GIT COMMANDS

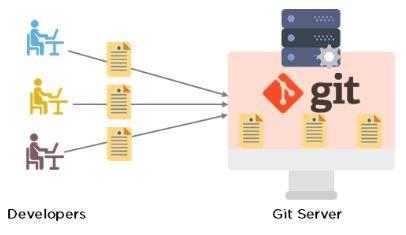
**What is Git?**

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

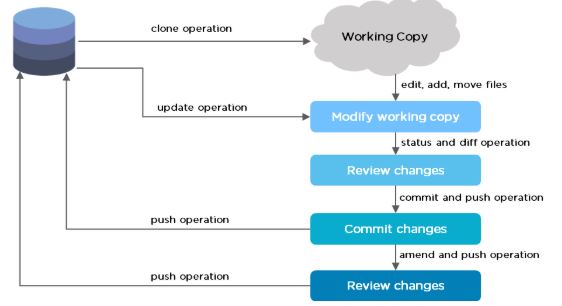
* Git is used to tracking changes in the source code
* It allows multiple developers to work together
* It supports non-linear development through its thousands of parallel branches

**Features of Git?**

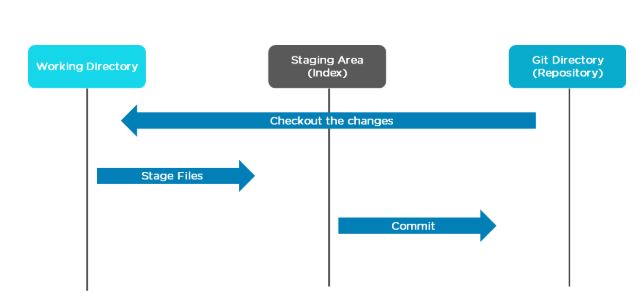
* Tracks history
* Free and open source
* Supports non-linear development
* Creates backups
* Scalable
* Supports collaboration
* Branching is easier
* Distributed development



Git Workflow

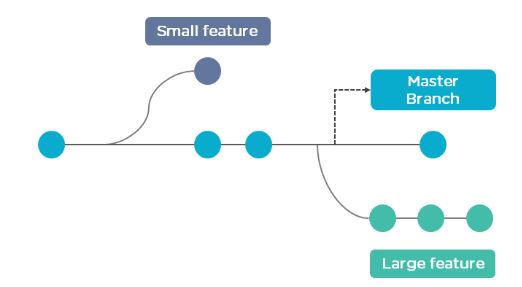


The Git workflow is divided into three states:

* Working directory - Modify files in your working directory
* Staging area (Index) - Stage the files and add snapshots of them to your staging area
* Git directory (Repository) - Perform a commit that stores the snapshots permanently to your Git directory. Checkout any existing version, make changes, stage them and commit.

**Branch in Git**

* Branch in Git is used to keep your changes until they are ready. You can do your work on a branch while the main branch (master) remains stable. After you are done with your work, you can merge it with the main office.



* The above diagram shows there is a master branch. There are two separate branches called “small feature” and “large feature.” Once you are finished working with the two separate branches, you can merge them and create a master branch.

**Commands in Git**

* Create Repositories  
  git init
* Make Changes  
  add  
  commit  
  status
* Parallel Development  
  branch  
  merge  
  rebase
* Sync Repositories  
  push  
  pull  
  add origin

**Git Commands**

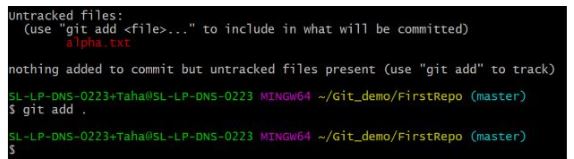
* git init
* The command git init is used to create an empty Git repository.
* After the git init command is used, a .git folder is created in the directory with some subdirectories. Once the repository is initialized, the process of creating other files begins.

|  |
| --- |
| git init |



* git add
* Add command is used after checking the status of the files, to add those files to the staging area.
* Before running the commit command, "git add" is used to add any new or modified files.

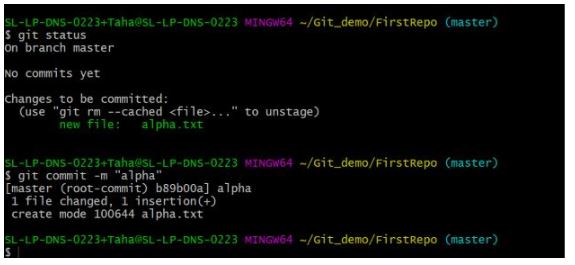
|  |
| --- |
| git add . |



git commit

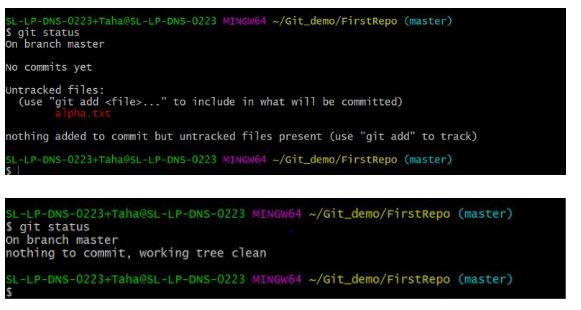
* The commit command makes sure that the changes are saved to the local repository.
* The command "git commit –m <message>" allows you to describe everyone and help them understand what has happened.

|  |
| --- |
| git commit -m “commit message” |



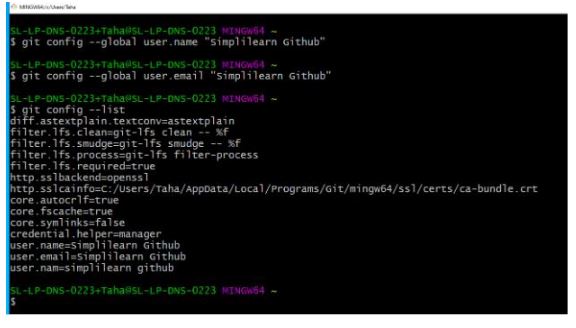
* git status
* The git status command tells the current state of the repository.
* The command provides the current working branch. If the files are in the staging area, but not committed, it will be shown by the git status. Also, if there are no changes, it will show the message no changes to commit, working directory clean.

|  |
| --- |
| git status |



* git config
* The git config command is used initially to configure the user.name and user.email. This specifies what email id and username will be used from a local repository.
* When git config is used with --global flag, it writes the settings to all repositories on the computer.

|  |
| --- |
| git config --global user.name “any user name”  git config --global user.email <email id> |



* git branch
* The git branch command is used to determine what branch the local repository is on.
* The command enables adding and deleting a branch.

|  |
| --- |
| # Create a new branch   git branch <branch\_name> |
| # List all remote or local branches   git branch -a |
| # Delete a branch   git branch -d <branch\_name> |

* git checkout
* The git checkout command is used to switch branches, whenever the work is to be started on a different branch.
* The command works on three separate entities: files, commits, and branches.

|  |
| --- |
| # Checkout an existing branch   git checkout <branch\_name> |
| # Checkout and create a new branch with that name   git checkout -b <new\_branch> |

* git merge
* The [git merge](https://www.simplilearn.com/tutorials/git-tutorial/merge-conflicts-in-git) command is used to integrate the branches together. The command combines the changes from one branch to another branch.
* It is used to merge the changes in the staging branch to the stable branch.

|  |
| --- |
| git merge <branch\_name> |

However, these are popular and basic git commands used by developers.

Git Commands: Working With Remote Repositories

* git remote
* The git remote command is used to create, view, and delete connections to other repositories.
* The connections here are not like direct links into other repositories, but as bookmarks that serve as convenient names to be used as a reference.

|  |
| --- |
| git remote add origin <address> |

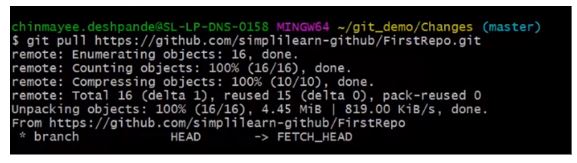


* git clone
* The git clone command is used to create a local working copy of an existing remote repository.
* The command downloads the remote repository to the computer. It is equivalent to the Git init command when working with a remote repository.

|  |
| --- |
| git clone <remote\_URL> |

* git pull
* The [git pull command](https://www.simplilearn.com/tutorials/git-tutorial/git-pull-request) is used to fetch and merge changes from the remote repository to the local repository.
* The command "git pull origin master" copies all the files from the master branch of the remote repository to the local repository.

|  |
| --- |
| git pull <branch\_name> <remote URL> |



* git push
* The command [git push](https://www.simplilearn.com/tutorials/git-tutorial/git-push-command) is used to transfer the commits or pushing the content from the local repository to the remote repository.
* The command is used after a local repository has been modified, and the modifications are to be shared with the remote team members.

|  |
| --- |
| git push -u origin master |

Some Advanced Git Commands

* git stash
* The git stash command takes your modified tracked files and saves it on a pile of incomplete changes that you can reapply at any time. To go back to work, you can use the stash pop.
* The git stash command will help a developer switch branches to work on something else without committing to incomplete work.

|  |
| --- |
| # Store current work with untracked files   git stash -u |
| # Bring stashed work back to the working directory   git stash pop |

* git log
* The git log command shows the order of the commit history for a repository.
* The command helps in understanding the state of the current branch by showing the commits that lead to this state.

|  |
| --- |
| git log |



**Basic Git Interview Questions and Answers**

**1. What is Git?**

Git is a decentralized version control system engineered for rapid and effective management of projects ranging from modest to vast in scale. It supports collaborative efforts among developers by monitoring file modifications and enhancing teamwork.

**2. What is a repository in Git?**

A [Git repository](https://www.simplilearn.com/tutorials/git-tutorial/what-is-a-git-repository) is a location where your project resides, acting as a storage area. This repository can exist locally within a directory on your computer or on a cloud-based platform like GitHub. It encompasses all the files associated with the project, along with a record of the modifications made to these files over time.

**3. What is the difference between Git and GitHub?**

Git is a tool for version control that enables you to monitor and record the evolution of your source code. GitHub is an online hosting service that facilitates the management of Git repositories. GitHub offers a user-friendly web interface along with functionalities such as permission management, task organization, error tracking, and the capability to handle feature suggestions.

**4. How does Git work?**

Git works by taking snapshots of a project's files. Unlike other version control systems, Git records the entire contents of each file and its changes every time a commit is made. This makes operations like branching, merging, and reverting changes more efficient.

**5. What is a commit in Git?**

In Git, a commit is a process that records a version of the project's presently prepared modifications. This record includes details on the modifications implemented, a distinctive identifier (a SHA-1 hash), the creator's identity, and the timestamp of the commit.

**6. What is branching in Git?**

In Git, branching allows you to veer off from the primary development path and proceed with separate tasks without impacting the main workflow. This technique enables the isolated development of features, bug fixes, or experimentation within a specific section of the repository, ensuring that each process remains distinct from the others.

**7. What is a merge in Git?**

Merging is a Git operation that integrates changes from one branch into another. It can be a fast-forward merge, where the target branch is updated to the latest commit of the source branch, or a three-way merge, where divergent branch histories are combined into a new commit.

8. **What is a conflict in Git?**

A conflict in Git occurs when two branches have made edits to the same line in a file or when one branch deletes a file while the other branch modifies it. Git cannot automatically resolve these changes; the developer must manually resolve the conflicts.

**9. What is a pull request?**

A pull request serves as a mechanism for contributing to a project, typically utilized in projects hosted on GitHub. In this process, a developer implements modifications within their own branch, uploads these changes to a repository, and then initiates a pull request. This action prompts the project's maintainers to examine the proposed changes, engage in discussions about possible adjustments, and ultimately integrate the pull request into the primary branch.

**10. What is `git fetch` vs. `git pull`?**

`git fetch` downloads updates from a remote repository to your local repository without integrating them. On the other hand, `git pull` not only fetches the updates but also incorporates them into your active branch.