Introduction to git

Joachim Desroches

GNU Generation

2018-10-01
This is Git. It tracks collaborative work on projects through a beautiful distributed graph theory tree model.

Cool. How do we use it?

No idea. Just memorize these shell commands and type them to sync up. If you get errors, save your work elsewhere, delete the project, and download a fresh copy.
1. The Problem
2. About git
3. Using git
4. Collaborate
5. Common mistakes
6. Practice time
Version Hell

- MyDocument_v2.docx
- MyDocument_v1.docx
- MyDocument_v1.1.docx
- MyDocument_final_v3.docx
- MyDocument_final.docx
- MyDocument.docx
Version Control

- Make the computer do the work.
Version Control

- Make the computer do the work.
- Several exist: git, SVN, CVS, Mercurial...
Make the computer do the work.

Several exist: git, SVN, CVS, Mercurial...

Git created to track Linux kernel, most commonly used.
Version Control

- Make the computer do the work.
- Several exist: git, SVN, CVS, Mercurial...
- Git created to track Linux kernel, most commonly used.
- Do NOT confuse with project management systems like Microsoft Github or Gitlab!
Tracking changes

Let us start locally. There are three things you want to do:

- Start using git! `git init` and `git status`
Tracking changes

Let us start locally. There are three things you want to do:

- Start using git! `git init .` and `git status`
- Add and save changes: `git add <file>` and `git commit`
Let us start locally. There are three things you want to do:

- Start using git! `git init .` and `git status`
- Add and save changes: `git add <file>` and `git commit`
- Review changes: `git log`, `git diff` and `git checkout`
And then they were two

- What happens if there are several of us?
- Needs some centrally accessible machine.
And then they were two

- What happens if there are several of us?
- Needs some centrally accessible machine.
- Helps to use a frontend like Gitlab.
And then they were two

- What happens if there are several of us?
- Needs some centrally accessible machine.
- Helps to use a frontend like Gitlab.
- Interacting: `git push`, `git pull`, merging.
And then they were two

- What happens if there are several of us?
- Needs some centrally accessible machine.
- Helps to use a frontend like Gitlab.
- Interacting: `git push`, `git pull`, merging.

We will see this in more detail in the practical part.
Some confusions that occur often

- Trying to remove things from git.
Some confusions that occur often

- Trying to remove things from git.
- Trying to modify a commit when history is shared.
Some confusions that occur often

- Trying to remove things from git.
- Trying to modify a commit when history is shared.
- Not fixing conflicts properly.
Please pair up! (Groups of 3 are ok) Then head over to http://gitlab.gnugen.ch
Practice time

One of you should create a new project, choosing to initialize it with a README and add the other as at least a developer. Then follow the instructions to clone the repository on your machines.
Make some changes each, and track them! Remember, you can use:

- `git add <myfile>` in order to set a changed file in this snapshot.
- `git commit` to snapshot the files.
- `git log` to see what has happened until now.
Now, one of you can use git push to send his work to the server. Go the webpage of your project, and see that you can indeed see the changes. The second should try to push as well: does this work?
- `git pull`: you will start a merge.
- `git status, git add` to resolve: open the files and look for lines with `>>>>>>` and `<<<<<<<`.
- Finally use `git commit` to finish, and push your changes!

Your partner can now pull: this time, there should be no conflict.
- **git pull**: you will start a merge.
- **git status, git add** to resolve: open the files and look for lines with `>>>>>>`, `=====` and `<<<<<<<<<`.
- Finally use **git commit** to finish, and push your changes!
- `git pull`: you will start a merge.
- `git status`, `git add` to resolve: open the files and look for lines with `>>>>>>`, `=====` and `<<<<<<<`.
- Finally use `git commit` to finish, and push your changes!
- Your partner can now pull: this time, there should be no conflict.
Conclusion

- Remember man pages are your friend!
Conclusion

- Remember man pages are your friend!
- Read the Git Book: more depth and more possibilities
Conclusion

- Remember man pages are your friend!
- Read the Git Book: more depth and more possibilities
- Most of all, use git! Practice makes perfect.
Conclusion

- Remember man pages are your friend!
- Read the Git Book: more depth and more possibilities
- Most of all, use git! Practice makes perfect.
- Questions?