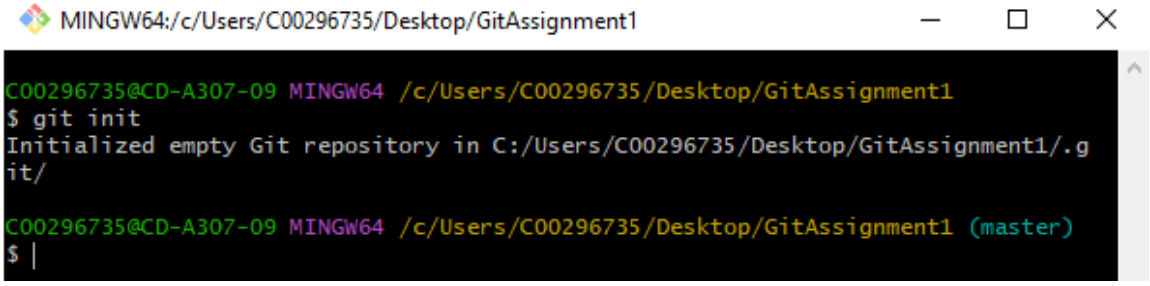
Git Init

Creates a new repository within a folder (appears as hidden .git file when view hidden files is enabled). This allows you to continue with other commands.

Syntax:

git init (Creates a new local repository within the current folder)



Git Status

Displays the status of files within the folder, such as new files, modified files and files that are added/staged. New/modified files are displayed as red and added/staged files are shown as green.

Syntax:

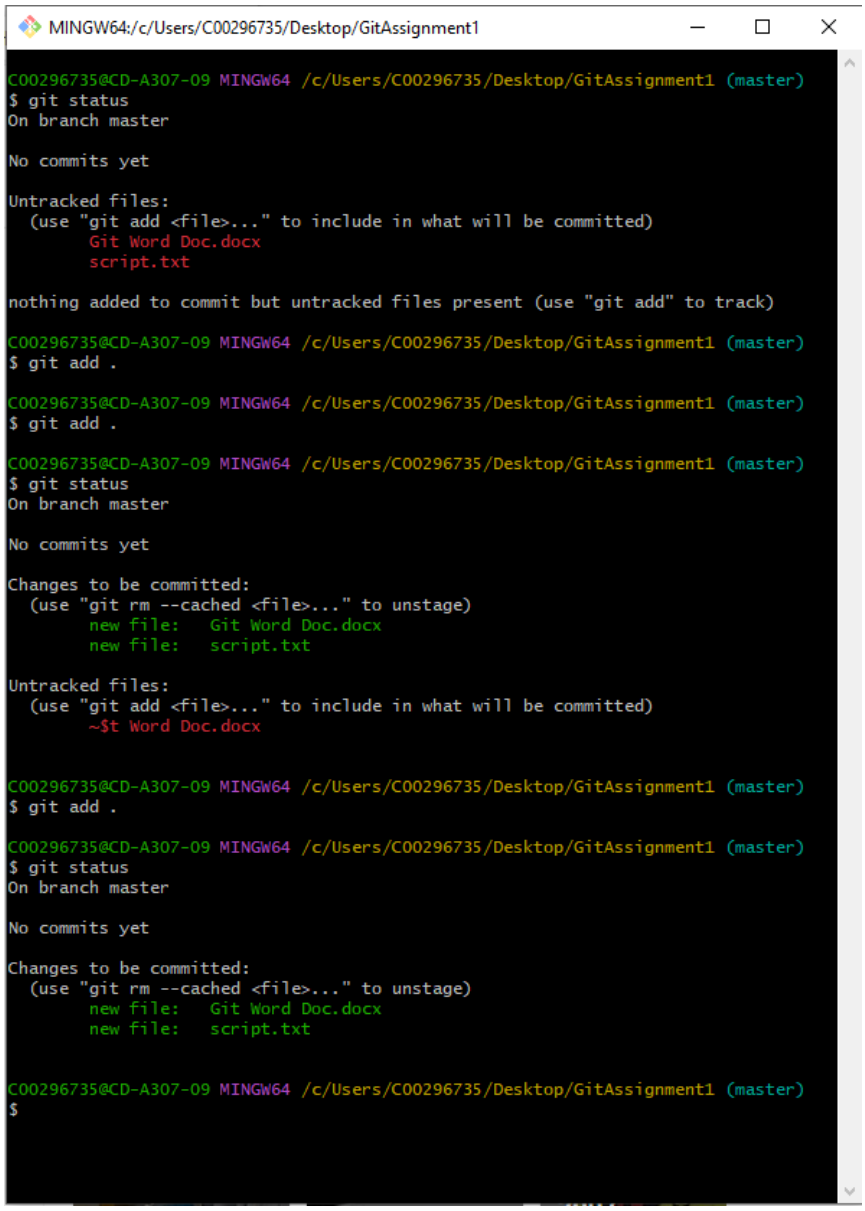
git status (Lists all modified, added and current files)

Git Add

Adds new/modified files to the index and stages them to be committed using the git init command. It is advisable to do another git status command to confirm that the git add was successful. MUST BE DONE BEFORE COMMIT.

Syntax:

git add <file name/.> (Adds the selected file or all files to be staged for commit)



Git Commit

Creates a new commit using the staged files within the index and a log message documenting the changes. This commit will be able to be seen on GitHub when git remote is used later.

Syntax:

git commit (Enters editor mode/multiline comment)

Git Log

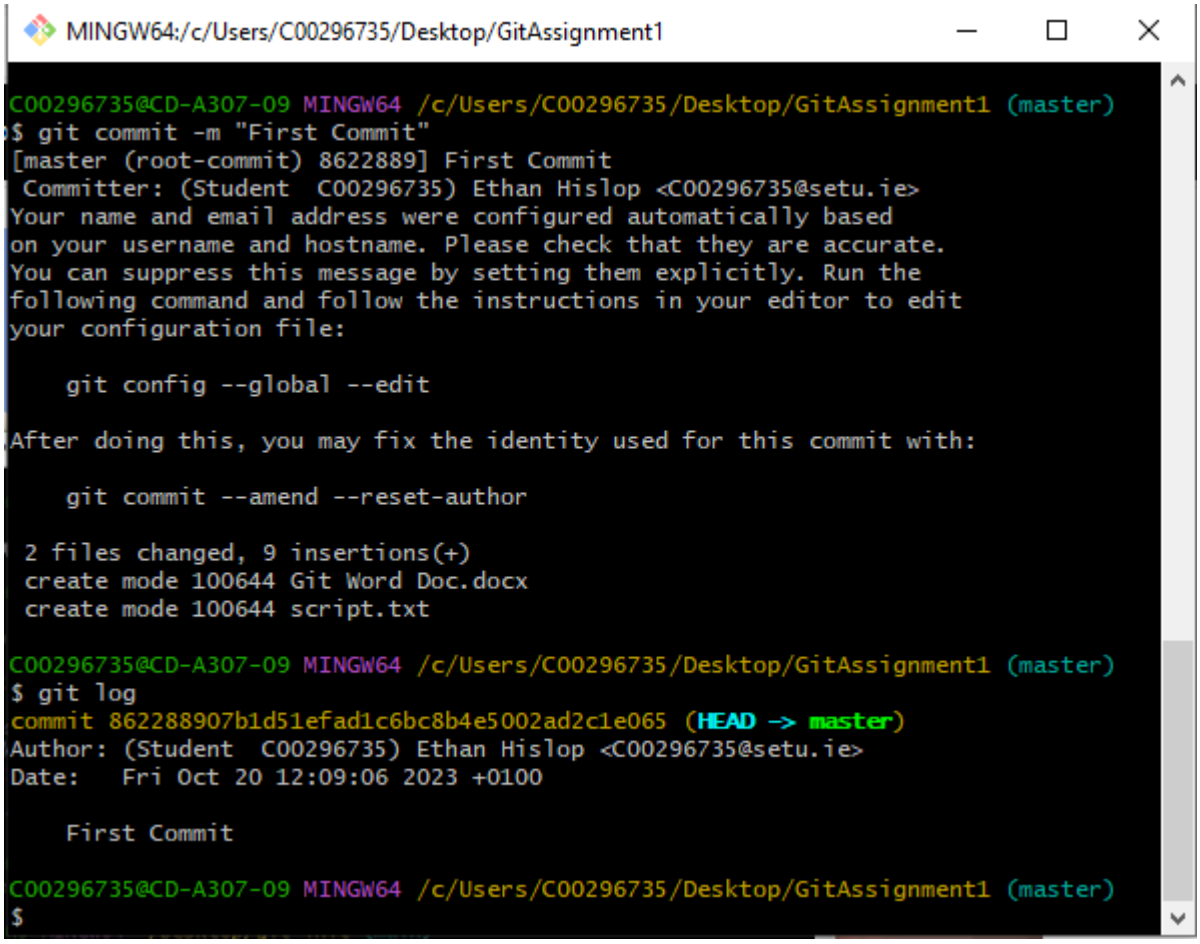
Shows the logs within a commit, such as message logs included with a commit command (as shown below).

Syntax:

git log (Display the entire commit history) [<space> for more and <q> to exit]

git log -stat (Shows the altered files and line changed)

git log -graph (Draws up branch paths)



```
MINGW64:/c/Users/C00296735/Desktop/GitAssignment1

C00296735@CD-A307-09 MINGW64 /c/Users/C00296735/Desktop/GitAssignment1 (master)
$ git commit -m "First Commit"
[master (root-commit) 8622889] First Commit
 Committer: (Student  C00296735) Ethan Hislop <C00296735@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

 2 files changed, 9 insertions(+)
 create mode 100644 Git Word Doc.docx
 create mode 100644 script.txt

C00296735@CD-A307-09 MINGW64 /c/Users/C00296735/Desktop/GitAssignment1 (master)
$ git log
commit 862288907b1d51efad1c6bc8b4e5002ad2c1e065 (HEAD -> master)
Author: (Student  C00296735) Ethan Hislop <C00296735@setu.ie>
Date:   Fri Oct 20 12:09:06 2023 +0100

    First Commit

C00296735@CD-A307-09 MINGW64 /c/Users/C00296735/Desktop/GitAssignment1 (master)
$
```

Git Branch

Creates a new branch within the current local repository. You can also use a command to list all the current branches within the current repository. The default branch is called either master or main, below it is remained from master to main.

Syntax:

git branch <name> (Creates a new branch in the current local repository)

git branch -a/-l/-r (Lists all branches within the repository)

```
MINGW64:/c/Users/C00296735/Desktop/GitAssignment1

C00296735@CD-A307-09 MINGW64 /c/Users/C00296735/Desktop/GitAssignment1 (master)
$ git branch -m main

C00296735@CD-A307-09 MINGW64 /c/Users/C00296735/Desktop/GitAssignment1 (main)
$ git branch -a
* main
```

Git Remote

Allows the creation of a remote repository within the current folder. Also allows you to list and delete these remote repositories if necessary. This will allow you to view your work and commits up on GitHub when they are pushed with the git push command (see next command).

Syntax:

git remote add <name> <URL> (Creates a new remote repository)

git remote -v (Lists all repositories on remote server)

git remote rm <name> (Delete an existing remote repository)

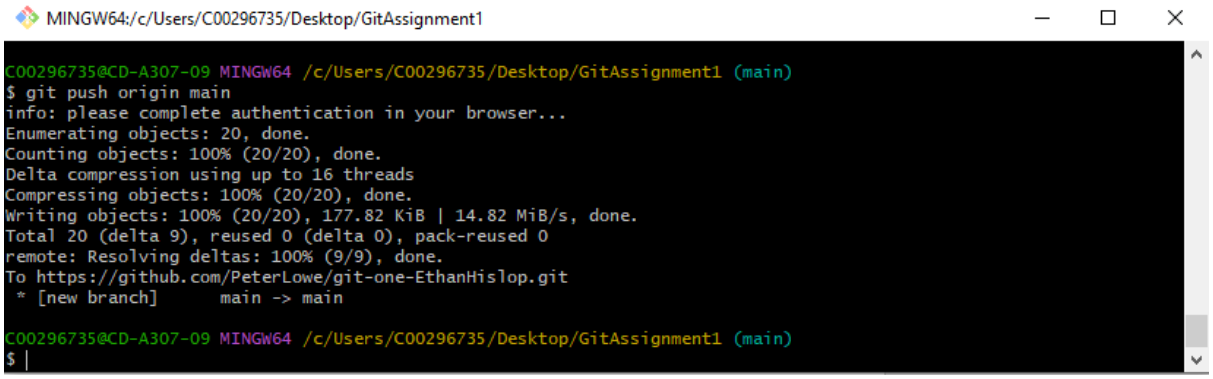
```
MINGW64:/c/Users/C00296735/Desktop/GitAssignment1

C00296735@CD-A307-09 MINGW64 /c/Users/C00296735/Desktop/GitAssignment1 (main)
$ git remote add origin https://github.com/PeterLowe/git-one-EthanHislop.git

C00296735@CD-A307-09 MINGW64 /c/Users/C00296735/Desktop/GitAssignment1 (main)
$ git remote -v
origin  https://github.com/PeterLowe/git-one-EthanHislop.git (fetch)
origin  https://github.com/PeterLowe/git-one-EthanHislop.git (push)

C00296735@CD-A307-09 MINGW64 /c/Users/C00296735/Desktop/GitAssignment1 (main)
$
```

Git Push

Uploads all local commits within the repository/branch to the matching remote repository/branch. These can then be viewed through GitHub if the command was successful.

Syntax:

git push <remote> <branch> (Pushes the given remote and branch commit up to GitHub)

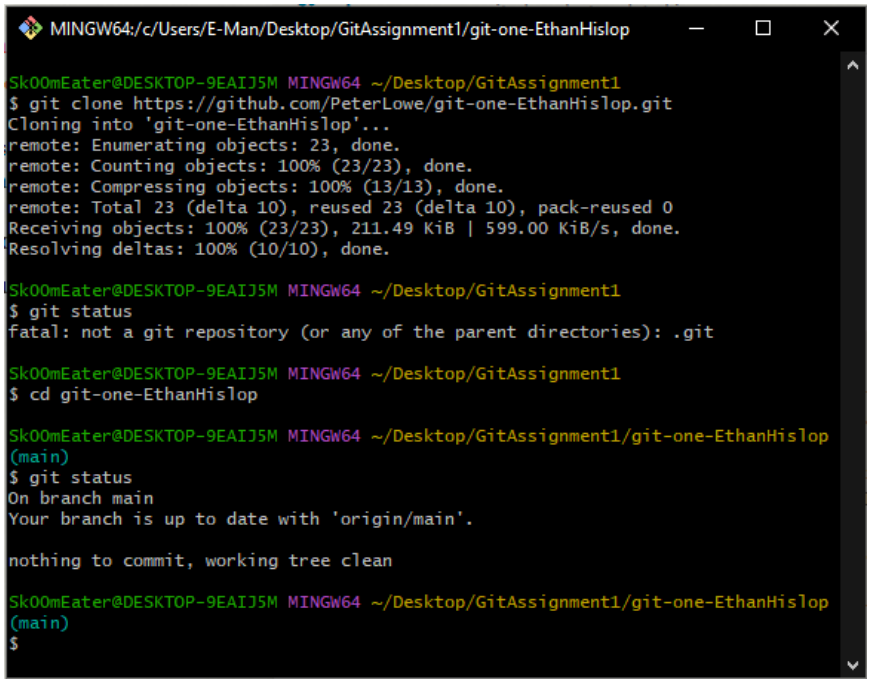


Git Clone

Clones an existing repository (usually a remote one) into a new directory (perhaps on another file system or computer).

Syntax:

git clone <URL> (Clones the remote repository within the indicated URL)



Git Checkout

Allows the switching between different branches within a repository or, more accurately, makes the branch/node the current head. As shown in the screenshot below the head is assigned to main so no change is needed to be made (there are no other branches within this repository).

Syntax:

git checkout <branch_name> (Makes this node the current head)

Git Merge

Allows the current assigned head/branch of the repository to be merged with the main/master branch. Useful for combining two different branches, which is what this command is most used for. The below screenshot shows that everything is up to date (as there is only one branch in this repository, which is main/master).

Syntax:

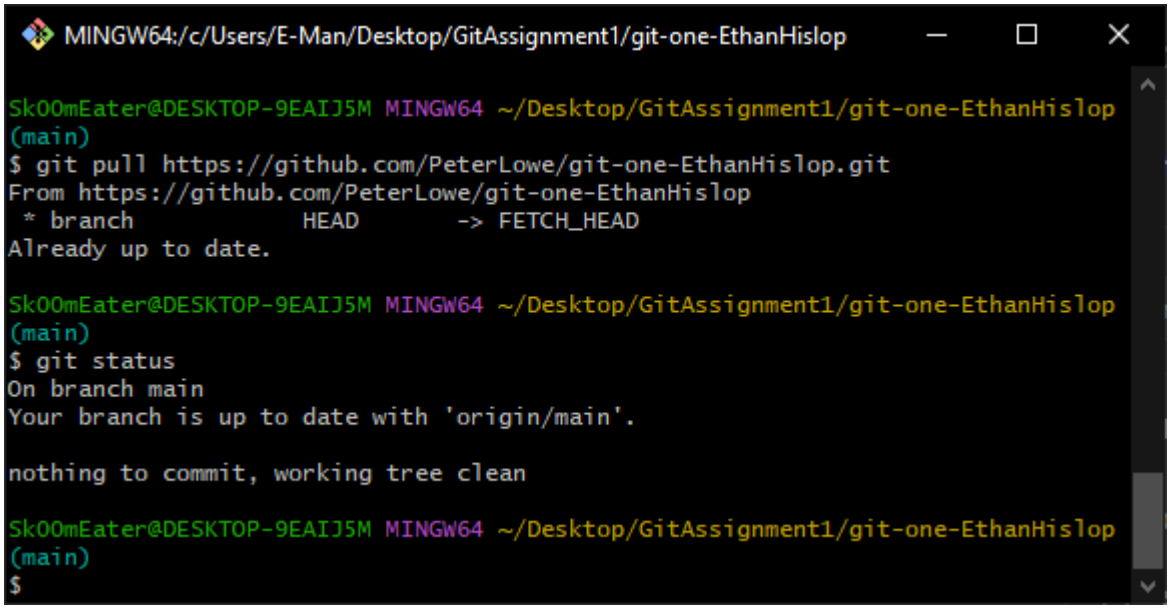
git merge master/main (Merge the current head with main/master branch, git auto applies changes)

Git Pull

This command performs a fetch and merge command at the same time. It will first fetch files from the remote repository then it will merge them with the current head/branch. As shown below, no changes were made in this case as the local repository and the remote repository are both the same and up to date.

Syntax:

git pull <remote> (Perform a fetch and a merge)



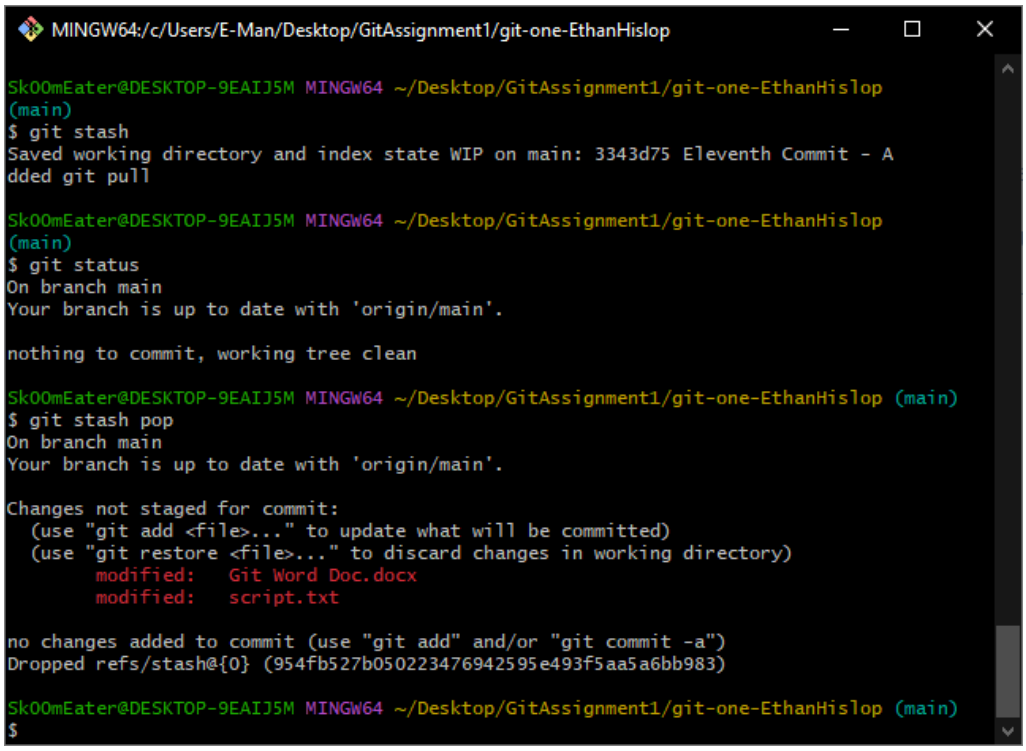
Git Stash

Shelves changes made to the current working directory so another thing can be worked on, allowing you to come back and reapply these changes later. CAN ONLY BE DONE BEFORE COMMIT NOT AFTER

Syntax:

git stash (Stashes away an un-staged or staged, but not committed, work to be reapplied later)

git stash pop (Removes the stash changes and applies them to your working directory)



```
MINGW64:/c/Users/E-Man/Desktop/GitAssignment1/git-one-EthanHislop

SkOOmEater@DESKTOP-9EAIJ5M MINGW64 ~/Desktop/GitAssignment1/git-one-EthanHislop
(main)
$ git stash
Saved working directory and index state WIP on main: 3343d75 Eleventh Commit - A
dded git pull

SkOOmEater@DESKTOP-9EAIJ5M MINGW64 ~/Desktop/GitAssignment1/git-one-EthanHislop
(main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

SkOOmEater@DESKTOP-9EAIJ5M MINGW64 ~/Desktop/GitAssignment1/git-one-EthanHislop (main)
$ git stash pop
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Git Word Doc.docx
        modified:   script.txt

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (954fb527b050223476942595e493f5aa5a6bb983)

SkOOmEater@DESKTOP-9EAIJ5M MINGW64 ~/Desktop/GitAssignment1/git-one-EthanHislop (main)
$
```

Git RM

This can be used remove one or multiple files from a directory, mainly tracked ones from the index, staging index and the local working repository. This applies to all of these and can't be used for only one.

Syntax:

git rm <file name> (Removes file from repository)

git rm -f <file name> (Removes file from repository by force, disregarding errors)