GIS4x07 Git Workflow

Intro

This document introduces version control with Git and outlines the process of using Git and GitHub for GIS4x07 Exercises. To make best use of this document, there are two prerequisites:

1. The reader has a GitHub Student account (see Appendix A: GitHub setup for Students)
2. Git for Windows is installed (see Appendix B: Installing Git on Windows)

This document covers the simplest possible Git Workflow. It does not yet cover some of the other fundamental version control concepts such as branching, merging, reverting, diffing, etc.

To practice this “pull – edit – add – commit – push” workflow, a GitHub repository could be cloned into two separate folders to simulate two different users on different computers.

Contents

[What is Git 1](#_Toc462868610)

[Create a new repository in GitHub 1](#_Toc462868611)

[Add Collaborators 3](#_Toc462868612)

[Clone the GitHub repository to a local repository 4](#_Toc462868613)

[Pull changes from remote repository to local repository 5](#_Toc462868614)

[Add and Commit new/updated files to the local repository 6](#_Toc462868615)

[Push changes in local repository to remote repository (GitHub) 7](#_Toc462868616)

[Appendix A: GitHub setup for Students 9](#_Toc462868617)

[Appendix B: Installing Git on Windows 10](#_Toc462868618)

# What is Git

Git is a version control system. The Git version control system has two main parts:

1. A repository – to store a set of files and the history of changes made to those files.
2. A Git client – a software that interacts with the repository

A local Git repository is a set of files in a .git sub-folder that keeps “snapshots” of the parent folder’s files and folders through time. The .git folder’s parent folder is called a “working folder/directory” or “working copy”.

GitHub is a web-based Git repository that stores the same set of files as the local Git repository. Changes to the local or remote repositories can be synchronized using Git client software.

The workflow for setting up remote and local repositories for exercises is:

1. **Create a new repository in GitHub**
2. **Add Collaborators**
3. **Clone the GitHub repository to a local repository**

After the setup of local and remote repositories, the workflow is:

1. **Pull changes from remote repository to local repository**
2. Make changes to your local repository (add/delete/modify files)
3. **Add and Commit changes to the local repository**
4. **Push changes from the local repository to the remote repository (GitHub)**

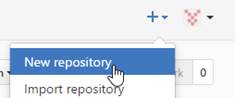
Pulling changes from the remote repository ensures you have any changes from your collaborators before you begin your work making changes to your local repository.

The bold items in the above will be covered in detail in the rest of this document.

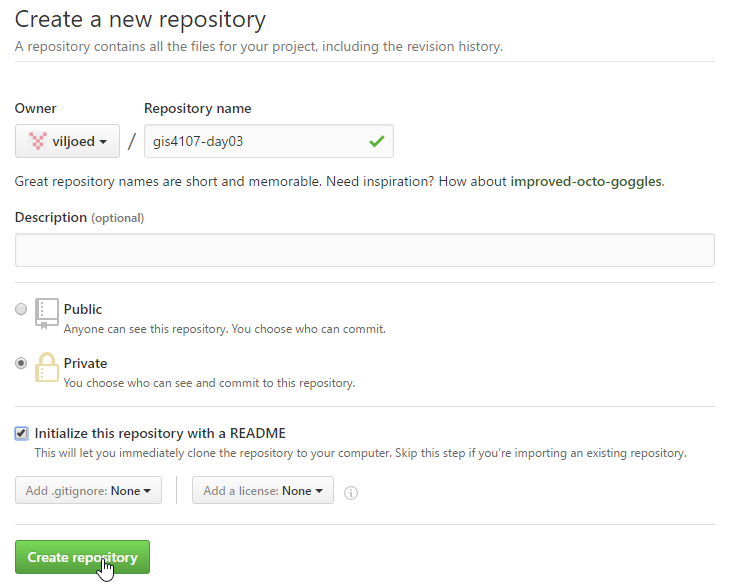
# Create a new repository in GitHub

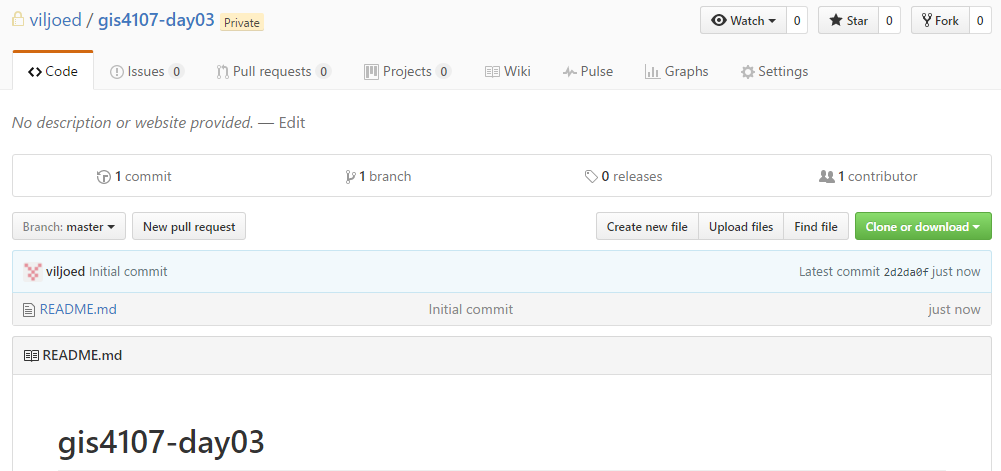
Login to GitHub

Select “New repository”:



1. For Repository name, use the pattern “course-dayN” (e.g. gis4107-day03)
2. Set it to be a Private repository
3. Check Initialize this repository with a README. This will allow you to clone this GitHub repository to a local repository.

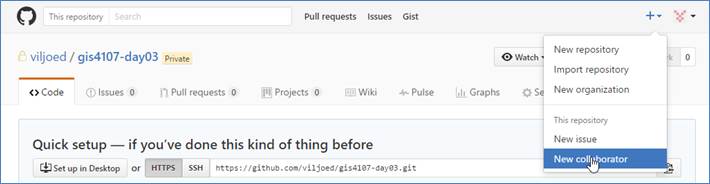




# Add Collaborators

You have now successfully created a Private repository and it will be the active repository.  Add viljoed and your partner as collaborators as follows:

Select New collaborator …



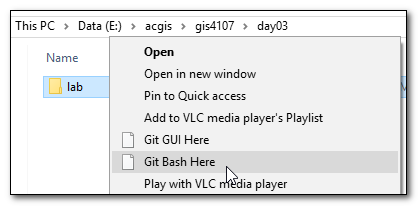
Search / select /add collaborator …



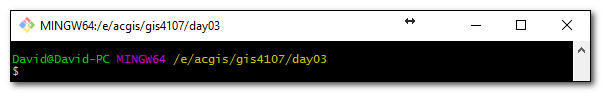
Search / select / add other collaborators

# Clone the GitHub repository to a local repository

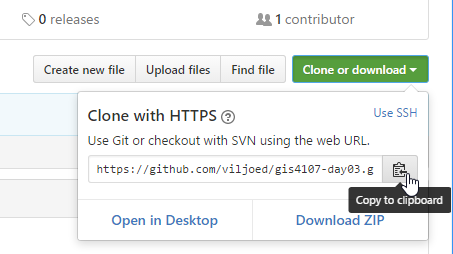
A local repository will be created as a .git hidden folder in a folder of your choosing. In this example, E:\acgis\gis4107\day03\lab. Right-click on its parent folder (i.e. E:\acgis\gis4107\day03) and select Git Bash …



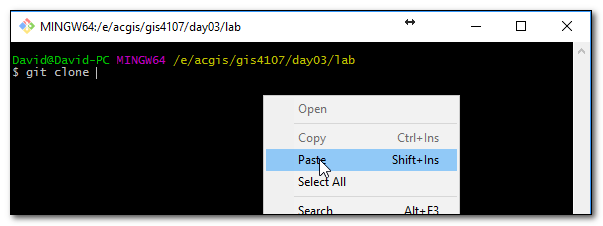
This will open the Git Bash shell …

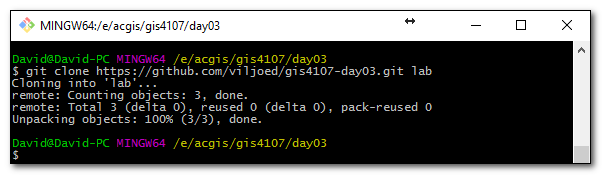


In GitHub, copy the GitHub URL by clicking Clone or Download and then clicking Copy to clipboard.

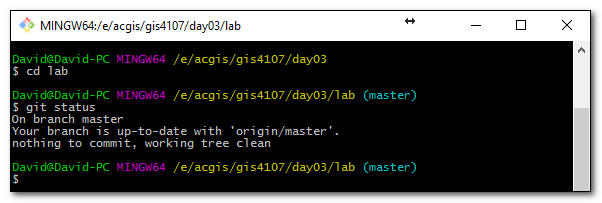


In the Git Bash shell, enter git clone in the Git Bash shell and then right-click and select Paste

 Then enter “lab” as the target folder and press Enter



Use the cd command to change to the lab folder and then enter “git status” to confirm the repository was cloned properly and everything is up to date.

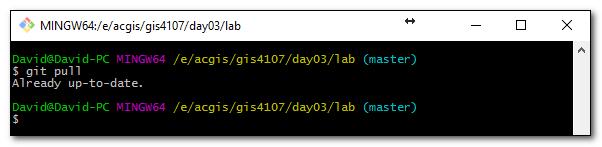


The local repository is now ready to pull/push updates from/to the GitHub repository.

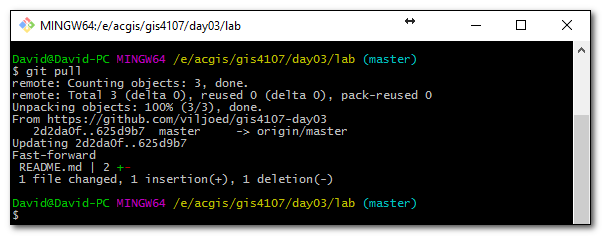
# Pull changes from remote repository to local repository

The Git pull command will update the local repository with any changes that have been made to the GitHub (remote) repository.

The following shows the local repository is up-to-date with GitHub



If changes had been made, the pull would display something like:



# Add and Commit new/updated files to the local repository

One of the first things to do in setting up exercises is to create an exercise sub-folder in the same folder as the .git folder (DavidV in the example shown below). Your folder name would be more like BobMandDougM.

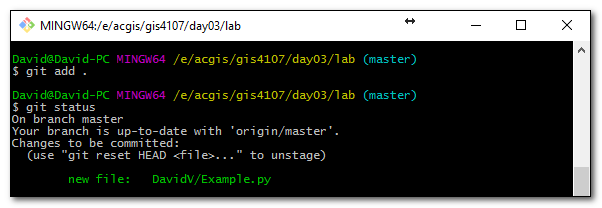


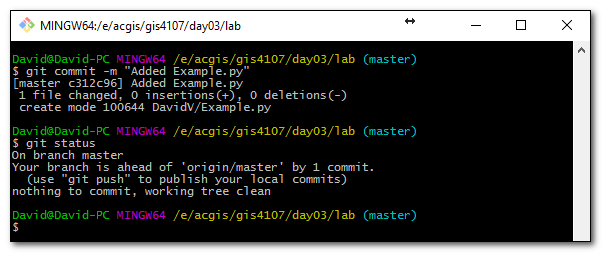
In the exercise folder, I created an Example.py file. If the folder was left empty, it would not be added/committed to the repository. The creation of Example.py in the DavidV folder is a change in the working copy. To add this change to the local repository, use the Git add and commit commands. The Git add command will add a file or multiple files to a staging index that keeps track of which files will part of the next commit. The Git commit command will add those changes to the repository. You can use a “.” to specify add all files that have changed to the staging area:  
$ git add .  
$ git commit -m "Added Example.py"

Alternatively, individual files can be specified:

$ git add DavidV/Example.py

$ git commit -m "Added Example.py"

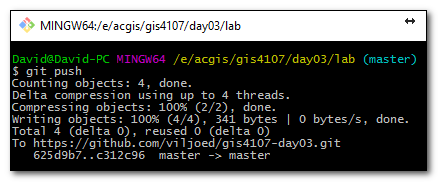




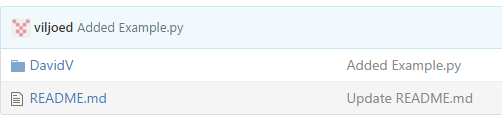
Now the working tree is clean, the local repository can be pushed to GitHub.

# Push changes in local repository to remote repository (GitHub)

The Git push command will push local changes in the repository to the remote repository.



Refreshing the GitHub web page, you can see the DavidV/Example.py file has been added.

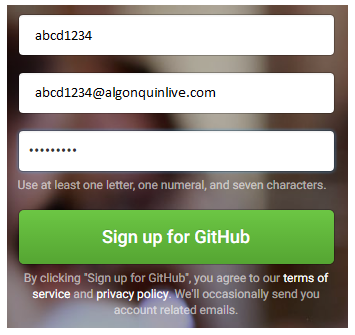


Now your collaborators can pull these changes to their local repository.

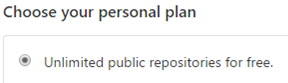
# Appendix A: GitHub setup for Students

Go to <https://github.com/>

Enter Algonquin login id, e-mail, and a password you want to use for GitHub



In Step 2, “Choose you plan”, select Unlimited public repositories for free.



Click Continue

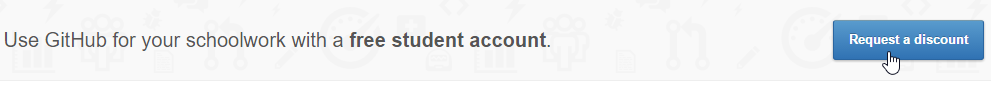
In Step 3, “Tailor your experience”:

* How would you describe our level of programming experience? Select whatever you want :o)
* What do you plan to use GitHub for: Select “School projects”
* Which is closest to how you would describe yourself: Select “I’m a student”
* What are you interested in? Python
* Click Submit

On getting started page, click “Skip and continue to your dashboard”

Go to <https://education.github.com/>

Scroll to



Click Request a discount

Step 1: Tell us what you need

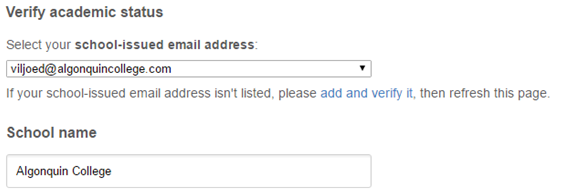
* Which best describes you? Select “Student”.
* What are you looking to get a discount for? Select “Individual account”.

Click Next

Step 2: Tell us about you

Name: enter your name

Your school-issued e-mail address should be selectable, e.g.



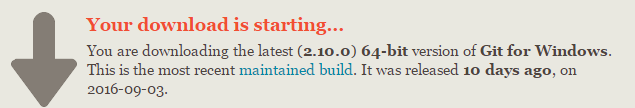
For “How do you plan to use GitHub?”, you can put whatever you like. Perhaps something like:

“Learning about version control by using GitHub for course exercises and my own projects.”

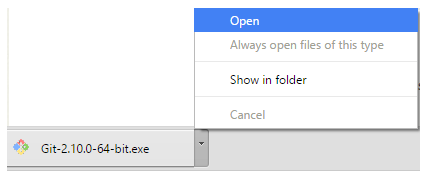
Click Submit request

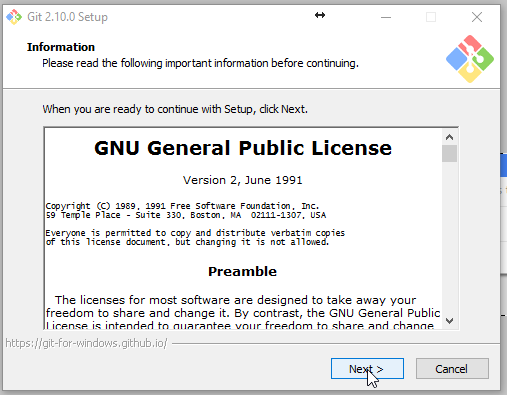
# Appendix B: Installing Git on Windows

Go to <https://git-scm.com/download/win>



The installer should appear in Downloads. Click to Open

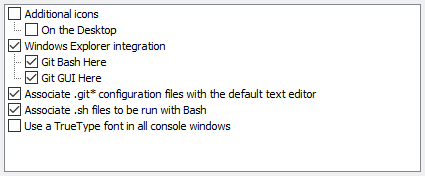




Click Next

Click Next to Accept default install path (e.g. C:\Program Files\Git)

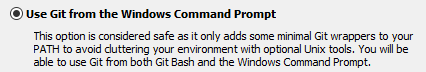
Make sure the following components are selected:



Click Next

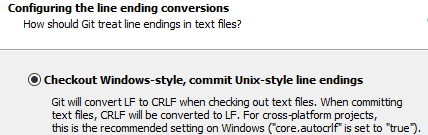
Click Next to accept Start Menu Folder (Git)

In “Adjusting your PATH environment”, select



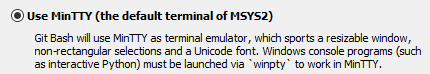
Click Next

In “Configuring the line ending conversions”, select



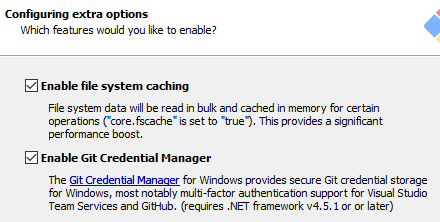
Click Next

For “terminal emulator”, select



Click Next

For “extra options”, leave defaults selected:



Click Install

Installation is complete.

After installing, the first task is to set some global configuration settings by right-clicking on any folder and selecting “Git Bash Here …” and issuing the following config commands:

$ git config –global user.name "FirstName LastName"

$ git config –global user.email "somebody@somewhere.com"

On Windows 7+, this information gets written to

C:\Users\<username>\.gitconfig