GitHub reference

This document provides a basic introduction and reference to the utilization of GitHub in an academic environment for collaboration on programming projects and/or providing course document to students in a course.

US Coast Guard Academy 2020

Table of Contents

[2 Introduction 2](#_Toc31839410)

[2.1 What is GitHub 2](#_Toc31839411)

[2.2 Setup 2](#_Toc31839412)

[2.2.1 Creating Your Repository 2](#_Toc31839413)

[2.2.2 Software 4](#_Toc31839414)

[3 Utilization of GitHub in the Classroom 8](#_Toc31839415)

[3.1 Sharing Documents With Peers and Students 8](#_Toc31839416)

[3.1.1 Putting Documents on the Repository 8](#_Toc31839417)

[3.1.2 Cloning Repositories 8](#_Toc31839418)

[3.1.3 Updating Documents 8](#_Toc31839419)

[3.2 Collaborating on Projects 8](#_Toc31839420)

[3.2.1 Collaborators 8](#_Toc31839421)

[3.2.2 Committing Changes 8](#_Toc31839422)

[3.2.3 Pushing and Pulling from the Origin 8](#_Toc31839423)

[3.2.4 Branching 8](#_Toc31839424)

[3.3 Providing a Template 8](#_Toc31839425)

[3.4 8](#_Toc31839426)

[4 Project Management 9](#_Toc31839427)

# Introduction

## What is GitHub

GitHub is software that has been developed for primarily programmers. Whenever more than one programmer must collaborate on a project, there has always been the issue of sharing code from a project with others. GitHub allows for others to replicate a programming environment while maintaining a certain level of integrity to the layout of the paths and the current state of the project. If using the software, you can keep the most up-to-date replica of the files stored on the GitHub server for your repository. Any changes on GitHub will reflect on your filesystem. Likewise, any changes made on your filesystem will reflect on GitHub. Both of these require pushing and pulling to the origin which will be covered later in this guide.

What does this mean for us as programmers? Let’s take the example of having two programmers working on a project together. Whenever an update to code is made by one programmer, there is an issue of getting the updated code to his or her partner. They could maintain a string of emails; but, this quickly gets cluttered and disorganized. Using GitHub, when an update is made to the code, the programmer could commit his changes to the GitHub project and the other programmer could pull this change from the project origin and have a copy of the most recent code directly to his or her system.

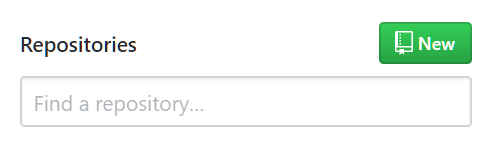
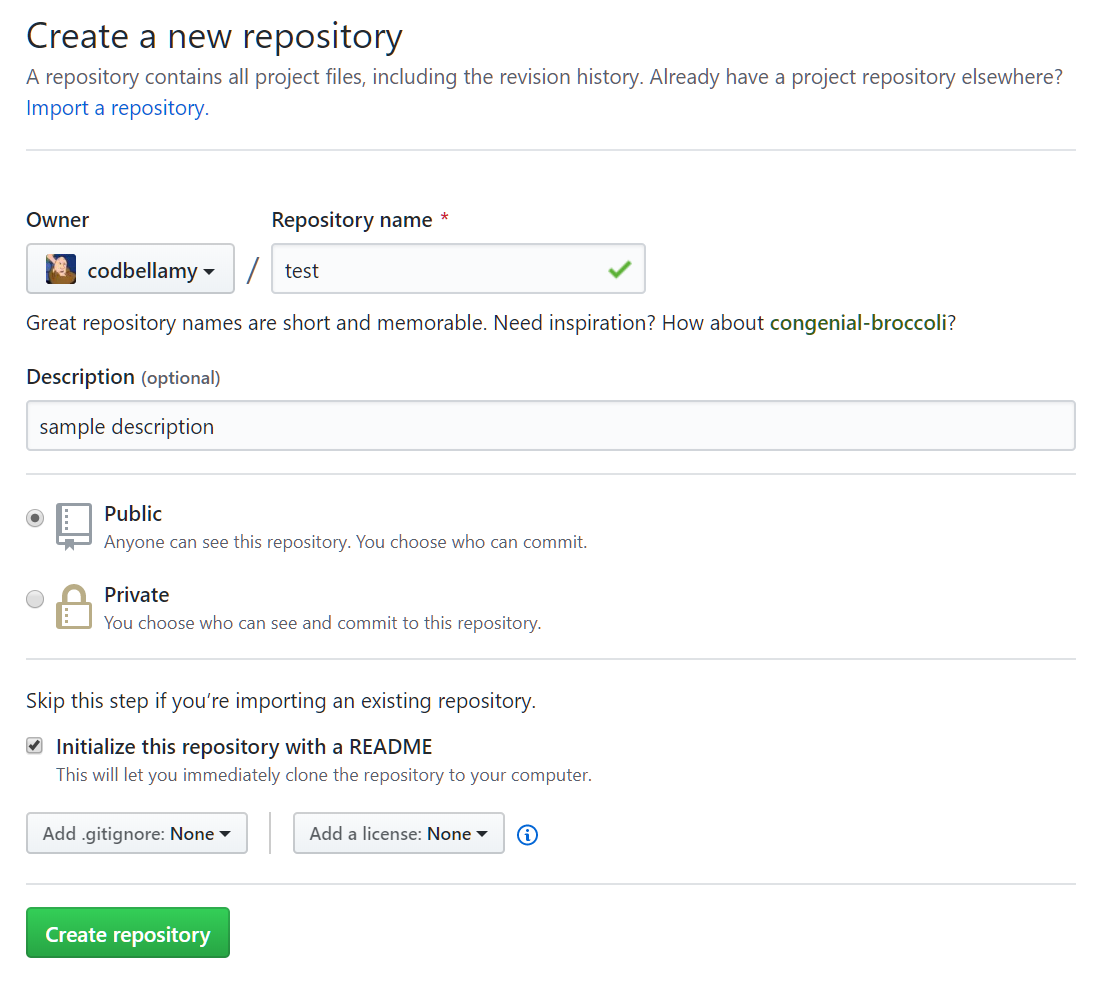
In an academic sense, this could be applied to a classroom setting. A professor could maintain administrative files in one folder and course materials such as code and power point presentations in another folder. If a student wanted a copy of these materials, the student could simply fork the branch to their own filesystem and have an exact copy of the entire project. This includes code, administrative documents, power points, and due dates.

## Setup

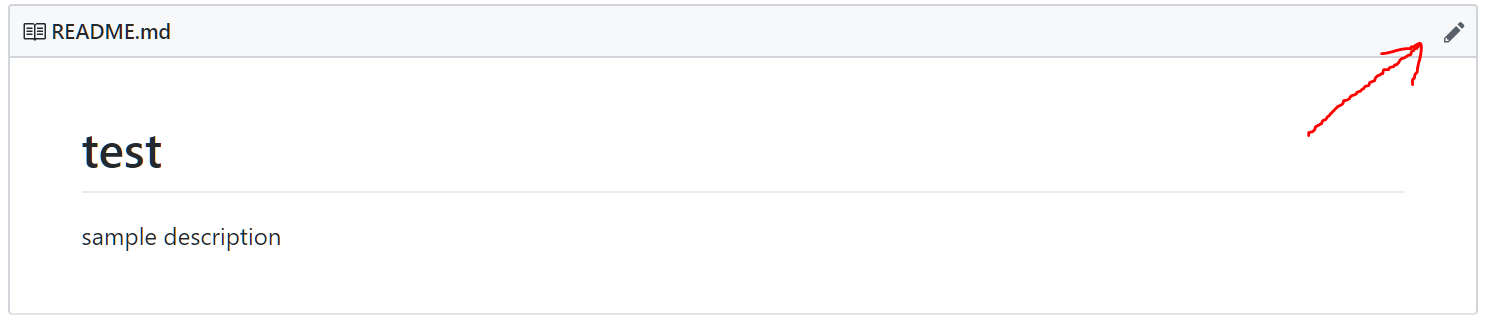
The following steps are to help you set up GitHub on your system. Although downloading the GitHub software is not necessary, it is highly recommended. Prior to completing any of these steps, it is mandatory for project collaborators to maintain an account. Standard users that will be cloning the project on their filesystems are not required to create an account. To create an account, go to <https://github.com/>.

### Creating Your Repository

Project collaborators have the highest authority when it comes to managing the project or course files. Like D2L, only teachers and TAs should be added as project collaborators.

1. Sign into your account
2. At the top left of the screen by the “Repositories” header, click “New.” 
3. Create a repository name, enter a basic description of the project, and select “Initialize this repository with a README” 
4. Select “Create repository”

That’s it! You’ve successfully created your repository in GitHub. The README file is a text file that typically contains information about the project/course. You can edit this by clicking the pencil icon on the top-right of the dedicated README box.

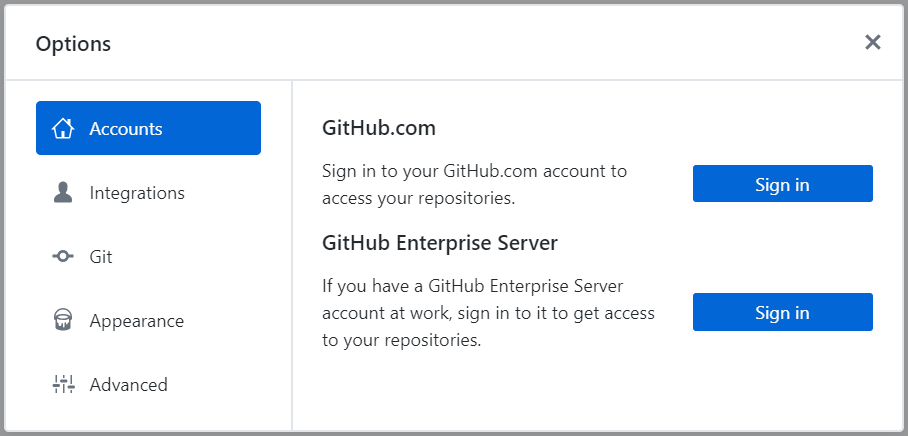
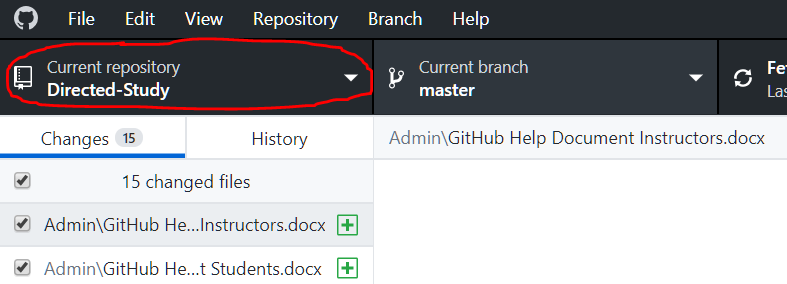


### Software

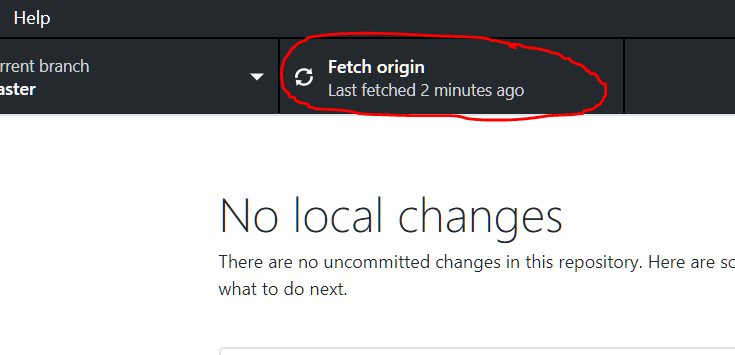
There is only one program that makes accessing, updating, and cloning projects marginally easier. This software is called “GitHub Desktop.” It is important to note that most of the tasks that can be done on the software can be done through GitHub’s website. The software can be used to keep an exact replica of the server’s files on your system. The software can be downloaded at <https://desktop.github.com/>. After installing GitHub Desktop, sign into your account if you are a project collaborator. All your project permissions will be translated to the software automatically. More in-depth tutorials on how to use the software will be covered in later sections.

#### Connecting GitHub to Your Repository

Connecting GitHub to your repository is very simple. Assuming you have created your repository as mentioned in 2.2.1, you can connect to your repository in a few simple steps.

1. Open GitHub Desktop
2. Go to File > Options > Accounts
3. Under “GitHub.com,” click the first “Sign In” option 
4. Enter your credentials and sign in
5. After being redirected to the main program interface, click the dropdown menu at the top-left to select a repository
6. Click “Add” > “Clone Repository”
7. Select the desired repository and click “Clone”

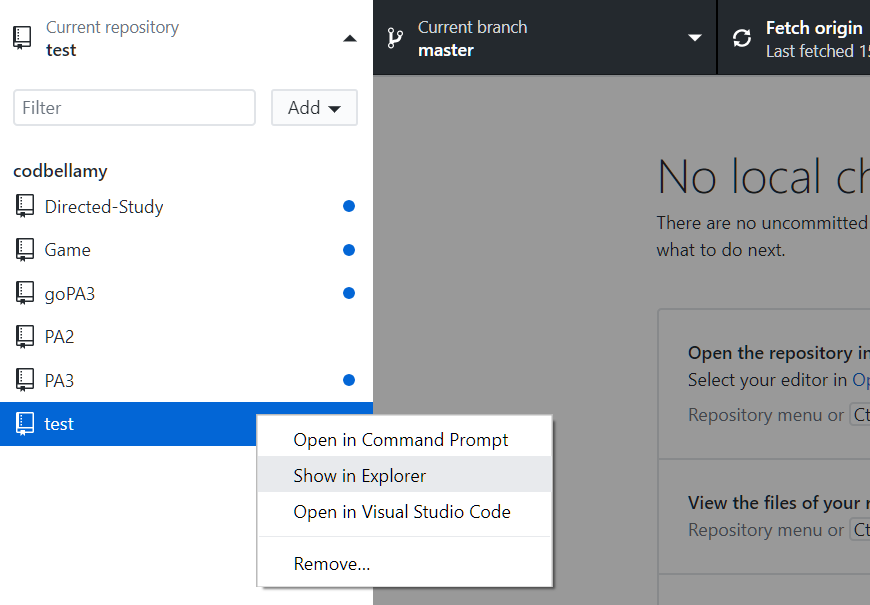
Since you are already signed in, a list of your current, active repositories with GitHub will be listed under the “GitHub.com” tab when selecting repositories to clone. You are now successfully connected to your repository.

1. At the top right, select “Fetch Origin” 

If there are any differences between the files posted on GitHub and your filesystem, a pull request will be generated. Click “Pull from origin” if prompted. A pull request is a request to the GitHub server to download any commit differences between your machine and the server. If your machine is behind, a pull request is generated and any files necessary to catch up will be downloaded upon clicking “Pull from origin.” If your machine is ahead of the origin, a push request will be generated. Push requests are covered in 3.1.1. If done successfully, your machine will have the exact same structure as the files stored on GitHub.

#### Accessing the Files (Windows)

It is important to note that the files that will be on your desktop will be a clone of the files in your repository. This means that any changes that are done on GitHub will reflect on your desktop. Likewise, any changes that are made on your desktop will reflect on your GitHub repository. To access the files on your machine, follow the steps provided.

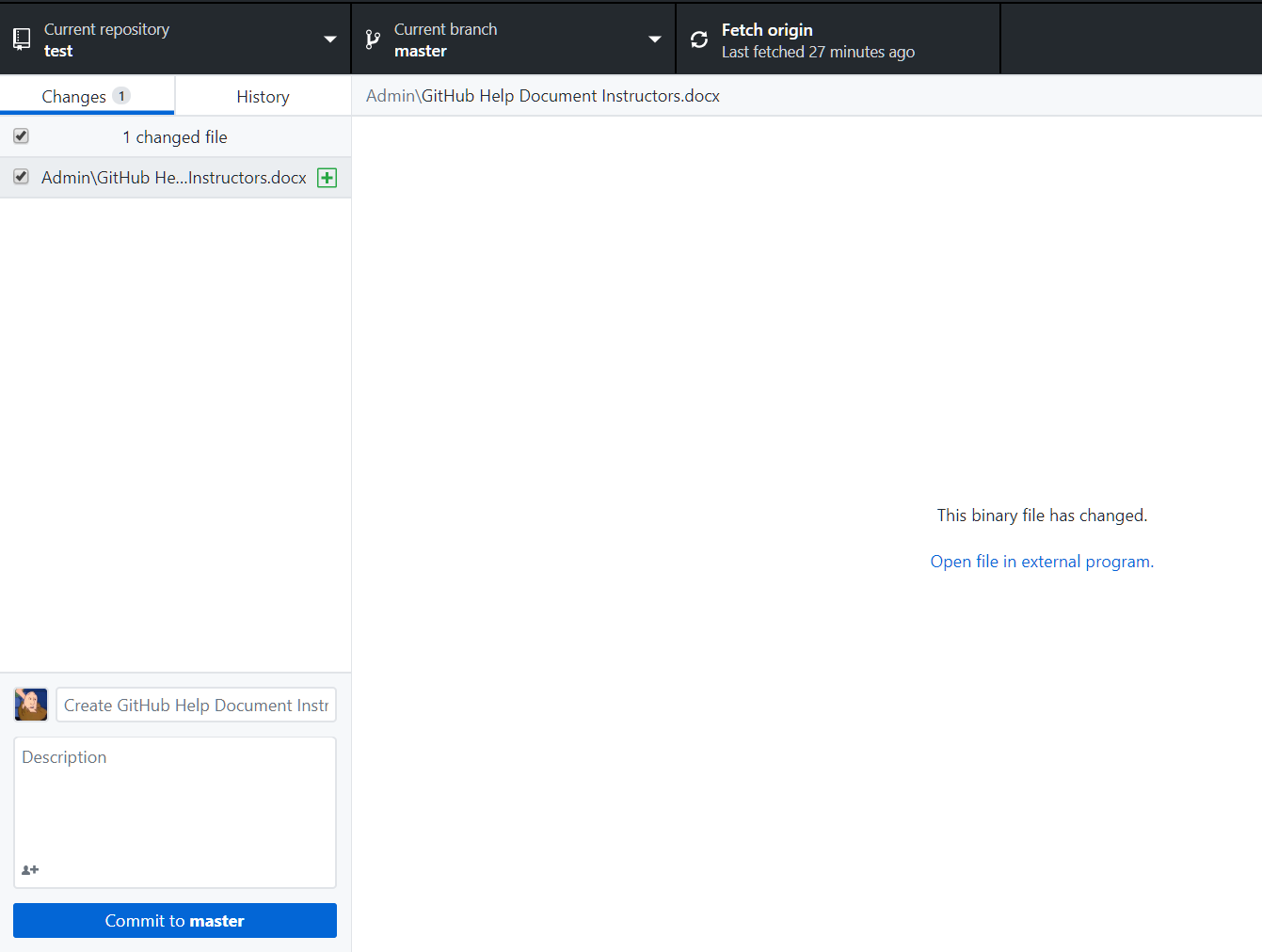
1. Click the “Current Repository” tab on the top-left.
2. Right click on the active repository.
3. Click “show in explorer” 

This will open the corresponding directory to the repository on your machine.

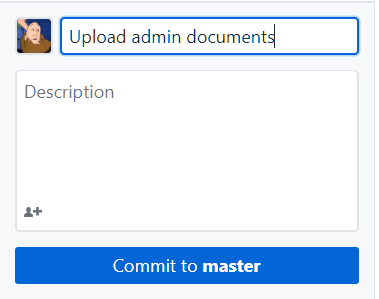
#### Maintaining Sync with GitHub

Now that you have successfully cloned your repository, we want to maintain a certain level of sync with your repository on GitHub’s servers. Any changes that are made on your filesystem **WILL NOT** automatically push to GitHub. In the following example, we will upload this document to our repository in a folder called “Admin.” These steps can be done to create any structure for your repository. All folders and files will maintain the same state that they are uploaded.

With our repository open, create a new folder and place the document inside. After doing this open the GitHub Desktop application. You will notice the changes that you’ve made reflect automatically within the software.



If you are satisfied with these changes, we need to commit these changes to master. To do this, we will create a title and an optional description in the bottom left of the screen. If there are any documents that you do not want to upload, you may simply uncheck them from the list on the left or remove them from the directory entirely.



After clicking “Commit to master,” we finally must push these changes to the server. This can be done by clicking “Push origin” on the top-right of your screen. Once completed, our changes will reflect on GitHub.com in our repository.

# Utilization of GitHub in the Classroom

## Sharing Documents With Peers and Students

Sometimes, it is useful to simply provide documents to your students like the format that D2L offers. When providing documents to students, it is not necessary for the students to have a GitHub account so long as your project is public. Create a repository strictly for your course and be sure to only add instructors and TAs as necessary as project collaborators. Your students will be able to view the files and download them as needed.

### Putting Documents on the Repository

Once your repository is set up, adding documents to the repository is relatively straight forward barring managing the file structure (directory layout). The following steps provided will walkthrough creating a folder and adding documents to your repository.

#### GitHub Desktop

See 2.2.2.2-3 for a detailed description on how to upload folders and files to your repository.

#### Website

Adding files to your repository is slightly more complex compared to uploading documents with the software. Additionally, creating folders is even more non-intuitive. Again, I highly recommend using the software for convenience. If downloading the software is not possible (using a .MIL or some .EDU machines) then the following steps will show how to create a folder and upload a file.

### Cloning Repositories

### Updating Documents

## Collaborating on Projects

### Collaborators

### Committing Changes

### Pushing and Pulling from the Origin

### Branching

## Providing a Template

## 

# Project Management