**GIT**

1. **GIT**

* It is source code management tool which is used by everyone to manage their source codes
* It is a distributed version control system
* Commonly used for tracking changes on source code during s/w development.
* Allows multiple developers to collaborate on a project simultaneously.

1. **Distributed version control system**

* Every user can work on the project independently and make commits to their local repo without

contact with central server.

1. **What is Repo**

* Collection of files & directories that make up a project.

1. **What is commit**

* Set of changes made to the files in repo.
* Each commit has unique identifier & information such as author, timestamp, etc.

1. **To create a local repo in GIT**

* CMD : **git init**
* This is a independent local repo.
* It serves as local working copy and later connect it to remote repo for any collaboration or backup purpose.

1. **How to configure GIT repo locally**

CMD : **git config --global user.name “user\_name”**

**git config --global user.email “user\_email”**

1. **How to add files from work area to index/staging/cache area**

CMD: **git add <filename>**

1. **Working Directory**
   * Where we modify our files.
   * Contains the current state of our project, including the recent changes we made.
2. **Staging Area**

* Where we assemble the changes that we want to include in next commit.
* It acts as **holding area,** where we can view and modify before the commit.

1. **Use of git log**
   * Displays the commit history of a Git repo such as author details, date, content or history.

**To display the commit history**: git log

**10. What is the git clone?**

Git clone to local - To download an existing repository from Centralized (Github) to local system.

Git clone to Jenkins – cloning to Jenkins workspace

**Cmd:**  git clone <url>

1. **How to edit an incorrect commit message in Git? Or how can you fix a broken commit?**



* git commit --amend -m “New commit message”

It reverts back to previous commit.

1. **How to get back a commit to staging area?**



* git reset –soft <previous\_commit id>

1. How to get back a file from staging area to working area?



* git reset head <file\_name>



1. **How to get back a commit to work area?**

* git reset --mixed <previous commit id>

1. What is git reset?



* Reset the current HEAD state to specific state.
* Undo changes from local repo by moving the current branch pointer to a different commit.

1. What is ‘head’ in git and how many heads can be created in a repository?

* A ‘head’ is simply a reference to a commit object.
* In every repository, there is a default head referred as “Master”.
* A repository can contain any number of heads.

1. What is .gitignore file?

* Files which contains some information which are required related to project or something
* If we keep the file name as .gitignore, it skips that files while adding and committing.

1. **How to see the difference between 2 commits?**

* git diff <commit\_id1>..<commit\_id2>

1. **What is fork in git repo (**downloading the git repo to local machine)

* Process of creating a personal copy of a repo owned by someone else.
* On repo page, click **“Fork”** on top right corner.
* We can use **“git clone”** to clone the repo to local machine by copying the git URL.
  + **Git clone [forked repo URL]**
* Then, if we have to contribute the changes back to original repo, we can create a **pull** request from forked repo to original repo.

1. How to create a branch?

# git branch <branch\_name>

1. **How to checkout to branch?**

# git checkout <branch\_name>

1. **How to create branch while checkout?**

# git checkout -b <branch\_name>

1. **How do you rename the local branch?**



# git branch -m <old\_branch\_name> <new\_branch\_name>

1. **How to see the branch list?**

# git branch

1. **How to see the remote branch list?**

# git branch -r

Or

# git remote show origin

1. **How to see the local and remote branch list?**



# git branch -a

1. **How to delete a branch?**

# git branch -d <branch\_name>

Or

# git branch -D <branch\_name>

1. **How to delete a Remote Branch?**

# git push origin -d <branch\_name>

1. **How to see the difference between 2 branches**

# git diff <branch1>..<branch2 >

1. **What is git push?**

git push is to push commits from your local repository to a remote repository.

1. **How do you push the files to master branch in remote repo?**

#git push origin master(you must be in master branch)



1. **How do you push files from local to particular branch in remote repo?**

#git push origin <branch\_name>

(or)

#git push --set-upstream <branch\_name>

1. **How to push new branch and its data to remote repository?**



#git push <github\_repository\_path> <branch\_name>

(or)

#git push --set-upstream <branch\_name>

1. **What is git pull?**

Git pull downloads and merges a ‘branch data’ from remote repository to local repository.

Use ‘git stash’ command to Hide your local changes before git pull.

* # git pull (git fetch + git merge.)

1. How do you pull a file from particular remote branch?



# git pull origin <branch\_name>



1. **How do you download a remote branch to local without merge?**

# git fetch origin <branch\_name>

# git checkout <downloaded\_branchname>

1. **Difference between git fetch nd git pull?**

|  |  |
| --- | --- |
| **Git Fetch** | **Git Pull** |
| Fetch new changes from remote repo without merging into current branch | Pulls the changes from remote repo & merges them into current branch |
| It reviews the commits and changes can be done | Update the changes to local repo immediately. |
| CMD : **git fetch<remote repo>** | CMD: **git pull<remote><branch>** |

1. **What is cherry pick in git?**

* Used to select & apply specific commits from one branch to another.

CMD: **git cherry-pick <commit\_id>**

1. **What is difference between git clone & git pull?**

**GIT CLONE**

* + - If you want to download whole existing repository than use Git Clone.

**GIT PULL**

* Used to fetch the latest changes from a remote repository and merge them into your current branch.
* It combines two actions: git fetch, which downloads the latest changes from the remote repository, and git merge, which integrates those changes into your branch.

1. **What is git merge?**

Git merge is used to combine two branches.

**# git merge <branch\_name>**

Note: you should be in target branch. Then run the command

1. **What is git conflict? What is the scenario you will get git conflict error?**

* For example, if you and another person both edited the same file on the same lines in different branches of the same Git repository, you'll get a merge conflict error when you try to merge these branches.
* You must resolve this merge conflict with a new commit before you can merge these branches.

1. **How do you skip from merge conflict?**

#git merge --abort

1. **What is the function of ‘git rm’?**

To remove the file from the work area/staging area and also from your disk ‘git rm’ is used. You can revert a deleted file.

1. **How will you know in GIT if a branch has been already merged into master?** git branch –all  
    git branch --merged master  
    git branch --no-merged master
2. **What is branching? What is the purpose of branching in GIT?**

* Developer can work on different versions of collection of files
* A branch allows the user to switch between these versions so that he can work on different changes independently from each other.
* Real time example –
  + Master
  + Development – developer have their code here and test on development(dev) server.
  + Production – code to be deployed on production server(live/prod/target serrver) after development server.
  + V2.1
  + Etc

**NOTE: Being DevOps we are used to deploy the codes which given by the developers**

1. **Branching strategy…**
2. **What is Git stash?**
   * It allows to temporarily save changes that don’t want to commit.

**When we use:**

* To switch to a different branch that requires clean working directory, but don’t want to commit the changes.

**How it works:**

* Git stash takes all the changes made in our working directory that are not staged for commit and saves in new stash.
* This includes both modified and untracked files.

**Git stash commands:**

1. **Git stash save “message” 🡪** Creates a new stash.
2. **Git stash list 🡪** Lists all the created stashes.
3. **Git stash apply [stash] 🡪** Applies the changes from specific stash to working directory and keep the stash intact.
4. **Git stash pop [stash] 🡪** Applies the changes from specific stash to working directory and removes the stash from stash list.
5. **Git stash clear 🡪** Removes all stashes from stash list
6. **What is another option for merging in git?**

* git rebase

1. What is difference between git merge and git rebase?

|  |  |
| --- | --- |
| **GIT MERGE** | **GIT REBASE** |
| Applies all unique commits from branch A into branch B in one commit | Gets all unique commits from both the branches & applies them one by one |
| It doesn’t rewrite commit history, just adds one new commit | It rewrites commit history, but doesn’t create extra commit for merging |

1. **How do you undo the last commit?**

# git revert <commit\_id>

1. **How to delete Repository in GitHub?**

• On GitHub, navigate to the main page of the repository.

• Under your repository name, click **Settings**.

• Scroll to the bottom of the page and you will find **Delete this repository** button

• When you click on that button, another pop up will appear, here you need to type in the name of your repository name and click on the button bellow which says: I understand the consequences, delete the repository.

1. **How to give an access to a specific person to repository?**

It’s a feature where we can grant access to a developer for our personal repo.

* Under your repository name, click **Settings.**
* In the left sidebar, click **Collaborators**.
* Under "Collaborators", start typing the collaborator's username.
* Select the collaborator's username from the drop-down menu.
* Click **Add collaborator**.

The user will receive an email inviting them to the repository. Once they accept your invitation, they will have collaborator access to your repository.