# 

# 

# GIT(Global Information Tracker)

**Linus Torvalds**

GitHub is a web-based version control and collaboration platform for software developers.

A version control system records the changes made to our code over time in a special database called **repository(storage space).**

By using version control system we can track our project history and work together.

Version control systems fall into two categories

1. Centralized : All team members connect to a central server to get the latest copy of the code and to share their changes with others. For example : Subversion and Team foundation server.

Problem: if the sever goes offline we cannot save the snapshots of our project. Until wait the sever come back.

1. Distributed : In distribute there is no problem every team member has a copy of the project with its history on their machine and save the snapshots of our local systems. If the central server is offline we can synchronize our work directly with others git.

### Git config : It is builtin Tools

* --global ( user info )
* --system ( system info )
* --local ( repo info )

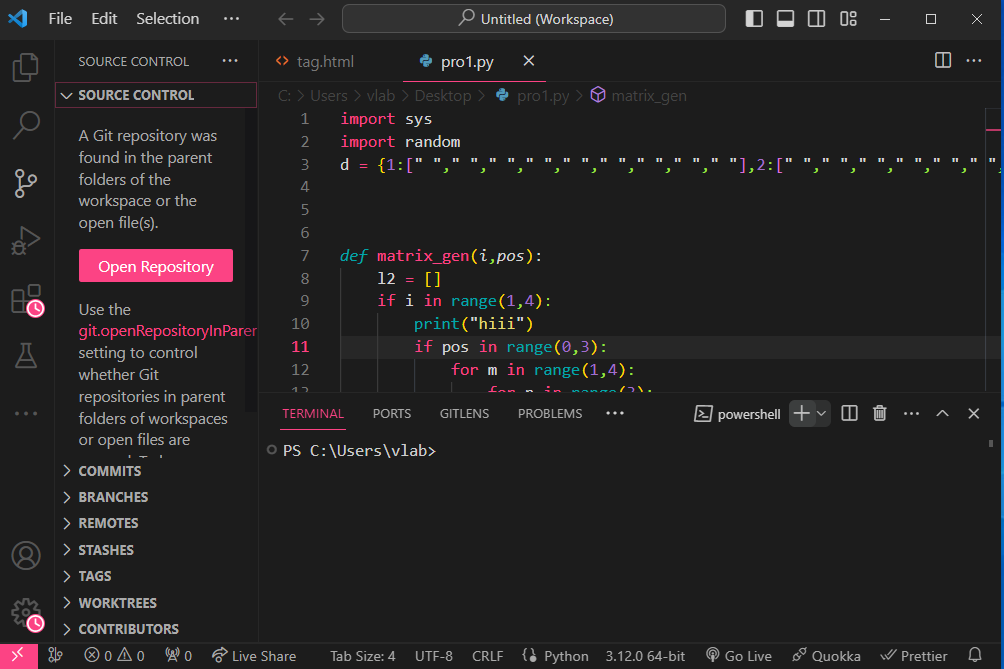
### Why GIT ?

1. Free
2. Distributed source control system
3. Open Source
4. Super Fast
5. Scalable Operations
6. Cheap branching and merging

### USING GIT

1. The command line : Open a terminal or command prompt window to execute git commands.
2. Code editors and IDEs (Integrated Development Environment):

For example : In vs code we have source control panel which gives us the essential git features.



1. Graphical user interfaces: Git comes with built-in GUI tools for committing (git-gui) like there are different platforms we have tools for windows , mac, linux, Android and iOS.

### Installation of Git:

Windows:

* Click the search icon and type cmd
* By knowing the version just type git --version
* Git version 2.44.0.windows.1 showing and install

### Terms:

* **Commit** : use , date and commit unique id
* **Clone**  : clone means Exact copy
* **Tracked/Untracked files :** .git folder is tracking the different files.
* **Branch**
* **Master(branch)**
* **Origin**

### Configuring Git:

#### Settings:

1. Name
2. Email
3. Default Editor
4. Line Ending

#### Three different levels:

1. System : all users
2. Global : All repositories of the current user
3. Local : The current repository or the repository in the current folder

**To specify the level at which we are defining these settings:**

git config --global user.name "varalaxmi ganta"

git config --global user.email [varalaxmiganta066@gmail.com](mailto:varalaxmiganta066@gmail.com)

For example :

By using vs code when we type Code in cmd then vs code will be open but it will close automatically after some time.

git config --global core.editor "code --wait" by using this command we tell the terminal window to wait until we close the new vs code instance.

git config --global -e This will open default editor to edit all the global settings.

In vs code : .gitconfig file will be open

For example

Git config --global core.editor “notepad”

Git config --get core.editor

[user]

    name = varalaxmi ganta

    email = varalaxmiganta066@gmail.com

[core]

editor = code --wait

After close the Configuration file then terminal will be terminate the waiting command.

**To configure how git should handle end of lines:**

In windows end of lines are marked with two special characters

abc \r\n

\r -> Carriage Return

\n -> Line Feed

To configure one property called **core.autocrlf** which is short form carriage return

git config --global core.autocrlf input

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git config --list

diff.astextplain.textconv=astextplain

filter.lfs.clean=git-lfs clean -- %f

filter.lfs.smudge=git-lfs smudge -- %f

filter.lfs.process=git-lfs filter-process

filter.lfs.required=true

http.sslbackend=openssl

http.sslcainfo=C:/Users/vlab/AppData/Local/Programs/Git/mingw

32/etc/ssl/certs/ca-bundle.crt

core.autocrlf=true

core.fscache=true

core.symlinks=false

core.fsmonitor=true

pull.rebase=false

credential.helper=manager

credential.https://dev.azure.com.usehttppath=true

init.defaultbranch=master

pack.packsizelimit=2g

user.name=varalaxmi ganta

user.email=varalaxmiganta066@gmail.com

core.editor=code --wait

core.autocrlf=input

diff.tool=vscode

core.repositoryformatversion=0

core.filemode=false

core.bare=false

core.logallrefupdates=true

:

#### Help Command

* git help <verb>
* git <verb> --help
* Man git -<verb> some times its working

git config -h or git config --help : It shows all options in the configuration file

### Initializing a Repository

Where does repository lives in ?

* Folder
* GitHub
* Online host
* First create a Directory

vlab@VEDA2F118 MINGW32 ~/GitProjects (master)

$ mkdir Moon

vlab@VEDA2F118 MINGW32 ~/GitProjects (master)

$ cd Moon

* Add one file to git repository

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ git init

Initialized empty Git repository in C:/Users/vlab/GitProjects/Moon/.git/

* List all the hidden files

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ ls -a

./ ../ .git/

* Open git directory

$ open .git

$ cd .git/

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon/.git (GIT\_DIR!)

$ ls

HEAD config description hooks/ info/ objects/ refs/

* Just for knowing this remove the .git

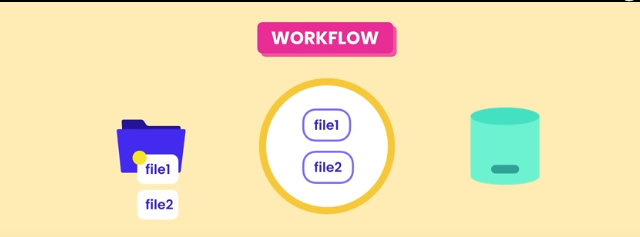
$ rm -rf .git/

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ git init

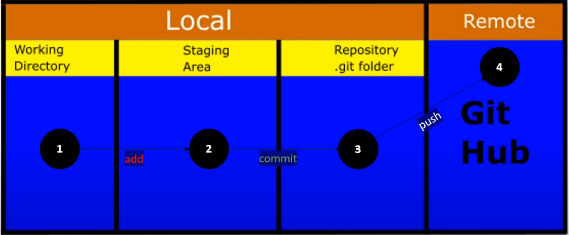
Initialized empty Git repository in C:/Users/vlab/GitProjects/Moon/.git/

### Git Workflow

**Staging Area** :

* It is a file, generally contained in your Git directory, that stores information about what will go into your next commit. Its technical name is index.
* By using **add** command to add these files to the staging area.
* By using the commit command to permanently store this snapshot in the repository.





* When fixing a bug we make some changes to file so note that what we currently have in the staging area is the old version of the file.
* If change any thing in the file then once again we use the add command to stage the changes.
* We have different commits in our repository each commit contains a unique

ID

Message

Date/time

Author

Complete snapshot : Snapshot waste a lot of space because git is very efficient in data storage it compresses file contents and Doesn’t store duplicate content

**Staging Files:**

For example:

Create two files

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ echo hello > file1.txt

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ echo hello > file2.txt

Here we have to created two new files but these files are not tracked by git because the first time initialize a git repository in a directory. Git is not going to automatically track your files.

**To see the status of the working directory :**

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

file1.txt

file2.txt

nothing added to commit but **untracked** files present (use "git add" to track)

These files are showing red color because they are not in the staging area.

**Add the files into the staging area :**

$ git add file1.txt

$ git add file2.txt

Or

$ git add file1.txt file2.txt

Or

$ git add \*.txt

Or

$ git add .

Here + symbol indicates staging area that are ready to be committed then run again git status

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: file1.txt

new file: file2.txt

Look at the files they are shows green color so,they are in the staging area

**To modify the file content :**

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ echo world >> file1.txt

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: file1.txt

new file: file2.txt

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: file1.txt

In the file1 will shows the modified in red color.

So again add the file1 to the staging area

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ git add file1.txt

vlab@VEDA2F118 MINGW32 ~/GitProjects/Moon (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: file1.txt

new file: file2.txt

### Cloning Existing repo:

git clone <repo-url> by default it create repo name if you want to change that repo name then type

Optional : git clone <repo-url> <preferred-name>

Open one public repo and copy the url

Move project to GitProjects directory

vlab@VEDA2F107 MINGW64 ~/GitProjects (master)

$ git clone https://github.com/vara221/project.git

Cloning into 'project'...

remote: Enumerating objects: 3, done.

remote: Counting objects: 100% (3/3), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Receiving objects: 100% (3/3), done.

vlab@VEDA2F107 MINGW64 ~/GitProjects (master)

$ ls

project/ repo1/

vlab@VEDA2F107 MINGW64 ~/GitProjects (master)

$ cd project/

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ ls

example1

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ ls -al

total 5

drwxr-xr-x 1 vlab 1049089 0 Mar 4 10:11 ./

drwxr-xr-x 1 vlab 1049089 0 Mar 4 10:11 ../

drwxr-xr-x 1 vlab 1049089 0 Mar 4 10:11 .git/

-rw-r--r-- 1 vlab 1049089 22 Mar 4 10:11 example1

Modify example1 then check the status

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ git status

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: example1

add and commit the changes

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ git add example1

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: example1

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ git commit -m "add one print statement example1"

[main 3efd5be] add one print statement example1

1 file changed, 1 insertion(+)

Move to the modified file to git repo

Origin is used to specified the remote branch to system

Origin or any name we can used on that place

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ git push origin main

Enumerating objects: 5, done.

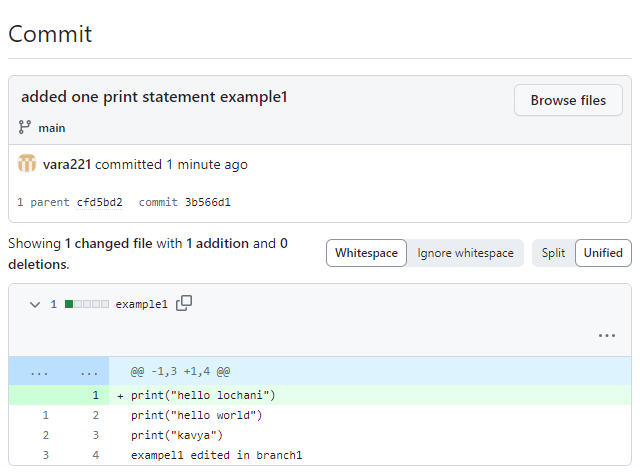
Counting objects: 100% (5/5), done.

Writing objects: 100% (3/3), 284 bytes | 284.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

To https://github.com/vara221/project.git

05ea802..3efd5be main -> main



Then check the example 1 into the github

### Cloning:

When we get one file from the repository to local then it is called as Cloning.

git clone <repo-url>

by using cloning we can give predefined name given that is my-project

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ ls

example1

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ git clone https://github.com/vara221/project.git my-project

Cloning into 'my-project'...

remote: Enumerating objects: 108, done.

remote: Counting objects: 100% (108/108), done.

remote: Compressing objects: 100% (64/64), done.

remote: Total 108 (delta 32), reused 104 (delta 31), pack-reused 0

Receiving objects: 100% (108/108), 37.44 KiB | 2.20 MiB/s, done.

Resolving deltas: 100% (32/32), done.

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ ls

example1 my-project/

Move local -> Git repository

vlab@VEDA2F107 MINGW64 ~/GitProjects/project (main)

$ cd my-project/

vlab@VEDA2F107 MINGW64 ~/GitProjects/project/my-project (main)

$ ls

Question3/ fact.py file2 helloworld.py

example1 file1.txt file2.txt sele\_project/

### Initializing GIT REPOSITORY in existing project:

Create a new repository in github and send files to the exisiting file to the

Then

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ ls

Question3/ example1 file1.txt file2.txt

README.md fact.py file2 sele\_project/

Change to one existing directory

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ cd sele\_project

Initialize that directory

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (main)

$ git init

Initialized empty Git repository in C:/Users/vlab/GitProject/

project/sele\_project/.git/

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (master)

$ ls

\_\_init\_\_.py test\_webelements.py

\_\_pycache\_\_/ webelement.py

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (master)

$ ls -la

total 24

drwxr-xr-x 1 vlab 1049089 0 Mar 4 14:00 ./

drwxr-xr-x 1 vlab 1049089 0 Mar 4 13:45 ../

drwxr-xr-x 1 vlab 1049089 0 Mar 4 14:00 .git/

-rw-r--r-- 1 vlab 1049089 0 Mar 4 13:43 \_\_init\_\_.py

drwxr-xr-x 1 vlab 1049089 0 Mar 4 13:43 \_\_pycache\_\_/

-rw-r--r-- 1 vlab 1049089 966 Mar 4 13:43 test\_webelements.py

-rw-r--r-- 1 vlab 1049089 1562 Mar 4 13:43 webelement.py

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (master)

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committ

ed)

\_\_init\_\_.py

\_\_pycache\_\_/

test\_webelements.py

webelement.py

nothing added to commit but untracked files present (use "git

add" to track)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (master)

$ git add .

warning: in the working copy of 'test\_webelements.py', CRLF w

ill be replaced by LF the next time Git touches it

warning: in the working copy of 'webelement.py', CRLF will be

replaced by LF the next time Git touches it

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: \_\_init\_\_.py

new file: \_\_pycache\_\_/\_\_init\_\_.cpython-312.pyc

new file: \_\_pycache\_\_/demo\_automation.cpython-312-p

ytest-8.0.0.pyc

new file: \_\_pycache\_\_/jyo.cpython-312-pytest-8.0.0.

pyc

new file: \_\_pycache\_\_/test\_demo\_automation.cpython-

312-pytest-8.0.0.pyc

new file: \_\_pycache\_\_/test\_webelements.cpython-312-

pytest-8.0.0.pyc

new file: \_\_pycache\_\_/tests-webelements.cpython-312

-pytest-8.0.0.pyc

new file: \_\_pycache\_\_/webelement.cpython-312.pyc

new file: test\_webelements.py

new file: webelement.py

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (master)

$ git commit -m "Initial commit"

[master (root-commit) d000009] Initial commit

10 files changed, 83 insertions(+)

create mode 100644 \_\_init\_\_.py

create mode 100644 \_\_pycache\_\_/\_\_init\_\_.cpython-312.pyc

create mode 100644 \_\_pycache\_\_/demo\_automation.cpython-312-p

ytest-8.0.0.pyc

create mode 100644 \_\_pycache\_\_/jyo.cpython-312-pytest-8.0.0.

pyc

create mode 100644 \_\_pycache\_\_/test\_demo\_automation.cpython-

312-pytest-8.0.0.pyc

create mode 100644 \_\_pycache\_\_/test\_webelements.cpython-312-

pytest-8.0.0.pyc

create mode 100644 \_\_pycache\_\_/tests-webelements.cpython-312

-pytest-8.0.0.pyc

create mode 100644 \_\_pycache\_\_/webelement.cpython-312.pyc

create mode 100644 test\_webelements.py

create mode 100644 webelement.py

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (master)

$ git remote add origin https://github.com/vara221/sw.git

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sele\_project (master)

$ git push -u origin master

Enumerating objects: 13, done.

Counting objects: 100% (13/13), done.

Delta compression using up to 4 threads

Compressing objects: 100% (12/12), done.

Writing objects: 100% (13/13), 8.39 KiB | 1.40 MiB/s, done.

Total 13 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0

)

remote: Resolving deltas: 100% (2/2), done.

To https://github.com/vara221/sw.git

\* [new branch] master -> master

branch 'master' set up to track 'origin/master'.

#### Local project:

* git init
* git add
* git commit
* git remote add <short-name> <repo-url>
* git push origin master

#### Git Lifecycle of status:

Untracked file : unknow to git

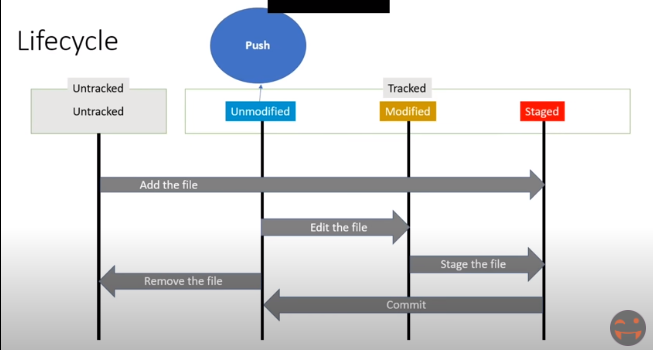
Tracked file : know to git

Git status : to know the stages

Unmodified

Modified

Staged



### Committing Changes:

Now we have a snapshot in the staging area ready to be permanently stored in our repository.

### Capture$ git commit -m "Initial commmit.

This command is used commit the files and display the Initial commit message (-m)

$ git commit

[master (root-commit) 278367e] Initial commit.

2 files changed, 3 insertions(+)

create mode 100644 file1.txt

create mode 100644 file2.txt

Then open one editor commit file in the vs code and edit that commit\_editing file then close.

### Skipping the staging area :

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo test >> file1.txt

-a indicates all files

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git commit -am "Fix the bug that prevented the users from signing up."

[master (root-commit) b2ed89f] Fix the bug that prevented the users from signing

up.

2 files changed, 3 insertions(+)

create mode 100644 file1.txt

create mode 100644 file2.txt

**Removing the Files :**

When we removing the file2.txt , It removed from the Directory but still its existing the staging area.

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ rm file2.txt

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

Changes not staged for commit:

(use "git add/rm <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

deleted: file2.txt

no changes added to commit (use "git add" and/or "git commit -a")

To prove that

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git ls-files

file1.txt

file2.txt

These files are at the staging area. So add file2.txt to stage this change deletion more accurately.

$ git add file2.txt

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git ls-files

file1.txt

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

deleted: file2.txt

It’s in the staging area so lets commit this changes

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git commit -m "Remove unsed code."

[master c95bd43] Remove unsed code.

1 file changed, 1 deletion(-)

delete mode 100644 file2.txt

So removing the file from both the working and directory

Instead of **rm** command in linux using the **git rm file\_name** So git remove this file from both the working directory as well as staging.

$ git rm file2.txt \*.txt

### Renaming or Moving Files:

**By using domain.js with this command we can rename or move**

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ ls

file1.txt

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ mv file1.txt main.js

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

Changes not staged for commit:

(use "git add/rm <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

deleted: file1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

main.js

no changes added to commit (use "git add" and/or "git commit -a")

**In the above case deleted shows red so both these stages are not staged**.

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git add file1.txt

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git add main.js

warning: in the working copy of 'main.js', CRLF will be replaced by LF the next

time Git touches it

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

renamed: file1.txt -> main.js

**File1.txt is renaming the main.js (.js extension used to change the file name)**

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git mv main.js file1.js

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

renamed: file1.txt -> file1.js

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git commit -m "Refactor code."

[master a863076] Refactor code.

1 file changed, 0 insertions(+), 0 deletions(-)

rename file1.txt => file1.js (100%)

### Ignoring Files :

For example :

Create a log directory

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ mkdir logs

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo hello > logs/dev.log

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

Untracked files:

(use "git add <file>..." to include in what will be committed)

logs/

nothing added to commit but untracked files present (use "git add" to track)

In the above logs are untracked but we don’t want to add this to the staging area because we don’t want git to track this so to prevent this we have to create a file

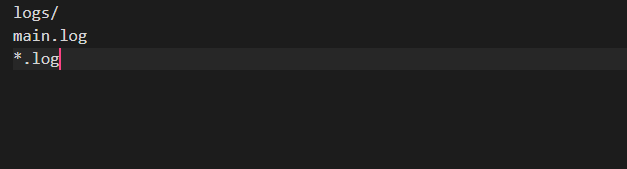
vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo logs/ > .gitignore

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ code .gitignore

File will be opened enter the all log files



vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git add .gitignore

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git commit -m "Add gitignore"

[master ee51386] Add gitignore

1 file changed, 3 insertions(+)

create mode 100644 .gitignore

If accidentally include file in your repository and then later added to git ignore git is not going to ignore let us consider one exmaple

* Create one directory

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ mkdir bin

Let’s imagine that this directory contains our compiled source code

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo hello > bin/app.bin

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

Untracked files:

(use "git add <file>..." to include in what will be committed)

bin/

nothing added to commit but untracked files present (use "git add" to track)

* Add all the changes and then commit

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git add .

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

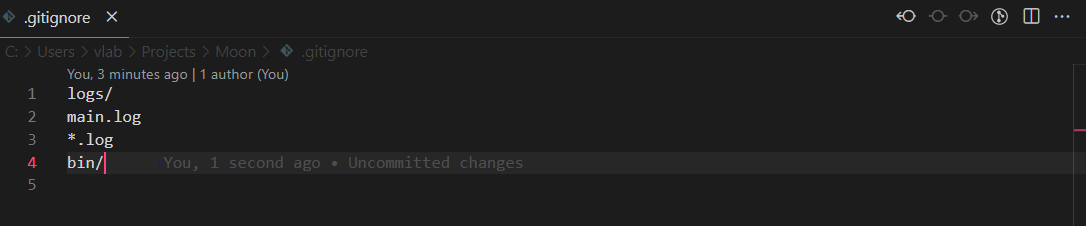
$ git commit -m "Add bin."

[master 0bf1878] Add bin.

1 file changed, 1 insertion(+)

create mode 100644 bin/app.bin

* Here app.bin is at bin directory if we want ignore then edit the **.gitignore** file



vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

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: .gitignore

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git add .

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git commit -m "Inculde bin/ in gitignore."

[master 3fa5187] Inculde bin/ in gitignore.

1 file changed, 1 insertion(+)

* Lets modify app.bin file

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo helloworld > bin/app.bin

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

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: bin/app.bin

no changes added to commit (use "git add" and/or "git commit -a")

Here file will be modified but we want to remove this file from the staging area.

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git ls-files

.gitignore

bin/app.bin

file1.js

* If we remove only from the staging area then

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git rm -h

--cached option : only remove from the index

-r : allow recursive removal

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git rm --cached -r bin/

rm 'bin/app.bin'

Removed the bin from the staging area and verify it

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git ls-files

.gitignore

file1.js

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

deleted: bin/app.bin

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git commit -m "Remove the bin directory that was accidentally committed."

[master a540c24] Remove the bin directory that was accidentally committed.

1 file changed, 1 deletion(-)

delete mode 100644 bin/app.bin

* Check the working directory if file is there are not

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo test > bin/app.bin

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

nothing to commit, working tree clean

### Short Status

* --short
* ?? - Untracked
* A - Staging Area
* M - Modified files

For example:

* Modify one existing file

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo sky >> file1.js

* Create a new file

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo sky > file2.js

* Check the status

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status

On branch master

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: file1.js

Untracked files:

(use "git add <file>..." to include in what will be committed)

file2.js

no changes added to commit (use "git add" and/or "git commit -a")

* Other form of the status

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

M file1.js

?? file2.js

It shows the simple information

file1.js is modified in the working directory that’s why it shows red in color M

File2.js is the new file created that why it shows ??

Then add the file1.js to staging area , it coverts to green

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git add file

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

M file1.js

?? file2.js

* Again modify the file1.js then again it shows red color

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ echo ocean >> file1.js

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

MM file1.js

?? file2.js

Here staging file is not added

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git add file1.js

warning: in the working copy of 'file1.js', CRLF will be replaced by LF the next

time Git touches it

One or more files with the same name they are converted to one single file

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

M file1.js

?? file2.js

* Add second file also

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git add file2.js

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

M file1.js

A file2.js

So, Both the files are in Staging area

### GIT Diff:

Status vs Diff

In status just specified modify or new or unmodifed where as diff shows the modified text

#### Viewing the staged and Unstaged Changes

It shows the difference between the newly created file and modified file

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ vi file1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ vi file2.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git status

On branch main

Untracked files:

(use "git add <file>..." to include in what will be committ

ed)

file1.txt

file2.txt

nothing added to commit but untracked files present (use "git

add" to track)

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git diff

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git add file1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git status

On branch main

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: file1.txt

Untracked files:

(use "git add <file>..." to include in what will be committ

ed)

file2.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git diff

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ vi file1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git status

On branch main

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: file1.txt

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: file1.txt

Untracked files:

(use "git add <file>..." to include in what will be committ

ed)

file2.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git diff

diff --git a/file1.txt b/file1.txt

index 92c5442..3089127 100644

--- a/file1.txt

+++ b/file1.txt

@@ -1 +1 @@

-hello veda

+hello lochani

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git diff --staged

diff --git a/file1.txt b/file1.txt

new file mode 100644

index 0000000..92c5442

--- /dev/null

+++ b/file1.txt

@@ -0,0 +1 @@

+hello veda

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git diff --cached

diff --git a/file1.txt b/file1.txt

new file mode 100644

index 0000000..92c5442

--- /dev/null

+++ b/file1.txt

@@ -0,0 +1 @@

+hello veda

### Viewing the History

* It display the different between each and every two commit

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git log -p

commit 0337202869ac9008756ec68df0632c4fdd4d0ae7 (HEAD -> main

)

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Mon Mar 4 15:56:12 2024 +0530

all files are removed

diff --git a/README.md b/README.md

deleted file mode 100644

index 6733c34..0000000

--- a/README.md

+++ /dev/null

@@ -1 +0,0 @@

-#sw

diff --git a/example1 b/example1

deleted file mode 100644

index 7a64dde..0000000

--- a/example1

+++ /dev/null

@@ -1,4 +0,0 @@

-print("hello lochani")

-print("hello world")

-print("kavya")

-exampel1 edited in branch1

diff --git a/fact.py b/fact.py

deleted file mode 100644

* By using log command to look at our history

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git log

commit a442eadeb945e4bf06135d8fcf9aebd315d1d6fc (HEAD -> master)

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Fri Mar 1 17:32:37 2024 +0530

ewferfrgreg

commit e1f153e9f9c79443b503a1accb7b32d991ea7e7d

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Fri Mar 1 17:23:29 2024 +0530

Initial the new file

commit 53eb36c1f16a49daf34b3ce56c6fb4920906eadc

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Fri Mar 1 15:54:22 2024 +0530

file2.js

commit a540c24a6bd5f507b56ea388fb9ba8f91fb07458

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Fri Mar 1 14:44:02 2024 +0530

Remove the bin directory that was accidentally committed.

: <space Enter> then showing the more information

To quit by using q (command)

**a442eadeb945e4bf06135d8fcf9aebd315d1d6fc :** Each and every commit has a unique identifier this is a 40 character hexadecimal string. That automatically generated the git.

Master is the main branch or main line of work in git.

In some other version control systems its called trunk.

By using this multiple bug fixes in parallel and then combine our code

* It shows the short summary info

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git log --oneline

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git log --pretty=oneline

a442ead (HEAD -> master) Remove the bin directory that was accidentaly committed.

e1f153e Initial the new file

53eb36c file2.js

a540c24 Remove the bin directory that was accidentally committed.

3fa5187 Inculde bin/ in gitignore.

0bf1878 Add bin.

ee51386 Add gitignore

a863076 Refactor code.

c95bd43 Remove unsed code.

b2ed89f Fix the bug that prevented the users from signing up.

* Reverse the information

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git log --oneline --reverse

b2ed89f Fix the bug that prevented the users from signing up.

c95bd43 Remove unsed code.

a863076 Refactor code.

ee51386 Add gitignore

0bf1878 Add bin.

3fa5187 Inculde bin/ in gitignore.

a540c24 Remove the bin directory that was accidentally committed.

53eb36c file2.js

e1f153e Initial the new file

a442ead (HEAD -> master) Remove the bin directory that was accidentaly committed.

* It shows the specific logs

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git log -2

commit 0337202869ac9008756ec68df0632c4fdd4d0ae7 (HEAD -> main

)

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Mon Mar 4 15:56:12 2024 +0530

all files are removed

commit 3b566d151c3770e73bd0d6ddd4fcc56917249198

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Mon Mar 4 13:46:26 2024 +0530

added one print statement example1

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git log --pretty=short

commit 0337202869ac9008756ec68df0632c4fdd4d0ae7 (HEAD -> main

)

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

all files are removed

commit 3b566d151c3770e73bd0d6ddd4fcc56917249198

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

added one print statement example1

commit cfd5bd225123c3a2e607aafbefaa7a62a39b4c44

Author: vara221 <159216502+vara221@users.noreply.github.com>

Delete helloworld.py

commit 73e6952d929fdd5ca39ac8a26747bc009c043182

Author: varalaxmiganta <varalaxmiganta066@gmail.com>

first commit

commit 7ae69df38abb30d7106869fa1a6130ae6e0b6c63

Author: veena <veenanjalitammina999@gmail.com>

commit to push file from branch1 to main

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git log --pretty=full

commit 0337202869ac9008756ec68df0632c4fdd4d0ae7 (HEAD -> main

)

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Commit: varalaxmi ganta <varalaxmiganta066@gmail.com>

all files are removed

commit 3b566d151c3770e73bd0d6ddd4fcc56917249198

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Commit: varalaxmi ganta <varalaxmiganta066@gmail.com>

added one print statement example1

commit cfd5bd225123c3a2e607aafbefaa7a62a39b4c44

Author: vara221 <159216502+vara221@users.noreply.github.com>

Commit: GitHub <noreply@github.com>

Delete helloworld.py

commit 73e6952d929fdd5ca39ac8a26747bc009c043182

Author: varalaxmiganta <varalaxmiganta066@gmail.com>

Commit: varalaxmiganta <varalaxmiganta066@gmail.com>

first commit

commit 7ae69df38abb30d7106869fa1a6130ae6e0b6c63

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ git log --pretty=fuller

commit 0337202869ac9008756ec68df0632c4fdd4d0ae7 (HEAD -> main

)

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

AuthorDate: Mon Mar 4 15:56:12 2024 +0530

Commit: varalaxmi ganta <varalaxmiganta066@gmail.com>

CommitDate: Mon Mar 4 15:56:12 2024 +0530

all files are removed

commit 3b566d151c3770e73bd0d6ddd4fcc56917249198

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

AuthorDate: Mon Mar 4 13:46:26 2024 +0530

Commit: varalaxmi ganta <varalaxmiganta066@gmail.com>

CommitDate: Mon Mar 4 13:46:26 2024 +0530

added one print statement example1

commit cfd5bd225123c3a2e607aafbefaa7a62a39b4c44

Author: vara221 <159216502+vara221@users.noreply.github.c

om>

AuthorDate: Mon Mar 4 13:42:49 2024 +0530

Commit: GitHub <noreply@github.com>

CommitDate: Mon Mar 4 13:42:49 2024 +0530

Delete helloworld.py

### Viewing Commit

Viewing the list of commit but what if you want to see exactly that commit.

There are two ways to reference the commit.

**Commit** :

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git commit -m "Initial commit"

[master (root-commit) 8dd5f18] Initial commit

3 files changed, 24 insertions(+)

create mode 100644 varalaxmi-techs/about.html

create mode 100644 varalaxmi-techs/address.html

create mode 100644 varalaxmi-techs/home.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log

commit 8dd5f180e1ea901b582024bf0055ad71b899b20e (HEAD -> mast

er)

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Mon Mar 4 16:52:27 2024 +0530

Initial commit

Unique identifier : c95bd43

Header point : Head~1 1 represent the list of first commit value

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git show HEAD~1

commit e1f153e9f9c79443b503a1accb7b32d991ea7e7d

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Fri Mar 1 17:23:29 2024 +0530

Initial the new file

diff --git a/file1 b/file1

new file mode 100644

index 0000000..ce01362

--- /dev/null

+++ b/file1

@@ -0,0 +1 @@

+hello

Give the full path also

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git show HEAD~1:.gitignore

logs/

main.log

\*.log

bin/

It shows the all files and directories in a commit

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git show HEAD

commit a442eadeb945e4bf06135d8fcf9aebd315d1d6fc (HEAD -> master)

Author: varalaxmi ganta <varalaxmiganta066@gmail.com>

Date: Fri Mar 1 17:32:37 2024 +0530

ewferfrgreg

diff --git a/file1.js b/file1.js

deleted file mode 100644

index 8f6afb1..0000000

--- a/file1.js

+++ /dev/null

@@ -1,4 +0,0 @@

-hello

-test

-sky

-ocean

diff --git a/file1.txt b/file1.txt

new file mode 100644

index 0000000..228b872

--- /dev/null

+++ b/file1.txt

@@ -0,0 +1,4 @@

+hello

Show the List of files and directories called as Tree manner because tree representing hierarchical information .

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git ls-tree HEAD~1

100644 blob de80a7b9265aa3e11b1a8804a3c62a9492a0e23d .gitignore

100644 blob ce013625030ba8dba906f756967f9e9ca394464a file1

100644 blob 8f6afb1e56e825df257cd855e35158128646380a file1.js

100644 blob f5e95e70e524ec32d0200e10ba179ab4c5f13884 file2.js

Files are represented using blobs

Directories are represented using trees

All these are objects that are stored in git’s database by using the show command we can view an object in git.

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git show de80a

logs/

main.log

\*.log

bin/

$ git show 64629

tree 64629

app.bin

#### GIT OBJECTS

* Commits
* Blobs (Files)
* Trees (Directories)
* Tags

### UNSTAGING FILES

In the previous cases file1.js is added to the staging area now we want to undo this operation

git reset --soft or git reset --hard

by using restore ,restore the working or in the staging area files.

If we want to restore file1 in the staging area SO,

$ git restore --staged file1.js file2.js

Or

$ git restore --staged file1.js \*.js

Or

$ git restore --staged .

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git --version

git version 2.44.0.windows.1

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git restore --staged file1.js

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

M file1.js

?? file2.js

Here m indicates red in color so that file in working directory not in staging area.

File2.js is also undo all the local changes

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git restore .

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

?? file2.js

To remove all these new untracked files

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git clean -h

usage: git clean [-d] [-f] [-i] [-n] [-q] [-e <pattern>] [-x | -X] [--] [<pathsp

ec>...]

-q, --[no-]quiet do not print names of files removed

-n, --[no-]dry-run dry run

-f, --[no-]force force

-i, --[no-]interactive

interactive cleaning

-d remove whole directories

-e, --exclude <pattern>

add <pattern> to ignore rules

-x remove ignored files, too

-X remove only ignored files

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git clean -fd

Removing file1.js

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

### Restoring a File to an Earlier Version :

How to undo local changes files

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git rm -rf file1.txt

rm 'file1.txt'

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git rm -rf file2.txt

rm 'file2.txt'

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

D file1.txt

D file2.txt

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git commit -m "Delete file1.txt and file2.txt"

[master 033b25e] Delete file1.txt and file2.txt

2 files changed, 7 deletions(-)

delete mode 100644 file1.txt

delete mode 100644 file2.txt

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git restore -h

usage: git restore [<options>] [--source=<branch>] <file>...

-s, --[no-]source <tree-ish>

which tree-ish to checkout from

-S, --[no-]staged restore the index

-W, --[no-]worktree restore the working tree (default)

--[no-]ignore-unmerged

ignore unmerged entries

--[no-]overlay use overlay mode

-q, --[no-]quiet suppress progress reporting

--[no-]recurse-submodules[=<checkout>]

control recursive updating of submodules

--[no-]progress force progress reporting

-m, --[no-]merge perform a 3-way merge with the new branch

--[no-]conflict <style>

conflict style (merge, diff3, or zdiff3)

-2, --ours checkout our version for unmerged files

-3, --theirs checkout their version for unmerged files

-p, --[no-]patch select hunks interactively

--[no-]ignore-skip-worktree-bits

do not limit pathspecs to sparse entries only

--[no-]pathspec-from-file <file>

read pathspec from file

--[no-]pathspec-file-nul

with --pathspec-from-file, pathspec elements are separ

ated with NUL character

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git restore --source=HEAD~1 file1.txt

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ git status -s

?? file1.txt

## 

### CONFIGURE NOTEPAD

To integrate the notepad++

vlab@VEDA2F118 MINGW32 ~/Projects/Moon (master)

$ notepad myfile

Open the notepad file then filename will be myfile

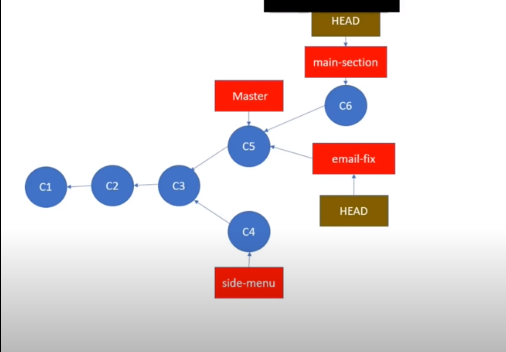
To convert default editor

$ git config --global core.editor “notepad.exe -multilnst -nosession”

$ git config --global --list

$ git config --global -e

### GIT Branching:



* Create a folder named varalaxmi-techs

vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ mkdir varalaxmi-techs

vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ cd varalaxmi-techs

* Create three files :

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ vi home.html

<html>

<body>

<h1>Varalaxmi Techs</h1>

<p>learning github</p>

</body>

</html>

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ vi about.html

<html>

<body>

<h2>About</h2>

<p>we learn C,C++,Python etc...</p>

</body>

</html>

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ vi address.html

<html>

<body>

<h3>Address</h3>

<pre>

varalaxmiTechs,

Everywhere,

tech.varaganta

</pre>

</body>

</html>

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ ls

about.html address.html home.html

* Initialize git:

$ git init

**Master:** vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master) **default master branch**

**Branch - Pointer :** Pointer to one of the commit’s

Creating a new branch

* Edit one home file

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ vi home.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git status

On branch master

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: home.html

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git add home.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git commit -m "updated home page"

[master e68ab89] updated home page

1 file changed, 1 insertion(+)

* **First commit doesn’t have any parent commit where as second commit has one parent commit**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ vi about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git status

On branch master

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: about.html

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git add about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git commit -m "updated about html page"

[master df89218] updated about html page

1 file changed, 1 insertion(+)

* **Third cpmmit is derived from second commit**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline

df89218 (HEAD -> master) updated about html page

e68ab89 updated home page

40fbffe updated about page

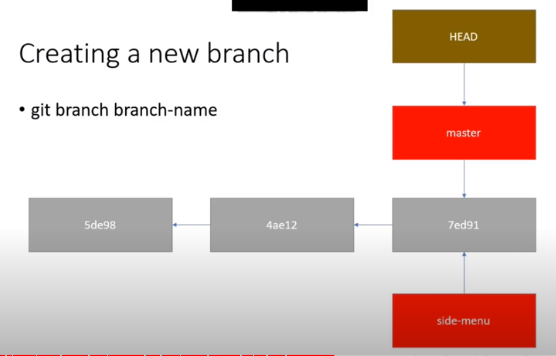
5357e8d adding one new file

8dd5f18 (side-menu) Initial commit

Branch is nothing but a pointer it shows to commit

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git branch side-menu



vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline

8dd5f18 (HEAD -> master, side-menu) Initial commit

Master and side-menu branchs are point to the same commit

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

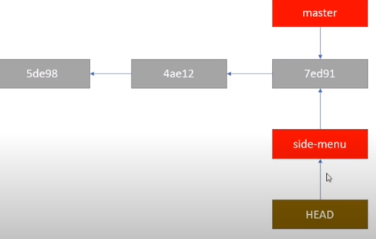
$ git checkout side-menu

Switched to branch 'side-menu'

**Switching Branches:**

git checkout branch-name

As here commit, new branch moves



Here head is point to the side-menu

Every time branch will be moved when commit will added

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git log --oneline

8dd5f18 (HEAD -> side-menu) Initial commit

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ vi nav\_bar

This change will be at side-menu branch

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git status

On branch side-menu

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

nav\_bar

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git add nav\_bar

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git commit -m "added nav bar"

[side-menu 765fcba] added nav bar

1 file changed, 2 insertions(+)

create mode 100644 varalaxmi-techs/nav\_bar

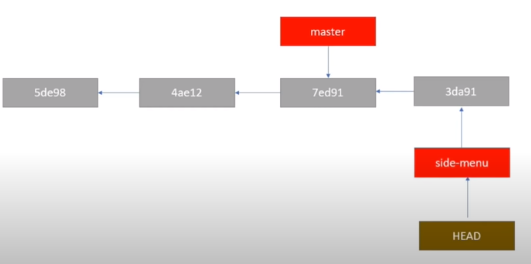
vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git log --oneline

765fcba (HEAD -> side-menu) added nav bar

8dd5f18 Initial commit

side-menu will be pointed to the new commit and master will be pointed to the previous commit



**Switch back to master:**

* git checkout master
* Note : Switching branches changes files in your working directory

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git checkout master

Switched to branch 'master'

But it doesn’t show the side-menu branch so that use --all command

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline

df89218 (HEAD -> master) updated about html page

e68ab89 updated home page

40fbffe updated about page

5357e8d adding one new file

8dd5f18 Initial commit

It shows all the branch whatever we created

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline --all

765fcba (side-menu) added nav bar

df89218 (HEAD -> master) updated about html page

e68ab89 updated home page

40fbffe updated about page

5357e8d adding one new file

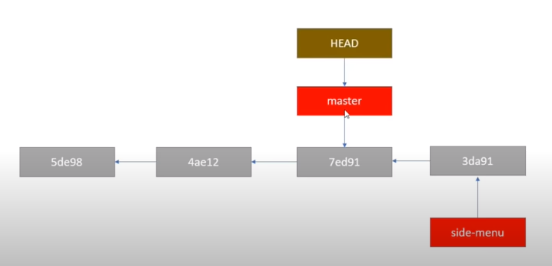
8dd5f18 Initial commit

Here we can’t see the nav\_bar file because we can do commits at side-menu branch

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ ls

about.html address.html file2.txt home.html



vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ vi about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git status

On branch master

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: about.html

Untracked files:

(use "git add <file>..." to include in what will be committ

ed)

../file1

../project/

no changes added to commit (use "git add" and/or "git commit

-a")

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git add about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git commit -m "updated about page"

[master 40fbffe] updated about page

1 file changed, 1 insertion(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline --graph

\* 4ea2aa6 (HEAD -> master) updated about page

\* df89218 updated about html page

\* e68ab89 updated home page

\* 40fbffe updated about page

\* 5357e8d adding one new file

\* 8dd5f18 Initial commit

Or

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline --graph --all

\* 4ea2aa6 (HEAD -> master) updated about page

\* df89218 updated about html page

\* e68ab89 updated home page

\* 40fbffe updated about page

\* 5357e8d adding one new file

| \* 765fcba (side-menu) added nav bar

|/

\* 8dd5f18 Initial commit

Master head pointed to the new commit



**Creating branch and switching same time :**

* git checkout -b branch-name
* git checkout -b main-section

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git checkout -b main-section

Switched to a new branch 'main-section'

Edit one file in the main-section

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ vi home.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git status

On branch main-section

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: home.html

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git add home.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git commit -m "updated main section"

[main-section 1dabad9] updated main section

1 file changed, 2 insertions(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git checkout master

Switched to branch 'master'

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git status

On branch master

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

nothing added to commit but untracked files present (use "git add" to track)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline --graph --all

\* 1dabad9 (main-section) updated main section

\* 4ea2aa6 (HEAD -> master) updated about page

\* df89218 updated about html page

\* e68ab89 updated home page

\* 40fbffe updated about page

\* 5357e8d adding one new file

| \* 765fcba (side-menu) added nav bar

|/

\* 8dd5f18 Initial commit

**Edit address.html files at other place after modify then merge into the master**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git checkout -b email-fix

Switched to a new branch 'email-fix'

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (email-fix)

$ ls

about.html address.html file2.txt home.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (email-fix)

$ vi address.html

<html>

<body>

<h3>Address</h3>

<pre>

varalaxmiTechs,

Everywhere,

tech.varaganta

</pre>

</body>

</html>

Tech.varaganta change to [tech.varalaxmiganta066@gmail.com](mailto:tech.varalaxmiganta066@gmail.com)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (email-fix)

$ git status

On branch email-fix

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: address.html

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (email-fix)

$ git add address.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (email-fix)

$ git commit -m "fixed email problem"

[email-fix 6f84ba1] fixed email problem

1 file changed, 1 insertion(+), 1 deletion(-)

### Merging :

Two branches are merging

Email-fix is merge to the master branch

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (email-fix)

$ git checkout master

Switched to branch 'master'

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ ls

about.html address.html file2.txt home.html

**Here we can’t saw the modified content because we are in master branch**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ cat address.html

<html>

<body>

<h3>Address</h3>

<pre>

varalaxmiTechs,

Everywhere,

**tech.varaganta**

</pre>

</body>

</html>

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git merge email-fix

Updating 4ea2aa6..6f84ba1

Fast-forward

varalaxmi-techs/address.html | 2 +-

1 file changed, 1 insertion(+), 1 deletion(-)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ cat address.html

<html>

<body>

<h3>Address</h3>

<pre>

varalaxmiTechs,

Everywhere,

**tech.varalaxmiganta066@gmail.com**

</pre>

</body>

</html>

Fast forward Merge :

If two branches are merge then they are comes under single commit then it is called as fast forward merge.

**Here master and email-fix are at point to the same commit**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline --graph --all

\* 6f84ba1 (HEAD -> master, email-fix) fixed email problem

| \* 1dabad9 (main-section) updated main section

|/

\* 4ea2aa6 updated about page

\* df89218 updated about html page

\* e68ab89 updated home page

\* 40fbffe updated about page

\* 5357e8d adding one new file

| \* 765fcba (side-menu) added nav bar

|/

\* 8dd5f18 Initial commit

### Deleting a Branch :

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline --graph --all

\* 6f84ba1 (HEAD -> master, email-fix) fixed email problem

| \* 1dabad9 (main-section) updated main section

|/

\* 4ea2aa6 updated about page

\* df89218 updated about html page

\* e68ab89 updated home page

\* 40fbffe updated about page

\* 5357e8d adding one new file

| \* 765fcba (side-menu) added nav bar

|/

\* 8dd5f18 Initial commit

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git branch -d email-fix

Deleted branch email-fix (was 6f84ba1).

If any commit can be done we can use git branch -D email-fix

**Deleted the email-fix branch**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline --graph --all

\* 6f84ba1 (HEAD -> master) fixed email problem

| \* 1dabad9 (main-section) updated main section

|/

\* 4ea2aa6 updated about page

\* df89218 updated about html page

\* e68ab89 updated home page

\* 40fbffe updated about page

\* 5357e8d adding one new file

| \* 765fcba (side-menu) added nav bar

|/

\* 8dd5f18 Initial commit

**Check modified content address.html file into the main-section branch**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git checkout main-section

Switched to branch 'main-section'

It doesn’t show any modified content

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ cat address.html

<html>

<body>

<h3>Address</h3>

<pre>

varalaxmiTechs,

Everywhere,

tech.varaganta

</pre>

</body>

</html>

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ vi home.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git status

On branch main-section

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: home.html

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git add home.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git commit -m "finished main section"

[main-section 1378a67] finished main section

1 file changed, 2 insertions(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git log --oneline --graph --all

\* 1378a67 (HEAD -> main-section) finished main section

\* 1dabad9 updated main section

| \* 6f84ba1 (master) fixed email problem

|/

\* 4ea2aa6 updated about page

\* df89218 updated about html page

\* e68ab89 updated home page

\* 40fbffe updated about page

\* 5357e8d adding one new file

| \* 765fcba (side-menu) added nav bar

|/

\* 8dd5f18 Initial commit

### Then merge the main-section branch into the master

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (main-section)

$ git checkout master

Switched to branch 'master'

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ cat home.html

vara laxmi

<html>

<body>

<h1>Varalaxmi Techs</h1>

<p>learning github</p>

</body>

</html>

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git merge main-section

Merge made by the 'ort' strategy.

varalaxmi-techs/home.html | 4 ++++

1 file changed, 4 insertions(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ cat home.html

vara laxmi

lochani

veenanjali

jyothsna

<html>

<body>

<h1>Varalaxmi Techs</h1>

<p>learning github</p>

</body>

</html>

### Merge Conflicts :

**By using git branch ---merge to know the what branch are merge**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git branch --merge

main-section

\* master

**To know what branches are bot merge**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git branch --no-merge

side-menu

**If we 2 members can do change the single line then merge here get some conflicts**

**To display all the branches**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git branch -a

main-section

\* master

side-menu

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ cat about.html

<html>

<body>

<h2>About</h2>

<p>we learn C,C++,Python etc...</p>

</body>

</html>

dsfsdfdsf

egresgrg

hjjhfvuysvyf

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ vi about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ cat about.html

<html>

<body>

<h2>About</h2>

<p>we learn C,C++,Pyt**hon,Machine Learning,pytest** etc...</p>

</body>

</html>

dsfsdfdsf hbvcghc sjkgajd

egresgrg

hjjhfvuysvyf

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: about.html

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: about.html

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git add .

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git commit -m "updated about -added new techn"

[master b9391ae] updated about -added new techn

1 file changed, 1 insertion(+), 1 deletion(-)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git checkout side-menu

Switched to branch 'side-menu'

**Here can’t see the change because changes can be done at master**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ cat about.html

<html>

<body>

<h2>About</h2>

<p>we learn C,C++,Python,Machine Learning etc...</p>

</body>

</html>

Edit the same line

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ vi about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ cat about.html

<html>

<body>

<h2>About</h2>

<p>we learn C,C++,**Python,Machine Learning, pychram** etc...</p>

</body>

</html>

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git status

On branch side-menu

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: about.html

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git add about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git commit -m "updated new techn"

[side-menu 237e110] updated new techn

1 file changed, 1 insertion(+), 1 deletion(-)

**Then Merge two branches**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (side-menu)

$ git checkout master

Switched to branch 'master'

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git merge side-menu

Auto-merging varalaxmi-techs/about.html

CONFLICT (content): Merge conflict in varalaxmi-techs/about.html

Automatic merge failed; fix conflicts and then commit the result.

Here master | merging means merging will be pausing stage

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master|MERGING)

$ git status

On branch master

You have unmerged paths.

(fix conflicts and run "git commit")

(use "git merge --abort" to abort the merge)

Unmerged paths:

(use "git add <file>..." to mark resolution)

both modified: about.html

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

no changes added to commit (use "git add" and/or "git commit -a")

**Cat about.html**

<html>

<body>

<h2>About</h2>

<<<<<<< HEAD

<p>we learn C,C++,Python,Machine Learning,pytest etc...</p>

=======

<p>we learn C,C++,Python,Machine Learning, pychram etc...</p>

>>>>>>> side-menu

</body>

</html>

dsfsdfdsf hbvcghc sjkgajd

egresgrg

hjjhfvuysvyf

**So u can delete the side-menu or head then side-menu will be deleted**

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master|MERGING)

$ vi about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master|MERGING)

$ cat about.html

<html>

<body>

<h2>About</h2>

<p>we learn C,C++,Python,Machine Learning,pytest etc...</p>

</body>

</html>

dsfsdfdsf hbvcghc sjkgajd

egresgrg

hjjhfvuysvyf

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master|MERGING)

$ git add about.html

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master|MERGING)

$ git status

On branch master

All conflicts fixed but you are still merging.

(use "git commit" to conclude merge)

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master|MERGING)

$ git commit

[master 33290d2] Merge branch 'side-menu'

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git status

On branch master

Untracked files:

(use "git add <file>..." to include in what will be committed)

../file1

../project/

nothing added to commit but untracked files present (use "git add" to track)

vlab@VEDA2F118 MINGW32 ~/GitProject/varalaxmi-techs (master)

$ git log --oneline --graph --all

\* 33290d2 (HEAD -> master) Merge branch 'side-menu'

|\

| \* efb5842 (side-menu) updated new techn

| \* 20b24b4 add one file

\* | b9391ae updated about -added new techn

\* | bbe591c merge both files Merge branch 'side-menu'

|\|

| \* 237e110 updated new techn

| \* 765fcba added nav bar

\* | ad6b85f updated about -added new lines

\* | df1bc7b Merge branch 'main-section'

|\ \

| \* | 1378a67 (main-section) finished main section

| \* | 1dabad9 updated main section

\* | | 6f84ba1 fixed email problem

|/ /

\* | 4ea2aa6 updated about page

\* | df89218 updated about html page

\* | e68ab89 updated home page

\* | 40fbffe updated about page

\* | 5357e8d adding one new file

|/

\* 8dd5f18 Initial commit

### Remote Branching :



vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ git clone https://github.com/vara221/sw.git

Cloning into 'sw'...

remote: Enumerating objects: 13, done.

remote: Counting objects: 100% (13/13), done.

remote: Compressing objects: 100% (10/10), done.

remote: Total 13 (delta 2), reused 13 (delta 2), pack-reused 0

Receiving objects: 100% (13/13), 8.39 KiB | 429.00 KiB/s, done.

Resolving deltas: 100% (2/2), done.

vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ ls

file1 project/ sw/ varalaxmi-techs/

vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ cd sw/

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git remote

origin

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ ls

\_\_init\_\_.py \_\_pycache\_\_/ ho test\_webelements.py webelement.py

These are all local changes

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ vi webelement.py

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git status

On branch master

Your branch is up to date with 'origin/master'.

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: webelement.py

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git log --oneline --all --graph

\* d000009 (HEAD -> master, origin/master, origin/HEAD) Initial commit

Origin/master , origin/HEAD these remote branching

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git add .

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git commit -m "updated webelement.py "

[master 2910fb9] updated webelement.py

1 file changed, 2 insertions(+), 1 deletion(-)

Here remote references are same but local pointer will be changed

Remote references are origin/master and origin/HEAD

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git log --oneline --all --graph

\* 2910fb9 (HEAD -> master) updated webelement.py

\* d000009 (origin/master, origin/HEAD) Initial commit

If added one new into the remote loaction (sw - testing.txt)

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ ls

\_\_init\_\_.py \_\_pycache\_\_/ test\_webelements.py webelement.py

It can’t be see here

So, we can move that change to here

#### FETCH AND PULL :

FETCH : To retrieve new work done by other people but doesn’t apply to the local system.

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git fetch origin

remote: Enumerating objects: 4, done.

remote: Counting objects: 100% (4/4), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 936 bytes | 6.00 KiB/s, done.

From https://github.com/vara221/sw

d000009..ef5322d master -> origin/master

Here we can’t see the testing.txt file but It is exist

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ ls

\_\_init\_\_.py \_\_pycache\_\_/ test\_webelements.py webelement.py

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git branch -a

\* master

remotes/origin/HEAD -> origin/master

remotes/origin/master

Manually we can merge

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git merge origin/master

Merge made by the 'ort' strategy.

testing.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 testing.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ ls

\_\_init\_\_.py \_\_pycache\_\_/ test\_webelements.py testing.txt webelement.py

PULL : To retrieve new work done by other people and It can be applied to the working directory.

Add one file to the git repo (sw)

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git pull origin master

From https://github.com/vara221/sw

\* branch master -> FETCH\_HEAD

Already up to date.

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ ls

\_\_init\_\_.py test.txt testing.txt

\_\_pycache\_\_/ test\_webelements.py webelement.py

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git log --oneline --all --graph

\* 524debf (HEAD -> master) Merge branch 'master' of h

ttps://github.com/vara221/sw

|\

| \* 6904d96 (origin/master, origin/HEAD) Create test.tx

t

\* | 52d15f8 Merge remote-tracking branch 'origin/master

'

|\|

| \* ef5322d Create testing.txt

\* | 2910fb9 updated webelement.py

|/

\* d000009 Initial commit

#### PUSH :

Here local has one commit then push to server

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git pull origin master

From https://github.com/vara221/sw

\* branch master -> FETCH\_HEAD

Already up to date.

vlab@VEDA2F118 MINGW32 ~/GitProject/sw (master)

$ git push origin master

Enumerating objects: 13, done.

Counting objects: 100% (11/11), done.

Delta compression using up to 4 threads

Compressing objects: 100% (7/7), done.

Writing objects: 100% (7/7), 786 bytes | 262.00 KiB/s,

done.

Total 7 (delta 4), reused 0 (delta 0), pack-reused 0 (f

rom 0)

remote: Resolving deltas: 100% (4/4), completed with 2

local objects.

To https://github.com/vara221/sw.git

6904d96..524debf master -> master

Then check once github

### ALIASES :

vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ git log --oneline --graph --all

\* a5b5af9 (HEAD -> master) hhjf

\* 8a5fff6 modified one new file

\* 3bf89f8 modified one new file

\* b0efa9e add one fileeeee1111

|\

| \* 540070c add one fileeeee

\* | c249945 add one fileeeee1111

|/

\* e012296 add one file

\* cdf9120 added other file

\* 6906b8f add one file

\* 89517df modified add one file1

\* c33eae6 modified add one file

\* b512b6d add one file

\* 33290d2 Merge branch 'side-menu'

|\

| \* efb5842 (side-menu) updated new techn

| \* 20b24b4 add one file

\* | b9391ae updated about -added new techn

\* | bbe591c merge both files Merge branch 'side-menu'

|\|

| \* 237e110 updated new techn

| \* 765fcba added nav bar

\* | ad6b85f updated about -added new lines

\* | df1bc7b Merge branch 'main-section'

vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ git config --global alias.allcommits "log --oneline --graph --all"

vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ git allcommits

\* a5b5af9 (HEAD -> master) hhjf

\* 8a5fff6 modified one new file

\* 3bf89f8 modified one new file

\* b0efa9e add one fileeeee1111

|\

| \* 540070c add one fileeeee

\* | c249945 add one fileeeee1111

|/

\* e012296 add one file

\* cdf9120 added other file

\* 6906b8f add one file

\* 89517df modified add one file1

\* c33eae6 modified add one file

\* b512b6d add one file

\* 33290d2 Merge branch 'side-menu'

|\

| \* efb5842 (side-menu) updated new techn

| \* 20b24b4 add one file

\* | b9391ae updated about -added new techn

\* | bbe591c merge both files Merge branch 'side-menu'

|\|

| \* 237e110 updated new techn

| \* 765fcba added nav bar

\* | ad6b85f updated about -added new lines

\* | df1bc7b Merge branch 'main-section'

### REBASE:

Rebase is same as merge but commit history will show the linearly

vlab@VEDA2F118 MINGW32 ~/GitProject (master)

$ cd project/

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ cd sim

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (main)

$ vi file

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (main)

$ vi home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (main)

$ git init

Initialized empty Git repository in C:/Users/vlab/GitProject/p

roject/sim/.git/

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committe

d)

file

home

nothing added to commit but untracked files present (use "git

add" to track)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git add .

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git commit -m "Initial commit"

[master (root-commit) a84d372] Initial commit

2 files changed, 3 insertions(+)

create mode 100644 file

create mode 10c0644 home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git status

On branch master

nothing to commit, working tree clean

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git allcommits

\* a84d372 (HEAD -> master) Initial commit

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git checkout -b bottle

Switched to a new branch 'bottle'

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (bottle)

$ ls

file home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (bottle)

$ vi home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (bottle)

$ git status

On branch bottle

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working d

irectory)

modified: home

no changes added to commit (use "git add" and/or "git commit -

a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (bottle)

$ git add .

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (bottle)

$ git commit -m "updated hone page"

[bottle b6ab1dd] updated hone page

1 file changed, 1 insertion(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (bottle)

$ git allcommits

\* b6ab1dd (HEAD -> bottle) updated hone page

\* a84d372 (master) Initial commit

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (bottle)

$ git checkout master

Switched to branch 'master'

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ ls

file home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ vi file

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git add .

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git commit -m "updated file page"

[master bbbccfe] updated file page

1 file changed, 1 insertion(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git status

On branch master

nothing to commit, working tree clean

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git allcommits

\* bbbccfe (HEAD -> master) updated file page

| \* b6ab1dd (bottle) updated home page

|/

\* a84d372 Initial commit

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git merge bottle

Merge made by the 'ort' strategy.

home | 1 +

1 file changed, 1 insertion(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/sim (master)

$ git allcommits

\* b89ede4 (HEAD -> master) merging Merge branch 'bottle'

|\

| \* b6ab1dd (bottle) updated hone page

\* | bbbccfe updated file page

|/

\* a84d372 Initial commit

#### By using rebase

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ mkdir rebase\_1

vlab@VEDA2F118 MINGW32 ~/GitProject/project (main)

$ cd rebase\_1

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (main)

$ vi home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (main)

$ vi about

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (main)

$ git init

Initialized empty Git repository in C:/Users/vlab/GitProject/p

roject/rebase\_1/.git/

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git add .

g

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git commit -m "Initial commit"

[master (root-commit) ecaf27c] Initial commit

2 files changed, 5 insertions(+)

create mode 100644 about

create mode 100644 home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git checkout -b new-feature

Switched to a new branch 'new-feature'

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (new-feature)

$ ls

about home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (new-feature)

$ vi home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (new-feature)

$ git status

On branch new-feature

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working d

irectory)

modified: home

no changes added to commit (use "git add" and/or "git commit -

a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (new-feature)

$ git add .

g

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (new-feature)

$ git commit -m "updated home page"

[new-feature 2ab0916] updated home page

1 file changed, 1 insertion(+), 1 deletion(-)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (new-feature)

$ git log --oneline --graph --all

\* 2ab0916 (HEAD -> new-feature) updated home page

\* ecaf27c (master) Initial commit

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (new-feature)

$ git checkout master

Switched to branch 'master'

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi about

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git add .

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git commit -m "updated about page"

[master 4aa6064] updated about page

1 file changed, 1 insertion(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

nothing to commit, working tree clean

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git allcommits

\* 4aa6064 (HEAD -> master) updated about page

| \* 2ab0916 (new-feature) updated home page

|/

\* ecaf27c Initial commit

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git merge new-feature

Merge made by the 'ort' strategy.

home | 2 +-

1 file changed, 1 insertion(+), 1 deletion(-)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --all --graph

\* 2b69751 (HEAD -> master) Merging branch 'new-feature'

|\

| \* 2ab0916 (new-feature) updated home page

\* | 4aa6064 updated about page

|/

\* ecaf27c Initial commit

Or

**In rebase brand new commit will not be formed it will last most command**

**Changes are displayed but commit will be changed**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git rebase new-feature

Successfully rebased and updated refs/heads/master.

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --all --graph

\* b459aa4 (HEAD -> master) updated about page

\* 2ab0916 (new-feature) updated home page

\* ecaf27c Initial commit

### Stash:

* Delete(backup)
* Shift+delete(permanent)

Git works based on stack

#### Commands :

* git stash push
* git stash list
* git stash apply
* git stash apply stash@{1}
* git stash drop
* git stash pop
* git stash -u

#### At staging Files :(Back up Delete)

Modify this home file

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi home

This file put into the backup by usng stash

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working d

irectory)

modified: home

no changes added to commit (use "git add" and/or "git commit -

a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash

Or

$ git stash push

Saved working directory and index state WIP on master: b459aa4

updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

nothing to commit, working tree clean

Previous changes are not displayed they are at stash bin

To display all the stash list

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash list

stash@{0}: WIP on master: b459aa4 updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi about

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working d

irectory)

modified: about

modified: home

no changes added to commit (use "git add" and/or "git commit -

a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash

Saved working directory and index state WIP on master: b459aa4

updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash list

stash@{0}: WIP on master: b459aa4 updated about page

stash@{1}: WIP on master: b459aa4 updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master

$ git stash apply

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working

irectory)

modified: about

modified: home

no changes added to commit (use "git add" and/or "git commit

a")

When stash apply after git stash again shows list are same

These changes are apply to the project but stash will not deleted

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash list

stash@{0}: WIP on master: b459aa4 updated about page

stash@{1}: WIP on master: b459aa4 updated about page

$ git stash apply stash@{1}

From stash bin we can delete by using drop

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash drop

Dropped refs/stash@{0} (fa4ff8ea432c9b125f69e29651e7840e3563

e20d)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash list

stash@{0}: WIP on master: b459aa4 updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash drop

Dropped refs/stash@{0} (164b585058eca767c54a858512d2a943f350

f9f7)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash list

**Again use git stash**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

modified: home

no changes added to commit (use "git add" and/or "git commit

-a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash

Saved working directory and index state WIP on master: b459a

a4 updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

nothing to commit, working tree clean

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (maste)

$ git stash list

stash@{0}: WIP on master: b459aa4 updated about page

**Here pop is used to applied the changes and droped that list**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (maste)

$ git stash pop

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed

(use "git restore <file>..." to discard changes in workin

directory)

modified: about

modified: home

no changes added to commit (use "git add" and/or "git commi

-a")

Dropped refs/stash@{0} (ac7af2e9b4240dca2a7b36a9c298393e2b5

ff3a)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (maste)

$ git stash list

#### Git stash is stashed at modified states and staged states but can’t stash untracked files :

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ ls

about home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed

)

(use "git restore <file>..." to discard changes in workin

g directory)

modified: about

modified: home

no changes added to commit (use "git add" and/or "git commi

t -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi testing.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed

)

(use "git restore <file>..." to discard changes in workin

g directory)

modified: about

modified: home

Untracked files:

(use "git add <file>..." to include in what will be commi

tted)

testing.txt

no changes added to commit (use "git add" and/or "git commi

t -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash

Saved working directory and index state WIP on master: b459

aa4 updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

Untracked files:

(use "git add <file>..." to include in what will be commi

tted)

testing.txt

nothing added to commit but untracked files present (use "g

it add to track)

#### In case Untracked states are also changed to stash by using :

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash -u

Saved working directory and index state WIP on master: b459

aa4 updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

#### nothing to commit, working tree clean

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash list

stash@{0}: WIP on master: b459aa4 updated about page

stash@{1}: WIP on master: b459aa4 updated about page

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash pop

Already up to date.

On branch master

Untracked files:

(use "git add <file>..." to include in what will be commi

tted)

testing.txt

nothing added to commit but untracked files present (use "g

it add" to track)

Dropped refs/stash@{0} (c20a6c73eae2f4d93a48014ea5cca092b7b

c0148)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash list

stash@{0}: WIP on master: b459aa4 updated about page

**If we work on old stash then create new branch :**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git stash branch oldidea

Switched to a new branch 'oldidea'

On branch oldidea

Changes not staged for commit:

(use "git add <file>..." to update what will be committed

)

(use "git restore <file>..." to discard changes in workin

g directory)

modified: about

modified: home

Untracked files:

(use "git add <file>..." to include in what will be commi

tted)

testing.txt

no changes added to commit (use "git add" and/or "git commi

t -a")

Dropped refs/stash@{0} (7cbb18f7b15f08179d580d37b35473e8306

332fe)

**Here created a new branch and apply all stash and also drop the list :**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (oldidea)

$ git stash list

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (oldidea)

$ ls

about home testing.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (oldidea)

$ cat home

hii

wel

come

bye

#### Git Cleaning : shift + delete(permanent delete)

Cleaning is done at only untracked files only but not in modified or staged state

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi about

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

no changes added to commit (use "git add" and/or "git commit -a")

Its not working because about file is in modified state

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean

fatal: clean.requireForce defaults to true and neither -i, -n, nor -f g

iven; refusing to clean

It doesn’t show any error

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean -f

When we check its again shows modified about file

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

no changes added to commit (use "git add" and/or "git commit -a")

**Create one untracked file:**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi testing1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

Untracked files:

(use "git add <file>..." to include in what will be committed)

testing1.txt

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ ls

about home testing.txt testing1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean -f

Removing testing1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ ls

about home testing.txt

**If there is a available folder in that folder having untracked files**

**if they are deleted or not check once**

new.txt files is untracked

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi new.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ mkdir test

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ ls

about home new.txt test/ testing.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ cd test

New1.txt files is also untracked

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1/test (master)

$ vi new1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

Untracked files:

(use "git add <file>..." to include in what will be committed)

new.txt

test/

no changes added to commit (use "git add" and/or "git commit -a")

**Here deleted only new.txt file only**

**Git clean deletes only files not folders**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean -f

Removing new.txt

Test directory is available

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ ls

about home test/ testing.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

Untracked files:

(use "git add <file>..." to include in what will be committed)

test/

no changes added to commit (use "git add" and/or "git commit -a")

**If directory also deleted by using git clean -f -d(directory)**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean -fd

Removing test/

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

no changes added to commit (use "git add" and/or "git commit -a")

**Git clean also delete .gitignore available special files also**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi automatic.orig

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

Untracked files:

(use "git add <file>..." to include in what will be committed)

automatic.orig

no changes added to commit (use "git add" and/or "git commit -a")

**Automatic.orig is untracked file and ignore**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi .gitignore

Added one expression that is \*.orig in gitignore file

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: about

Untracked files:

(use "git add <file>..." to include in what will be committed)

.gitignore

no changes added to commit (use "git add" and/or "git commit -a")

**Then it shows only .gitignore file only not automatic.txt file**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ ls

about automatic.orig home testing.txt

-x : used at to deleting ignore files

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean -f -d -x

Removing .gitignore

Removing automatic.orig

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ ls

about home testing.txt

**To know what happened after git clean**

**Git clean -f -n**

**Or**

**Git clean -f --dry-run**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean -f --dry-run

Would remove testing1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean -f -n

Would remove testing1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ ls

about home testing.txt testing1.txt

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git clean -f

Removing testing1.txt

### CaptureTAGGING

Tagging is known as marking , mark the particular

commit .

To know the by default list of tags:

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git tag --list

Or

$ git tag -l

Or

$ git tag

**Create a new tag:**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --all --graph

\* 7e6c51c (HEAD -> master) modified all files

\* b459aa4 (oldidea) updated about page

\* 2ab0916 (new-feature) updated home page

\* ecaf27c Initial commit

At latest commit will be modified files on that commit mark the tag

V1 .0 is a light weight tag it has no description

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git tag v1.0

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --all --graph

\* 7e6c51c (HEAD -> master, tag: v1.0) modified all files

\* b459aa4 (oldidea) updated about page

\* 2ab0916 (new-feature) updated home page

\* ecaf27c Initial commit

Add description to a tag it is called as annotated Tags

Expression commit means -am

Modified about file

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi about

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git commit -am "updated"

[master 39d3b91] updated

1 file changed, 2 insertions(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --graph --all

\* 39d3b91 (HEAD -> master) updated

\* 7e6c51c (tag: v1.0) modified all files

\* b459aa4 (oldidea) updated about page

\* 2ab0916 (new-feature) updated home page

\* ecaf27c Initial commit

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ vi home

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git status

On branch master

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: home

no changes added to commit (use "git add" and/or "git commit -a")

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git commit -am "updated home page"

[master 6dbf0c5] updated home page

1 file changed, 1 insertion(+)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --graph --all

\* 6dbf0c5 (HEAD -> master) updated home page

\* 39d3b91 updated

\* 7e6c51c (tag: v1.0) modified all files

\* b459aa4 (oldidea) updated about page

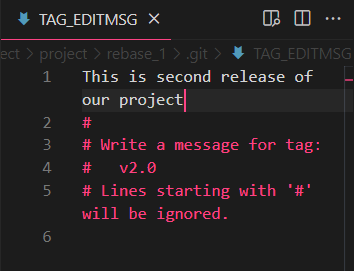
\* 2ab0916 (new-feature) updated home page

\* ecaf27c Initial commit

Gives a annoted tag by using -a

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git tag -a v2.0



Open one default editor and edit their

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git tag

v1.0

v2.0

Second tag will be created

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --graph --all

\* 6dbf0c5 (HEAD -> master, tag: v2.0) updated home page

\* 39d3b91 updated

\* 7e6c51c (tag: v1.0) modified all files

\* b459aa4 (oldidea) updated about page

\* 2ab0916 (new-feature) updated home page

\* ecaf27c Initial commit

**Compare two tags:**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git diff v1.0 v2.0

diff --git a/about b/about

index 7273a96..8761819 100644

--- a/about

+++ b/about

@@ -2,3 +2,5 @@ why

what

where

whom

+hello

+hiii

diff --git a/home b/home

index 8c71a74..dad240d 100644

--- a/home

+++ b/home

@@ -2,3 +2,4 @@ hii

wel

come

bye

+jgjdsg

**Delete a Tag :**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git tag -d v1.0

Deleted tag 'v1.0' (was 7e6c51c)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --graph --all

\* 6dbf0c5 (HEAD -> master, tag: v2.0) updated home page

\* 39d3b91 updated

\* 7e6c51c modified all files

\* b459aa4 (oldidea) updated about page

\* 2ab0916 (new-feature) updated home page

\* ecaf27c Initial commit

**Give a tag with Already available commit :**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git tag -a v1.0 2ab0916

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --graph --all

\* 6dbf0c5 (HEAD -> master, tag: v2.0) updated home page

\* 39d3b91 updated

\* 7e6c51c modified all files

\* b459aa4 (oldidea) updated about page

\* 2ab0916 (tag: v1.0, new-feature) updated home page

\* ecaf27c Initial commit

**By mistake you give a tag at one commit but I have to change that tag to other commit:**

**So that update a tag or delete a tag :**

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git tag -a v1.0 -f 7e6c51c

Updated tag 'v1.0' (was fc46818)

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git log --oneline --graph --all

\* 6dbf0c5 (HEAD -> master, tag: v2.0) updated home page

\* 39d3b91 updated

\* 7e6c51c (tag: v1.0) modified all files

\* b459aa4 (oldidea) updated about page

\* 2ab0916 (new-feature) updated home page

\* ecaf27c Initial commit

**If you push any commit only that commit will be moved tags are not moved by using this command**

$ git push origin

Push only one tag

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git push origin v1.0

Push all tags

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git push origin master --tags

Pattern search

vlab@VEDA2F118 MINGW32 ~/GitProject/project/rebase\_1 (master)

$ git tag --list "v1.\*"