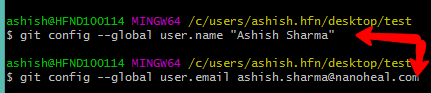
Step 1- Configuring Git

$ git config --global user.name "My Name"

$ git config --global user.email myEmail@example.com

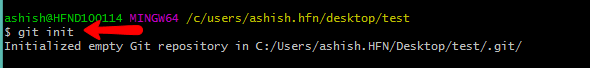


Step 2- Initializing a new repository

* Navigate to the project/repo directory in the git bash terminal.



* Execute the command – git init (This will enable Git for this particular folder and create a hidden .git directory where the repository history and configuration will be stored.)

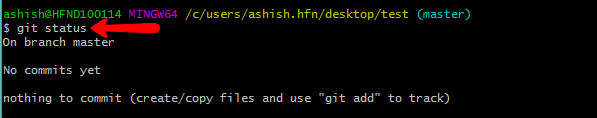


Now our git repository has been successfully created but is still empty.

Step 3- Check status of your repository

-returns information about the current state of the repository.

Execute this command – git status (to see current status of the project)

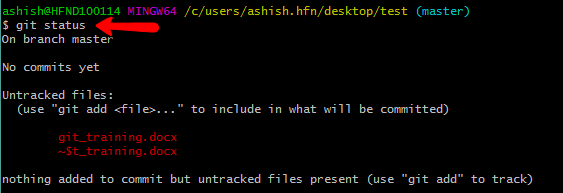


Step 4- Adding some files into project

* Add some txt files in the repository now.

This is done to have hands-on-experience.

Now again check status of the project:



Untracked means that the file is new and Git doesn’t know yet those if it should keep track of changes happening to the file or just ignore. To acknowledge the new file we need to stage it.

Step 5- Staging the files

This is done to add files to the staging area so Git can track changes made into the file.

Execute the command to stag the file: git add file\_name



\*\*Git\_training.docx is my filename.

In our case we have only one file so let's add that:

$ git add hello.txt

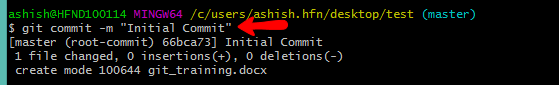
If we want to add everything in the directory, we can use:

$ git add –A

Again check status of the project and notice the difference.

Step 6 -Committing the changes (Saving the changes made in the project)

Execute the command: git commit –m “Commit message depicting some info about the commit”

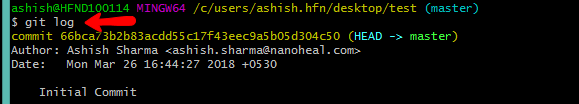


This will create a new commit with all the changes from the staging area.

Step 7-Checking different commits

Every commit has its unique id in the form of a string of numbers and symbols. To see a list of all commits and their ids we can use git log:

$ git log

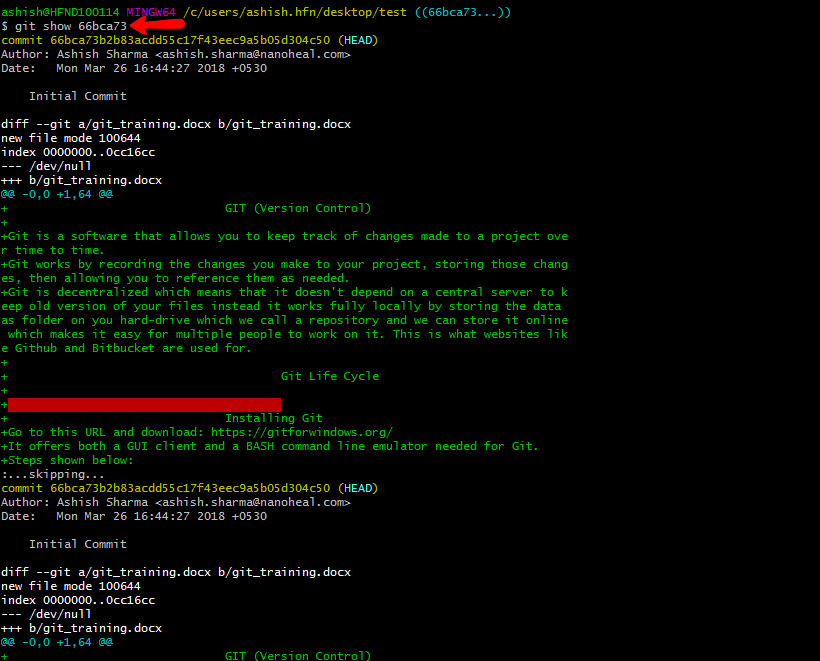


As you can see the ids are really long, but when working with them it's not necessary to copy the whole thing - the first several symbols are usually enough.

\*\*To see what was new in a commit we can run git show [commit]:

git show b10cc123 🡨 this is the first seven letters of the Sha-id.

A detailed log containing every information about that particular commit with the given sha-id will be shown.



\*\*To see the difference between any two commits we can use git diff with the [commit-from]..[commit-to] syntax:

git diff 09bd8cc..ba25c0ff

Usually it's easier to do this using the git difftool command which brings up a graphical client showing all differences side-to-side.