

1] Basics of Git ?

Git is a version control system.

Git helps you keep track of code changes.

Git is used to collaborate on code.

2] How to check git version and which version you used in your project?

C:\Users\qqq>git --version

git version 2.25.0.windows.1

**3 ] How do you create a Repository in Git?**

**Answer:** To create a repository, you need to create a directory for the project if it does not already exist, and then simply execute the command “**git init**”. By executing this command, a .git directory will be created inside the project directory i.e. now your project directory has turned into a Git repository.

**4 ] What are the advantages/main features of Git?**

**Answer: Enlisted below are the various features of Git.**

**(i) Free & Open Source:**

Git is issued under GPL’s (General Public License) open source license. You need not pay anything to use Git.

It is absolutely free. As it is open-source, you can modify the source code according to your needs.

**(ii) Speed:**

As you are not required to connect to any network for executing all the actions, it performs all the tasks quickly. Obtaining version history from a locally stored repository can be one hundred times speedier than obtaining it from the remote server.

**(iii) Scalable:**

Git is highly scalable. So, if the number of collaborators increases in the coming time, then Git can easily accommodate this change.

Despite the fact that Git represents an entire repository, the data kept on the client’s side is very small as Git compacts the entire vast data through a lossless compression technique.

**(iv) Reliable:**

As every collaborator has its own local repository, on the instances of a system crash, the lost data can be recuperated from any of the local repositories. At all times, you will have a backup of all your files.

**(v) Secure:**

Git utilizes the SHA1 (Secure Hash Function) to name and identify objects inside its repository. Each artifact and commit are check-summed and recovered through its checksum during checkout.

The Git history is saved in a manner in which the ID of a specific version (a commit in terms of Git) relies on the total development history running up to that commit. Once a file version is pushed to Git, then there is no way to change it without being noticed.

**(vi) Economical:**

In the case of a centralized version control system, the central server must be strong enough to attend requests of the entire team. This is not a problem for smaller teams, however as the team expands, the hardware limitations of the server can be an impediment for performance.

In the case of distributed version control systems like Git, the team members don’t require interaction with the server expect when they are required to push or pull changes. All the heavy lifting occurs at the client end, thus the server hardware can be kept quite simple certainly.

**(vii) Supports Non-linear Development:**

Git provides rapid branching & merging and contains particular tools for envisaging and traversing a non-linear development history. A basic notion in Git is that a change will be merged more frequently than it is written as it is sent across different reviewers.

Git Branches are extremely lightweight. A branch in Git refers only to a single commit. The complete branch structure can be created, with the help of parent commits.

**(viii) Easy Branching:**

Branch management through Git is very straightforward and easy. It requires just a few jiffies to create, delete, and merge branches. Feature branches give an insulated environment to each change to your codebase.

When a developer requires to begin working on something, irrespective of the size of work, they create a new branch. This makes sure that the master branch constantly holds a production-quality code.

**(ix) Distributed Development:**

Git provides every developer a local copy of the whole development history, plus the changes get cloned from one such repository to another. These changes are introduced as added development branches and can be merged in the same manner as a locally developed branch.

**(x) Compatibility along with present Systems or Protocol:**

Repositories can be published through HTTP, FTP or a Git protocol on top of either a plain socket or ssh.

### **5 ] What is a Git repository?**

[Git repository](https://www.simplilearn.com/git-tutorial-article) refers to a place where all the Git files are stored. These files can either be stored on the local repository or on the remote repository.



### 6 ] **How can you initialize a repository in Git?**

If you want to initialize an empty repository to a directory in Git, you need to enter the “**git init”**command. After this command, a hidden **.git** folder will appear in the folder.

7 ] Name a few Git commands with their function.

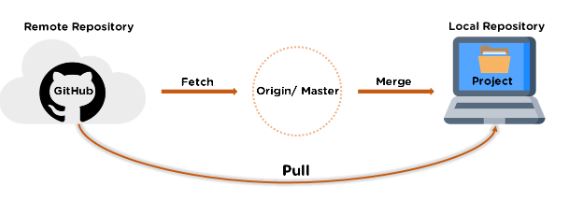
* Git config - Configure the username and email address
* Git add - Add one or more files to the staging area
* Git diff - View the changes made to the file
* Git init - Initialize an empty Git repository
* Git commit - Commit changes to head but not to the remote repository

### **8 ]  Explain the git push command.**

The [Git push command](https://www.simplilearn.com/tutorials/git-tutorial/git-push-command" \t "_blank) is used to push the content in a local repository to a remote repository. After a local repository has been modified, a push is executed to share the modifications with remote team members.

9 ] Explain the git pull command.

Git pull is used to fetch and merge changes from the remote repository to the local repository. Git pull is a combination of two commands: git fetch; followed by git merge.



13. Difference between git fetch and git pull.

|  |  |
| --- | --- |
| Git fetch | Git pull |
| * Git fetches only downloads new data from a remote repository. * It does not integrate any of these new data into your working files. * Can be done any time to update the remote-tracking branches   Command - git fetch origin                      git fetch –-all | * Git pull updates the current HEAD branch with the latest changes from the remote server. * Downloads new data and integrate it with the current working files. * Tries to merge remote changes with your local ones.   Command - git pull origin master |