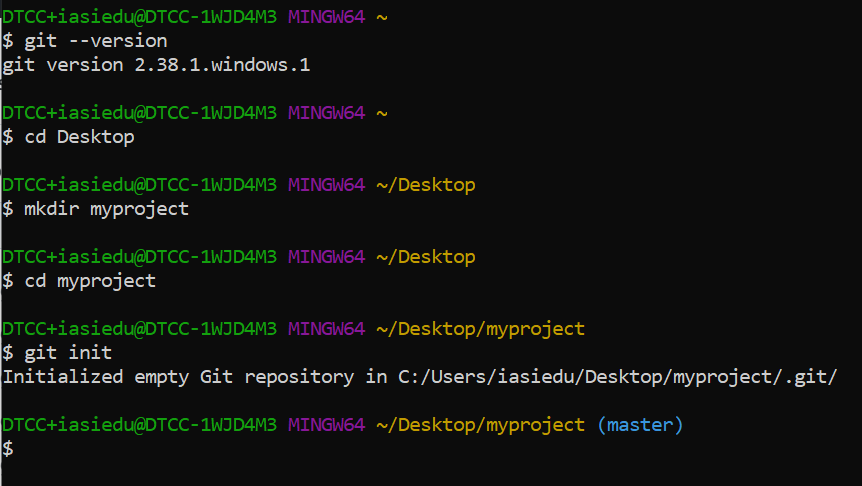
Notes:

* Get the version of git.
* Changed current directory to Desktop.
* Created a new directory(mkdir) ‘myproject’.
* Changed directory into ‘myproject’.
* Created git repository(git init).



* List files in the directory

Text

Description automatically generated

* Create html file in sublime text3 and save it as index.html

Text

Description automatically generated

* List files
* Check git status and see if it is part of the repo.
* Git is aware of the file but has not yet added it to the repo.
* Notes.doc is this word document saved in the repo. AS I update it, different versions will be created.

Text

Description automatically generated

Git staging environment.

Staged files are ready to be committed to the repo.

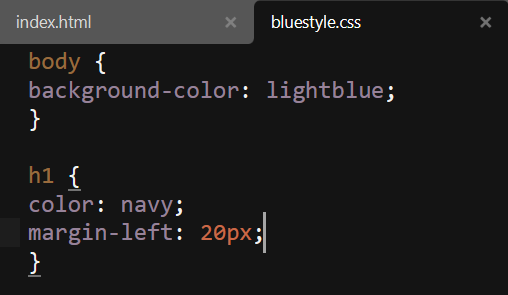
* Git add to add the html file to the staging area.

Text

Description automatically generated

Add more Files.

* A README.md file to describe the repo.(use a txt doc)
* This word document.
* A css file
* An updated html file.

 Text

Description automatically generated

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

Git Commit

Adding commits keeps tracks of your progress. Git considers each “commit” as a save point. It is a point in the project where you can go back to if you find a bug or want to make a change. When we commit we should use a message to tell everyone what changes have been made.

The commit command performs a commit, and the -m “message” adds a message.

Text

Description automatically generated

Commit Without Stage

It is possible to commit changes directory without staging. The -a option will stage every changed and tracked file.

Let’s update the index.html and this document which is always been updated.

Use –short to check status, then commit.

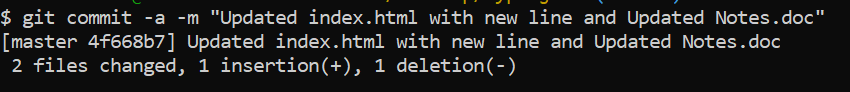
The index.html and the constantly updated notes have been updated.

Text

Description automatically generated with medium confidence

Now commit directly without staging.

Do not always do this as it can cause troubles.



Git Help:

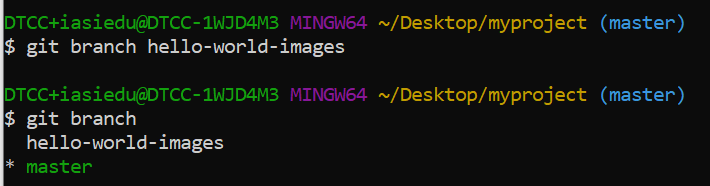
* git ‘command’ -help :will give you all available options for the specific command.
* git help –all :will provide all commands.

Git Branch

* A branch allows you to work on different parts of a project without impacting the main branch.
* You can have several branches and merge them with the main branch later.
* You can switch between different branches and work on different parts of the project.
* Branching is fast.

Now assume we do not want to disturb the main project: Create a new branch called “hello-world-images”

* Git branch <branch name>
* Git branch: to confirm the created branch.
* The \* beside the master means we are currently on the master.



* Use git checkout to move from the master to the branch that we currently created.

Text

Description automatically generated

Now we are on the new branch so we open a new text editor and make some changes and save. Lets add a hello world image and modify the index.html file.

Text

Description automatically generated

Git status will tell us the current state of our documents on the new branch.

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Now add the files to the staging area and find the status.

Text

Description automatically generated

Now Commit them to the branch since everything is as required.

Text

Description automatically generated

We now have a branch different from the master.