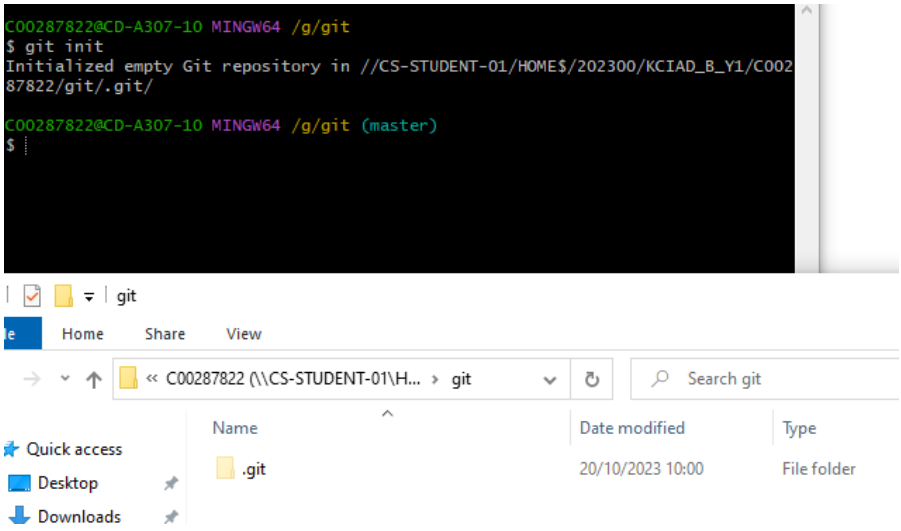
Git Guide

By Shane Moran

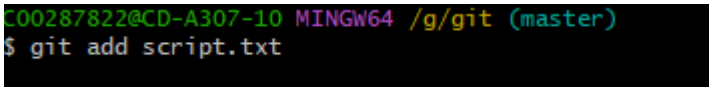
1: git init

The “git init” command creates a new empty git repository. You need to start an empty repository in order to access any further commands. As shown in the image below, this command will create a .git folder within the folder that you have opened GitBash in.



2: git add

After you have created a new repository, you can add files to this repository with the “git add” command. This will only add the selected files to your next commit, you will usually add and then commit straight after.



3: git status

The “git status” command lists all files in your working directory, staging area and your untracked files. The purpose of this command is to show you what files have been committed, and what files are due to be committed. As seen in the image below, I have not made a commit, I have one file ready to be committed, and I have two untracked files.

```
C00287822@CD-A307-10 MINGW64 /g/git (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   script.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        gitShane.docx
        ~$tShane.docx
```

4: git commit

When you are ready to transfer your stages files to your repository, you do this through the “git commit” command. You need to make sure to add “ -m “comment”” at the end of the git commit command, if you don’t add it, it will bring you into the VI window.

```
C00287822@CD-A307-10 MINGW64 /g/git (master)
$ git commit -m "Second commit"
[master 14cd974] Second commit
 Committer: (Student  C00287822) Shane Moran <C00287822@setu.ie>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 gitShane.docx
```

5: git log

After a git commit, you can check if the commit was successful with the “git log” command. This will bring up a portion of text that includes the node ID, the person who done the commit, and the date and time that the commit was done. The main purpose of this command is so you and your colleagues can pinpoint who made what commit.

```
C00287822@CD-A307-10 MINGW64 /g/git (master)
$ git log
commit 14cd97418d7c100a37d296e84781c87626c8ed15 (HEAD -> master)
Author: (Student  C00287822) Shane Moran <C00287822@setu.ie>
Date:   Fri Oct 20 10:32:29 2023 +0100

    Second commit
```