**Working with git and github.**

1. For windows system, install git from [www.git-scm.com](http://www.git-scm.com)
2. Check the version after installation.

**$ git –version**

1. We can link git to cmd prompt as well as to a separate git bash terminal. Choose whatever needed.
2. Create a folder.
3. Get inside it using git bash. Git bash uses basic linux cmmands.
4. Select the folder as the staging environment.

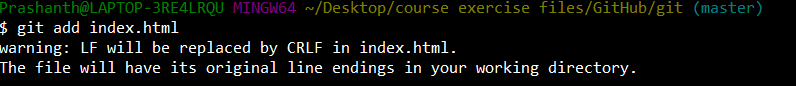
**$ git init**

1. We can see the invisible .git folder using.

**$ ls -la**

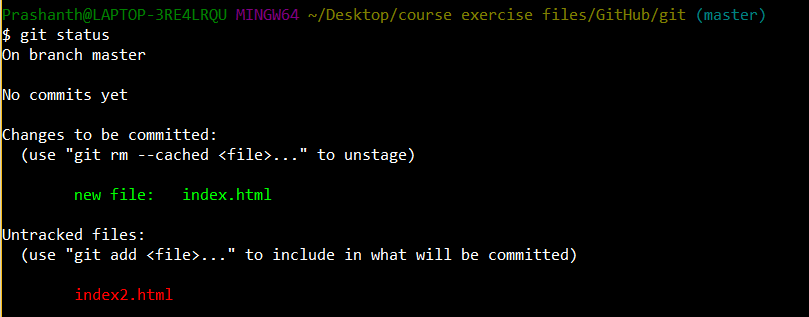
1. Create two random html files, index1.html, index2.html.
2. Add any one file to the staging environment.

**$ git add file\_name**



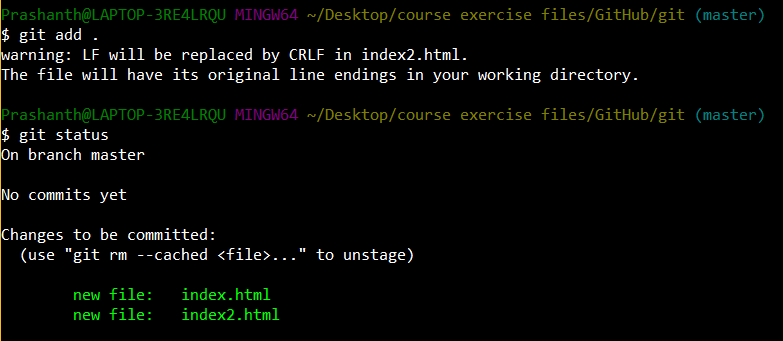
1. See the status of the git staging folder.

**$ git status**



1. Above we can see that git keeps a track on the unattended files also.
2. For adding all the files to the staging environment, type the following command and check the status.

**$ git add**

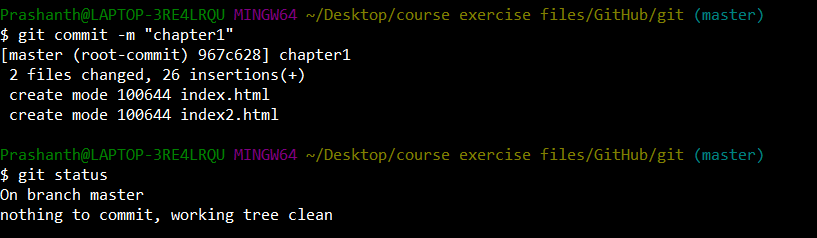


1. Now configure your git using the following command

**$ git config --global user.email “**[user\_name@google.com](mailto:user_name@google.com)”

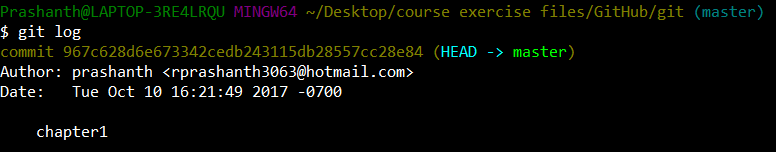
**$ git config --global user.name "your\_name"**

1. Now let’s try to commit and see the status of the files available in the staging folder-

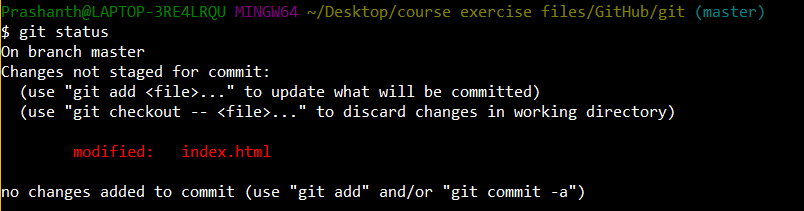
**$ git commit -m “task\_name/any\_name”**

1. Do check the log do-

**$ git log**

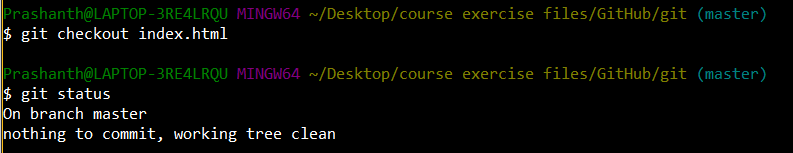
****

1. Try modifying index.html file and then see the status. It will show as modified.

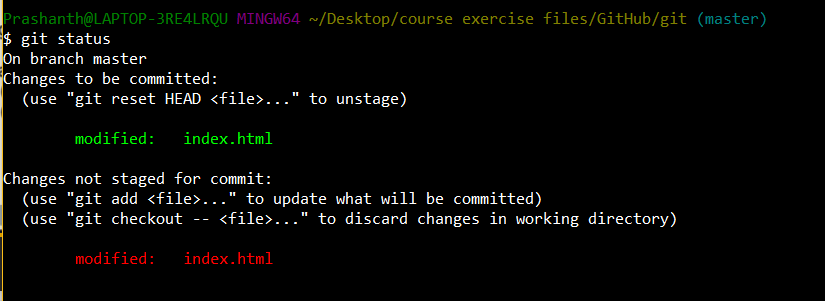


1. If you want to switch back do the following and then see the git status. It switches back to previous state.

**$ git checkout index.html**

****

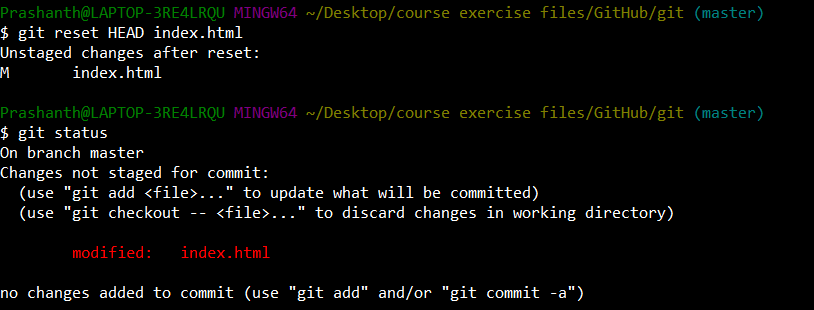
1. try modifying the index.html file and try to add it. Then see the status. Then again modify the index.html file and see the status. This shows that index.html file in the staging environment and staging folder are not the same.



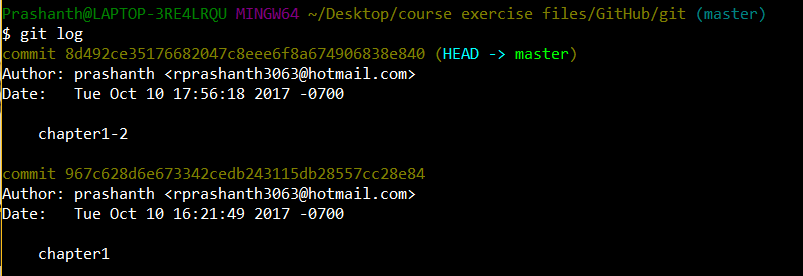
1. do reset to revert the changes and then see the status.

**$ git reset HEAD index.html**

Once the commands are executed, we see that changes made in the staging folder exists not the staging environment changes.

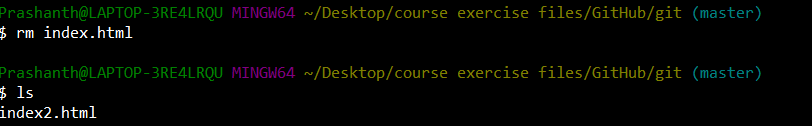
****

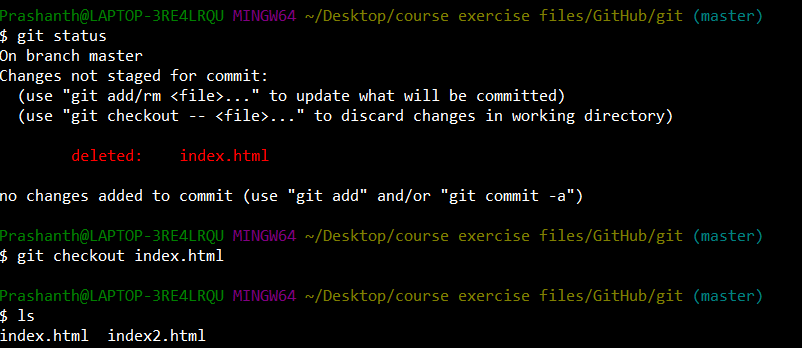
1. commit the changes and check the log. There will be a new log added.



1. now delete the index.html file manually and then check status. U can pull the file back by-

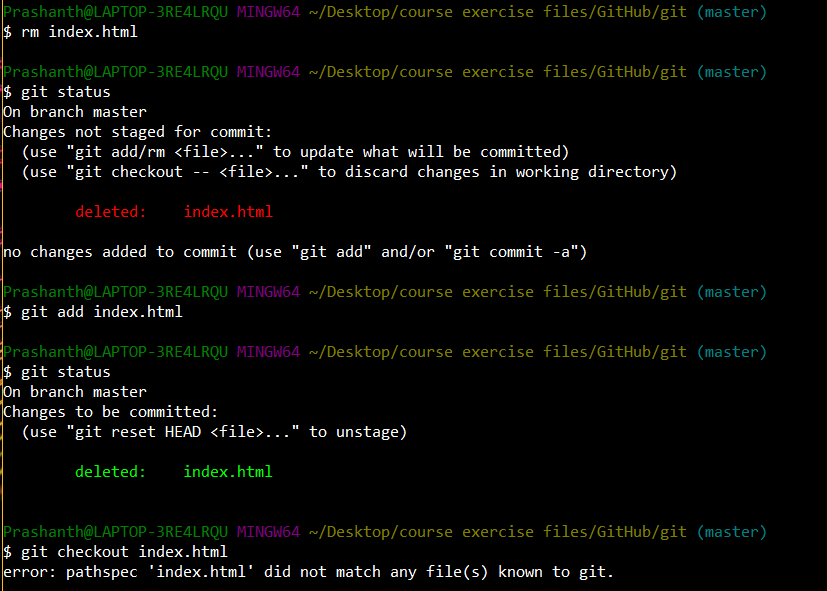
**$ git checkout index.html**

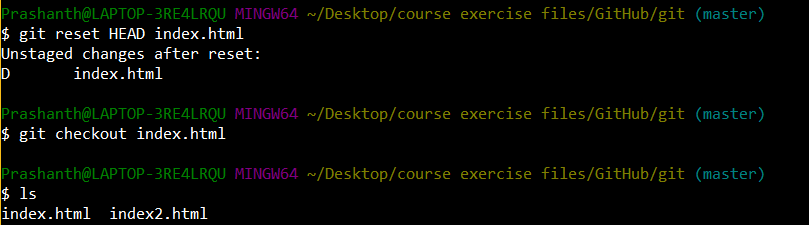




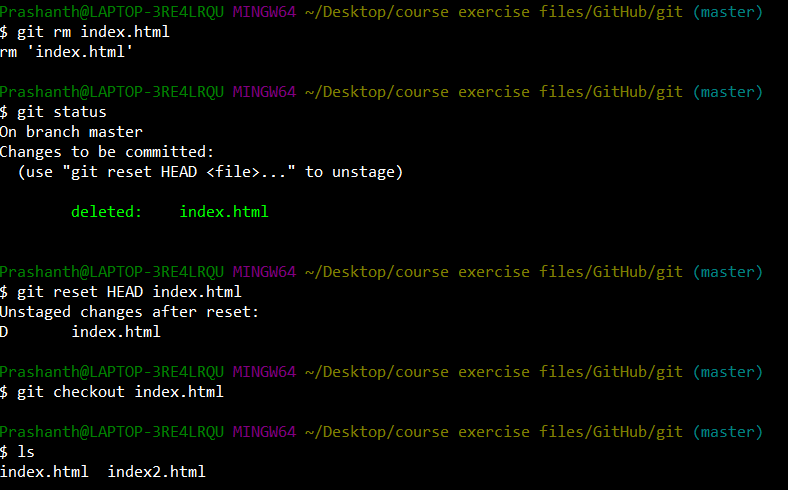
1. Again delete the file and try to stage it. So that state has been saved in the staging environment.

Now if u try to revert back the file using the checkout command, it wont work.



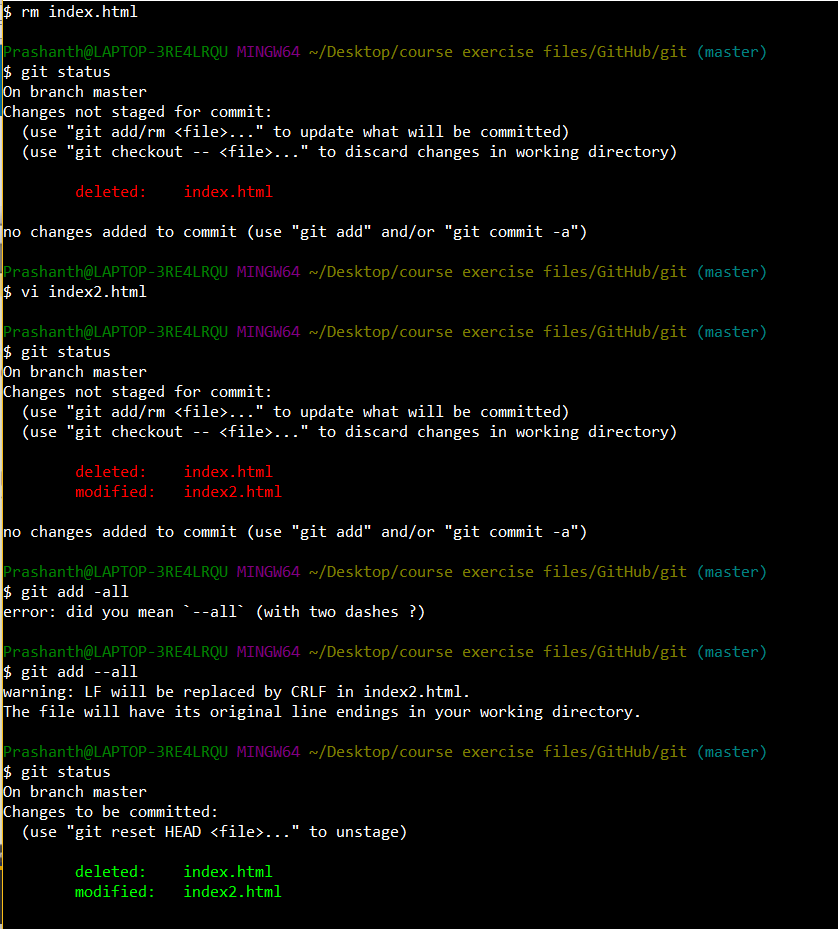
1. Now if u want to revert the state, then reset it and do checkout. You will get your file back.
2. If u want to perform 2 tasks at a time, like – for deleting a file and staging it we do rm command and then add command to save the state. These 2 commands can be performed by using the command-

**$ git rm index.html**



1. If u have made changes in many different files and u want to stage all. Then do-

**$ git add --all**

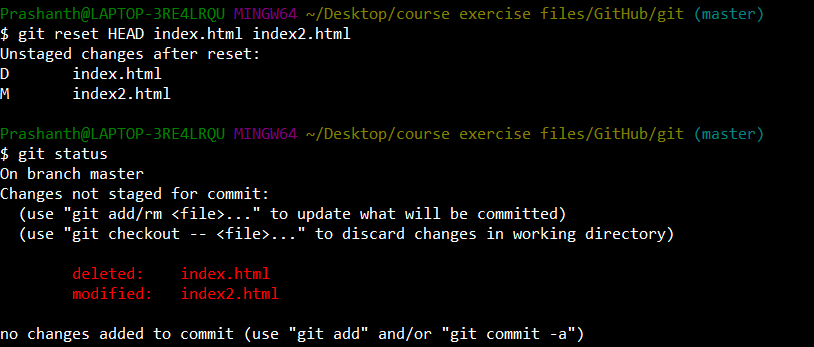


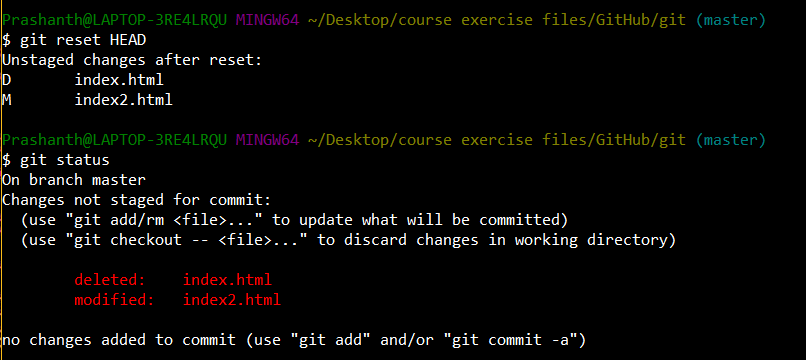
1. If u want to revert back multiple changes on multiple files then do-

**$ git reset HEAD index.html index2.html**

or if u want to revert all the staged changes completely then do –

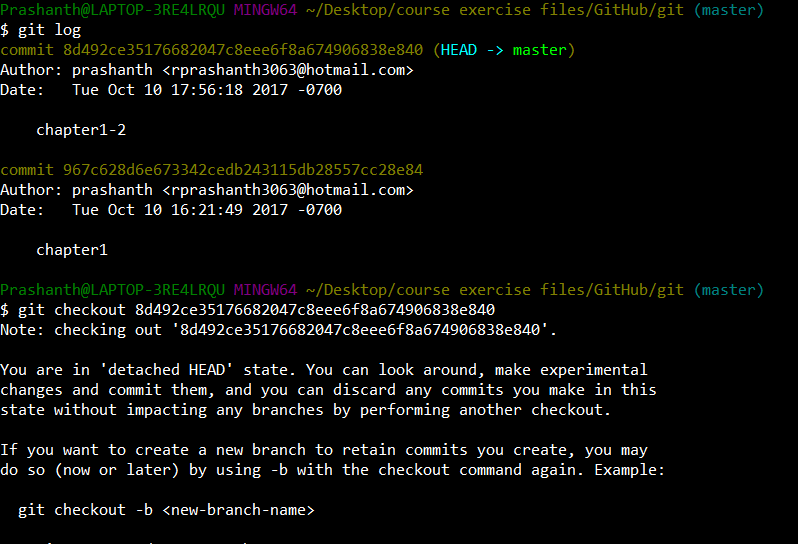
**$ git reset HEAD**

****

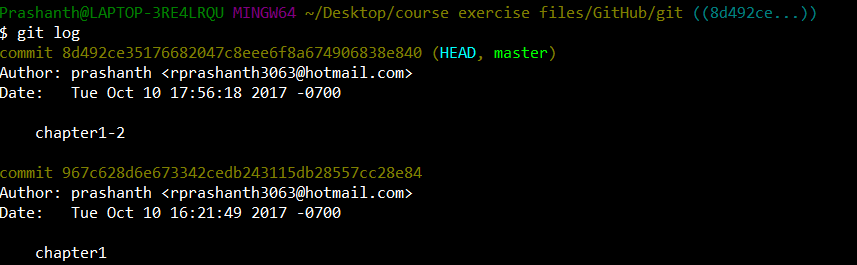
****

1. If u want to revert back to a committed point. Then do-

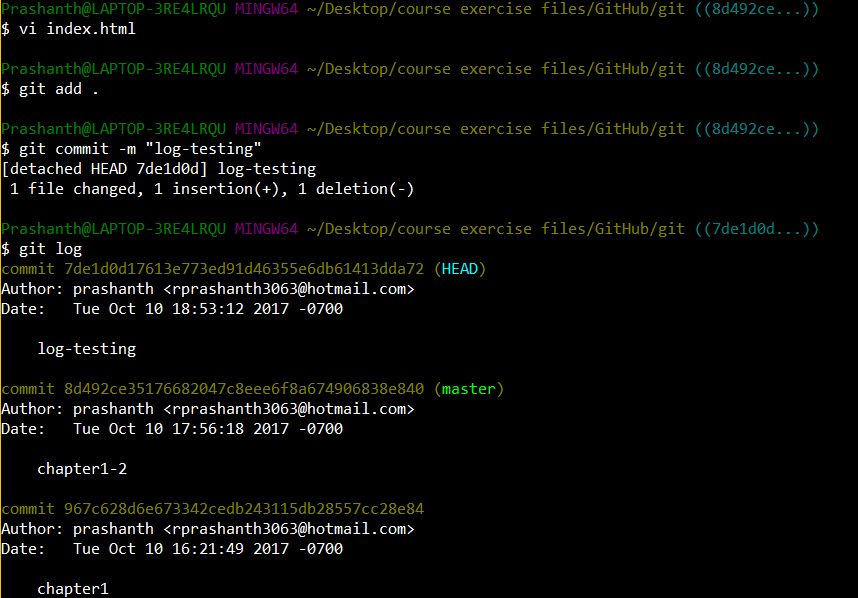
$git checkout #HASH#



1. Now if u check the logs again u can see that, the stage to which reverted back becomes the HEAD and the master. So it tells us that we are at that point.

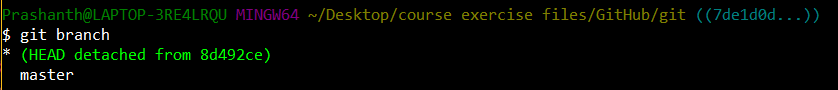


1. Now if u make changes and then do commit then a new log gets added and he new one becomes the head but the master remains the previous one.



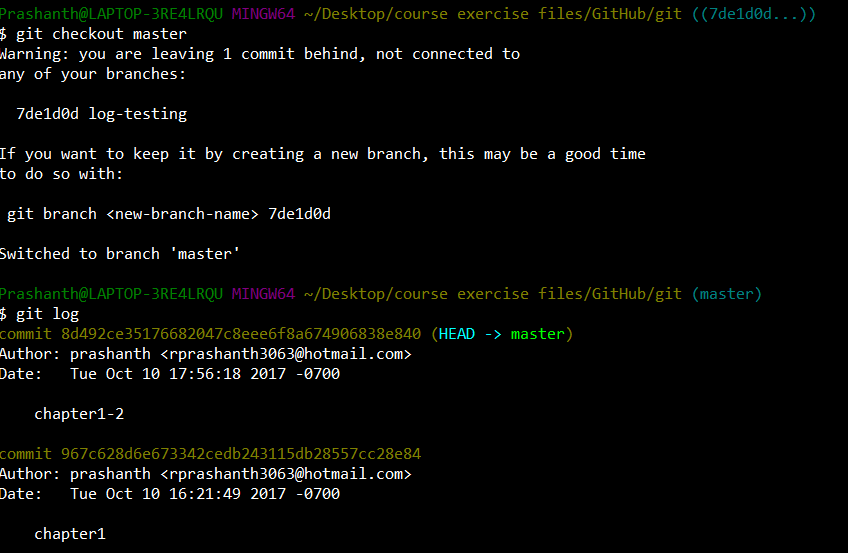
1. U can see all the branches by-

**$ git branch**

****

1. Now if u want to shift the HEAD back to master then.

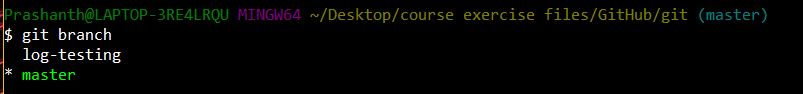
**$ git checkout master**

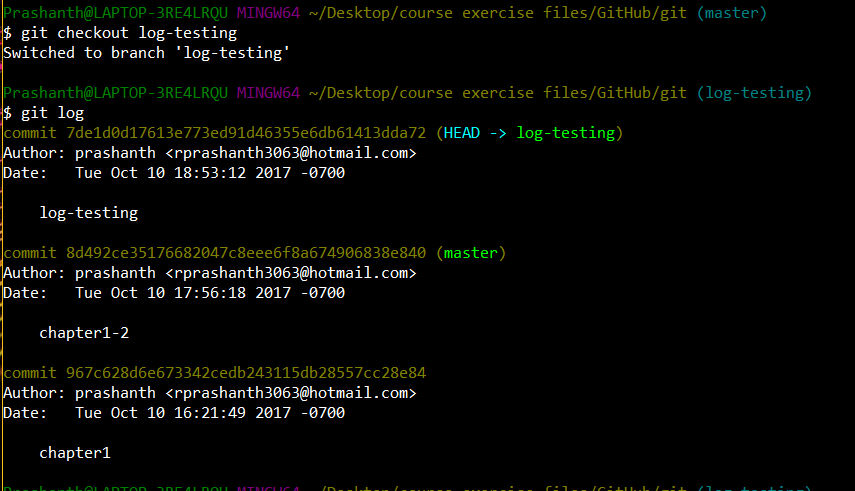
****

1. So all the logs that u see are the alternate realities. We can switch to any one of these.
2. We can create a new branch by-

**$ git branch <banch\_name> #HASH#**

**C:\Users\Prashanth\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Capture.png**

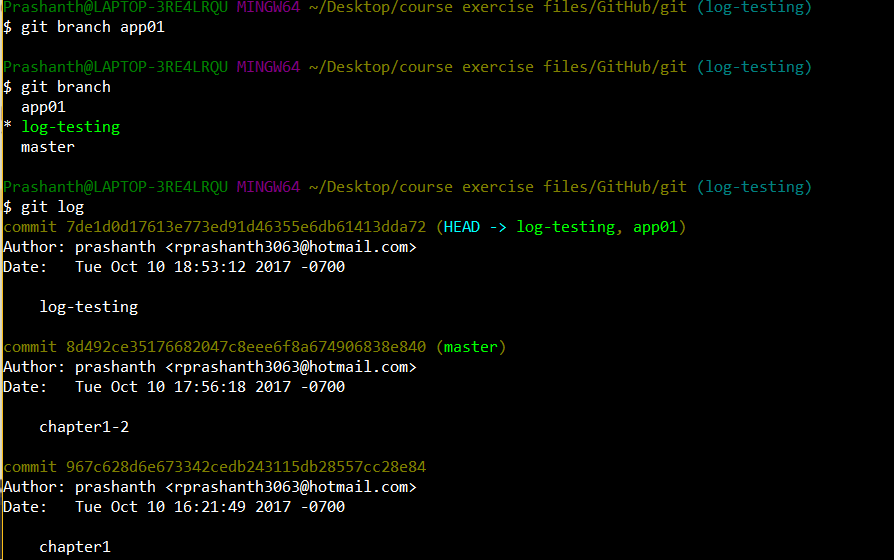




1. Now we are at the log-testing reality. These commands are useful if u commit some mistake.
2. If we want to create a branch within an existing branch, then do

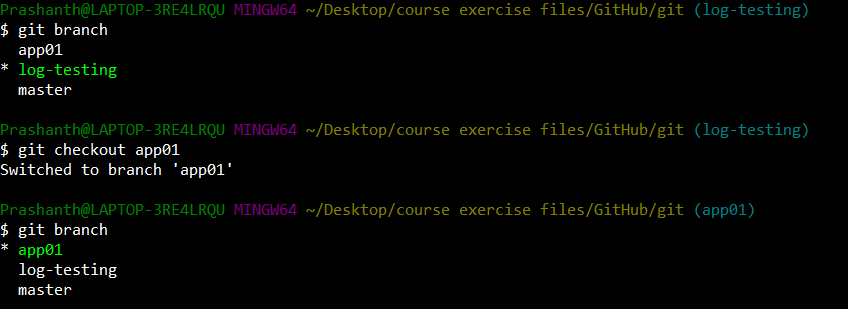
**$ git branch subbranch\_name**

So a new branch will get added within the aren’t branch where we are residing.



1. Now if u want to jump to a newly created branch then do checkout to that branch.

From now the state will be getting saved within the branch to which we have switched to.



Now all the above commands can be repeated inside this branch. Now do make a change in index.html file and try to commit the state. Now do check the log. The head will be shifted to the newly added branch which is our current reality.

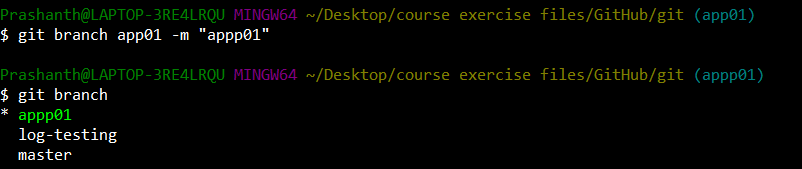
1. If we want to take the change made in a different reality and add it to the current branch where we are residing, then do-

**$ git merge app01**

This will merge the current state and the previously created state.

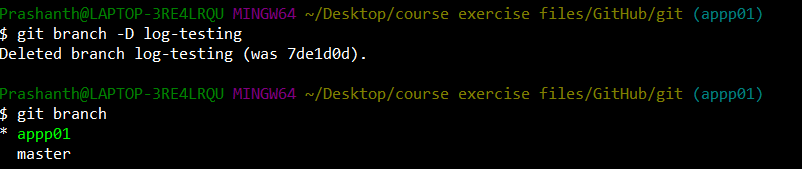
1. Now if u want to rename a branch then do-

**$ git branch <branch\_name> -m “new\_name”**



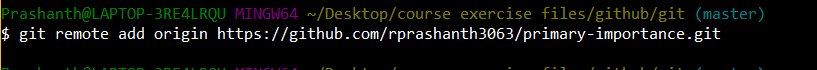
1. If u want to delete a branch then do-,,,,,,,, when we delete it that state will also get deleted.

**$ git branch -D <branch\_name>**



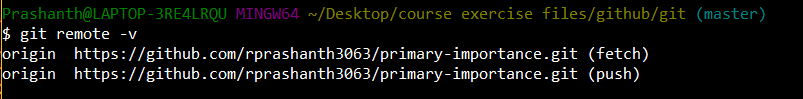
1. Create a github account and give a name to a repository.
2. In your git bash terminal type

**$ git remote add origin <URL>**

****

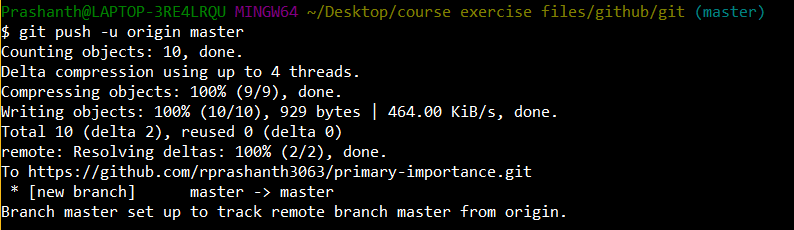
1. Verify the remote url-

**$ git remote -v**

****

1. Push your files to the online repository

**$ git push -u origin master**

****

1. Thankyou