GIT ---- https://www.youtube.com/watch?v=xqwoTHL6CJ0&list=PLH1ul2iNXl7vkfIFF2BxLA5xpkbvWtFWf&index=1

------------------------------------------------------------GIT ---------------------------------------------------------------------

**Git secrets?**

Git-secret is a bash tool that is used to store your private files and information inside of a git repo on your server. The git-secret tool allows us to encrypt sensitive files that are stored inside a secure, encrypted repository.

**How to push only particular commit to remote repo?**

git push origin 2e5e6f5:master

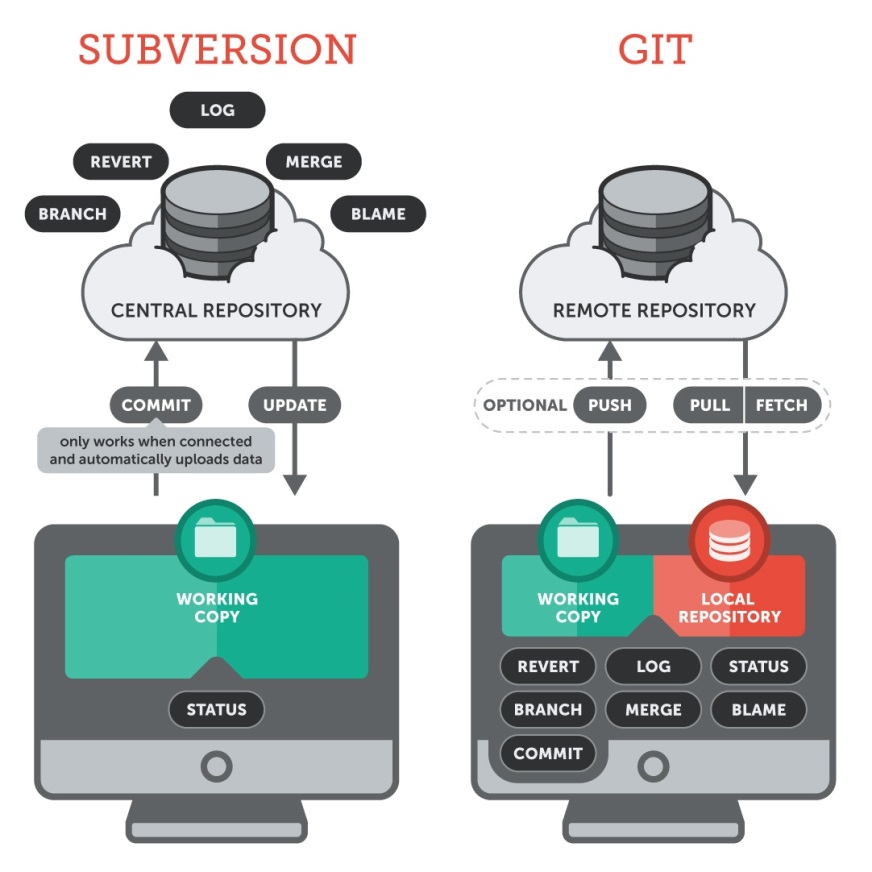
**Command to update the local repo?**

Git pull

**Why we need git? What makes git unique from other tools like SVN?**

* Git is a distributed version-control system for tracking changes in the source code.
* Git tracks the changes you made to the files ,so you have a record of what has been done, and you can revert to specific versions.

**Diff b/w GIT and SVN**



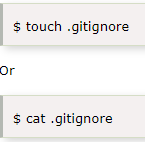
* GIT is a Distributed version control system ,where as SVN is centralized version control system
* GIT creates local repo to store everything locally ,where as SVN stores everything on centralized server.
* Network access is not mandatory for GIT operations ,Network access is required for SVN operations
* Speed of fetching/accessing application code from GIT is much better compared to SVN bcz developers will fetch the code ones from remote repo and have copy on local repo , whereas in SVN speed of access may affect the developers operations bcz all the developers have to fetch the code from central repo.
* If the central repo in SVN crashes for some reason there won’t be any backup for recovery ,whereas in GIT if the remote repo crashes ,every developer will have their own working copy.

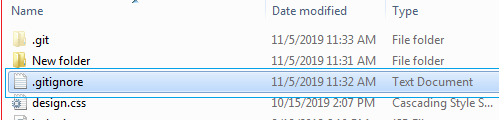
**Let's say i have maven repo cloned on to my local repo, did some changes and i have build the code and now we’ve target folder. So now when i do git operations like git add, git commit or any other git operations target folder should not be considered, how would you achieve the same?**

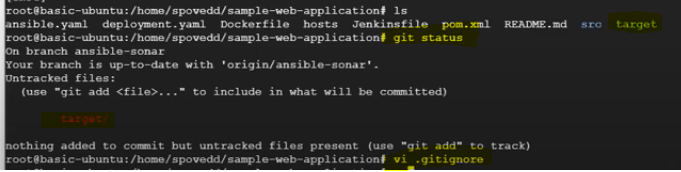
.gitignore

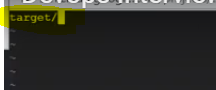
Steps :: create a .gitignore file and mention the target folder on to it as shown below.

Ignored files are such files that are explicitly ignored by git. We have to tell git to ignore such files.





1)vi .gitignore 



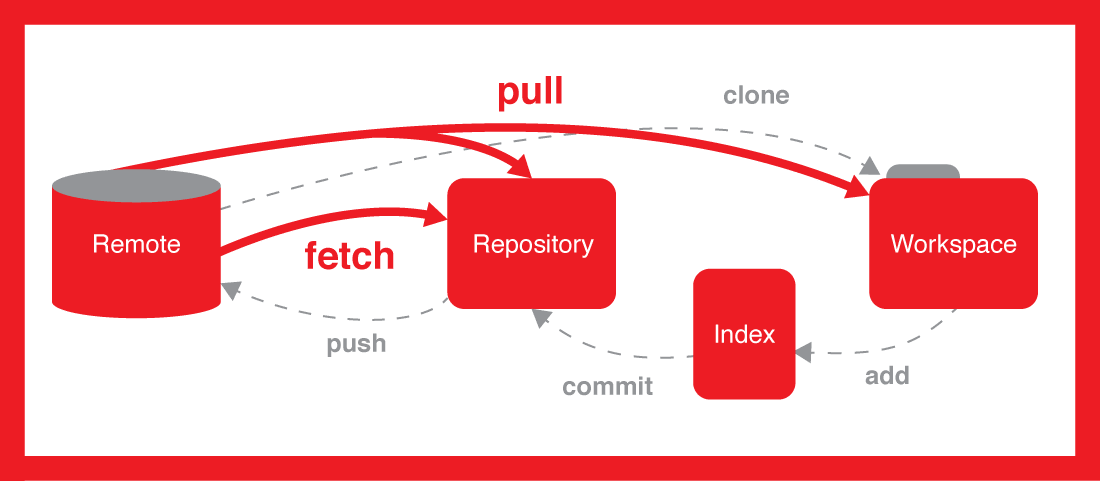
**What are the differences between git pull and git fetch?**

Git pull measuredly does two functions ,it fetches the code from remote repo to local repo and it merges that onto working copy.

Git pull= git fetch + git merge

[enter image description here](https://i.stack.imgur.com/zUInQ.png)

**Git fetch** :: Git fetch do only half of the work of git pull. It fetches the code from remote to the local repo. But it won’t merges the code to working copy.

[](https://i.stack.imgur.com/nWYnQ.png)

**How to clone specific branch in git?**

**Git clone –b branch1 --single-branch repo ulr**

Where

branch1—specific branchname



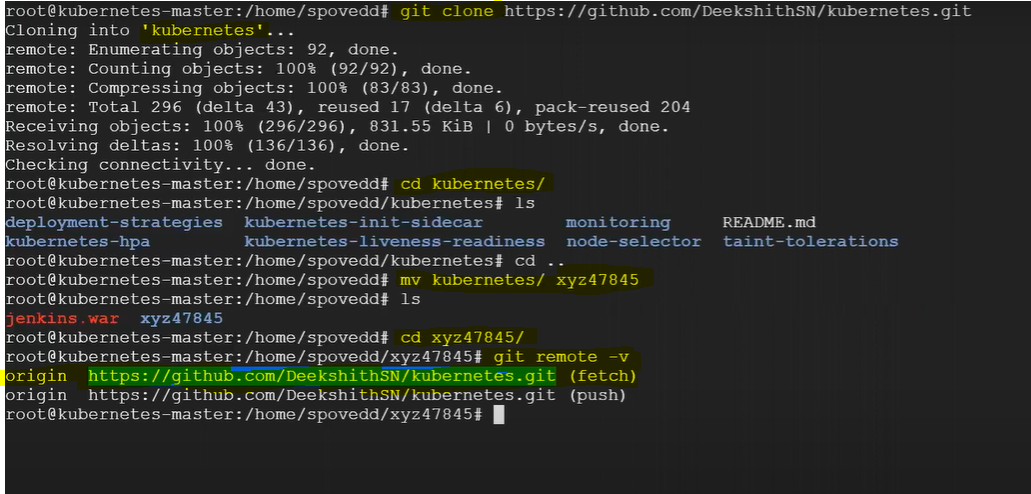
**What is Branch in GIT?**

A branch in GIT is an independent line of development taken from the same source code

**Let’s say your organization has github and bitbucket store code, you have cloned a repo on to your local and changed the directory name. After some days one of your team member asks you to share the cloned link, how would you share the same?**

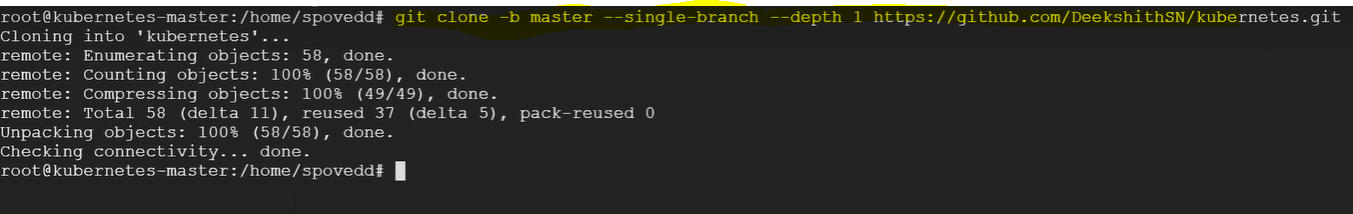
Git remote –v

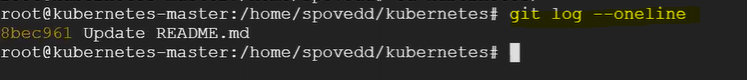
Where V is verbose



**I have shell script to delete particular dependancy in repo.Before running this script ,I need to clone repo to my local,here point to note is I should only clone master and only last commit(as it contains full code) how would you do this?**

--depth 1

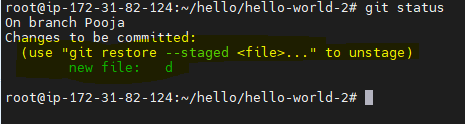


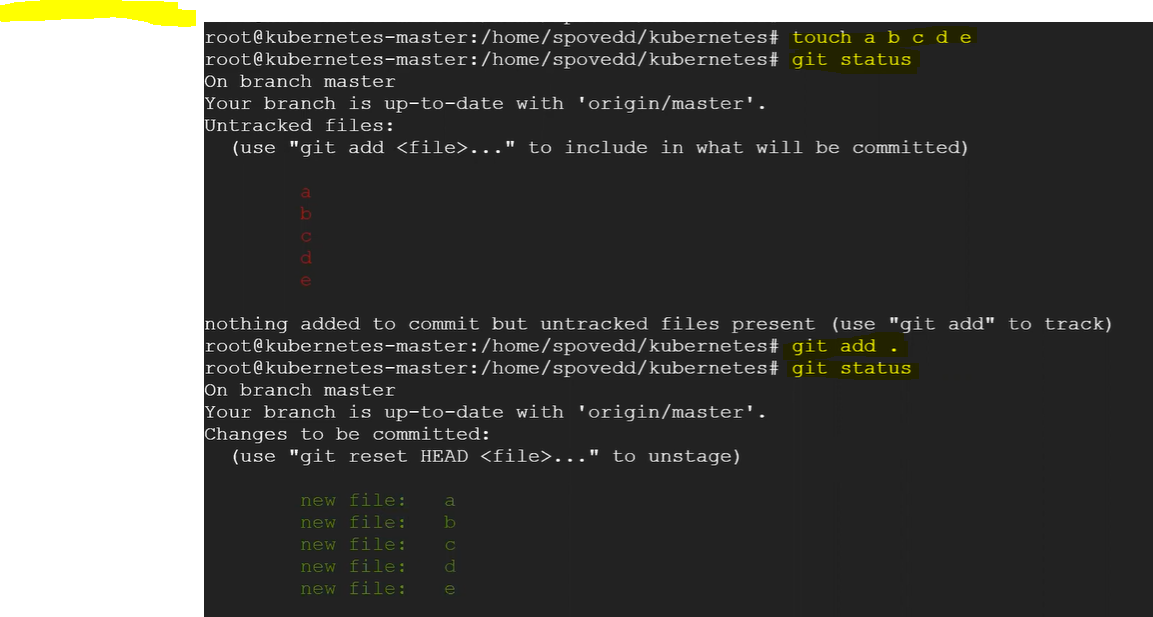


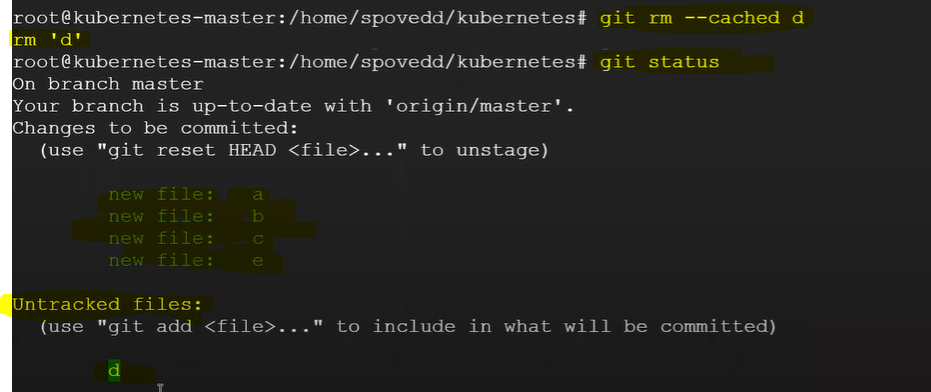
**Let’s say you have changed 5 files a,b,c,d,e in repo and you did git add . ,now all files are in staging area,now I decided not to commit file d. how would I delete from staging area to local?**

“**git rm –cached d** “ –to delete

“**git restore –staged d**” --to unstage



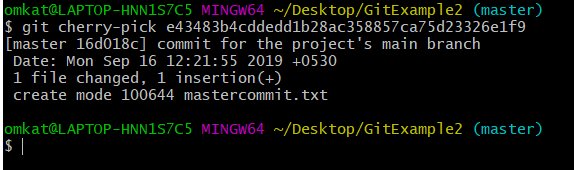




**What is cherry pick? Why we use it?**

**Cherry-pick** :: It helps us to apply changes to only particular branch by using commit ID.

Cherry-picking in Git is meant to apply some commit from one branch into another branch.



For example, say a commit is accidently made to the wrong branch. You can switch to the correct branch and cherry-pick the commit to where it should belong.

### Use of Git init?

It is used to initialize empty git repo.

### [What](https://www.javatpoint.com/git-head) is HEAD?

HEAD is the representation of the last/latest commit in the current checkout branch. When you switch branches with git checkout, the HEAD revision changes, and points the new branch**.(Fast-forwarding)**

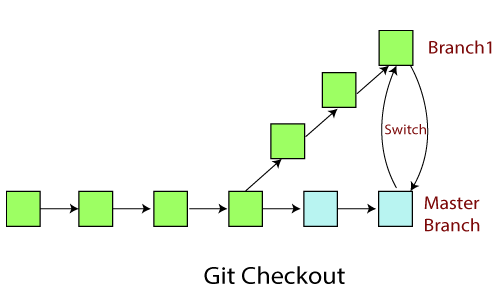
# What are Upstream and Downstream?

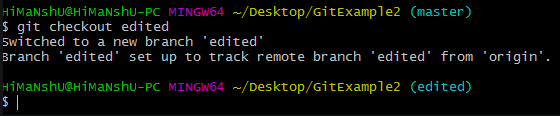
The term upstream and downstream refers to the repository. Generally, upstream is from where you clone the repository, and downstream is any project that integrates your work with other works.

**What is the use of GIT checkout?**

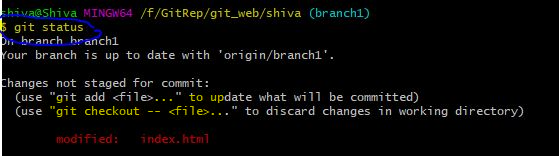
The **git checkout** command is used to switch between branches in a repository. Be careful with your staged files and commits when switching between branches.

**Note** :: If we switch to the other branch without add/commit ,the changes will be available to all the branches.(But GIT doesn’t allow you to switch to other branch if there are any untracked/unstaged files)





**And also it is useful to discard the changes in working area>>**



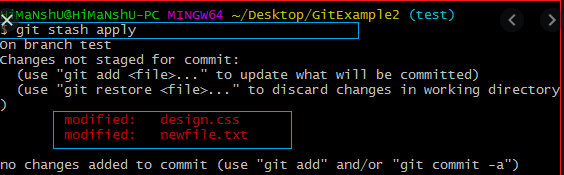
**Let’s say you’re working on new feature in some branch,now your manager come and says stop working on that and change few other things on an old code. Here after changing the old code,I need to work on new code,so I need to place my new changes in some place,how would you handle this?**

Will use Git stash

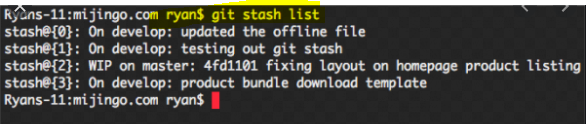
The **git stash command** enables you to switch branches without committing the changes in the current branch.(By saving working directory and index)

Generally, the stash's meaning is "**store something safely in a hidden place**." The sense in Git is also the same for stash; Git temporarily saves your data safely without committing.

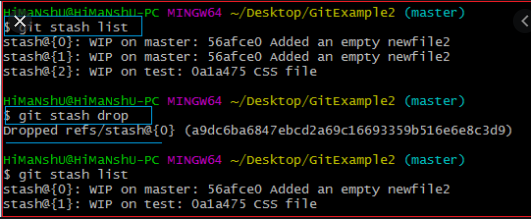
**Git stash apply**



Git stash list

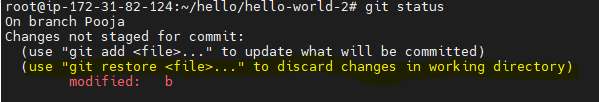


Git stash drop



**What is the use of git restore?**

The "restore" command helps to **unstage or even discard changes which are in staging area.**



**Note** :: **git switch & git restore** meant to replace **git checkout**

**What is the use of Git checkout?**

Git checkout is used to **unstage or even discard changes which are in working area.**

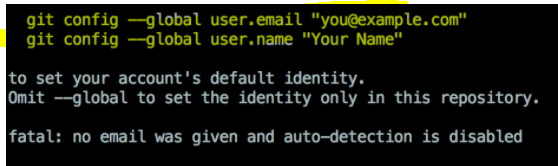
**Command to list all branches?**

Ans ::: Git branch –a

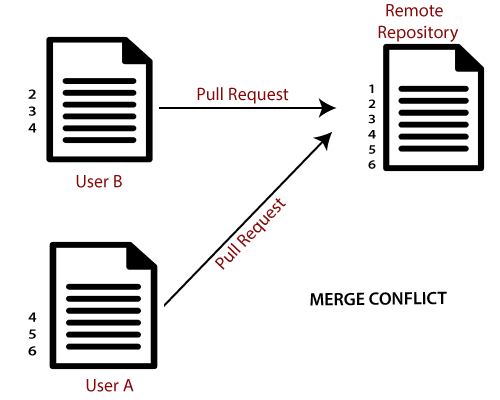
* To see local branches, run this command:
  + git branch
* To see remote branches, run this command:
  + git branch -r
* To see all local and remote branches, run this command:
  + git branch -a

### What is the purpose of 'git config'?

Used to set accounts default identity, and user information**. (Before pushing changes to the remote repo,Git should force us to set user information ,to track who is committing the changes.)**

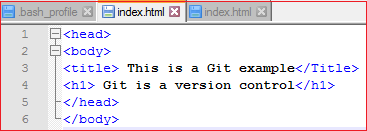


**What is merge conflict in GIT?**



When two users have edited the same file at the same time and are trying to merge the code onto the remote repo,then Git won't be able to identify which changes to take. Then this will results in situation called **merge conflict**.

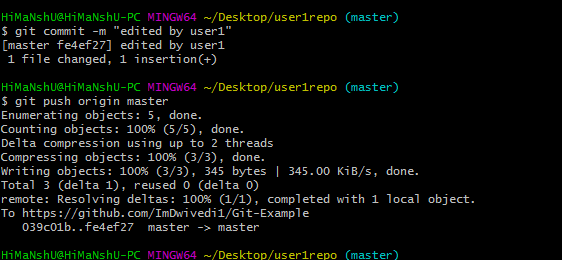
Two users (A & B) have the repository. The user A made changes as below in my projects index file.



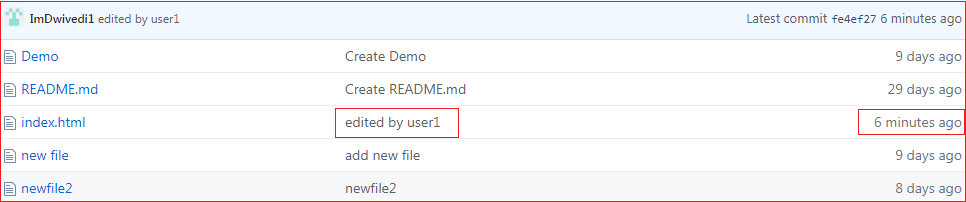
Update it in the staging and local repository with the help of git add & git commit command.

Git Merge and Merge Conflict

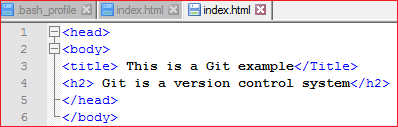
Now commit the changes and push it onto remote repository using GIT push. See the below output:



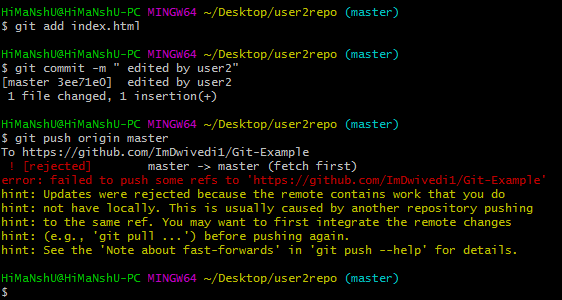
Now, UserA remote repository will look like this: It will show the status of the file like edited by whom and when.



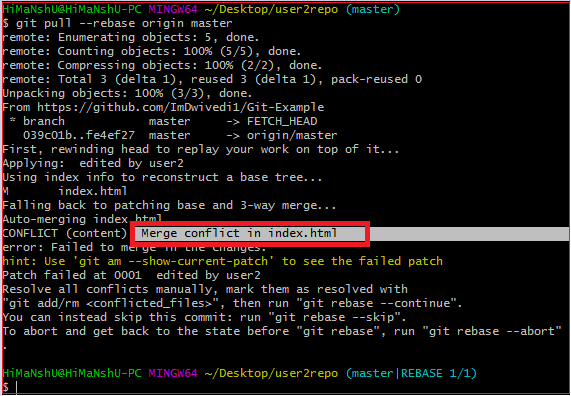
Now, at the same time, user2 also update the index file as follows.



User B has added and committed the changes in the local repository. But when he tries to push it to remote server, it will throw an error. See the below output:



In the above output, the server knows that the file is already updated and not merged with other branches. So, the push request was rejected by the remote server. It will throw an error message like [rejected] failed to push some refs to <remote URL>. It will suggest you to pull the repository first before the push. See the below command:

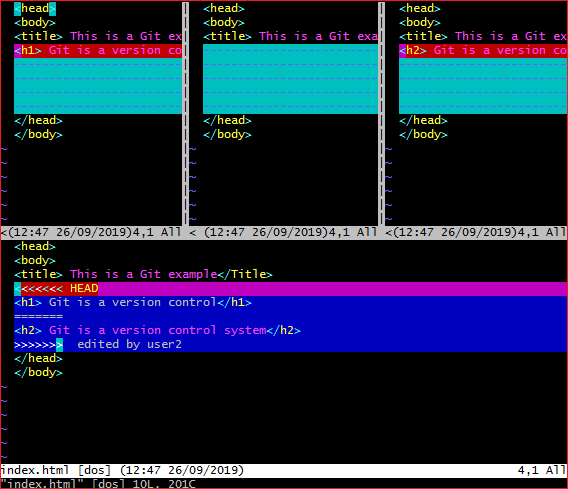


In the given output, git rebase command is used to pull the repository from the remote URL. Here, it will show the error message like merge conflict in <filename>.

**Resolve Conflict:**

To resolve the conflict, it is necessary to know when the conflict occurs and why it occurs. Git merge tool command is used to resolve the conflict. The merge command is used as follows:

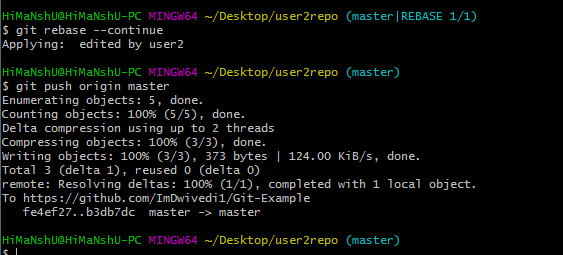
1. **$ git mergetool**



The above output shows the status of the conflicted file. To resolve the conflict, enter in the insert mode by merely pressing **I key** and make changes as you want. Press the **Esc key**, to come out from insert mode. Type the: **w!** at the bottom of the editor to save and exit the changes. To accept the changes, use the rebase command. It will be used as follows:

1. $ git rebase --continue

Hence, the conflict has resolved. See the below output:



In the above output, the conflict has resolved, and the local repository is synchronized with a remote repository.

**What is a commit message?**

The commit command is used to save changes to a local repository after staging in Git,,while doing we need to give message so that it’ll be easy for identification ,.

### By what method will you get to know whether a branch has just been merged into master or not?

**git branch - merged** It records the branches that have been merged into the present branch.

**git branch - no merged** It records the branches that have not been merged.

**What is GIT reset? Types of GIT reset?**

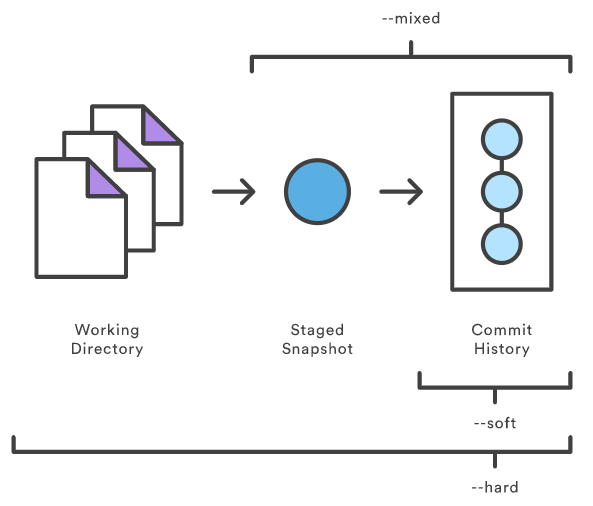
The git reset command is used to undo the changes in the local. The git reset command has three core forms.

**Note**:With git reset we can remove recent 2,3,etc commits,but we can’t remove particular commit like 5th or 6th commit.

**Git reset HEAD~1**—recent 1 **Git reset HEAD~3**—recent 3 commits

* **Soft**— It only resets the HEAD, and keeps the changes in staging area and working copy.(Git reset –soft)
* **Mixed**—(Default one) It resets the HEAD and undoes the changes in the staging area (and keeps the changes in working copy).(Git reset –mixed HEAD~1)
* **Hard**— Resets the HEAD and also it removes changes in Staging and Working copy.

To discard the changes from the files which are already staged and committed we use “GIT reset”



**How to delete local branch and remote branch in GIT?**

## Deleting local branches

1. First, we print out all the branches (local as well as remote), using the git branch command with -a (all) flag.
2. To delete the local branch, just run the git branch command again, this time with the -d (delete) flag, followed by the name of the branch you want to delete (test branch in this case).

**Note:** Comments are the output produced as a result of running these git commands

git branch -a

# \*master

#  test

#  remote/origin/master

#  remote/origin/test

**git branch -d test**

# Deleted branch test (was ########).

## Deleting remote branches

To delete a remote branch, you can’t use the git branch command. Instead, use the git push command with --delete flag, followed by the name of the branch you want to delete. You also need to specify the remote name (origin in this case) after git push.

git branch -a

# \*master

#  test

#  remote/origin/master

#  remote/origin/test

**git push origin --d test**

# To <URL of your repository>.git

#  - [deleted]         test

**Diff b/w git diff and git status?**

**Diff** ---It shows what is the changes in the file from last commit/to the current commit.

**Status** –Shows what are all the unstaged/untracked files in the current branch which needs to be add or commit.

**What are GIT hooks?**

Git hooks are scripts that run automatically whenever a particular event occurs in a Git repository.

**What is .GIT? and the contents of it?**

It is a repository (.git/ folder) inside a project. This repository tracks all changes made to files in your project.

git contains all information required for version control. If you want to clone your repo, copy .git is enough.

**4 sub-directories:**

* hooks/ : example scripts
* info/ : exclude file for ignored patterns
* objects/ : all "objects"
* refs/ : pointers to commit objects

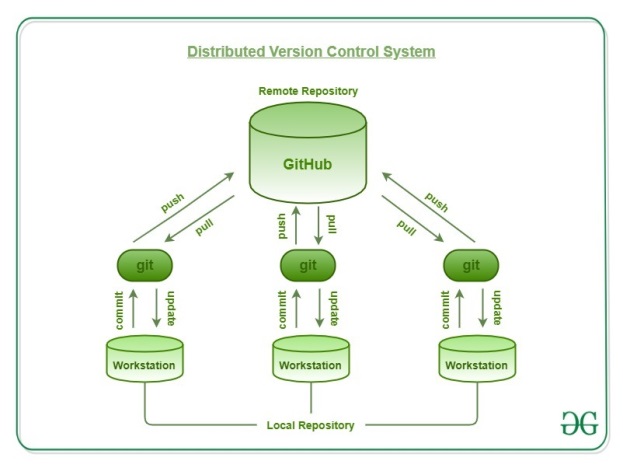
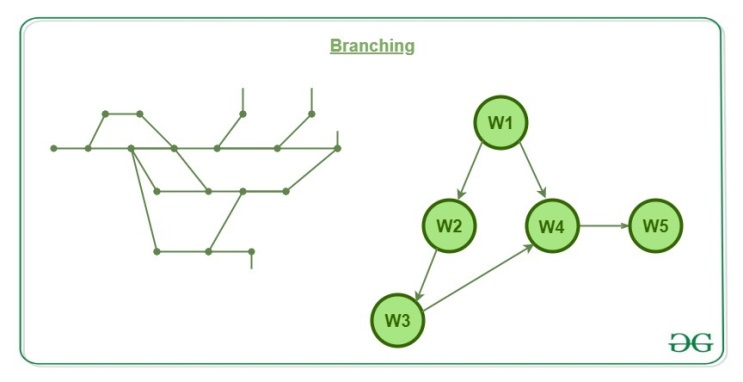
**4 files:**

* HEAD : current branch
* config : configuration options
* description
* index : staging area

**Here "object" includes:**

* blobs(files)
* trees(directories)
* commits(reference to a tree, parent commit, etc)

**What are the features of GIT:**

* **Distributed System:**  
  A distributed system holds a local repository that can be accessed by many remote collaborators by using a Version Control System.
  + **\*\* These not rely on a remote server,bcz every developer will have local copy.**
  + 
* **Compatibility:**  
  Git is compatible with all the Operating Systems that are being used these days. Git repositories can also access the repositories of other Version Control Systems like SVN, CVK, etc.
* **Non-linear Development:**  
  Git allows users from all over the world to perform operations on a project remotely. A user can pick up any part of the project and do the required operation and then further update the project. This can be done by the Non-linear development behavior of the Git.
* **Branching:** Branches in Git provide a feature to make changes in the project without affecting the original version.
  + 
* **Lightweight:**  
  One might be worried that cloning that much data into local machines might result in system failure but Git has already taken care of such a problem. Git follows the criteria of lossless compression that compresses the data and stores it in the local repository occupying very minimal space.
* **Speed:**  
  Since Git stores all the data related to a project in the local repository by the process of cloning, it is very much efficient to fetch data from the local repository instead of doing the same from the remote repository.
* **Reliable:**  
  Providing a central repository that is being cloned each time a User performs the Pull operation, the data of the central repository is always being backed up in every collaborator’s local repository. Hence, in the event of crashing of the central server, the data will never be lost as it can be gained back easily by any of the developer’s local machine.
* **Secure:**  
  Git keeps a record of all the commits done by each of the collaborators on the local copy of the developer. A log file is maintained and is pushed to the central repository each time the push operation is performed.

**What is a staging area?**  
It’s an intermediate area where commits can be formatted and reviewed before completion. When you make a commit, Git takes changes that are in the staging area and make them as a new commit.

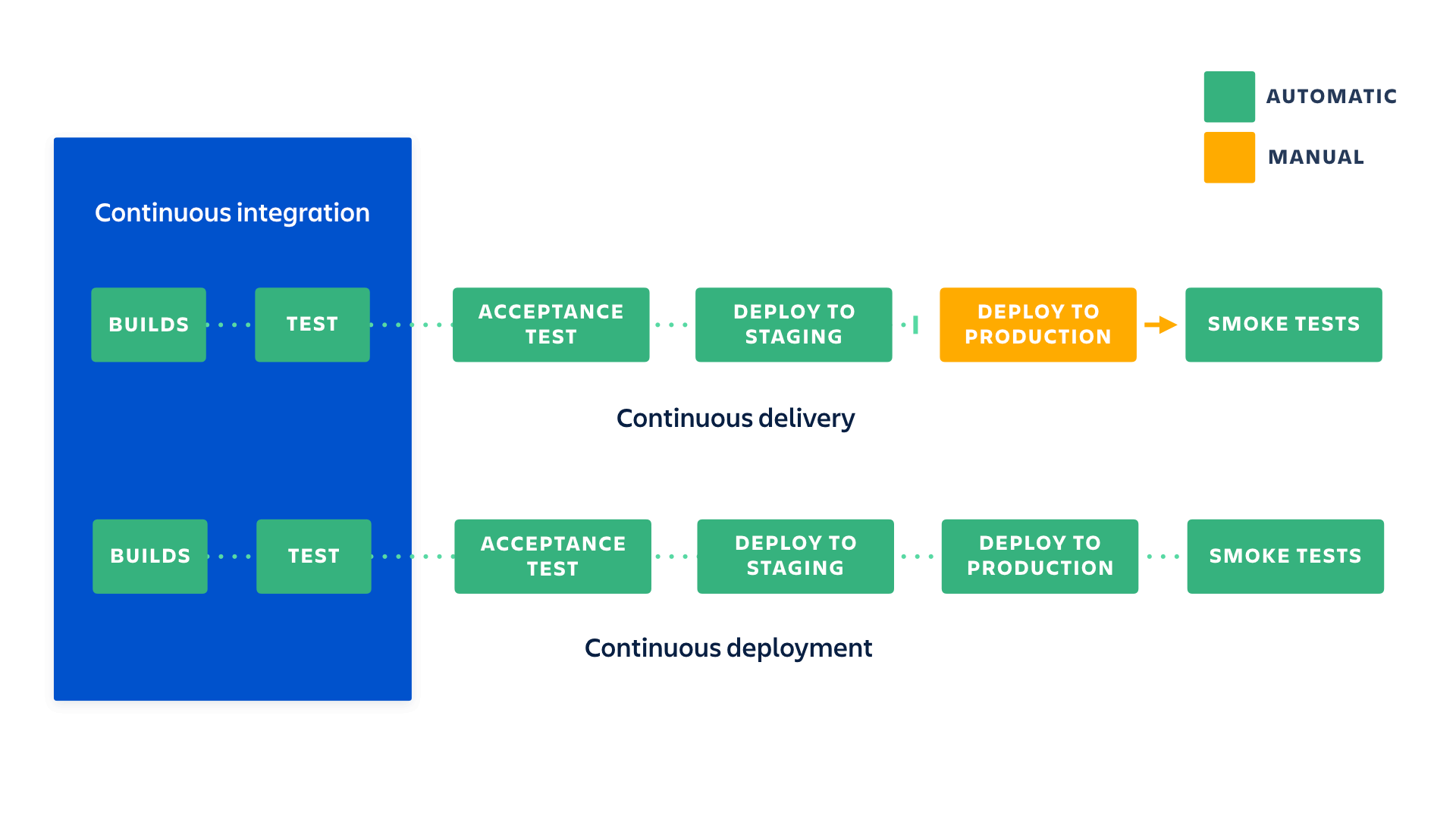


## List the Benefits of Git?



**What Is Continuous integration ,Delivery and Deployment?**

<https://www.atlassian.com/continuous-delivery/principles/continuous-integration-vs-delivery-vs-deployment>



These are software development and software release practices,which most of the companies have adopted according to their requirements. Before they used to follow waterfall,agile methodologies which had some issues ,in case of waterfall model there will not be continuous feedback and in contrast with that in agile will have continuous feedback but no continuous integration testing.

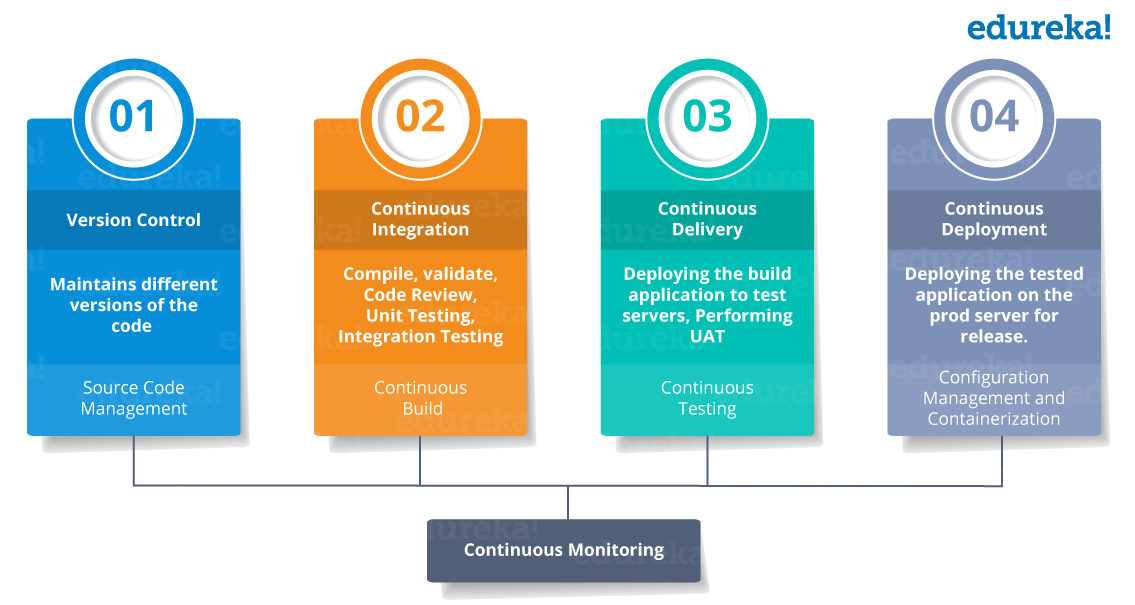
CI==Build+test+code analysis+package+release onto to artifactory

## Continuous integration :: Nothing but building the application continuously ,whenever a developer made commit on source code a continuous integration server will pull the code and prepare a build, validate,code review ,unit testing ,integration testing ,packaging the application and it also includes pushing the artifacts to the central repo(Nexus/jfrog).

## Continuous Delivery :: [Continuous delivery](https://www.atlassian.com/continuous-delivery) is an extension of continuous integration.The application code which was build is deployed to the test environment for testing if everything gets successful here ,the tested application will be deployed onto the prod environment manually.

## Continuous deployment : is the next step of continuous delivery: Every change that passes the automated tests is deployed to production automatically without human intervention.

## Continuous monitoring :: The deployed application will be monitored and feedback will be given to avoid failures ,using monitoring tools like Prometheus and Graphana.

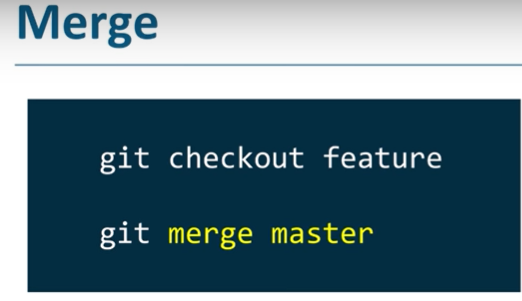
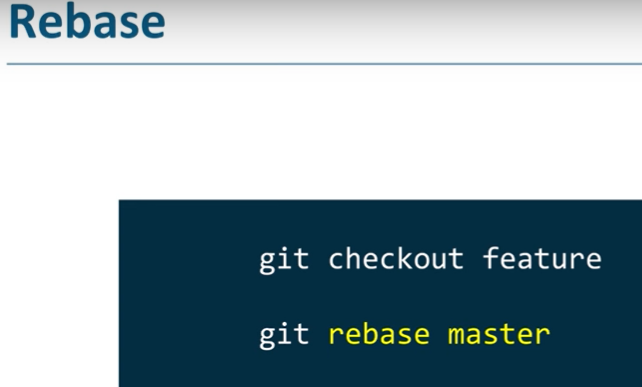
****

**Diff b/w GIT merge and GIT rebase?**

GIT rebase solves the same problem as GIT merge does,but it does it in a diff way.

Consider if I am working in a new feature in a dedicated branch .lets suppose my another team member updates the master branch with new commits ,if these new commits in the master are relevant to the feature branch which I’m working on ,now to incorporates these changes into the feature branch ,I have two options.

* 1. Either Merging the feature branch with the master
  2. Or rebasing with the master

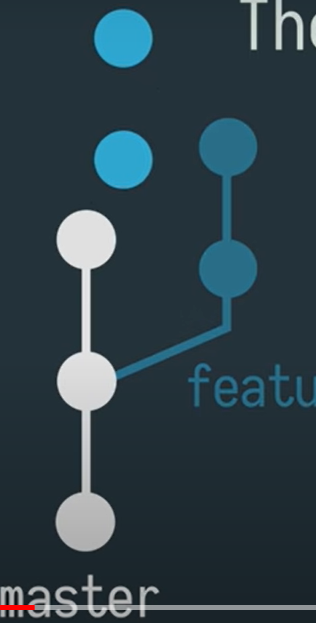
 



**Git merge ::** It’ll take all of the feature branch changes (2commits as shown in the figure above) and it’ll combine into one commit, which looks like this.



**Git rebase**: It takes all of the commits from feature branch and it moves them on top of the master branch ,instead of combining ,so it looks linear.

**What is tag in a branch?**

A tag represents version of a perticular branch at a moment in time.

**How to add only particular file to staging area?**

git add filename

**Explain Branching strategy In your project? (MDFRH)**

Currently in our project we’ve master,Dev,feature ,release and hot fix branches.

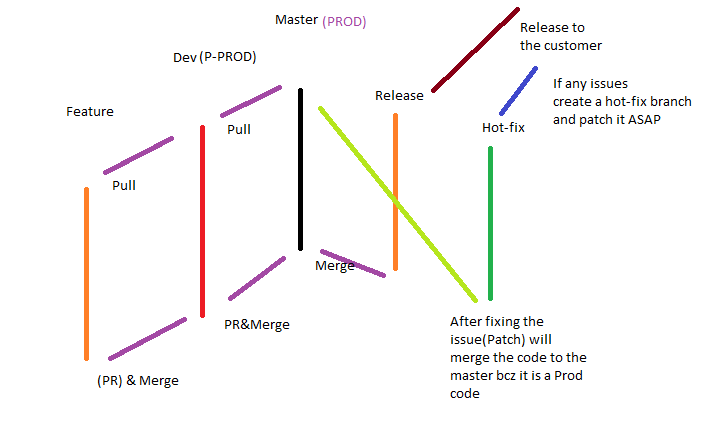
Master basically contains production code which is stable ,Dev as a pre-production taken from the Master and when we assign with new feature to add on, then will create feature branch from Dev branch ,when we’re done with new feature will create pull request and that will be merged into Dev branch .And release branch is to support preparation of new production release and hotfix is just to patch production releases if there is any bug.

**Feature branch names may contain JIRA issues name or sprint backlog ID’s**

**Unit tests happens at feature branch.**

**Feature---day build,,**

**Dev,Prod –Night build**



**What is pull request ?**

Pull requests are a mechanism for developers to notify team members that they have completed a feature.

**Use of Git revert ?**

To discard the changes from the remote repo.

Using which we can remove any commit particularly.

>>**Git revert commit ID**

Note :: It undo all the changes and it will add fresh commit.

**How to find list of files that changed during a commit?**

Git diff-tree

**How do you fix broken commit?**

Git commit –amend

**What is Git hub ?**

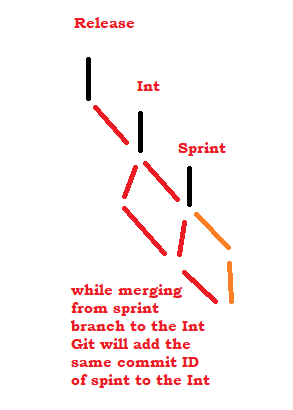
Github is a git repository hosting service.

**Variuos Git repository hosting functions?**

* Github
* Gitlab
* Bitbucket
* Git enterprise

**What is fast forward merge ?**

Whenever we merge code from one branch to another , GIT will add the latest commit ID to same branch. This is called Fast forward merge.



**what is .git ignore ?can you explain internal process?**

* a file which Git has been explicitly told to ignore.
* If you want to ignore a file that you've committed in the past, you'll need to delete the file from your repository and then add a . gitignore rule for it. Using the --cached option with git rm means that the file will be deleted from your repository, but will remain in your working directory as an ignored file.

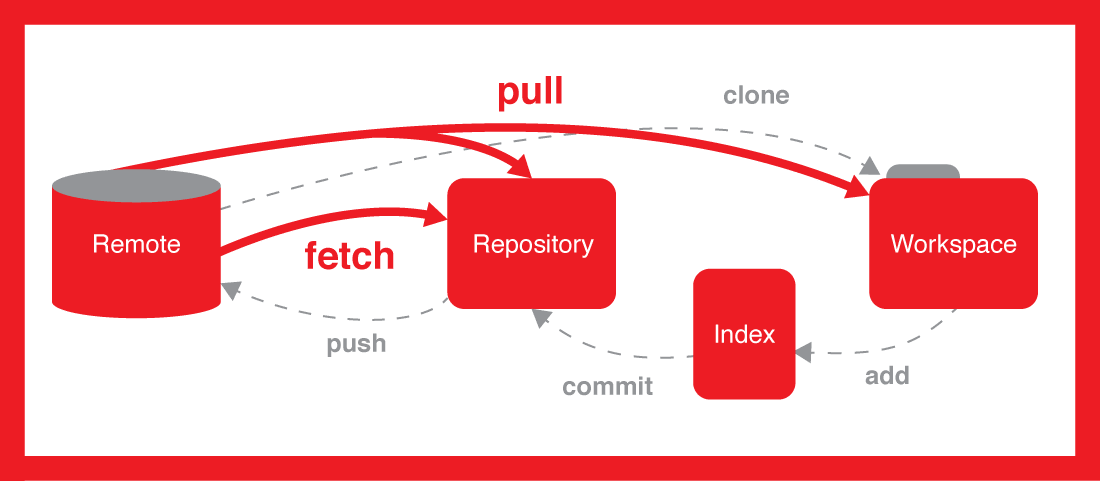
1. **what is git pull ,push and fetch and clone?**

**Push**: Send the local repo content to remote repo.

**Fetch**: it will bring remote changes to local repo and you have to again merge.

**Clone**: Bring total remote repo to Working area (Workspace).

**Git pull**= fetch + merge (FM)

What is the difference between 'git pull' and 'git fetch'? - Stack Overflow 

**how will you create another branch in git repository?\***

**Git branch**

We can also use **GIT checkout –b branchname**” it will create a branch and jump onto that branch.

**How do you setup a script to run every time a repository receives new commits through push?**

Using Git hooks

**How to clone the repository?**

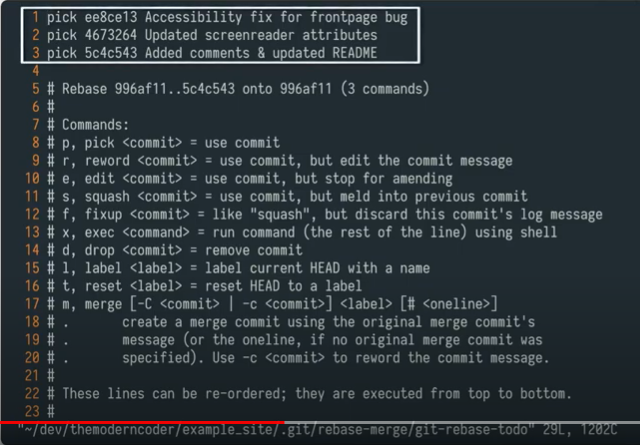
Git clone repo url

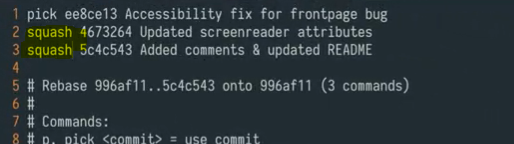
**What is** **Git squash?**

 squash is used to squash the previous commits into one.

 squashing all the commits into a large commit is the right choice than pushing

**$ git rebase -i HEAD ~3   (Opens up vim editor) and manually change “pick” to “squash”**





**What is GIT workflow model?**

Before you **merge** a feature branch back into your main branch (often master or develop ), your feature branch should be squashed down to a single buildable commit, and then rebased from the up-to-date main branch

**What is sprint and duration of it?**

A Sprint is a defined time period for developing feature for a product.

It’s a rule of scrum that a sprint shouldn’t be longer than one month.

**What is a tag in branch?**

It represents version of a particular branch at a moment in time.

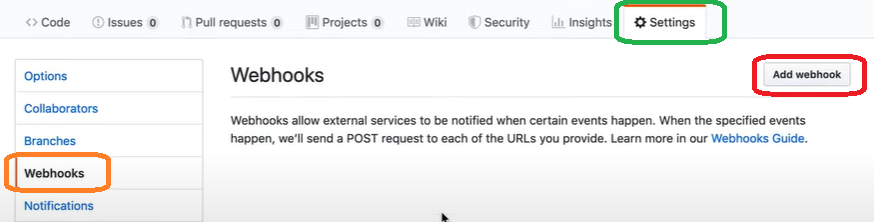
**How to delete branches in one shot?**

git branch -d branch1 branch2 branch3

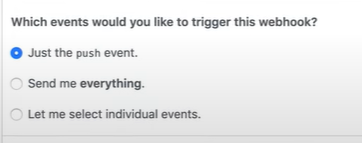
**Integration of GIT with hooks?**

Git hooks are shell scripts that trigger when you perform a specific action in Git.

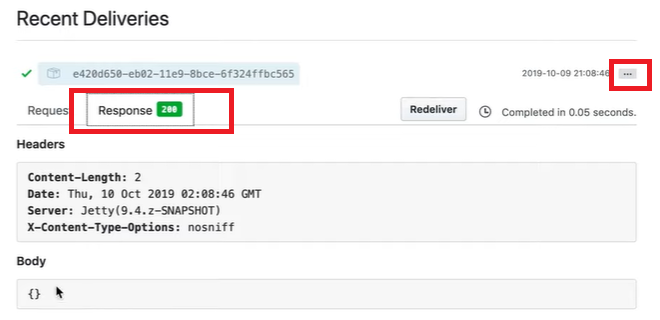
* 1. Install **Github integration plugin**
  2. In the Jenkins console ,while configuring a job in the Build trigger section select the option “**Githubhook trigger GITscm polling”**
  3. Go to the Github repository and click on settings 🡪webhooks 🡪 add webhook🡪put the password



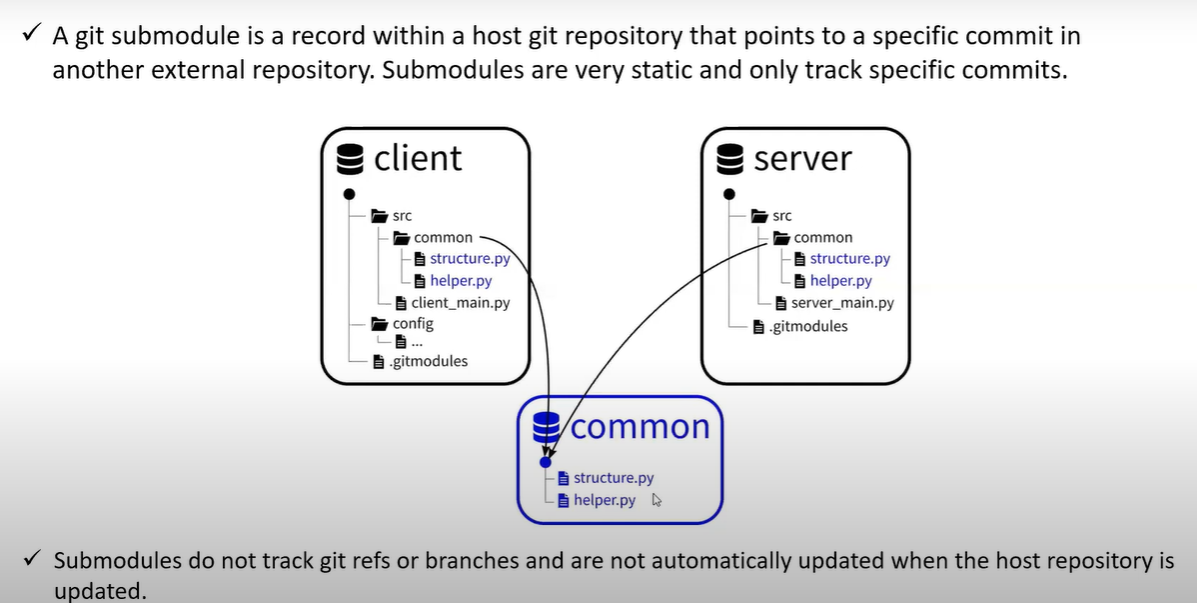
* 1. Give the payload URL (**Jenkins url/github-webhook/.**) and select the event type like just to push the event or send everything. 🡪 Add webhook



* 1. To check whether the webhook configuration is correct ,goto “**Recent deliveries** “ if the response is green with the code like 200



7**) What is a sub module and why we need it? And subtree?**



$ git submodule add <https://github.com/chaconinc/DbConnector>

Cloning into 'DbConnector'...

remote: Counting objects: 11, done.

remote: Compressing objects: 100% (10/10), done.

remote: Total 11 (delta 0), reused 11 (delta 0)

Unpacking objects: 100% (11/11), done.

Checking connectivity... done.

By default, submodules will add the subproject into a directory named the same as the repository, in this case “DbConnector”. You can add a different path at the end of the command if you want it to go elsewhere.

If you run git status at this point, you’ll notice a few things.

$ git status

On branch master

Your branch is up-to-date with 'origin/master'.

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

new file: .gitmodules

new file: DbConnector

First you should notice the new .gitmodules file. This is a configuration file that stores the mapping between the project’s URL and the local subdirectory you’ve pulled it into: