Source code Management tool (SCM)

* There are many SCM tools in market

1. Git (most popular as of today )
2. Git hub
3. Gitlab
4. Bit bucket.
5. Aws codecommit
6. They provide git as a service.
7. SVN
8. CVS
9. Microsoft TFs.
10. Rational ClearCase(IBM)

NOTE: SCM tool is also called as version control system.

Why SCM Tools

* It is hard to maintain project related files without Scm tool.
* SCM tool helps to developers to collaborate and work on same project. (changes done by diff developers
* We can setup granular permissions on the source code
* We can configure a backup. (we can setup a standby copy of your repository)

Types of version control systems: (imp for interviews)

* **Distributed version control system:**

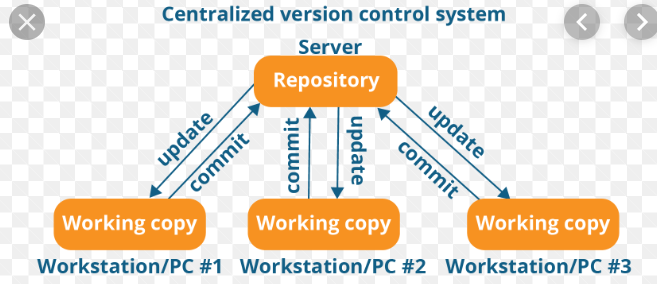
The SCM repository is distributed across multiple servers. then we call it distributed version control system.



* Git follows distributed control approach.

**Central version control system:**

In case of central approach, only remote server has SCM capabilities and local machines will not have SCM capabilities.



**WHY GIT?**

* Git is distributed
* Fast
* Open source
* Lots of modern features which speeds up our development activities.
* Light weight (consumes less resources – less disk, less cpu , than other version control systems) in Git new branch is a pointer pointing to snapshot

**Setup Git remote server**

* **Hosted Git**

a. you procure your server

b. Install configure and manage git software

**- Cloud:**

Third party will maintain all

for ex :

* Github
* Gitlab
* Bitbucket
* Codecommit
* Etc.

**Create an account on Git HUB:**

* Is going to be our remote repository
* It is free to use
* Githib.com

**Create new repository in Github**

**Storing our files in to Git:**

We need to have Git client on our local desktop.

We see various options like CLI & GUI

In interviews the questions on Git commands is expected so lets use CLI

Install Git bash on our desktop

**Configuring Git bash:**

We must configure our name & mail id which is used by Git in every commit we make.

Git config --global user.name “lokesh beere”

Git config --global user.email “lokesh[beere1995@gmail.com](mailto:beere1995@gmail.com)”

Take project from git repository in to local.

We have to clone our remote repository to local.

Git clone https://github.com/lokeshbeere/dec-930-2019.git

* Git clone supports 2 protocols

1. Https
2. SSH

Cloning using SSH:

* For SSH we need public & private keys.

Open Git Bash and type following cmd>

* ssh-keygen
* go to git hub homepage in the browser, right top corner expands the link click on settings,
* ssh and gpg keys- new ssh key – give title
* adding a new file and pushing that file to remote repository.

In git anything you want to push must go through local repository. You can’t directly bypass local repository.

**Working area:**

Any changes we do in our local repository those changes are automatically placed in working area by GIT.

**Staging area:**

After your work is done, the files you want to commit must be moved to staging area.

Git commit – it moves **files from staging area** to local repository (only files in staging is committed)

Git add info.txt i.e. file name – moves file to staging area. ( space separated file name , or \*

Instead of 1 file , we can specify space separated multiple files, we can use \* or .if we want to stage all the files.

We also can use regular expressions. Like git add \*.java

Ex: git add .

Integrating our local changes with remote repository.

Git push origin master

In this cmd origin returns to url in the background

Git automatically creates origin entry at the time of clone

Master is a branch

Pull & Fetch –(IQ)

* git fetch: it fetches all new commits in the remote to local without merging.

Where commits will reside

* git pull: it will fetch + it will merge

Git conflict: conflict is a scenario where multiple developers working on same file and at same line that causes conflict.

**How do you resolve conflicts in Git?**

We resolve conflicts using a tool – Diff merge.

Diff merge is 3rd party tool and we can integrate this with GIT.

Using this tool, we can resolve conflicts and you can see a diff b/w 2 versions

Installing Diff merge:

* integrating diff merge with Git.
* After conflict use Git merge tool
* > git mergetool
* Adjust according to your code – prepend – before that , append – after that

Then save and commit

**Undoing:**

Git restore info.txt - undo changes

Git restore --staged info.txt -- To unstage

Git log - -oneline ( to show in one line)

How to undo a local commit:

1. We can Remove that commit permanently or
2. Undo:

* Git reset d670234 (it resets to this commit id, anything top of it to be deleted)
* Git reset d670235 (it removes commits) (ex: git reset c3 🡪 all commits above c3 is deleted

1. Using reset we can remove only recent n commits, we can’t delete only specific commit, it deletes from that top of commit id. (
2. use reset for only local commits, (don’t use reset if commit is already pushed to remote)

RESET Modes:

**Soft reset:**

Git reset - -soft d67944

* Removes commits and changes made part of those commits is kept in staging area

**Hard reset**: be careful using this

* Removes commits and changes part of those commits are permanently deleted.
* Git reset hard - - d579696

**Mixed reset:**

* Deletes commits and changes part of those commits are kept in working area.

I have 5 files and I did commit for 4 files and 1 file I am doing again, so I don’t have 2 commits at that case I use soft reset then 4 files come to staging area and you can commit all at an once to be 1 commit

Git revert:

It is one more cmd to undo commits

Git revert d57686

Git revert will not remove a commit instead it removes changes in the commit, we can undo orbitary commits

**Git branch:**

[[ I created new repo , we can do it in local( create folder and git init that folder becomes local repo) or central ,,,,, default branch is master , I added info,txt , then I staged and committed it , by default all commits u made is kept under master ,

For same repo I want to add one.txt , the moment u cmmit the file is kept in repos , there will be a link b/w c2 & c1 and master will be moved to c2 ,, If any new comit done again , master is moved to c3 like wise., c1 🡨c2 🡨c3

I create a branch , switch to that branch start commiting the changes to that branch ( any new work can be done in separate branch )]

* Branches provide isolation to our work.
* Git comes with default branch i.e master ( master is called as main branch we can think of it as super branch)
* Any new task must be implemented in a separate branch. (that is a common practice)
* In real life no one will directly work on master and only limited people will have write access to master.
* In git branch is light weight it consumes less storage.

Creating a new branch:

* Git branch my branch ( git creates a branch from current branch) git branch branch name , now master and mybranch are pointing to same.

Switching the branch:

* Git checkout mybranch

To list my branches:

Git branch

In real time we cant directly push commits to remote master

Git branch -a – to display all branches I.e local + remote

Git branch -r -- to display remote branches

Git branch feature-1 --- from current branch( i.e from master) It crates another branch

Git checkout feature-1 ( swithching a branch – now switching from master to feature-1)

I want to add new txt file: add that file and commit it then it goes to feature-1

If u see in master now it is not visible , we need to integrate to master

* We can merge changes to locally but it is not standard, the standard way of merging is push local branch to remote
* Create pull request
* Get it reviewd and approved
* Merge pull request

Git push origin feature-1 , earlier we are using git push origin master ncoz we have only master

Pull request – it highlights the reviewr what changes you have done

Click on compare and pull request – now our changesa are in feature-1 –

Give title ---- and write comment if u want

If some one is there to review then assign review

Click on Create pull request

We can have conversation over pull rqst ,comits cjecks ,no.of files changed

Approver should approve ,after approver approves , we get an option called

Merge pull request , when you can merge , all changes are intergrated to master

* After closing pull request, we must delete our branch.

Git branch -d feature-1, we need to go to master (git checkout master)and then u can delete feature , bcoz u r in current branch.

Git merge feature-1 ( or ) git pull origin master ,,,, since u have merged in remote once fetch that is merged in locally

Git branch -d feature-1 - deleted locally.

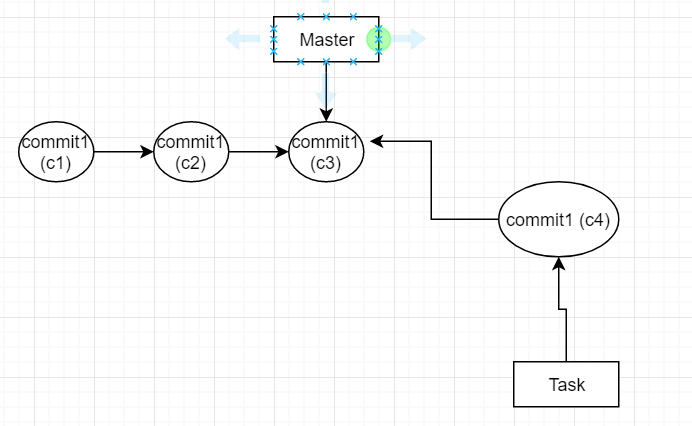
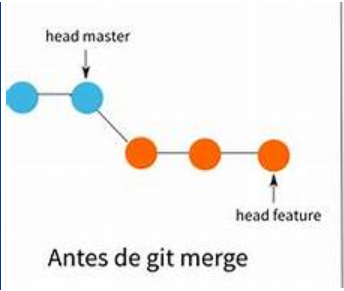
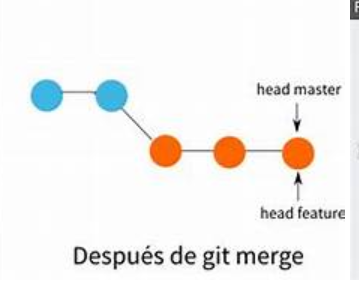
Git push -d origin feature-1 -push that to remote

* Create branch , checkout branch , add,commit,push origin feature-2,pull request.
* After merging changes to master , you can delete feature-2 once merged

$ git push origin -d Lokesh --- if Lokesh is feature

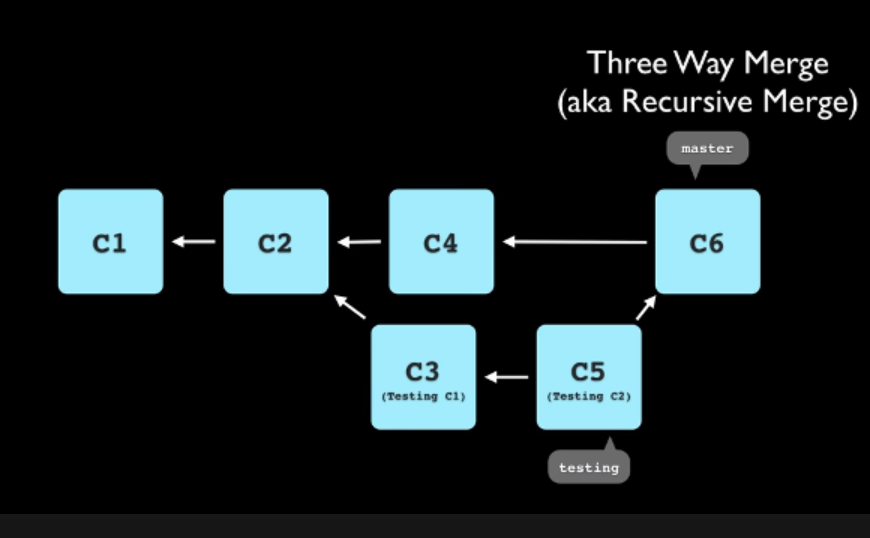
If someone is already committed to remote it won’t allow us to merge, but we can still create pull request at that time we pull from master to local feature(then all data is in local feature ) and then push that feature to remote master , merge ((( we need to resolve conflicts)

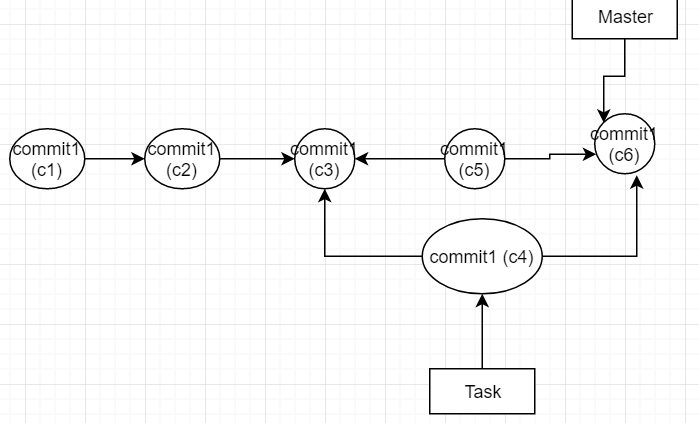
**Git Merging strategies**

* Fast forward: In above diag feature -1 is created from master when you were at commit C2 after feature -1 is created there are no new commits in the master, this makes git to choose Fast forward merge.
* 
* After fast forward merge commit history looks as follows.
* 

Recursive or 3 way merge:

* Feature-1 created from master at c2,
* After feature-1 created master got 2 new commits, bcoz of this 2 additional commits we cant use fast forward so , it follows recursive merge.

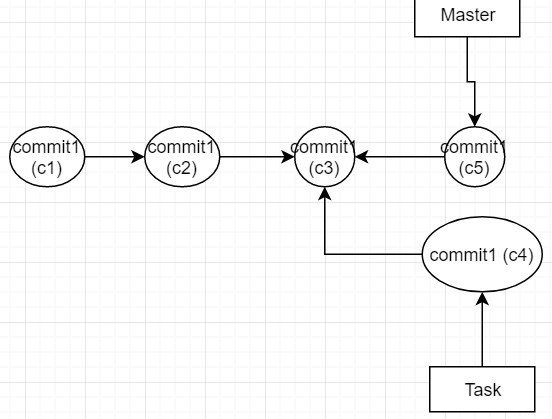




It creates new commit called as commit merge

IQ) what is Git rebase: (VVVimpa)

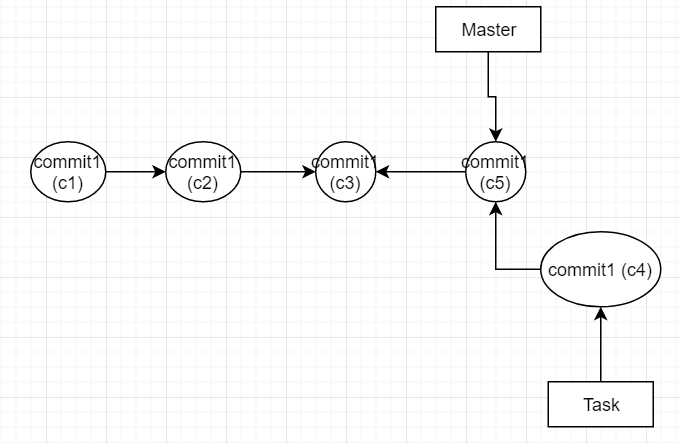
Rebase:



Feature-1 is based out of c2 , if we rebase feature-1 to master the new base commit becomes latest commit of master a2

At the time of rebasing it will merge.

After rebase:



Note: rebase is used to keep our history linear ( it’s not recommended to use) and we use rebase only in the local.

**What is cherry pick:**

Cherry pick pics only specific commit from the other branch and integrates with current branch

it is related to merging

git branch demo

gitt add

git commit

git add

git commit

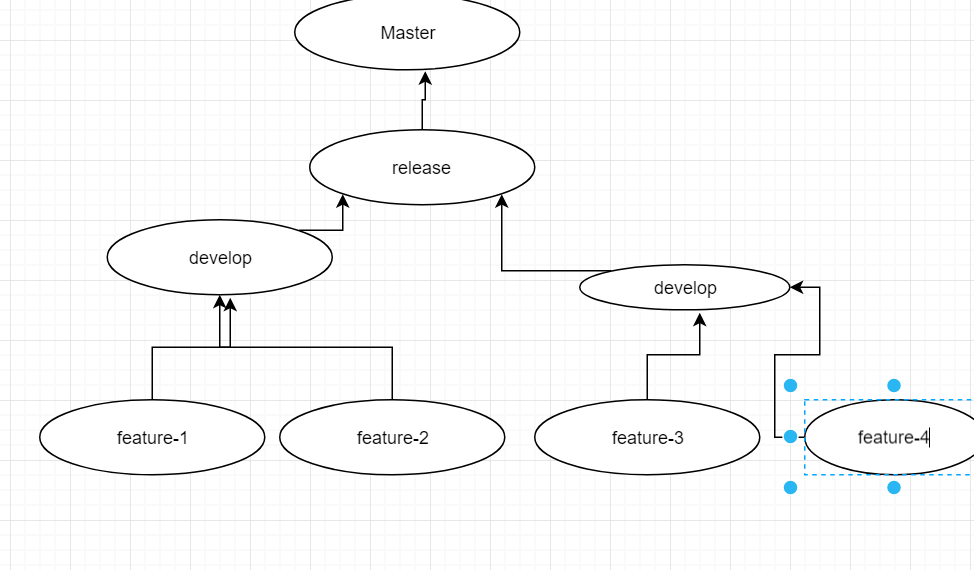
git check out master

git merge demo

git cherry-pick commit id

* Picks

IQ) which branching strategy are you using in your project?



**IQ)** what is bare repository in git:

It’s a repository which does not contain working area and staging area (i.e we can’t do changes to bare repository)

Bare repository (IQ)

It is also called as Read only repository

Which don’t have working or staging area

Git clone –bare repository paste git hub url

**What is git blame:**

**Git blame info.txt**

**Cat info.txt**

It tells us every commit who made.

What is Git init? (IQ)

It converts local folder in to git repository:

We are starting new project , I setup my template or framework in local and do git init

There is no remote, I need to create remote repo and sync it , create new repo in git hub -create repo

Git add

Git commit -m “ffd”

We cant push now , coz

Git remote add origin github url of new repo that u created -----------------

Git push -u origin master -🡪 u stands for upstream , default it takes origin master.

Q) what is Git stash ?

Q) How to check which branches are merged locally

