

GIT Hub Repository

iTerm2 -Replacement of terminal in Mac and works in macOSX 10.10 and later

```
//check git installed version  
git --version
```

```
//Check where is git  
which git
```

```
//Resolving Xcode Path error  
sudo xcode-select -switch /Applications/Xcode.app/  
Contents/Developer
```

```
//If brew not installed install home-brew // https://  
brew.sh/  
/usr/bin/ruby -e "$(curl -fsSL https://  
raw.githubusercontent.com/Homebrew/install/master/  
install)"
```

```
//MacOS install git with home-brew  
brew install git  
//Upgrade once installed  
brew upgrade git
```

```
//Install xcode command line tools which includes git  
xcode-select --install
```

```
//Other OS  
https://git-scm.com/
```

```
//Reference Public Repository  
https://github.com/raywenderlich/rwTODOs
```

//Account Creation and credentials

//Clonning rey's repository (using HTTPS)

cd desktop

git clone https://github.com/raywenderlich/rwTODOs.git

cd rwTODOs

ls //compare files in git url repo files

//To show most recent comments

git log

//Fork in URL to take and make your own copy of git  
code in your account

//Once done URL

//https://github.com/NareshBMV/rwTODOs.git

//Remove directory

rm -rf rwTODOs

//Showing the history

ls -la

//showing clone of the remote

git remote -v

///-----Creating Our Own  
Repository Locally

//Creating new repo

cd myTODOs //go into the directory to be in git

git init

//checking the history

ls -la //contains .git file which maintains all the repository  
content history

//Best practice to commit is to include readme and license

//<https://choosealicense.com/>

//gives untracked files  
git status

//Add file to git  
git add LICENSE

//Commit  
git commit

//Check out commit  
git log

//Get all the git configuration global  
git config --list

//Git configuration local level  
git config --local --list  
git config --global --list //for global

//Changing configuration  
git config --global user.name "NareshBMV"  
git config --global user.email  
"inareshbabumv@gmail.com"

///-----Creating Our Own  
Repository Remote

- Create Repository on account after login
- Give Repository Name

- //Add new remote to local repository
- git remote add origin https://github.com/NareeshBMV/SampleTODOs.git

//Local branch to track the remote master branch  
 git push --set-upstream origin master //will ask for user name and password  
 //Instead of password generate authentication token which will be stored in to keychain and no need to input again  
 //Steps of authentication token.  
 -Settings->Developer Settings -> Personal Access Token ->Generate Token  
 //Paste in place of password  
 //Successfully pushed the code to Git Repo

//Committing Code\*\*\*\*\*  
 git status //check status of uncommitted file  
 git diff //To check the changes made  
 git add .(dot) //adds all the changes in current directory  
 git add directory/\* //adds any files inside the directory folder  
 git log -p //Shows the changes in each of the commit

//When created the new directory git will not register the directory  
 //Git only tracks off the files  
 touch keep //creates empty.keep file in the directory  
 git commit -m "Enter commit message" //Typing commit message in command line rather than file

//Staging Area\*\*\*\*\*

git add . //adding to staging area  
git diff //difference in working directory and staging area  
git diff --staged //staging area and repo difference  
git reset HEAD~1 //reset one commit before  
git reset HEAD filename //reset staged content  
git add -i //interactive adding or staging  
Options  
S- status then type file code  
U-update  
R-Revert to upstage the content  
P-Patching each of the changes individually - will split the files into hunks (?-will expand the details of patching individually)  
git log -p //compares previous commits  
git add -p //shortcut for patch staging with split hunks

//Move files  
mv FileToBeMoved ToWhichItShouldBeMoved  
//won't keep track of movement of files and folders  
git add .

//Reset the changes  
git reset HEAD .  
//undo the deleted file in source of move  
git checkout -- .  
//remove moved file destination  
rm filepath

//git mv fileToMove ToDestinationLocation  
//git rm fileToRemove

//Ignoring Files\*\*\*\*\*  
secrets about any credentials  
which no need to be committed to Git

Metadata of build which no need to be tracked

//git ignore file and add paths to be ignored

//Global git ignore

touch allTODOs.html

git status

vim .gitignore

add line \*.html //ignores all html irrespective of their path and location of html files

//Vim editor deleting shortcut dd

Note - To ignore files in only subfolders or directories the add gitignore file and set trapped individual folder with parent directory

//to ignore except in root directory add line

\*/\*.html

//!/\*.html -- ignore previous rule

//Global git ignore

git config --global core.excludesfile

//if not exists the add where it should exists

git config --global core.excludesfile ~/.gitignore\_global

cat ~/.gitignore\_global //Details of file contents

//Git ignore files for different platforms

<https://github.com/github/gitignore>

//Mac OS global git ignore

<https://github.com/github/gitignore/blob/master/Global/macOS.gitignore>

//Its all because of git not to track any of the system files

//Git History\*\*\*\*\*

git log -4 //Specifying how many commits wanted to see

```
git log --oneline
git log --decorate //branch names
git log --graph //ASCII representation
git log --online --graph --decorate --all //all is for all the
branches
git log -p //summary followed by diff representation
git shortlog //group of change logs with author
git shortlog -4
git shortlog origin/master..HEAD //between origin master
and now
```

//Filtering commits

```
git log --author="authorName"
git log --grep="file" //to find commit message where term
grep is used
git log -S"Fortran" //Find all the commits that include
commits fortran "S- search term"
git log -S"Fortran" -p //diffs of that commit
git log -- books/ //To changes that exists in books
directory
git log -p --all -S"Windows 10" //"all" is for all the branch
git log -p -S"Windows 10" //It is not able to fetch bcoz it
is in the same branch
```

//

Branching\*\*\*\*\*  
\*

//Creating branch, Checkout, Deleting branch

```
git branch myBranch // Create a new branch
cat .git/refs/heads/MyBranch //Contains hash code for
my branch
git log --oneline -1 //Checking the same hash
git branch //list of all branch
git checkout myBranch //Switch to branch
git branch //demonstrate in which branch we are
```

```
git branch -d myBranch //Delete particular branch
ls .git/refs/heads //There will not be myBranch present
git branch --all //List out all the branch including remote
branch
git checkout --track origin/clickbait //To create new
branch to take origin and checkout that branch
git branch --all -v //Different branch list with details
git status //Check with home branch an origin uptodate
git log --oneline --graph //shows what need to be
merged
//Additional
git checkout -b myBranch //Creates and checkout
myBranch
```

## //Merging

\*\*\*\*\*

```
//Merge Conflicts in mastering git
git merge clickbait //specify the branch that need to be
merged in current branch
//Merge by recursive strategy
//Fast forward merge--Taking master directly into sub
branch
git merge --no-ff readme(branch which no need fast
forward merger commit : recursive strategy)
```

## //Syncing with Remote

Repository\*\*\*\*\*

```
git remote -v //List out all the remote associated with
code repo
git branch -vv //remote branches tracking
git branch -vv --all //all the branches
git log --oneline //List of all the commits
git push origin master //Push to master
```

```
//Adding remote for my repo
```



```
git remote add iwantmyrealname https://github.com/
iwantmyrealname/rwTODOs
git remote -v //shows new remote added
git log --oneline --all --decorate --graph // show no
changes
git fetch iwantmyrealname //To fetch all the information
related to commits
git remote show origin //Gives details about particular
remote branch
git remote show iwantmyrealname
git checkout clickbait //if not created follow below
git checkout --track origin/clickbait
//Merge iwantmyrealname branch into clickbait branch
git merge iwantmyrealname/clickbait
//Checkout to master
git checkout master
//merge clickbait branch into master branch
git merge clickbait
```

//Pull

Request\*\*\*\*\*  
\*\*\*\*\*

//Discussion changes, Adding continuous integration,  
Testing, Change Review

```
git checkout master
git checkout -b status_update
vim tutorials/ios_article_ideas.md
git add tutorials/ios_article_ideas.md
Commit -m "Adding checkmark in iOS_article_ideas"
vim books/books_ideas.md
git add books/books_ideas.md
git commit -m "Adding checkmark in books_ideas"
git push --set-upstream origin status_update
git branch --all -vv //Check all the branch
//In Git login follow steps to add pull request to merge
```

origin/status update into master  
====>Pull request tab====>New pull request====>Select  
Base and branch to be merged to Base(here base is  
Master and branch to merge is status update)====>Click  
pull request button if things reviewed are fine====>Add  
required details(Assign reviewers and label  
them)====>Created pull request ==>Confirm merge and  
delete if not needed for branch

git fetch //Update local git  
git log --oneline --all --graph --decorate  
git pull //Update local master branch with origin master  
git log --oneline --all --graph --decorate  
git branch -d status\_update  
git push origin --delete status\_update  
git remote prune origin //deleting the remote branch  
where local already deleted

//when error relating to fast forward  
git fetch  
git rebase branchWantedToRebase  
//We can send pull request for different fork head