- 😇 Meet Patel 😇

GIT-2

```
> mkdir project2
> cd project2

> git init

> git checkout -b main

> echo "first file" > file1.txt
> git add .
> git commit -m "file1"
> git log --oneline

> echo "second file" > file2.txt
> echo "third file" > file3.txt
> echo "first file -- new line" >> file1.txt

> git add file2.txt
> git add file3.txt
> git status

> git reset
  # restored staging area to working directory
  # Unstage files added into staging area -- file2.txt and file3.txt
> git status

> git reset --hard
  # for all the changes of file from repository to working directory
  # Checkout all the changes from last commit
  # New files in working directory are not modified or deleted
> git status
```

Branching

```
> git log

> git log --oneline
  # ae4d283 (HEAD -> main) file1

> vim demo1.py
  # def add(a,b):
  #     return a + b
  # n1 = 10
  # n2 = 4
  # n3 = add(n1, n2)
  # print("add",n3)
```

```
> python3 demo1.py

> git add .
> git commit -m "add"
> git log --oneline

# create new branch
> git branch branch1
> git log --oneline
  # curruntly main branch

> git checkout branch1
  # switch branch main to branch1

> cp demo1.py demo2.py
> vim demo2.py
  # add subtract code
> python3 demo2.py

> git add .
> git commit -m "subtract"

> vim demo2.py
  # add multiply code
> python3 demo2.py

> git add .
> git commit -m "multiply"

> git log --oneline

# go back to main branch
> git checkout main
  # branch1 file gone
> git log --oneline

> git checkout branch1
  # branch1 file back
> git log --oneline

> git checkout main
> git log --oneline

> ls
> cp demo1.py demo3.py
> vim demo3.py
  # add division code
> python3 demo3.py
> git add .
> git commit -m "divide"
> git log --oneline

# show graphical formate
```

```
> git log --all --oneline --decorate --graph
  #  * d46cf5e (HEAD -> main) divide
  # | * a12cbad (branch1) sub
  # | * 6af7857 sub
  # |/
  # * 8ce0d57 add
  # * 9398371 file1

# create new branch "branch2" and checkout it
> git checkout -b branch2
  # > git branch branch2
  # > git checkout branch2
> git log --oneline

# create new file demo4.py -- rect area - commit
> vim demo4.py
> git add .
> git commit -m "rect area"
  # commit
> git log --oneline

# demo4.py -- rect peri - commit
> vim demo4.py
> git add .
> git commit -m "rect peri"
# commit
> git log --oneline

> git log --all --oneline --decorate --graph
# * 094d2db (HEAD -> branch2) rect peri
# * d70586e rect area
# * d46cf5e (main) divide
# | * a12cbad (branch1) sub
# | * 6af7857 sub
# |/
# * 8ce0d57 add
# * 9398371 file1

# to show GUI graph
> sudo apt install git-gui
> gitk --all
```

Branching merge

```
# show list of all branches
> git branch
  # * represent current branch
  # branch1
  # * branch2
  # main
```

```
# goto main branch
> git checkout main

# branch1 is merge with main
> git merge branch1
> git log --all --oneline --decorate --graph
> gitk --all
```

imp Command

```
# list all branches
> git branch

# create "b1" branch
> git branch b1

# switch to "b1" branch
> git checkout b1

# create "b2" branch and switch
> git checkout -b b2

# switch to main branch
> git checkout main

# delete branch "b1"
> git branch -d b1

# merge "b2" branch to main
> git merge b2
```

- how to setup gitlab server on premise

GitHub

```
# Create a new repository -> name: project3 -> public
# Add .gitignore = python
# create it

> mkdir dev1
> cd dev1

> git clone https://github.com/meets-patel/project3.git
> cd project3
> ls
> ls -a
> vim .gitignore
    # observe file contents and exit
> cat README.md
```

```
# create new file in working dir
> vim demo1.py

# add file into staging area
> git add .

# commit file into local repo
> git commit -m "demo1"

> git push origin main
  # push file on remote repo
  # username: git-username (nilesh-g)
  # password: git classic token

> mkdir dev2
> cd ../../dev2
> git clone https://github.com/meets-patel/project3.git
> cd project3
> ls
> ls -a
> vim demo2.py
> git add .
> git commit -m "demo2"
> git push

> cd ../../dev1
> cd project3
> ls -a
> git pull origin main
> ls -a
```

```
> in github > goto project > settings > collabarators > add people >...
> add
```

```
# new
# create an empty project on github -- name = project2
> create new repo > project2 > public > create
> push existing

# go to "project2" directory
git checkout main

git remote -v

git remote add origin git@github.com:nilesh-g/project2.git

git remote -v
```

```
git push -u origin main
# upload main branch on github repo.

git branch

git push --all
# upload all branches on github repo.
```

- 😊 Game Over 😊