Git & GitHub Interview Questions

What is git?

Ans)

Git is a DevOps tool used for source code management. It is a free and open-source version control system used to handle small to very large projects efficiently. Git is used to tracking changes in the source code, enabling multiple developers to work together on non-linear development.

Git is a version control system used for tracking changes in computer files. It is generally used for source code management in software development.

Git is used to tracking changes in the source code

The distributed version control tool is used for source code management

It allows multiple developers to work together

It supports non-linear development through its thousands of parallel branches

Features of Git

Tracks history

Free and open source

Supports non-linear development

Creates backups

Scalable

Supports collaboration

Branching is easier

Distributed development

What is the difference between git and GitHub?

Ans)

While Git is a tool that's used to manage multiple versions of source code edits that are then transferred to files in a Git repository, GitHub serves as a location for uploading copies of a Git repository.

S.No., Git, GitHub

1., Git is a software., GitHub is a service.

2., Git is a command-line tool, GitHub is a graphical user interface

3., Git is installed locally on the system, GitHub is hosted on the web

4., Git is maintained by linux., GitHub is maintained by Microsoft.

5., Git is focused on version control and code sharing., GitHub is focused on centralized source code hosting.

6., Git is a version control system to manage source code history.
, GitHub is a hosting service for Git repositories.

7., Git was first released in 2005.
, GitHub was launched in 2008.

8., Git has no user management feature.
, GitHub has a built-in user management feature.

9., Git is open-source licensed., GitHub includes a free-tier and pay-for-use tier.

10., Git has minimal external tool configuration., GitHub has an active marketplace for tool integration.

11., Git provides a Desktop interface named Git Gui., GitHub provides a Desktop interface named GitHub Desktop.

12., Git competes with CVS, Azure DevOps Server, Subversion, Mercurial, etc., GitHub competes with GitLab, Git Bucket, AWS Code Commit, etc.

What is the abbreviation of VCS?

Ans)

Version Control Systems (VCS) have seen great improvements over the past few decades and some are better than others. VCS are sometimes known as SCM (Source Code Management) tools or RCS (Revision Control System). One of the most popular VCS tools in use today is called Git.

Git is a distributed version control system and is definitely the most used SCM today. Many developers may argue that Git is one of the most difficult VCS and this is somewhat true: it has a higher learning curve, but it also makes branch management a lot easier.

When developers are creating something (an application, for example), they are making constant changes to the code and releasing new versions, up to and after the first official (non-beta) release.

Version control systems keep these revisions straight, and store the modifications in a central repository. This allows developers to easily collaborate, as they can download a new version of the software, make changes, and upload the newest revision. Every developer can see these new changes, download them, and contribute.

What is the abbreviation of SCM?

Ans)

Source code management (SCM) is used to track modifications to a source code repository. SCM tracks a running history of changes to a code base and helps resolve conflicts when merging updates from multiple contributors.

Write down 5 SCM names?

Ans)

Git Lab
Bit Bucket
Git
Giyhub
Team Foundation Server

What is Branch?

Ans)

In Git, branches are a part of your everyday development process. Git branches are effectively a pointer to a snapshot of your changes. When you want to add a new feature or fix a bug—no matter how big or how small—you spawn a new branch to encapsulate your changes.

A branch in Git is simply a lightweight movable pointer to one of these commits. The default branch name in Git is master. As you start making commits, you're given a master branch that points to the last commit you made. Every time you commit, the master branch pointer moves forward automatically.

Branches are used to create another line of development.

By default, Git has a master branch, which is same as trunk in Subversion (SVN).

Usually, a branch is created to work on a new feature.

Once the feature is completed, it is merged back with the master branch and we delete the branch.

Git checkout –b branch name

Git checkout branchname

What is tag?

Ans)

Tags similar to branches, but the difference is that tags are immutable.
it means, tag is a branch, which nobody intends to modify. Once a tag is created for a particular commit, even if you create a new commit, it will not be updated.

Usually, developers create tags for product releases.

When we will create the branch and when we will the create tag?

Ans)

You should create a new branch when you're doing development work that is somewhat experimental in nature. So in your scenario definitely create a new branch and not a folder within master. If you created your sandbox work as a directory in the master, it's going to reside there until you remove it using git.

Tagging is generally used to capture a point in history that is used for a marked version release (i.e. v1. 0.1). A tag is like a branch that doesn't change. Unlike branches, tags, after being created, have no further history of commits.

What is “git add” command will do?

Ans)

The git add command adds a change in the working directory to the staging area. It tells Git that you want to include updates to a particular file in the next commit.

WORKING AREA STAGING AREA LOCAL REPO

FILE

What is the command to revert the code from staging area to working area?

Ans)

Git reset is a powerful command that is used to undo local changes to the state of a Git repo.

What is the git revert command?

Ans)

Git reset – unstage a file ! From staging area to working area! Bringing them back, only works on recent commits.

Git revert – undo changes in any commits unlike reset! Undoes even pushed commits to github

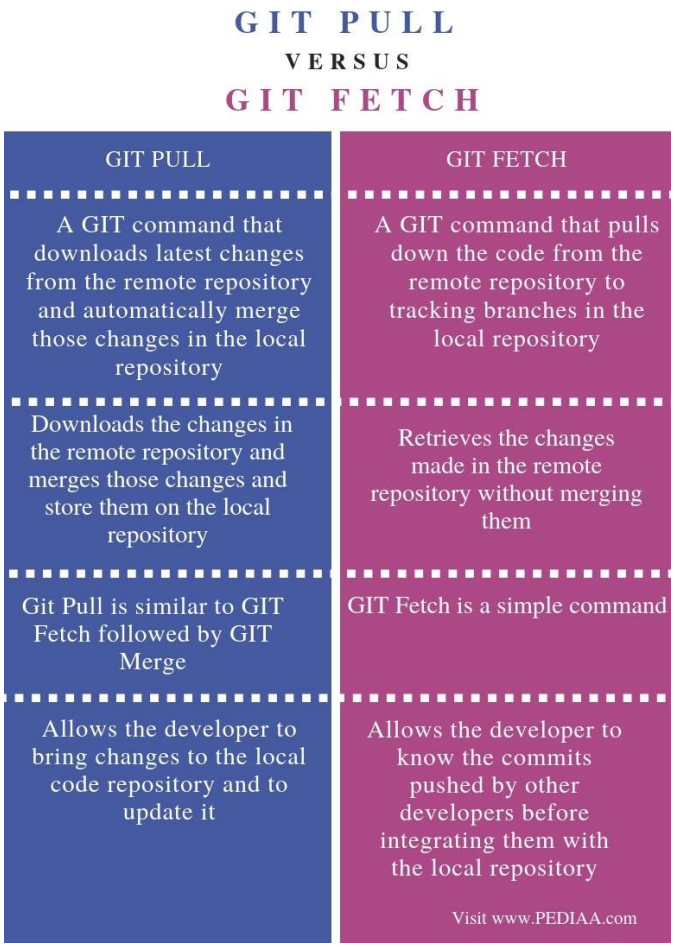
What is the difference between fetch and pull in Git?

Ans)

git fetch is the command that tells your local git to retrieve the latest meta-data info from the original (yet doesn't do any file transferring. It's more like just checking to see if there are any changes available), safer option.

git pull on the other hand does that AND brings (copy) those changes from the remote repository.

Git pull = git fetch + git merge



How to see how many branches are currently available in git repo?

Ans)

Git branch –a

What is the command to create the branch?

Ans)

Git checkout –b branch name

How to switch from branch to another branch?

Ans)

Git checkout branch name

What is the command to delete the branch in local repo?

Ans)

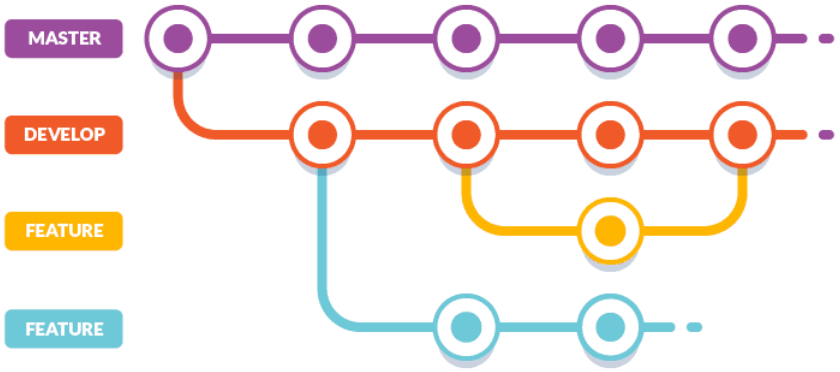
git branch -d <branch-name>

What is the branching strategy you are following?

Ans)

Simply put, a branching strategy is something a software development team uses when interacting with a version control system for writing and managing code. As the name suggests, the branching strategy focuses on how branches are used in the development process.

Git Flow. One well-known branching strategy is called Git Flow. The main branch always reflects the current production state. There is a second long-running branch, typically called develop. All feature branches start from here and will be merged into develop



What is git stash command?

Ans)

Git stash saves the uncommitted changes locally, allowing you to make changes, switch branches, and perform other Git operations. You can then reapply the stashed changes when you need them. A stash is locally scoped and is not pushed to the remote by git push.

Stashing is handy if you need to quickly switch context and work on something else, but you're mid-way through a code change and aren't quite ready to commit.

$ git status
On branch main
Changes to be committed:

new file: style.css

Changes not staged for commit:

modified: index.html

$ git stash
Saved working directory and index state WIP on main: 5002d47 our new homepage
HEAD is now at 5002d47 our new homepage

$ git status
On branch main
nothing to commit, working tree clean

At this point you're free to make changes, create new commits, switch branches, and perform any other Git operations; then come back and re-apply your stash when you're ready.
Note that the stash is local to your Git repository; stashes are not transferred to the server when you push.

You can reapply previously stashed changes with git stash pop

What is git cherry-pick?

Ans)

Cherry picking in Git means to choose a commit from one branch and apply it onto another. This is in contrast with other ways such as merge and rebase which normally apply many commits onto another branch. Make sure you are on the branch you want to apply the commit to.

Cherry picking in git means to choose a commit from one branch and apply it onto another.

What does git rebase?

Ans)

Rebase is an action in Git that allows you to rewrite commits from one branch onto another branch. Essentially, Git is deleting commits from one branch and adding them onto another.

What is PAT?

Ans)

Personal access tokens (PATs) are an alternative to using passwords for authentication to GitHub when using the GitHub API or the command line. If you want to use a PAT to access resources owned by an organization that uses SAML SSO, you must authorize the PAT.

What is SSH key and how to generate ssh-key?

Ans)

An SSH key is an access credential for the SSH (secure shell) network protocol. This authenticated and encrypted secure network protocol is used for remote communication between machines on an unsecured open network. SSH is used for remote file transfer, network management, and remote operating system access.

Ssh keygen command

What is the default path for ssh keys?

Ans)

Ls –la ~/.ssh to check wether we have any ssh files or not

~/.ssh directory
By default, the keys will be stored in the ~/. ssh directory within your user's home directory. The private key will be called id_rsa and the associated public key will be called id_rsa.

How many file will generate if you create ssh keys and what are the file names?
Ans)

Id_rsa – private key
Id_rsa.pub - public key

What are the git best practices while committing the code need to follow?

Ans)

Use branching strategy and pull requests
Commit once you finish the task.
Don't Commit Half-Done Work
Test your code before commit.
Write Good Commit Messages before you are committing
Try to use git commands rather than GUI tools.
Avoid merge commits.

How to change git commit message after push (given that no one pulled from remote)?

Ans)

#git commit --amend -m "an updated commit message" : Change most recent Git commit message.

How to rename the branch name?

Ans)

Git branch –m <oldname><newname>

What is the command to get the code from remote git repo?

Ans)

Git fetch

Remote repositories are versions of your project that are hosted on the Internet or network somewhere. You can have several of them, each of which generally is either read-only or read/write for you.

git fetch : It will get the update from git remote repo and will update your local repo. But it will not merge with Local working copy.
git pull : It will get the update from git remote repo and will update your local repo as well it will merge with Local working copy also.

How to check the particular branch from github?

Ans)

Change to the root of the local repository. $ cd <repo_name>
List all your branches: $ git branch -a. ...
Checkout the branch you want to use. $ git checkout <feature_branch>
Confirm you are now working on that branch: $ git branch.

What is the command to see all git messages in local repo?

Ans)

Git log

How to check if a branch is already merged with any branch or not?

Ans)

You can use the git merge-base command to find the latest common commit between the two branches. If that commit is the same as your branch head, then the branch has been completely merged.

Git merge-base branch 1 branch 2

What is SubGit?

Ans)

SubGit is stress-free tool for migrating from SVN to Git. It can be used with any Git server whether it is Github, Gitlab, Gerrit, or Bitbucket.It is developed by Tmate software. Answered by devquora. It is software for migrating SVN to GIT.

What is Git SVN?

Ans)

The git-svn tool is an interface between a local Git repository and a remote SVN repository. Git-svn lets developers write code and create commits locally with Git, then push them up to a central SVN repository with svn commit-style behavior.

Below is a table of differences between GIT and SVN:

GIT, SVN

Git is open source distributed vice control system developed by Linus Torvalds in 2005. It emphasis on speed and data integrity, Apache Subversion is an open source software version and revision control system under Apache license.

Git has a Distributed Model., SVN has a Centralized Model.

In git every user has their own copy of code on their local like their own branch., In SVN there is central repository has working copy that also make changes and committed in central repository.

In git we do not required any Network to perform git operation. , In SVN we required Network for runs the SVN operation.

Git is more difficult to learn. It has more concepts and commands. , SVN is much easier to learn as compared to git.

In git we create only .git directory., In SVN we create .svn directory in each folder.

It does not not have good UI as compared to SVN., SVN has simple and better user interface .Git is complex for beginners.

In Which language Git is developed?

Ans)

C Langugae

What is the difference between git merge and git rebase?

Ans)

Rebase is an action in Git that allows you to rewrite commits from one branch onto another branch. Essentially, Git is deleting commits from one branch and adding them onto another.

Git merge is a command that allows you to merge branches from Git. Merging is a common practice for developers. Whether branches are created for testing, bug fixes, or other reasons, merging commits changes to another branch. It takes the contents of a source branch and integrates it with a target branch.

What is the difference between git reset and git revert?

Ans)

Git reset – unstage a file ! From staging area to working area! Bringing them back, only works on recent commits. (local repo)

Git revert – undo changes in any commits unlike reset! Undoes even pushed commits to github (remote repo)

What is the git workflow?

Ans)

A Git workflow is a recipe or recommendation for how to use Git to accomplish work in a consistent and productive manner. Git workflows encourage developers and DevOps teams to leverage Git effectively and consistently. Git offers a lot of flexibility in how users manage changes.

How you will migrate the projects from SVN to GitHub?

Ans)

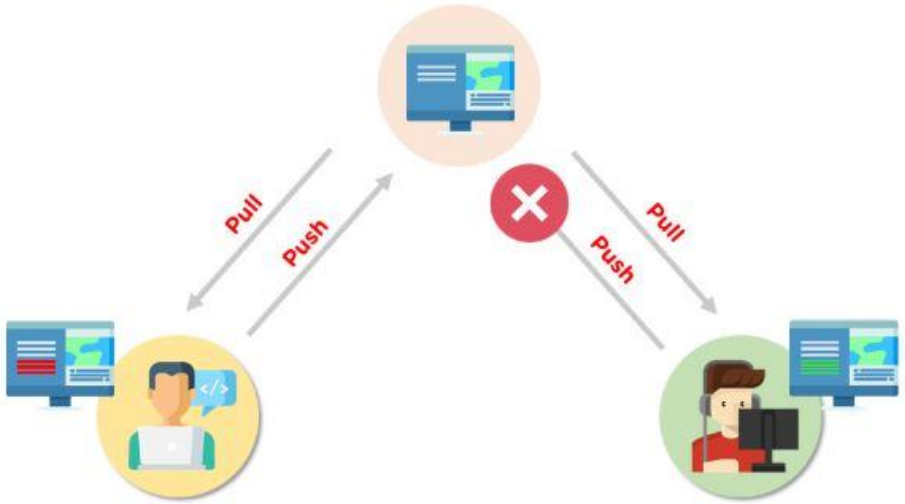
Prepare your environment for the migration.
Convert the SVN repository to a local Git repository.
Synchronize the local Git repository when the SVN repository changes.
Share the Git repository with your developers via Bitbucket.
Migrate your development efforts from SVN to Git.

SubGit is stress-free tool for migrating from SVN to Git. It can be used with any Git server whether it is Github, Gitlab, Gerrit, or Bitbucket.It is developed by Tmate software. Answered by devquora. It is software for migrating SVN to GIT.

What is merge conflicts? Have you resolved the merge conflicts?

Ans)

Merge conflicts occur when competing changes are made to the same line of a file, or when one person edits a file and another person deletes the same file. Merge conflicts happen when you merge branches that have competing commits, and Git needs your help to decide which changes to incorporate in the final merge. Git can often resolve differences between branches and merge them automatically..



Let’s assume there are two developers: Developer A and Developer B. Both of them pull the same code file from the remote repository and try to make various amendments in that file. After making the changes, Developer A pushes the file back to the remote repository from his local repository. Now, when Developer B tries to push that file after making the changes from his end, he is unable to do so, as the file has already been changed in the remote repository.

To prevent such conflicts, developers work in separate isolated branches. The Git merge command combines separate branches and resolves any conflicting edits.

There are a few steps that could reduce the steps needed to resolve merge conflicts in Git.

The easiest way to resolve a conflicted file is to open it and make any necessary changes

After editing the file, we can use the git add a command to stage the new merged content

The final step is to create a new commit with the help of the git commit command

Git will create a new merge commit to finalize the merge

Let us now look into the Git commands that may play a significant role in resolving conflicts.

Git Commands to Resolve Conflicts

1. git log --merge

The git log --merge command helps to produce the list of commits that are causing the conflict

2. git diff

The git diff command helps to identify the differences between the states repositories or files

3. git checkout

The git checkout command is used to undo the changes made to the file, or for changing branches

4. git reset --mixed

The git reset --mixed command is used to undo changes to the working directory and staging area

5. git merge --abort

The git merge --abort command helps in exiting the merge process and returning back to the state before the merging began

6. git reset

The git reset command is used at the time of merge conflict to reset the conflicted files to their original state

How to see how many files are committed in one particular commit id?

Ans)

Very useful command that you can use is git show command. Using this method, you can check all the changes done on a Specific Commit ID. Syntax to use the command is git show <Commit ID> . You can get all the Commit ID from git log command.

What is git hooks? Can you name some git hooks?

Ans)

Git hooks are scripts that run automatically every time a particular event occurs in a Git repository. They let you customize Git's internal behavior and trigger customizable actions at key points in the development life cycle.

There are Two types of git hooks.

- **Client-Side Hooks**
- **Server-Side Hooks**

Client-Side Hooks are trigged by operations such as committing and merging.

Server-Side Hooks are triggered by network operations such a receiving pushed commits.

After you are initialising the git repo, using git init command, it will create a one folder called. git and it contains some sub folders called hooks, branches, info, objects... etc

There are some predefined client-side hooks are available in hooks folder.

applypatch-msg.sample

post-update.sample

pre-push.sample

prepare-commit-msg.samp1e

What is .gitignore file?
Ans)

gitignore file tells Git which files to ignore when committing your project to the GitHub repository. gitignore is located in the root directory of your repo. The .gitignore file itself is a plain text document.

Example - Sometimes we don't want to commit the files which are generated by IDE like .project. and .classpath. To ignore these files and folders to commit. We will create one file called .git ignore. And we will keep the filenames and directory names which we don't want to commit. Not too bad.

What is git bare and git non bare repository, use cases?
Ans)

A bare repository is one that contains nothing but the . git folder; in other words, it has the index but lacks the actual working files. A non-bare repository is what you're used to working with, which includes both the git index and the checked out copy of working files.

A bare Git repository is typically used as a Remote Repository that is sharing a repository among several different people. You don't do work right inside the remote repository so there's no Working Tree (the files in your project that you edit), just bare repository data. And that's it.

What is the git command to create the achieve files?

Ans)

You can specify a permanent file by using git archives output option.

What is git prune, Mirroring and git repack commands?

Ans)

The git prune command is an internal housekeeping utility that cleans up unreachable or "orphaned" Git objects. Unreachable objects are those that are inaccessible by any refs. Any commit that cannot be accessed through a branch or tag is considered unreachable. git prune is generally not executed directly.

Repository Mirroring is a way to mirror repositories from external sources. It can be used to mirror all branches, tags, and commits that you have in your repository. Your mirror at GitLab will be updated automatically.

This command is used to combine all objects that do not currently reside in a "pack", into a pack. It can also be used to re-organize existing packs into a single, more efficient pack.

What are the git objects?

Ans)

Git stores content in a manner similar to a UNIX filesystem, but a bit simplified. All the content is stored as tree and blob objects, with trees corresponding to UNIX directory entries and blobs corresponding more or less to inodes or file contents.

There are 3 main types of objects that git stores:

Blob: This object as we have seen above stores the original content.

Tree: This object is used to store directories present in our project.

Commit: This object is created whenever a commit is made and abstracts all the information for that particular commit.

What is Version control system or Source code manager?

Ans)

Version control, also known as source control, is the practice of tracking and managing changes to software code. Version control systems are software tools that help software teams manage changes to source code over time.

Benefits of VCS or SCM?

Ans)

Traceability

Traceability is a mechanism that provides evidence of all revisions and changes made over a while. It enables users to identify the development of the file through its various stages. It tracks the contributions made by several developers. Tracking changes from the original copy to the many improved versions and, finally, to the final version. When a developer works on the latest draft, having the history of the document readily accessible enables the developer to understand the purpose of the dataset.

Document History

The history of the document provides invaluable information about the author and the date of editing. It also gives on the purpose of the changes made. It will have an impact on the developer that works on the latest version as it will help to solve problems experienced in earlier versions.

Branching And Merging

Version control allows team members to work on the same document concurrently but independent of each other without affecting the contribution of fellow collaborators. Each contributor works on an independent stream of change – commonly referred to as a branch.

Creating a branch enables each team member to work on the same project using multiple streams (branches). These are autonomous of each other. It empowers the team to merge their independent work.

Efficiency

Version control promotes an efficient progression of the document. Teams work to simplify complex processes and, thus, create greater scope for automation and consistency.

What is a Git Repository?

A Git repository tracks and saves the history of all changes made to the files in a Git project. It saves this data in a directory called .git, also known as the repository folder. (remote repo)

Git uses a version control system to track all changes made to the project and save them in the repository. Users can then delete or copy existing repositories or create new ones for ongoing projects.

Difference between Git and SVN?

Ans)

Below is a table of differences between GIT and SVN:

GIT, SVN

Git is open source distributed vice control system developed by Linus Torvalds in 2005. It emphasis on speed and data integrity, Apache Subversion is an open source software version and revision control system under Apache license.

Git has a Distributed Model., SVN has a Centralized Model.

In git every user has their own copy of code on their local like their own branch., In SVN there is central repository has working copy that also make changes and committed in central repository.

In git we do not required any Network to perform git operation. , In SVN we required Network for runs the SVN operation.

Git is more difficult to learn. It has more concepts and commands. , SVN is much easier to learn as compared to git.

In git we create only .git directory., In SVN we create .svn directory in each folder.

It does not not have good UI as compared to SVN., SVN has simple and better user interface . Git is complex for beginners

What are the two types of git authentication?

Ans)

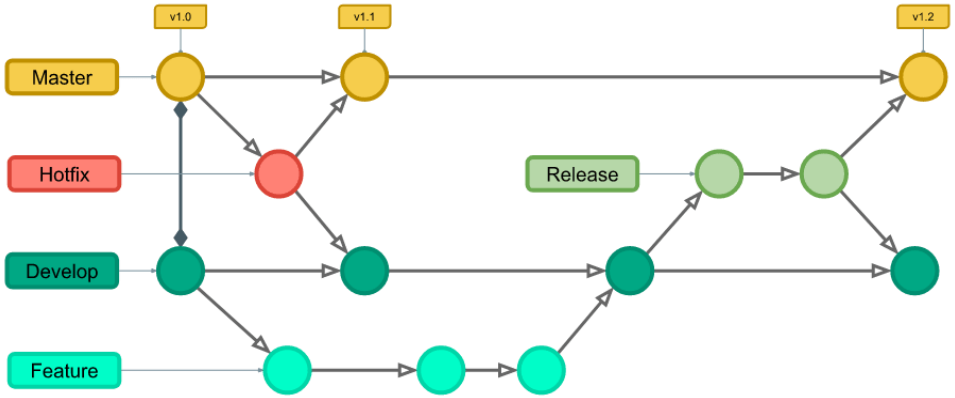
Git supports two types of remotes: SSH and HTTPS. These two use completely distinct authentication systems. For HTTPS remotes, git authenticates with a username + password. With GitHub, instead of a password, you can also use a Personal Access Token (PAT).

For ssh it's sshkeygen

What is the common branching pattern in GIT?

Ans)

GIT FLOW



Master

This is production code - simple as that. The big change that git flow adds to the mix is that you don't actually code ever on master. Everything here is done via merging or pull requests. This ensures that your code has gone through the process and is ready to go!

Hotfix

Need to fix production? Easy, start a hotfix branch and code away. When you're done something different happens, a double merge - one into master and one into develop, ensuring that the bug you just squashed does not come back!

Develop

This branch-like master is not committed on directly, it is a direct copy of the master branch when you enable it though. This is a one time thing, so keeping the pattern will ensure that the branches stay synced together. Merges will happen to it from both hotfix and feature branches - develop acts as an integration branch to that extent. The beauty of this is that all of the code comes together before a deployment, saving you time and lots of unneeded stress.

Feature

This is the closest to normal git branching as you will get in git flow. Essentially, it's a place to try out and implement new functionality. This is great, because you can have virtually unlimited features in progress and never have to worry about them breaking the main branches. On that note, you really want to start and end a feature as quickly as possible to avoid merge conflicts, because no one like them.

Release

This is one of the greatest built-in features of the git flow pattern. This gives you a place to start staging your release to production. While that is awesome on its own, it like hotfix has a double merge at the end of it. This means that if a bug or other issue is found during a release you can still fix it and have that code automatically flow into both your master and develop branches.

Overall

While git flow will not solve all of the problems that you face in a day, it can help you with a simple pattern for branching. There are a lot of add-ons out there, which make use of it, and now you can as well. Don't worry too much though if you cannot use a built-in git flow tool either.

What does pull request mean?

A pull request – also referred to as a merge request – is an event that takes place in software development when a contributor/developer is ready to begin the process of merging new code changes with the main project repository.

How to setup repository though command line?

Ans)

Gitinit

What is git clone command used for?

Ans)

git clone is primarily used to point to an existing repo and make a clone or copy of that repo at in a new directory, at another location. The original repository can be located on the local filesystem or on remote machine accessible supported protocols. The git clone command copies an existing Git repository.

What is git config command used for?

Ans)

The git config command is a convenience function that is used to set Git configuration values on a global or local project level. These configuration levels correspond to . gitconfig text files. Executing git config will modify a configuration text file.

Git config data is stored in what location?

Ans)

On Linux, the config file will remain in the /etc/gitconfig .

Content of git config file?

Ans)

The Git configuration file contains a number of variables that affect the Git commands' behavior. The . git/config file in each repository is used to store the configuration for that repository, and $HOME/. gitconfig is used to store a per-user configuration as fallback values for the . It contains the username and password credentials of the remote repo we like to push the code into.

Git add command's purpose?

Ans)

The git add command adds a change in the working directory to the staging area.

How to remove/rename files in local git repo?

Ans)

The easiest way to delete a file in your Git repository is to execute the “git rm” command and to specify the file to be deleted. Note that by using the “git rm” command, the file will also be deleted from the filesystem.

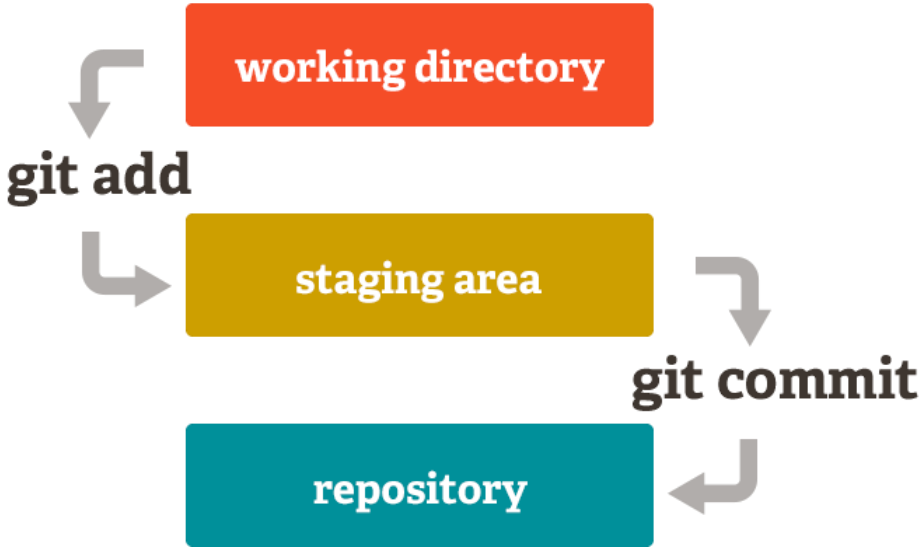
git mv old_filename new_filename

git status

$ git commit -m "Rename file"
 # Commits the tracked changes and prepares them to be pushed to a remote repository.

What is the git commit command purpose?

Ans) **git commit –m “first commit”**



How to sync local git repo data with GitHub?

Ans)

WORKING AREA **STAGING AREA** **LOCAL REPO**

FILE

USE COMMAND GIT ADD . TO PUSH THE FILE TO STAGING AREA

WORKING AREA **STAGING AREA** **LOCAL REPO**

FILE

THEN USE GIT COMMIT -M "SOME COMMENT" TO PUSH TO LOCAL REPO

ORKING AREA **STAGING AREA** **LOCAL REPO**

FILE

BEFORE DOING THIS FIRST CONFIG YOUR GITHUB DETAILS IN THE GLOBAL CONFIG

GIT CONFIG --GLOBAL USER.EMAIL "SRIKAR.AWESOME@GMAIL.COM"

GIT CONFIG --GLOBAL USER.NAME "SRIKAR19953"

THEN WE HAVE TO ADD THE GITHUB REPOSITORY TO HERE , CAN ADD ANY YOUR REPO

GIT REMOTE ADD ALIAS-NAME HTTPS://GITHUB.COM/SRIKAR19953/FB.GIT (HTTPS WITH PASSWORD CONNECTION)

GIT REMOTE –V (to view the aliases present)

GIT PUSH ALIAS NAME BRANCH NAME

#IT WILL ASK FOR USERNAME AND PASSWORD , GENERATE A PAT ! GHP_JSG6V0N16VISYAZZVOXAQZFW3RI2Q40JN3FX - PAT GIT (here PAT token is password)

GIT PUSH SOMETIMES WON'T WORK WE HAVE TO FORCE IT !! REFERENCE - HTTPS://STACKOVERFLOW.COM/QUESTIONS/28429819/REJECTED-MASTER-MASTER-FETCH-FIRST

GIT PUSH ALIAS NAME MASTER –FORCE

(OR)

#sometimes git wont allow to push the code to remote repo then try these commands before pushing

git fetch upstream

```
git merge upstream/master
```

How to change branches in local git repo?

Ans) **git checkout (branch name)**

Git status?

Ans)

The git status command is used to display the state of the repository and staging area. It allows us to see the tracked, untracked files and changes. This command will not show any commit records or information. Mostly, it is used to display the state between Git Add and Git commit command.

What is the function of ‘git rm’?

Ans)

git rm can be used to remove files from both the staging index and the working directory.

What is the use of ‘git log’?

Ans)

Generally, the git log is a record of commits.

Explain what is commit message?

Ans)

A commit in GitHub is described as a saved change. A commit message explains what change you made to your project. It is greatly important to learn how to make a good commit message no matter if it is a personal or professional project.

How to setup github ssh authentication?

Ans)

ADDING SSH KEY WITH GITHUB / PASSWORD-LESS CONNECTION

ls -la ~/.ssh #to check if theres an ssh key alrdy if not then
ssh-keygen #to generate the ssh key , once generated then
cat ~/.ssh/id_rsa.pub #copy the contents of the key completely then open github settings ssh to paste it over there after that
ssh -T git@github.com #succesfully authenticated to github ssh

What is the difference between "git checkout " and "svn checkout " commands?

Ans)

Git checkout

git checkout refers to the action of swaping between different repository branches/files/commits. It helps in switching between different branches that have been created by git branch. It can be understood as the method of selecting the current line of development one has to work on.

Svn checkout

The Checkout command is used to copy the files from the SVN repository to the working copy. If we want to access files from the SVN server, then we have to check out it first. The checkout operation creates a working copy of the repository where we can edit, delete, or add contents.

What is the difference between ‘git remote’ and ‘git clone’?

Ans)

They are two completely different things. git remote is used to refer to a remote repository or your central repository. git clone is used to copy or clone a different repository.

Advantages of git over with svn?

Ans)

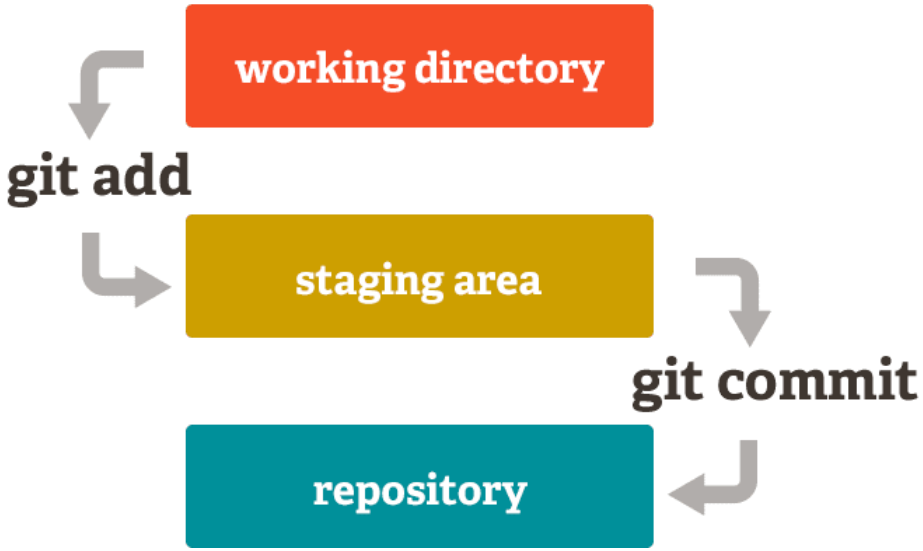
Git is a distributed version control system. That means that there's no requirement for a central server, though in practice there usually is a central server. It's faster to commit. Because you commit to the central repository more often in SVN, network traffic slows everyone down. Whereas with Git, you're working mostly on your local repository and only committing to the central repository every so often.

Git supports offline work, meaning that you can do all of the local operations (saving work, reverting work, changing branches, etc) without needing to be connected to a central server. Back In The Day when internet connectivity wasn't quite as ubiquitous as it is now, that was a bigger deal, but it's still a very nice feature. It also contributes significantly to the other two advantages.

No more single point of failure. With SVN, if the central repository goes down or some code breaks the build, no other developers can commit their code until the repository is fixed. With Git, each developer has their own repository, so it doesn't matter if the central repository is broken. Developers can continue to commit code locally until the central repository has been fixed, and then they can push their changes.

Explain about staging area in git? Ans)

The staging area is like a rough draft space, it's where you can git add the version of a file or multiple files that you want to save in your next commit



Pull request in git hub and git pull differences?

Ans)

Git Pull

Remote Repo======>local repo

git pull=Git Fetch + Git Merge

Pull Request

It's a Github thing.

Remote Github Repo<========Pull Request from=====Your Github Repo

Whether the maintainer of the remote github repo will accept your pull request or not, is on her.

Forking a repository in git hub?

Ans)

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

What Is the Git HEAD?

Head is the most recent commit in your working branch.

Before understanding how resets work, we need to talk about the Git HEAD.

“HEAD” is simply an alias for your current working commit. You can think of it like a record player head, where the position of it determines what data is being used. If you’re currently using the master branch, the HEAD will be at the latest commit in that branch.

Despite being an alias for a commit, the HEAD doesn’t actually point directly to a commit most of the time. It points towards a branch, and automatically uses the latest commit. It can also point towards a commit directly, which is known as “detached HEAD,” though that doesn’t matter for git reset.

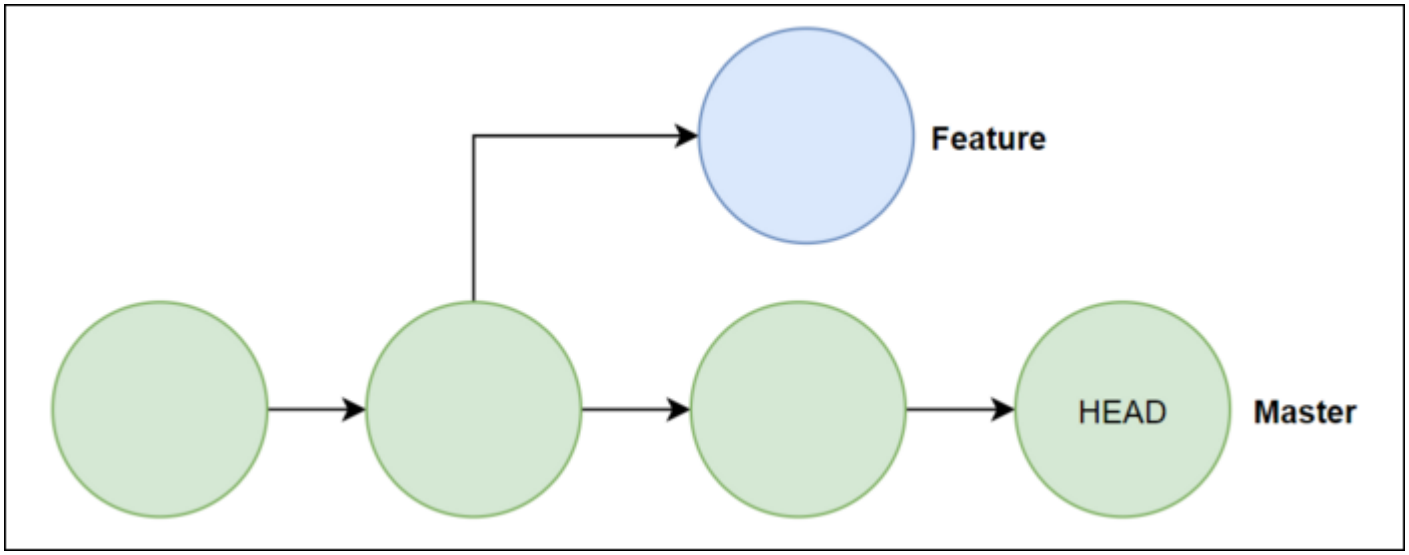
How Does Git Reset Work?

Ref link - https://www.howtogeek.com/devops/how-does-git-reset-actually-work-soft-hard-and-mixed-resets-explained/

https://medium.com/@keshshen/the-difference-between-git-reset-mixed-soft-and-hard-299704b12a7a

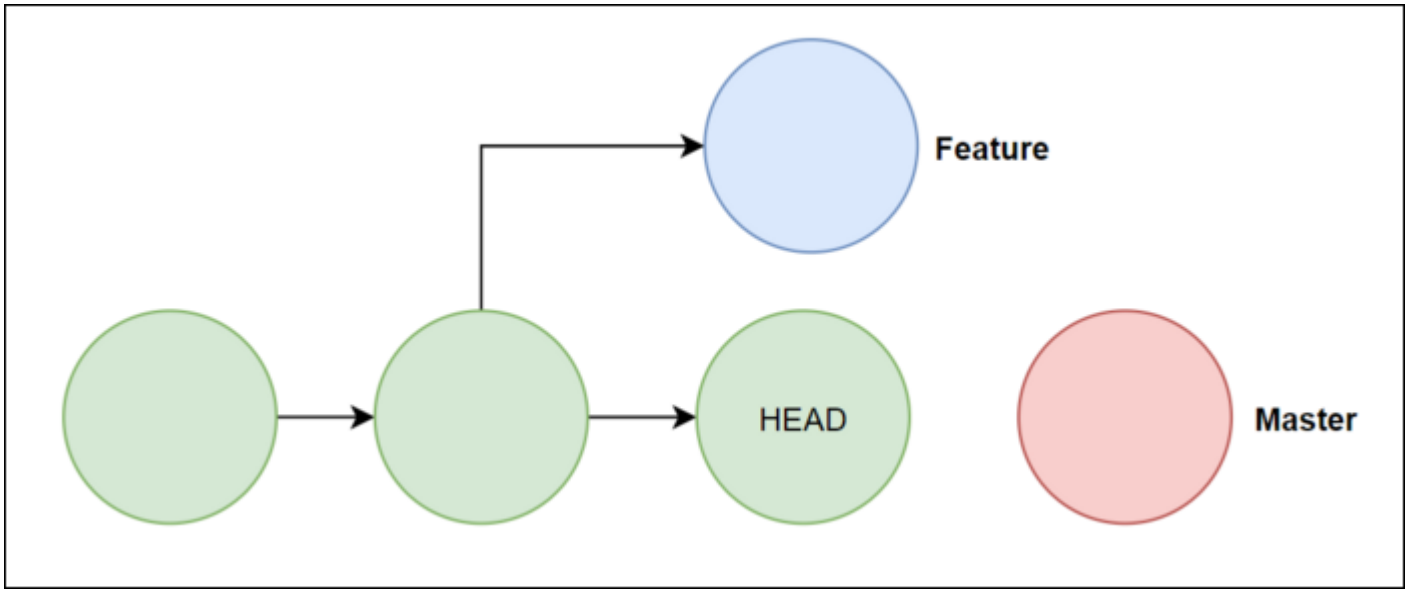
Git's commit history is stored as a tree of commits, and is intended to be immutable for the most part. However, sometimes it's necessary to modify this history, and that's where git reset comes into play.

Each commit links to the commit before it, and can branch off into different limbs that will eventually get merged back into the master branch. In either case, the HEAD will usually point towards the latest commit in whatever branch you're working on.



So what happens when you make a commit you want to revert?

Well, running `git reset` basically moves the HEAD back, and leaves all the commits in front of it hanging. This rewrites the usually immutable Git history to get rid of the commits in front of the HEAD.



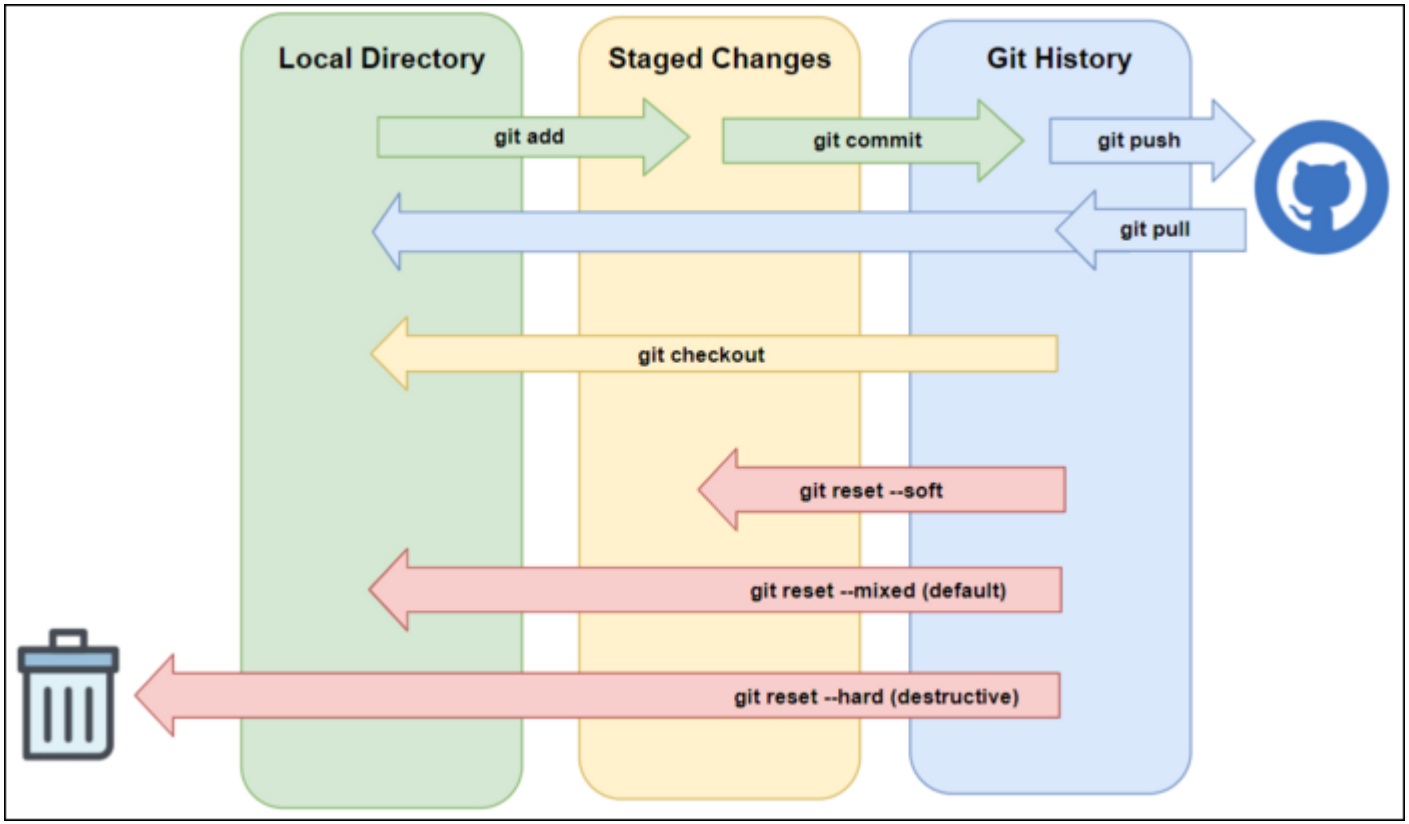
This can be quite useful for a lot of reasons. Perhaps you made a commit, then made some additional changes, and wanted to push the whole thing as one commit. You could `git reset` back to the previous commit, and then re-commit correctly.

Or perhaps you accidentally made a commit that included some changes you didn't want to track. This can be very hard to figure out if you don't know how `git reset` works, but resetting the HEAD and then only staging the correct changes will achieve this. Note that this is different from `git revert`, which reverses commits.

There are three different kinds of resets in Git, and they all differ based on how they handle the commits that get left hanging. They all rewrite Git history, and they all move the HEAD back, but they deal with the changes differently:

- **`git reset --soft`, which will keep your files, and stage all changes back automatically.**
- **`git reset --hard`, which will *completely destroy any changes and remove them from the local directory*. Only use this if you know what you're doing.**
- **`git reset --mixed`, which is the default, and keeps all files the same but unstages the changes. This is the most flexible option, but despite the name, it doesn't modify files.**

The difference between soft and mixed is whether or not the changes are staged. Staged is basically an in-between zone between the local directory and Git history. `git add` stages files, and `git commit` writes them to the history. In either case, the local directory is unaffected, it just changes the state of Git's tracking of those changes.



Basically, Soft and Mixed resets are mostly the same and allow you to keep the changes, and Hard resets will completely set your local directory back to where it was at the time of the commit.

Tags vs Branches

Both tags and branches point to a commit, they are thus aliases for a specific hash and will save you time by not requiring to type in a hash.

The difference between tags and branches are that a branch always points to the top of a development line and will change when a new commit is pushed whereas a tag will not change. Thus tags are more useful to "tag" a specific version and the tag will then always stay on that version and usually not be changed.