Introduction to Git

Alessandro Sciarra

Z02 – Software Development Center

December 21, 2021

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

CRC-TR 211
Strong-interaction matter under extreme conditions

Outline of the talk

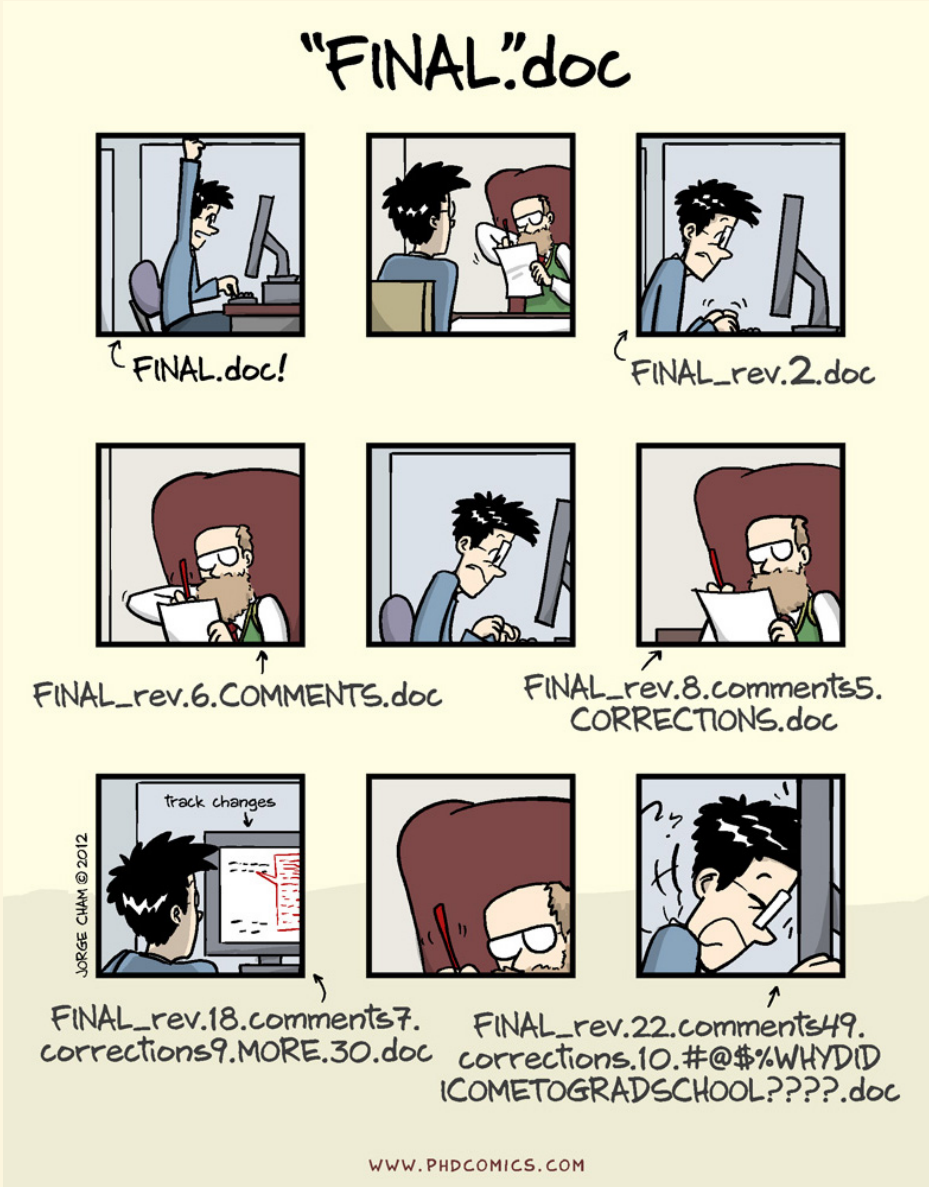
1. Why should you use it?
2. What is Git?
3. How does it work?

Why should you use it?

OK, let's do it without git

A possible workflow

Writing a review or a Ph.D. thesis



A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?

A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?
  - You make a backup of your file
  - You comment out a block of text in your source
  - If the old version was better, you restore by hand
  - If the new version is better, you clean by hand

A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?
- How do you create/view checkpoints?

A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?
- How do you create/view checkpoints?
  - Create a .tar or .zip file
  - Copy it somewhere and uncompress if needed

A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?
- How do you create/view checkpoints?
- Which version did you send to your supervisor/colleagues?

A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?
- How do you create/view checkpoints?
- Which version did you send to your supervisor/colleagues?
  - Put a copy of the PDF file or of the compressed folder somewhere
  - Keep the sent email for later use

A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?
- How do you create/view checkpoints?
- Which version did you send to your supervisor/colleagues?
- How long did it take to write this section?
- When did I start writing this chapter?
- How much did I write on average per day?

A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?
- How do you create/view checkpoints?
- Which version did you send to your supervisor/colleagues?
- How long did it take to write this section?
- When did I start writing this chapter?
- How much did I write on average per day?

Everything by hand, error-prone and big overhead!

Another possible workflow

Collaborating on a project

- How can you work on the same project?

Another possible workflow

Collaborating on a project

- How can you work on the same project?
  - You work on separate parts at the same time
  - Only one person works at the same time

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
  - You send the changed files per email and put them in the folder by hand
  - Copy/Rsync in some shared place the new status of the project
  - If only one person at once works, a compressed archive can be exchanged

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?
  - You don't, use `SSH`
  - Different machines are as different people, see above

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?
- How do you know who did what?

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?
- How do you know who did what?
  - This information is not important
  - Sending work around per email allows to trace this...
  - Put comments into the source!

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?
- How do you know who did what?
- How do you give credit to authors?

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?
- How do you know who did what?
- How do you give credit to authors?
  - Detailed information is not important
  - A rough idea about who worked on what is enough
  - See comments into the source!

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?
- How do you know who did what?
- How do you give credit to authors?
- How do you go back in history e.g. in case of a bug?

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?
- How do you know who did what?
- How do you give credit to authors?
- How do you go back in history e.g. in case of a bug?
  - Again, use the archive sent around per email
  - Using a shared place, this is not possible → debug!

Another possible workflow

Collaborating on a project

- How can you work on the same project?
- How do you merge work from other people in the team?
- How do you work on different machines?
- How do you know who did what?
- How do you give credit to authors?
- How do you go back in history e.g. in case of a bug?
  - Again, use the archive sent around per email
  - Using a shared place, this is not possible → debug!

OK, and how would it be with Git?

A possible workflow

Writing a review or a Ph.D. thesis

- How do you make writing experiments?
  - Just do them (staging/stash area)
  - `git-branch`
- How do you create/view checkpoints?
  - `git-log` `git-tag` `git-checkout`
- Which version did you send to your supervisor/colleagues?
  - `git-log` `git-tag`
- How long did it take to write this section?
- When did I start writing this chapter?
- How much did I write on average per day?

} `git-shortlog` `git-log` `gitstats`*

* This is just one of the pletora of libraries to make statistics based on a git repository.

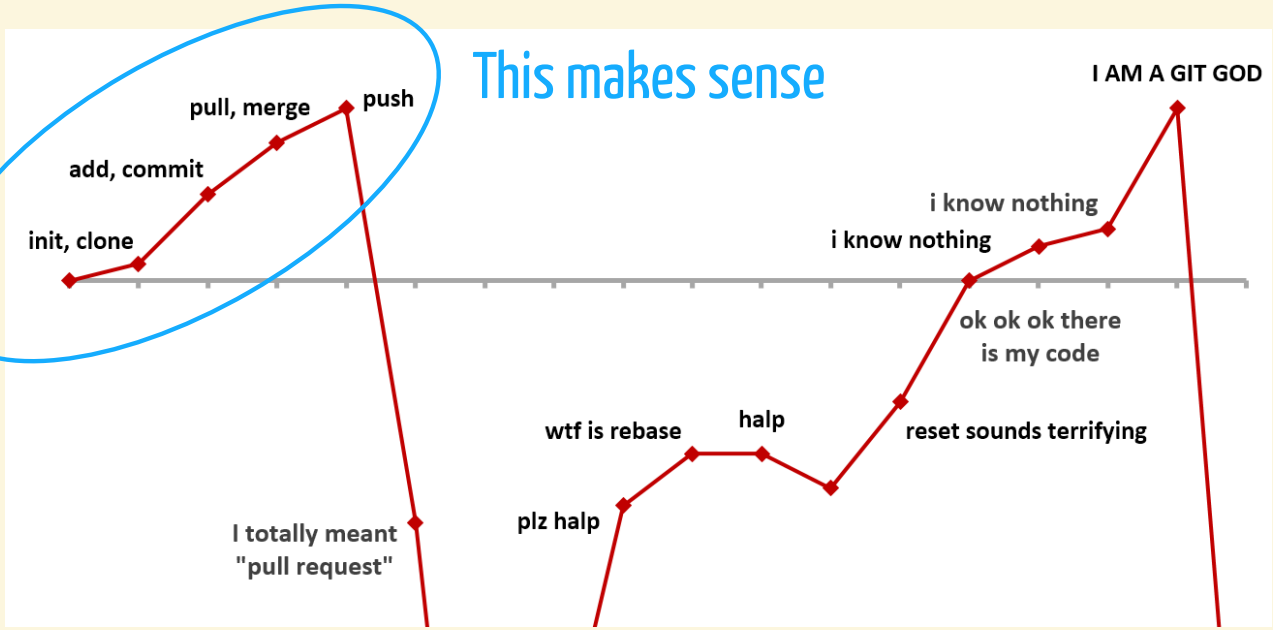
Another possible workflow

Collaborating on a project

- How can you work on the same project?
  - `git-pull` `git-push` `git-branch`
- How do you merge work from other people in the team?
  - `git-merge`
- How do you work on different machines?
  - `git-pull` `git-push`
- How do you know who did what?
  - `git-blame`
- How do you give credit to authors?
  - `git-shortlog`
- How do you go back in history e.g. in case of a bug?
  - `git-checkout` `git-bisect`

Yes, but I have to learn all those commands!

There are many jokes on the web...



...but after all it is about having the correct mental set up!

Git was not designed to be user-friendly, but it is by far the most used version control system in the world.

Yes, but I have to learn all those commands!

- As any new tool, it needs some practice
- The short- to long-term payoff is worth the effort
- It is plenty of 🔗GUI clients
  - 🔗Sourcetree: A Free GIT Client For Windows And Mac
  - 🔗Aurees: Easy-Fast-Free { Windows, Mac & Linux }
  - 🔗Git-Cola: Powerful GUI For GIT { Windows, Mac, Ubuntu & Linux }
  - [...]
- You can work in the terminal

  → after this (and next) talk it will be possible!

Git was not designed to be user-friendly, but it is by far the most used version control system in the world.

Last but not least

Which large famous products are developed using Git?

Linux, Homebrew, Windows, Tensorflow, Angular, Inkscape, ...

Last but not least

Which large famous products are developed using Git?

Linux, Homebrew, Windows, Tensorflow, Angular, Inkscape, ...

And if I do not have so large projects?

It doesn't matter! There are too many advantages having a writing or coding project under a source code management tool. Even alone. **Simply use one (Git). Now.**

Last but not least

Which large famous products are developed using Git?

Linux, Homebrew, Windows, Tensorflow, Angular, Inkscape, ...

And if I do not have so large projects?

It doesn't matter! There are too many advantages having a writing or coding project under a source code management tool. Even alone. **Simply use one (Git). Now.**

For collaborative projects like maintaining code in a group, handing it over from person to person and so on, Git is simply a must. **As project leader, you should think about requiring everybody to work in a Git repository.**

What is Git?

How does Git define itself?

> «Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. Git is easy to learn and has a tiny footprint with lightning fast performance.»
>
> Git homepage

1. Free and open
2. **Distributed version control system**
3. From small to very large projects
4. With speed and efficiency
5. Easy to learn

How does it work?

C