Juan Partida

11/10/2024

Assignment: Version Control Guidelines

Version control is essential in software development because it helps teams manage changes to their code over time. It lets you track revisions, work together smoothly, and record all the changes. So, learning about version control guidelines from reputable sources is essential, keeping all the most important ones in mind and applying them to whatever version control system you use.

One source I found, Version Control Concepts and Best Practices, by the University of Washington, highlights some vital version control practices. The page gives some insight into two types of version control: distributed and centralized. However, the guidelines apply to both kinds of version control types. One guideline is to use descriptive commit messages. Taking a moment to write a good commit message helps others understand the purpose of the change. It makes it easier to search for changes related to a specific concept, such as changes associated with a particular feature or bug. Descriptive commit messages make this process much more efficient. The article also notes another guideline to incorporate others' changes frequently. Always work with the most up-to-date version of the files by running git pull often. This tip helps avoid conflicts and saves time by ensuring your working copy includes others' changes before you start editing.

I also found an excellent article by Nulab, The Ultimate Guide to Document Version Control. This article also highlights the importance of committing descriptive messages and frequent commits. The article points out one should avoid making one large commit after hours or days of work. Smaller, frequent commits are easier to manage and help track when specific changes were made. Another guideline that the article emphasizes is the significance of using branching wisely. Having too many can clutter the repository. Keeping the number of branches low makes the repository clear and organized. It's easier to see what each branch is for and how it's progressing, which helps everyone work together better. It also reduces the chances of problems when merging changes, as fewer branches mean fewer conflicts.

Lastly, an article by Tower, Version Control Best Practices, gives some great insight into guidelines for using version control. The article also had guidelines similar to those of the other sources. Still, it highlights another crucial practice in version control, which is to always test your code before committing it. Thoroughly test it to ensure it works as intended and doesn't cause bugs and issues that affect the entire team, leading to potential disruptions and additional work. The article also asserts that version control is not a backup system. This guideline is vital because it can lead to messy and unorganized commits. When you treat it like a backup, you might commit large, unrelated changes all at once, making it harder to understand the history and purpose of each change. The primary purpose of a VCS is to manage changes to your codebase, track history, and facilitate collaboration among team members.

All these sources are great and similar except for a few differences in guidelines. However, all the guidelines I highlight in my essay are essential because they can help ensure everyone follows the same procedures and standards. Following these guidelines will make collaboration easier, assist in understanding the changes made, and maintain a clean project history. These guidelines are still helpful and can apply to people who use VCS. As projects and tools get more specific for a person or team, it will help to look for other sources that provide specialized guidance and best practices to ensure that you are using the most effective methods and staying up to date.

Sources:

[The ultimate guide to document version control | Nulab](https://nulab.com/learn/collaboration/document-version-control/)

[Version Control Best Practices | Tower Blog](https://www.git-tower.com/blog/version-control-best-practices/#:~:text=3.-,Don%27t%20Commit%20Half%2DDone%20Work,to%20commit%20early%20and%20often.)

[Version control concepts and best practices](https://homes.cs.washington.edu/~mernst/advice/version-control.html)