Lab 0: Learning to use Git

Due Friday, 12 April 2019 11:59 PM

Minimum Submission Requirements

- Create 6 folders in your repository labeled: Lab0, Lab1, Lab2, Lab3, Lab4, and Lab5 **(note the capitalization convention)**
- Add README.txt files in each of the 6 folders
- Commit and push your repo to the GitLab server

Lab Objective

This lab will introduce you to version control (Git) and how to use it within the context of this class. In addition to setting up your repository directory structure, you will read the the syllabus and personal responsibility document and write a statement confirming that you have read the documents.

Version control is an extremely important part of engineering; it is the basis for most engineering projects. Git is an industry standard tool for version control, and you should know how to use it as part of your engineering education (for example, Microsoft Windows development is version controlled using Git).

Note

Failure to meet the minimum submission requirements listed will result in failing the class regardless of any other scores. This will be true for all subsequent labs as well.

Preparation

To complete this lab you need to register for a GitLab @ UCSC account. There is a delay between signing up for an account and getting a repository. The repository (or repo) is the set of files that Git is keeping track of.

Familiarize yourself with the command line and Git by reading the documentation posted on Piazza, watching videos 2-6 posted in the CE 12 video playlist, or going through one or more of the git tutorials (Git Immersion and Learn Git Branching).

Lab Procedure

Once you have your GitLab account and a repo for the class, you are going to verify that you have read both the class syllabus and personal responsibility documents. You will be "digitally signing" the documents by creating a README.txt that includes a sentence stating you have read and understood these documents.

Due to the size of the class, this and other labs will be machine graded. That is, a program will run that will check the files and generate your grade. It is extremely important to **match the names and directories exactly (NOTE THE CAPITALIZATION CONVENTION STATED ABOVE!!!)**, as that is how the grading program needs to find your work. As in all engineering, in this course **attention to detail is paramount**. Stories abound where small errors lead

to catastrophic failures; in this class it will only lead to a failing grade, but you might find that catastrophic. Pay attention!

While these requirements may seem somewhat arbitrary, note that they flow from previous problems with the class (and other classes at UCSC) especially in the realm of academic misconduct. This lab serves to demonstrate that you are aware of what we expect of you such that there is no misunderstanding about what does and does not constitute improper use of material.

Steps

1. Register for an SOE GitLab account: https://gitlab.soe.ucsc.edu/gitlab/users/sign_in
2. Read the git_cmpe012 document available on Piazza. – this goes over the basics of Git and how to use it.
3. Clone your repository. If you are using a lab computer, make sure to use the X: drive so it doesn't get wiped when you log out.
4. Create the following directories in your repository: Lab0, Lab1, Lab2, Lab3, Lab4, and Lab5. **Note the capitalization convention**.
5. Use the "touch" command to create a README.txt file in each of the 6 folders.
6. **Carefully** read the syllabus and personal responsibility document available on canvas (personal_responsibility_cmpe012.pdf).
7. Modify README.txt in the Lab0 directory to include a statement that you have read and understood the syllabus and personal responsibility document. Use this format exactly (replace Rebecca Rashkin with your first and last name, rrashkin with your CruzID, and make sure the date **matches the date of the commit**):

   I, Rebecca Rashkin, have read and understood the CE12 syllabus and Personal Responsibility Document. rrashkin 8 April 2019

8. Add all the README files to your Git repository.
9. Commit your repository.
10. Push your commit to the server.
11. Go to the GitLab web interface and verify that the files have the right names and are in the right folders – i.e., verify that you have met the minimum submission requirements for this lab.

SSH Keys

To avoid having to enter your Git credentials every time you add a file or commit, you may set up an SSH Key on your own personal computer. Instructions on how to do this are posted here.

On the lab computers, the directory where you would need to save the ssh key gets wiped when you log out, so these instructions are only relevant to using your own personal computer.

Notes

Be familiar with adding one file at a time to Git, in addition to adding multiple files at a time. Students sometimes accidentally add files that they don't mean to (like adding files for previous labs) which end up costing them late hours.

This lab is quite short, but if you are unfamiliar with Git it can take some time. Start early and there should be no issues with finishing it well before the due date. This lab is really a check on the prerequisite knowledge required to succeed in this class; **if you cannot complete this easily, we strongly recommend you take a different class and come back another term when you are ready**.

This is how you will submit **ALL** labs in this class, and they will all have a similar set of minimum submission requirements. It is important that you learn to do this right at this point in time.

No further work in the class will be graded unless this lab is completed; this is our verification that you have been informed of our expectations of you. Following instructions carefully may seem arbitrary (to you) but is absolutely necessary both from an engineering and pedagogical viewpoint.

You should **commit early and often** (**at the very least once per day** you work on any lab). This ensures you lose minimal work in case of a malfunctioning computer or other unfortunate event. For this reason, extensions will **never** be issued due to a malfunctioning personal computer.

In addition, having many commits demonstrates your incremental progress and is your best defense against allegations of cheating.

Every time you make a change, add a file, modify a file, go ahead and commit and push. You will always be able to return to that place and never lose your work.

Grading Rubric

1 pt created all Lab{0,1,2,3,4,5} directories
1 pt created all Lab{0,1,2,3,4,5}/README.txt files
1 pt read syllabus and personal responsibility document
1 pt correct statement in Lab0/README.txt (check your spelling!)
1 pt included CruzID
1 pt date in the correct format, matches date of commit