**GIT:** Global information tracker

It was created by Linus Torvalds in 2005, and has been maintained by Junio Hamano

* Git is a version control system.
* Git helps you keep track of code changes.
* Git having repository
* Git is a distributed version control system (DVCS) that allows developers to track changes in their codebase, Collaborate with others, and manage different versions of their projects.
* It can record all the modifications done by timestamp.
* Git is a distributed version control system .

**What does Git do**

* Manage projects with Repositories
* Clone a project to work on a local copy
* Control and track changes with Staging and Committing
* Branch and merge to allow for work on different parts and versions of a projects
* Pull the latest version of the project to a local copy
* Push local updates to the main projects.

**Type of Version Control System:**

1. **Local Version control system**

* It is one of the simplest forms and has a database that kept all the changes to files under the version control

1. **Centralized Version Control System:**

* It contains current data only.
* Centralized version control system contain just one repository globally and every user need to commit for reflecting one’s changes in the repository
* It is possible for others to see your changes by updating

Two things are required to make your changes visible to others

**You commit**

**They update**

**Advantages of Centralized Version Control System**

* Easier to set up and manage for small teams.
* Better Access Control: Administrators can restrict access to different parts of the repository.
* Consistent Data Management: Since everything is centralized, it's easier to enforce policies and track changes.
* Easier to Learn: Simpler workflows compared to distributed systems.

**Disadvantages of Centralized Version control System**

* Single Point of Failure: If the server crashes, all version history may be lost unless backups exist.
* Requires Internet Connection: Users need a connection to commit changes or retrieve versions.
* Limited Offline Work: Users cannot commit changes without server access.

1. **Distributed version control systems**

* Distributed version control systems contain multiple repositories
* Each user has their own repository and working copy.
* Just committing your changes will not give others access to your changes.
* This is because commit will reflect those changes in your local repository and you need to push them in order to make them visible on central repository. Similarly, When you update, you do not get other’s changes unless you have first pulled those changes into your repository.

To make your changes visible to others, 4 things are required

**You commit**

**You push**

**They pull**

**They update**

**Advantages of Distributed version control system**

* Users can commit, branch, and view history without an internet connection
* More Redundant & Secure: If the central server fails, any user’s copy can restore the repository.
* Encourages flexible development and parallel workflows.

**Disadvantages of Distributed version control system**

* Higher Storage Use: Since every user has a full copy of the repository, it requires more local storage.
* Requires more knowledge, especially for beginners, to manage branching and merging.
* Developers must push changes to a shared repository for collaboration.
* Access Control Challenges: Since users have full copies, controlling permissions is harder.

**Why Git**

* Over 70% of developers use Git
* Developers can work together from anywhere in the world
* Developers can see the full history of the projects
* Developers can revert to earlier version of a project

**Version Control System (VCS) vs. Collaboration**

|  |  |
| --- | --- |
| **Version Control System** | **Collaboration** |
| A system that tracks changes to files, enabling version history and rollback. | The process of multiple users working together on a project. |
| Helps manage changes, maintain history, and avoid conflicts in code or documents. | Enhances teamwork, communication, and coordination among team members. |
| Possible with **DVCS** (e.g., Git), where changes can be made locally and synced later. | Often requires real-time communication, though asynchronous collaboration is possible. |
| Uses merging, branching, and commit history to handle conflicting changes. | Requires team discussions, peer reviews, and decision-making to resolve disagreements. |
| VCS can set permissions on who can commit, merge, or modify files. | Collaboration tools (e.g., Slack, Google Docs) manage access through roles and permissions |
| Git, SVN, Mercurial | Git, SVN, Mercurial |