Git Usefulness

Git is a version control system. It maintains a history of all changes made to the code. The changes are stored in a special database called “repository”, also known as “repo”.

Two main advantages of using Git at software development:

* Tracking the changes and updates. We are able to see who made which changes. Git also provides when and why a change was made.
* Allowing to work collaboratively. Software development projects usually require many people to work together. Git provides the developers with a systematic way of doing that. Thus, the developers focus on the project instead of extensive communication sessions between the other developers.

When we make some changes on a file and want to add the updated file in the git repository.

The first step is to add the updated file to the staging area. After review, it is committed to the git repository.The word “commit” means a lot and it is a fundamental operation in Git. Git takes a snapshot of the project and stores it in the repository along with some other identifying information.Each time we commit, Git records the information above. Thus, we have a highly useful history of the project.

When we commit a file, we provide a message that describes the change. This message should be as informative as possible in order to build a decent history.

Note: The advantage of saving all these information for every change is that we can easily revert back to an earlier version in case of a mistake.

 When there are lots of changes, it is easier to record a snapshot instead of the changes. Besides, when going back to a previous version, it becomes very complicated to track down all the modifications.One issue with recording the snapshot of entire project might be the space. We don’t have to worry about it because Git is quite efficient in storing the snapshots. Git does not store the duplicate content and also compresses the content.

Also Git's open-source nature has given rise to a vast community of developers and users who actively contribute to its development. This collective effort has led to a wide range of tools, plugins, and integrations that enhance Git's functionality and make it even more versatile.

Branching Model: Git has a different branching model than the other VCS. Git branching model lets you have multiple local branches which are independent of each other. Having this also enables you to have friction-less context switching (switch back and forth to new commit, code and back), role-based code (a branch that always goes to production, another to testing etc) and disposable experimentation (try something out, if does not work, delete it without any loss of code).

Staging Area: Git has an intermediate stage called "index" or "staging area" where commits can be formatted and modified before completing the commit.

Distributed: Git is distributed in nature. Distributed means that the repository or the complete code base is mirrored onto the developer's system so that he can work on it only.

Open Source: This is a very important feature of any software present today. Being open source invites the developers from all over the world to contribute to the software and make it more and more powerful through features and additional plugins.

Git is not limited to just software development. It's also used in research and data science projects to track data analysis scripts, datasets, and research papers. This ensures transparency and reproducibility in scientific research.

While Git is incredibly useful, it's not without its challenges. Some common issues developers face include merge conflicts, large repositories, and difficulty in learning the system. To maximize its usefulness, best practices should be followed, such as maintaining concise commit messages, regularly pulling changes from the remote repository, and utilizing .gitignore files to prevent the tracking of unnecessary files.

Conclusion

Git has established itself as an essential tool for modern software development and many other fields. Its usefulness extends beyond version control to enable efficient collaboration, distributed development, and automation. By understanding its features, real-world applications, and best practices, developers can harness Git's full potential to streamline their work and enhance their productivity. As the software development landscape continues to evolve, Git remains a steadfast and invaluable resource for developers worldwide.