****

GIT_HUB

****

Git is source code management tool
git is a software & github is a remote repository

- To install git pkg
- yum install -y git

create one directory with any name at any location

- mkdir /dir1
- cd /dir1
- git init

when we hit git init cmd then our local folder becomes git folder
and this git folder is divided into 3 parts

1) workspace/working dir/working area
2) staging area
3) local repository

To see files in the workspace & staging area

- git status

- If filename is in red means this file is in working area
- If filename is in green means this file is in staging area

touch file file2 file3 file4 file5
So all files will get created in working area

To send/add file1 to staging area

- git add filename
- git add file1

To see files in the workspace & staging area

- git status

To remove file from staging area

- git rm --cached filename
- git rm -- cached file1

To remove all files from the staging area

- git rm --cached *

To send/add all files to staging area

- git add *

To send file1 to the local repository

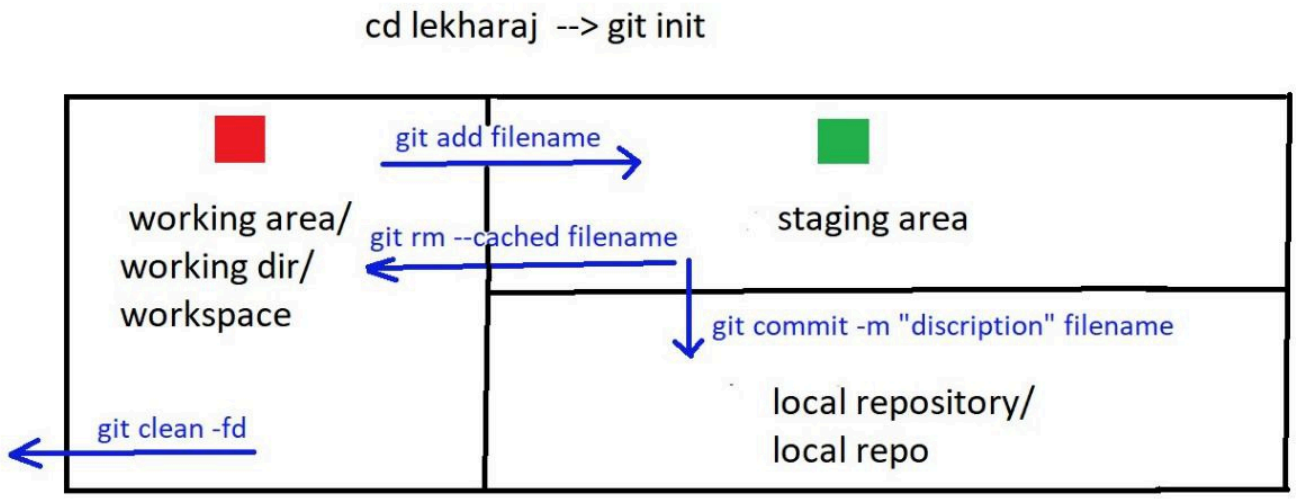
- git commit -m "description" file1

To send all files to local repository

- git commit -m "description"

To delete all files from workspace

- (only new files)
- git clean -fd



To see information or logs related to files

- git log
- git log -1
- git log -2
- git log --oneline
- git log --oneline | head -2
- git log --oneline -2
- git log filename

To change description of latest commit

- git commit --amend -m "new description"

To create username & userid

- git config --global user.name "lekharaj"
- git config --global user.id "lekharaj567@gmail.com"

To edit username & userid, use the same command

- git config --global user.name "sumit"
- git config --global user.id "sumit555@gmail.com"
- OR
- git config --global --edit

Login to GitHub account and create a remote repository

To connect local repo to remote repo

- git remote add origin url_of_remote_repo

To send/push all data from local repo to remote repo

- git push origin master
- (enter username and token)

Create a second instance

- Connect to this instance
- Download the Git package by using command yum install -y git
- To download a remote repository to our local machine:
- git clone url_of_remote_repo

To pull changes from the remote repository to our local machine:

- git pull url_of_remote_repo

Difference between Git Clone & Git Pull

1. Git clone copies the entire remote repository.
2. Git pull updates from the remote repository.
3. Git clone is used once when starting with a project.
4. Git pull is used regularly to stay updated with changes.
5. Git clone creates a new local repository.
6. Git pull updates an existing local repository.

Difference between Git Push & Git Clone

1. Git push sends existing local commits to the remote repository.
2. Git clone creates a new local copy of a remote repository.
3. Git push needs an initialized and committed local repository.
4. Git clone starts from scratch by copying a remote repo.
5. Git push is used after making changes.
6. Git clone is used before starting any changes

Difference between Git Push & Git Pull

1. Git push uploads changes from local to remote repository.
2. Git pull downloads changes from remote to local repository.
3. Git push shares your work with others.
4. Git pull gets others' changes into your local branch.
5. Git push requires committed local changes.
6. Git pull requires a remote repository with new changes.

Difference between Local Repository & Remote Repository

1. Local repository is stored on our local machine.
2. Remote repository is stored in GitHub or GitLab.
3. Local repository is used for development and testing.
4. Remote repository is used for sharing and collaboration.
5. Changes are committed locally in the local repository.
6. Changes must be pushed to appear in the remote repository

Difference git clone & git fork

1. git clone Creates a local copy of an existing Git repository (public or private, if you have access).
2. git fork Creates a copy of a repository under your own GitHub account (or other Git hosting service).
3. Git clone run on our local machine via command line.
4. Git fork run on the GitHub

- We can add folders to both the local and remote repositories, but Git does not support empty folders.
- .gitignore is used to temporarily hide or exclude files from being tracked by Git.
- It can also be used to upload empty folders (by including a placeholder file) or to remove specific files from the workspace by listing them in the .gitignore file

The git reset command in Git is used to undo changes in our repository

- **git reset command**

. Types of reset

1) Mixed reset
2) Soft reset
3) Hard reset

(Try this practical with at least 2 files in the local repository.)

1) Mixed reset
   (To move files from the local repository to the workspace)
   git reset HEAD~1

2) Soft reset
   (To move files from the local repository to staging)
   git reset HEAD~1 --soft

3) Hard reset
   (To delete files from the local repository)
   git reset HEAD~1 --hard

==========

git stash command

The git stash command is used when we are working on something, but suddenly need to switch to a different task without committing our current changes.

It temporarily hides (or stashes) our uncommitted changes, so we can work on something else.

Later, we can bring those changes back and continue where we left off.

git stash works only for modified files.

git stash list
Shows the list of all stashed changes.

git stash
Saves our uncommitted changes and gives us a clean working directory.

git stash apply <number>
Restores the most recent stash.

git stash apply stash@{1} Restores a specific stash by its number.

git stash drop stash@{1}
Deletes a specific stash.

git stash clear
Deletes all stashes.

==========

Tags

Tags in Git are like labels that we attach to specific commits

Types of tags

1) Lightweight tag

2) Heavyweight tag
(allows adding an additional message to the tag)

To see tags:
git tag

To create a lightweight tag for a commit:
git tag tagname commitid

To delete a tag:
git tag -d tagname

To see detailed information about a tag:
git show tagname

To create a heavyweight tag for a commit:
git tag tagname commitid -m "description"

To connect the remote repository 1 with the local machine:

git remote add origin URL_of_remote_repository_1

To push data from the local repository to remote repository 1:

git push origin master

To connect the remote repository 2 with the local machine:

git remote add origin2 URL_of_remote_repository_2

To push data from the local repository to remote repository 2:

git push origin2 master

Before pushing data from the local repository to the remote repository,
we should first pull the latest changes from the remote repository to avoid conflicts,
and then push our local changes.

Branches in Git

To see all branches:
git branch

To create a branch:
git branch branchname

To delete a branch:
git branch -d branchname

To switch to another branch:
git checkout branchname

To create and switch to a new branch:
git checkout -b branchname

To push changes from a particular branch:
git push origin branchname

To push changes from all branches:
git push --all

Branches in git

Branches are independent lines of code of development.

The default branch in Git is the master branch.

We can create multiple branches in Git.

Branching Strategies:

1) A new branch is created for every new release.

2) The master branch always contains the latest code.

To create a new branch, we start from the master branch.

If we create a new branch from the master branch, the new branch will contain the files present in the master at that time.

However, if we later add new files to the master branch, those files will not automatically appear in the new branch.

How to merge two branches:

git merge branch-name

git merge master

git merge branchA

Git

Knowledge of using Git to track and manage changes in code.

Understanding of creating and managing local and remote repositories for pushing and pulling changes.

Familiarity with Git commands such as git reset, git tag, and git stash.

Understanding of creating and managing branches, performing merges, and resolving conflicts.

Roles & responsibilities in Git

Used Git to track and manage changes in code

Created & managed local & remote repositories for pushing & pulling changes.

Used different commands in git like git reset, tag & git stash

Experienced in creating and managing branches, performing merges, and resolving conflicts