GIT

## 

## VERSION CONTROL SYSTEM (VCS)

* GIT was created by Linus Torvalds in 2005.
* Version Control System (VCS) is a fundamental tool for managing changes in a project over time. It enables collaboration, tracks modifications, and provides the ability to revert to previous versions.
* Git is one of the most widely used distributed version control systems.
* There are many features in GIT like it is open source, high speed, more secure and more scalable.
* GIT is distributed under GNU (general public license).

Git Basics

### Installation

1. Visit [git-scm.com](https://git-scm.com/) to install Git.
2. Open the command prompt and enter **git** to ensure successful installation.

### Git Repositories

* **Local Repository:** The version of the project on your machine.
* **Remote Repository:** A project folder hosted on a distant server, enabling collaborative development.

**Important terms:**

1. **WORKING DIRECTORY**

* The files which we created and not in the tracking status or untracked files will be in the area working directory**.**
* From this working directory, the files will be add to the staging area.
* Simply, Working directory is the working space where we store our files.
* Here we can select which file we want to push into remote repository.

**2.STAGING AREA:**

* **Staging area is a preview of our next commit**, moreover, 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.
* We are allowed to add and remove changes from the staging area.
* The staging area can be considered as a place where Git stores the changes.

|  |
| --- |
| 3.[Checkout](https://www.javatpoint.com/git-checkout) In Git, the term checkout is used for the act of switching between different versions of a target entity. The **git checkout** command is used to switch between branches in a repository. 4.[Cherry-Picking](https://www.javatpoint.com/git-cherry-pick) Cherry-picking in Git is meant to apply some commit from one branch into another branch. In case you made a mistake and committed a change into the wrong branch, but do not want to merge the whole branch. You can revert the commit and cherry-pick it on another branch. 5.[Clone](https://www.javatpoint.com/git-clone) The **git clone** is a Git command-line utility. It is used to make a copy of the target repository or clone it. If I want a local copy of my repository from GitHub, this tool allows creating a local copy of that repository on your local directory from the repository URL. 6.[Fetch](https://www.javatpoint.com/git-fetch) It is used to fetch branches and tags from one or more other repositories, along with the objects necessary to complete their histories. It updates the remote tracking branches. 7.[HEAD](https://www.javatpoint.com/git-head) HEAD is the representation of the last commit in the current checkout branch. We can think of the head like a current branch. When you switch branches with git checkout, the HEAD revision changes, and points the new branch. 8.[Index](https://www.javatpoint.com/git-index) The Git index is a staging area between the working directory and repository. It is used as the index to build up a set of changes that you want to commit together. 9.[Master](https://www.javatpoint.com/git-origin-master) Master is a naming convention for Git branch. It's a default branch of Git. After cloning a project from a remote server, the resulting local repository contains only a single local branch. This branch is called a "master" branch. It means that "master" is a repository's "default" branch. 10.[Merge](https://www.javatpoint.com/git-merge) Merging is a process to put a forked history back together. The git merge command facilitates you to take the data created by git branch and integrate them into a single branch. 11.[Origin](https://www.javatpoint.com/git-origin-master) In Git, "origin" is a reference to the remote repository from a project was initially cloned. More precisely, it is used instead of that original repository URL to make referencing much easier. 12.[Pull/Pull Request](https://www.javatpoint.com/git-pull) The term Pull is used to receive data from GitHub. It fetches and merges changes on the remote server to your working directory. The **git pull command** is used to make a Git pull.  Pull requests are a process for a developer to notify team members that they have completed a feature. Once their feature branch is ready, the developer files a pull request via their remote server account. Pull request announces all the team members that they need to review the code and merge it into the master branch. 13.[Push](https://www.javatpoint.com/git-push) The push term refers to upload local repository content to a remote repository. Pushing is an act of transfer commits from your local repository to a remote repository. Pushing is capable of overwriting changes; caution should be taken when pushing. 14.[Remote](https://www.javatpoint.com/git-remote) In Git, the term remote is concerned with the remote repository. It is a shared repository that all team members use to exchange their changes. A remote repository is stored on a code hosting service like an internal server, GitHub, Subversion and more.  In case of a local repository, a remote typically does not provide a file tree of the project's current state, as an alternative it only consists of the .git versioning data. 15.[Repository](https://www.javatpoint.com/git-repository) In Git, Repository is like a data structure used by VCS to store metadata for a set of files and directories. It contains the collection of the file as well as the history of changes made to those files. Repositories in Git is considered as your project folder. A repository has all the project-related data. Distinct projects have distinct repositories. 16.[Stashing](https://www.javatpoint.com/git-stash) Sometimes you want to switch the branches, but you are working on an incomplete part of your current project. You don't want to make a commit of half-done work. Git stashing allows you to do so. The **git stash command** enables you to switch branch without committing the current branch. 17.[Git Rm](https://www.javatpoint.com/git-rm) In Git, the term rm stands for **remove**. It is used to remove individual files or a collection of files. The key function of git rm is to remove tracked files from the Git index. Additionally, it can be used to remove files from both the working directory and staging index. |

**Git Workflow**

1. Initialize a new Git repository:

git init

1. Check the status of your repository:

git status

1. Add changes to the staging area:

git add <file name>

git add <file>

1. Commit changes to the local repository:

git commit -m “commit message”.

git commit -

1. Push changes to a remote repository:

git push origin <branch>

git push origin <branch>

### Branching and Merging

* **Branch:** A parallel version of a repository, allowing for isolated development.
* **The default branches of the git are main/master.**
* **Every branch has a copy of main/master project.**
* Create a new branch:

git checkout -b <branch name>

git checkout -b <branch-name>

* Switch between branches:

git checkout <branch name>

git checkout <branch-name>

* Merge changes from one branch into another:

git merge <branch name>

git merge <branch-name>

### GitHub Collaboration

GitHub is a popular platform for hosting Git repositories and offers additional collaboration features such as bug tracking, feature requests, task management for every project. Git hub offers both distributed version control and source code management.

1. Create a remote repository on GitHub.
2. Link the local repository to the remote repository:

git remote add origin <repository-url>

3.Push changes to the remote repository for the first time:

git push -u origin main

git push -u origin main

### Pull Requests and Forking

* **Pull Request (PR):** Propose changes from one branch to another, facilitating code review.
* **Fork:** Create a personal copy of a repository for independent development.

1. Create a new branch for a feature or bug fix:

git checkout -b <feature-branch>

1. Resolve merge conflicts if needed:

git merge origin/main

git merge origin/main

1. Push the changes to the remote repository:

git push origin <feature-branch>

git push origin <feature-branch>

1. Open a Pull Request on GitHub.
2. Fetching changes from remote repository:

git fetch

1. Pulling changes from remote repository:

git pull

**Additional Tips**

* **Readme (README.md):** Provide essential information about the project.
* **git ignore:** Specify files or directories to be ignored by Git.
* **SSH Keys:** Ensure secure authentication for remote repository access.

Understanding these Git and GitHub basics empowers effective version control and collaborative development.