1. **git config**
2. **git --version**
3. **git init**
4. **git clone**
5. **git add**
6. **git commit**
7. **git diff**
8. **git reset**
9. **git status**
10. **git rm**
11. **git log**
12. **git show**
13. **git tag**
14. **git branch**
15. **git checkout**
16. **git merge**
17. **git remote**
18. **git push**
19. **git pull**
20. **git stash**
21. **Esc : wq**
22. **Ls –lart** (all files)
23. **Pwd -** location check git
24. **Cd -** location desktop/…
25. **git clone** link
26. **ls –** check folder list
27. **git status –** all file status
28. **git diff –** changes files
29. **git diff file\_name –** check in file
30. **git add** \*
31. **git log**
32. **git branch** master/new
33. **git branch new**
34. **git checkout** master/new(changes)

**git config**

$ git config --global user.name “username”

$ git config --global user.email “adarsh@gmail.com”

$ git config --global --edit

All the name and email edit.

**git init**

$ git init [repository name]

**git clone**

$ git clone [url]

**git add**

$ git add [file]

$ git add \*

$ git add -A

This command adds **one or more to the staging area**.

**git commit**

$ git commit -m “[Type in the commit message]”

$ git commit -am “[Type in the commit message]”

commits any files you’ve added with the git add command and also commits any files you’ve changed since then.

**git diff**

$ git diff

This command shows the **file differences which are not yet staged.**

$ git diff --staged

shows the differences between the **files in the staging area** and **the latest version present.**

$ git diff [first\_branch] [second\_branch]

This command shows the **differences** between the **two branches** mentioned.

**git reset**

$ git reset [file]

This command **un-stages** the file, but it **preserves the file contents**.

$ git reset [commit]

undoes all the commits after the **specified commit and preserves the changes locally.**

$ git reset –hard [commit]

This **command discards all history and goes back** to the specified commit.

**git status**

$ git status

$ git status -s

This command lists all the files that have to be committed.

**git rm**

$ git rm [file]

This **command deletes the file** from your working directory and **stages the deletion**.

**git log**

$ git log

$ git log –follow[file]

This command **lists version history for a file**, including the renaming of files also.

**git show**

$ git show [commit]

This command shows the **metadata and content changes** of the specified commit

**git tag**

$ git tag [commitID]

This command is used to **give tags to the specified commit**.

**git checkout & merge**

$ git checkout [branch\_name]

$ git branch [branch\_name]

$ git checkout -b [branch\_name]   creates a **new branch & entered**

$ git merge [branch\_name]

This command merges the specified branch’s history into the current branch.

**git remote**

$ git remote add [remote] [Link]

$ git remote (status remote)

$ git remote -v (all status remote)

This command is used to connect your local repository to the remote server.

**git push**

$ git push -u [remote] [branch]

$ git push

$ git push --force [remote] [branch]

$ git push –all [remote]

$ git push [remote] : [branch] (delete branch)

$ git branch -d [branch] (delete branch)

**git pull**

$ git pull [Repository Link]

$ git pull [Remote] [Branch]

$ git pull

**fetches and merges changes on the remote server** to your working directory (Desktop).

**git stash**

$ git stash save

This command **temporarily stores all** the modified tracked files.

$ git stash pop

This command **restores the most recently** stashed files.

$ git stash list

This command **lists all stashed changesets**.

$ git stash drop

This command **discards the most recently stashed changeset.**