☰　　⊡(/learn)　　　　　　　　　　　　　　　　　　　⚙　📋

How Git Differs from Other Version Control Systems (VCS's)

Understand how Git is different from other Version Control Systems (VCS's).

We'll cover the following	⌃

- History is more malleable
- Branching is cheap
- Commits are made across the whole project
- No version numbers

If you're already familiar with other VCSes, Git has some key differences you should bear in mind as you learn about it.

History is more malleable

You can change the history in your own copy of the repository and others' assuming you have the appropriate permission to push to them.

Branching is cheap

In most traditional VCS's (i.e., CVS and Subversion), it's very slow to branch a repository ($O(n)$ (order n) to a number of files).

In Git, it's an $O(1)$ (an "order 1" step).

> **Order notation (e.g., $O(1)$)**
> Here, *O(1)* notation means that branching takes the same amount of time, regardless of the size of the repository that you're doing the branching in.

This makes experimentation with branching much easier. Branch deletion is also a common and cheap operation. This changes the typical workflow in many cases.

Commits are made *across the whole project*

In contrast to other source control tools, changes are made across the whole project but not per file.

One consequence of this is that moving/renaming files does not involve loss of history for that file. For example, this is a massive win over CVS, which is file-oriented.

No version numbers

Git does not automatically number versions of files/changes. Instead, it assigns a hash (effectively random) to the change to the repository (commits), which is then used to refer to those changes.

← **Back**

Next →

What about GitHub?

Introduction: Git Basics

✅ Mark as Completed

⊘ Report an Issue

⍰ Ask a Question (https://discuss.educative.io/tag/how-git-differs-from-other-version-control-systems-vcss__introduction-to-git__learn-git-the-hard-way)