**CS469 Data Structures and Algorithms**

**HOS00B: Git and GitHub**

3/29/2023 Updated by Christopher Sharp

01/04/2024 Reviewed by Anh Nguyen

09/25/2024 Reviewed by Shahid Khan

School of Technology and Computing (STC) @City University of Seattle (CityU)



**Learning Outcomes**

* Understand the concept of version control system (VCS)
* Create GitHub account
* Create GitHub repository
* Use Git, GitHub
* Do 1 add, commit and push git operation

**Before you start**

Version control system (VCS) is created to make it easy to store different versions of the project you are working on at any time (without any fixed rules, time or rules). You can easily restore an earlier record and compare the content with the current data to find the differences. In addition, it is also an effective tool to cooperate with many other people in the same project. In this module, we will learn about one of the most popular VCS which is Git and GitHub.

First, Git and GitHub are not the same. Git is a VCS while GitHub is a hosting server for software development version control. The way Git works is that it creates a **repository** to store each version of the project every time developers modify or add some features into the project. Developers use Git through terminal or command line.

However, Git only allows you to control your project version on your local machine. Therefore, if there are 2 or more people working on the same project, how do we connect the works of all the members? That is why we need a hosting server GitHub.

To learn more about Git and GitHub visit:

<https://codeburst.io/git-and-github-in-a-nutshell-b0a3cc06458f>

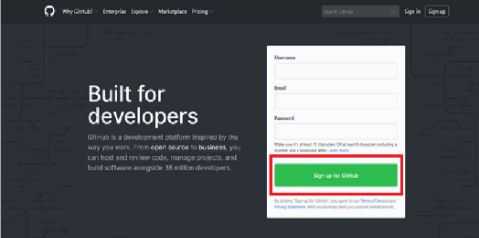
<https://git-scm.com/book/en/v2>

**Create GitHub account**

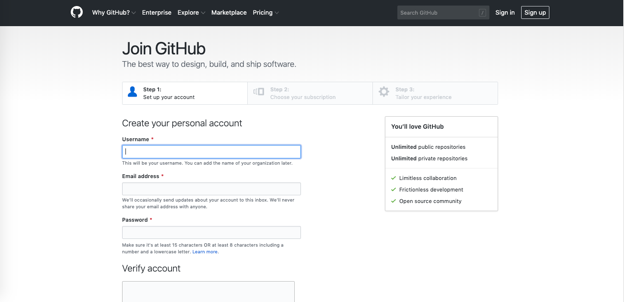
**Registering a GitHub account and GitHub repository**

If you already have a GitHub account, skip to step 7.

1. Visit <https://github.com/> and click on “Sign up for GitHub”



1. Filling in the form with Username, Email Address, Password, etc.



1. Click “Create Account” to proceed to the next page.
2. On Step 2, Choose your subscription as “Free” and click “Continue”.



1. On the welcoming page, choose your options then click “Submit”, or click on “Skip this step”



1. You will be asked to check your email and verify your GitHub account.
2. Go to github.com and login to your GitHub account.
3. Please submit your username to GitHub owner / maintainer.

**Joining as a collaborator in a GitHub repository**

You’ll gain access to your **CS469\_ON\_Winter\_2024** GitHub repository by clicking on the link under the HOS assignments in Brightspace.

You then follow the instructions to set up and reach your classroom repository.

Click on “Code” to go to the base level of the repository (this is the history page), copy and paste the URL of this web page to use below.

**Check Git**

Check if your local machine has git.

>>> **git --version**



If you have git installed (a version number comes up) then skip to the section “Cloning a Repository.”. Else, you’ll need to install git first.

*Note: Most Mac and almost all Linux users will already have Git installed, but not Windows users.*

**Installing Git**

**Linux**

Follow the instructions on [this page.](https://git-scm.com/download/linux)

If you’re on Ubuntu, use “apt-get install git” with “sudo” in front if needed, and for Fedora use “yum install git” or “dnf install git” – all in the terminal.

**Mac**

You should be prompted to install Git if you typed “git --version” above and it failed to find Git. If that doesn’t work, then try “brew install git” if you have Homebrew installed. Otherwise, follow the instructions on [this page](https://git-scm.com/download/mac).

**Windows**

You’ll need to download a dedicated Git for Windows app developed by the official Git maintainers. Follow the instructions on [this page](https://git-scm.com/download/win).

Once installed, you can use the included terminal Bash environment to input codes below; it fakes a Linux environment translating your commands to Windows command line inputs.

**VS Code for all operating systems**

We recommend using the VS Code environments for our assignments and most instructions and support assume you are using it. It provides an integrated development environment for productivity that lets you input terminal commands – including Git commands – without leaving the window.

Once you have Git installed, open VS Code. Go to Settings (File -> Preferences -> Settings in Windows or Code -> Settings -> Settings in Mac) and type “git: enabled” into the search bar. Look for **Git: Enabled** and check the box.

You may have to restart VS Code for the changes to take place. Once they do, you can bring up the integrated terminal with control + ` (the ~ key) on all three OSs.

**Cloning a repository**

1. Clone (copy) the remote (online) repository to your local machine

>>> **cd ~**

>>> **git clone** [URL from above]

The URL will be something like https://github.com/cityuseattle/[repository-name]-[yourGHname]

(If there’s anything after your GitHub username at the end, like you’re another directory down, then either go up or just delete that extra part.)

*Note: It may give an error at this step if you’ve never set up Git before. Try these two commands first and then retry cloning the repository. Use your name and school email to configure these global variables.*

>>> **git config --global user.name “[Your name]”**

>>> **git config --global user.email “[Your email]”**

Below is an example output of a successful download with a git clone command, though yours won’t have a “.git” at the end.



1. Go to the project folder

>>> **cd [repository-name]**

**Git commands**

**Saving your credentials**

To avoid logging to your username and password every session, please execute the command.

>>> **git config credential.helper store**



**Branching out from master branch**

1. Create your own branch

>>> **git checkout -b** [new branch name]

Please replace [new branch name] with your preferred name. You can follow this [post](https://medium.com/@abhay.pixolo/naming-conventions-for-git-branches-a-cheatsheet-8549feca2534) for branch naming conventions.

For an easy example, you can use your Please replace your *firstname* and *lastname*.

Make sure you see console output “Switched to a new branch”

If you don’t do this step, that’s fine, but in all your assignment submissions use “git push origin” without the final “master” at the end, which is in the instructions.

*Note: Creating a master branch and other branches is good practice in real-world projects with multiple collaborators, as you can review and merge changes to files while tracking changes, reverting if needed.*

**Creating your own content**

1. You may wish to create a folder with your name inside the HOS assignment folder.

>>> **mkdir [FirstnameLastname]**



1. Go to your folder

>>> **cd [FirstnameLastname]**



**Push your first file**

1. Here’s an example of how to submit using an empty README.md file.

>>> **touch README.md**



The “touch” command creates empty files. Use the “ls” command to double check it was created.



1. Add the file to staging

>>> **git** **add .**

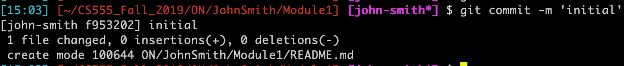
*Note: The period is to say, “all of the files in here.”*



1. Update your local repository. Commit the file and add a message

>>> **git commit -m “[your commit message]”**

*Note: Typically keep your commit message short but basically descriptive of the change.*

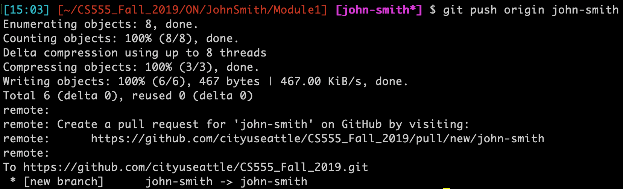


1. Push your branch to GitHub repository

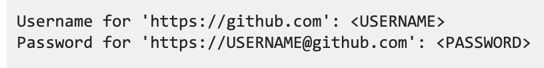
>>> **git push origin**

*Note: Add “master” to the end if you created a master branch earlier which is good practice in*

*real world projects.*



1. When prompted for GitHub username (your email with GitHub) and password. Enter them



You MADE it! 🚀