GitHub looked a little different than the screen shots. I did not see anywhere to verify my email. Perhaps it was already done. In the end, everything worked.
* Step 8 – I’m not sure which options to choose. I chose: repo, admin:repo-hook, delete\_repo, project.
* Step 9 – Save the token in a text file somewhere you will remember. You can’t access it on GitHub except when you generate it. If you lose it, or it expires, you’ll have to create a new one.
* Step 11 – Choose: All Repositories.
* Step 12 – Ignore
* Step 13 – Under:
  + Repository permissions, choose: Read and write for all permissions. There are a bunch of them. I don’t know which ones we actually need, but do them all and it should work.
  + Account permissions – Ignore. If something doesn’t work right, go back and set all these to Read and write.
* Step 14 – Copy the PAT and save it somewhere you will remember. You’ll need it.

**Note:**

* **If things don’t go correctly in subsequent steps in this tutorial or future ones, you may need to create a new PAT**

1. Now, we are ready to push our local repository to the remote. Issue this command:

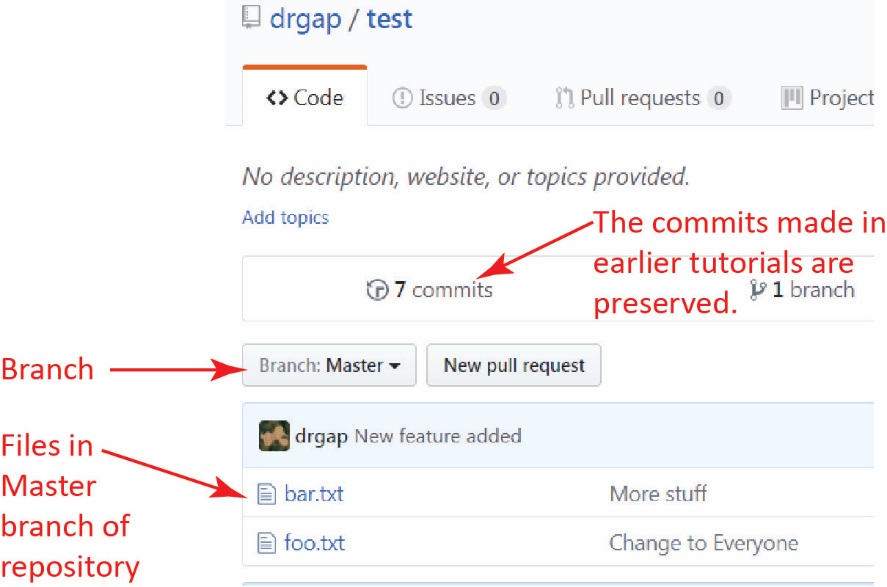
**λ**  git push –u origin master

**You will be prompted for your GitHub username and password. The password is the PAT (the password displays nothing so you can’t actually see that it is there)**

If the command is successful, the result is shown on the right (yours may look different as the screen shot on the right was generated when using SSH)

(Read, no action required) The standard of using “origin” as an alias for the remote URL is a bit counter-intuitive to me, but in the context of this statement, “origin” is the *destination* you are pushing to*.* Also counter-intuitive to me is the order of “origin” and “master”. “master” refers to local repository branch you are pushing to the remote “origin”; it seems more intuitive if the order were “master origin”.

1. Return to GitHub. Select the Octocat logo and then select your repo.. You should see your files displayed as shown below. (Yours will probably show 3 commits and only one file, *foo.txt*)



1. Select the link that says, “7 commits” (your number could be different). Drill down into a commit that was made earlier and note how it is displayed. When done exploring, return to the previous page by using the back arrow or selecting the link in the upper left that has the name of your repository (*i.e.* test).
2. Select *foo.txt* and it will display the file. For now, DO NOT EDIT the file. We will do that in the next tutorial.
3. (Read, no action required) **Note, this could be very useful in a pinch:**

* **You can also drag files from your local computer into the view above in GitHub that shows the files in a branch. When you do this, GitHub provides a dialog to stage and commit the change.**
* **You can’t drag a single file from GitHub to your local computer. However, you can choose: *Clone or Download* and then *Download ZIP*, then open the zip file and drag the file(s) you want to the local folder.**

Essentially, this is a different workflow, and I don’t know if it is acceptable in practice. Essentially, you are not utilizing a local repository.

1. **Do the following:**
2. Make a screen shot similar to the one in Step 8 above. Make sure it shows your User ID.
3. Place the image in the *HW VCS* document in the appropriate place.
4. The image should easily readable without zooming in or out.