Git Lab 1 – Git Commands

git init

Creates a new repository in the folder we created in step one.

Creates a hidden file in the root directory .git which can be brought to view using the view tab and then checking the “Hidden Folders” check box.

This creates a master repository on the machine that the user is sitting at.

A screenshot of a computer

Description automatically generated

Git add

This command prepares a file to the next commit by sending it to the staging area. This staging area is where all the files that you want to include in your next commit need to be added to. Using the git status command you can check what files have and haven’t been added to the staging area.

I have added both my text documents and they are now ready to be committed



git status

Gives us a readout of what files are in the staging area and are ready to be committed.

Files shown in Green have successfully been added using the git add command and are in the staging area now ready to be committed. Files shown in Red have not been added to the staging area and won’t be committed with the next commit.

This is before adding, the file is in red

A screen shot of a computer

Description automatically generated

This is after using the add command to add the file to the staging area

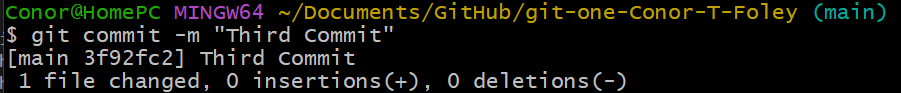
A screen shot of a computer code

Description automatically generated

git commit

This command creates a snapshot of the staged files. In other words, a new revision point, a point along the projects timeline where the changes made between commits can easily be viewed.

Putting -m after the commit command allows me to add a comment to the commit, in this example “Third Commit” which allows me to keep track of what has been changed and at what time along the timeline



git branch -M master

This command has allowed me to change the name of my main branch to “master”

The git branch command is use for branch management. It can list the branches I have, create a new branch, delete a branch or rename an existing branch.

Note the branch name has switched to master at the end of the second line.

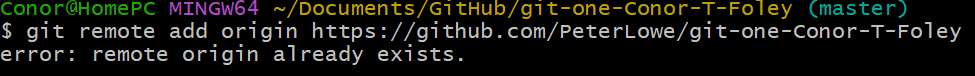
A black background with yellow text

Description automatically generated

git remote

This command creates a connection between my local repository that I created with git init and the remote repository created by Pete on GitHub. Note the error is because this connection has already been established at an earlier time and this screenshot is for demonstration purposes only.

The command reads as follows, git remote tells git you intend to setup a remote repository connection, add origin tells git this is where the remote repository will be and finally you add the url ofd the remote repository at the end.



git push

The git push command sends content from the local repository that we created with git init to the remote repository which was created on git hub for us by Pete.

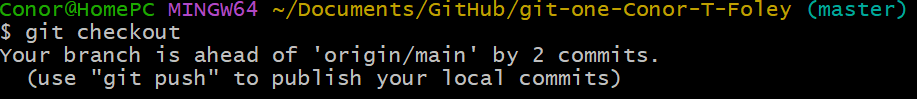
We can do this because we have established the connection to the remote repository in the last step using the git remote command.

A screenshot of a computer screen

Description automatically generated

git checkout

This command allows me to see if the branch I am currently working on is up to date with the main branch. This will tell me if there are “dirty files” or files which have been edited and differ from the version of those files stored along the main branch. You can see the checkout command is telling me the branch I’m working on is 2 commits ahead of main. These are the two commits I have done from home today so far.



By using the command git checkout main , I have switched back to the main branch and now when I commit and push, the files will be in the correct repository.

A screen shot of a computer

Description automatically generated

git merge

This command allows me to merge branches to the main branch of my repository.

Since I have been working on both a main and master branch, I have now merged them together so all files and changes appear along one timeline.

