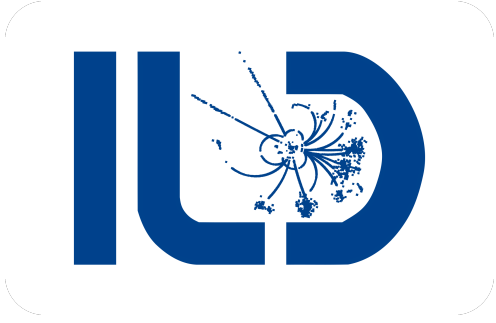
Programming tips. Git (2).

Git or CTRL+S evolution

Rémi Ete

DESY

November 26, 2018



Today: Git locally (no github)

Some git definitions (from Git ep1)

- **Working directory**: local directory where you have content that you want to manage with git
- **Commit**: full snapshot of your working directory contents. Uniquely identified by a 40 character SHA1 hash. Refer to the exact state of your project and can be used for copying or restoring your project.
- **Index**: local staging area in your working directory for your changes before being commited.
- **Branch**: a series of commits (history). A branch has a name (default is *master*).
- **HEAD**: a commit pointer stored by Git. Points on the current revision/version of your files. Should always point on the latest commit of the current branch, else your are in a state called *DETACHED HEAD*.
- **Remote repository**: remote location where a copy of your working directory can be stored (i.e Github). On your computer, a remote location associate a name (i.e *origin*) to a repository URL (e.g https://github.com/*username*/*reponame*.git).

First commands

Creating a new repository

- Create a repository:

```
$ mkdir MyRepo; cd MyRepo
$ git init
Initialized empty Git repository in /home/toto/MyRepo/.git/
```

- MyRepo is a standard UNIX repository but ...

```
$ ls -la
.
..
.git
```

- The `.git` directory keep track of all git information.
- Never delete this directory!

First commands

Configuring your repository

- The repo configuration is written in .git/config file

```
$ cat .git/config
[core]
repositoryformatversion = 0
filemode = true
bare = false
logallrefupdates = true
```

- To configure your repo, use `git config` (286 options on my machine)
- Start with basic settings:

```
$ git config –global user.name "Sarah Connor"
$ git config –global user.email sarah.connor@sky.net
$ git config –global core.editor emacs # please no vi or vim ...
$ git config –list
...
```

- Use `--global` for all repositories
- Use `--local` for this repository only

First commands

Your first file

- Create your first file, edit it

```
$ touch ReadmE.md
# and edit !
```

- The most important command with git: `git status` !

```
$ git status
On branch master

Initial commit

Untracked files:
(use "git add <file>..." to include in what will be committed)

ReadmE.md

nothing added to commit but untracked files present (use "git add" to track)
```

This is a subliminal message

The most important command with git:

```
git status
```

First commands

Your first commit

- Instruct git to start tracking your Readme.md file
- This file will go in your *index*

```
$ git add ReadmE.md
```

- Now you can either create/edit/add new files
- Or commit files in your index

```
$ git commit -m "Added Readme.md file"
[master (root-commit) bc2c8b6] Added ReadmE.md file
1 file changed, 1 insertion(+)
create mode 100644 ReadmE.md
```

- If you omit the option *-m <message>* your configured editor will popup in the terminal to get your commit message

In case you did not know

The most important command with git:

```
git status
```

First commands

Moving (move/rename) a file

- You have noticed the typo: Readme.md / ReadmE.md
- Use *git mv* to rename your file

```
$ git mv ReadmE.md Readme.md
```

- It tells git
  - To rename your file without any changes
  - To add this file to your index

```
$ git status
On branch master
Changes to be committed:
(use "git reset HEAD <file>..." to unstage)

renamed: ReadmE.md -> Readme.md
```

- Then add more stuff and/or commit your changes

You got it right ?

The most important command with git:

```
git status
```

First commands

Removing a file

- Files can be removed normally with the `rm` command
- Will appear as:

```
$ rm Readme.md
$ git status
On branch master
Changes not staged for commit:
(use "git add/rm <file>..." to update what will be committed)
(use "git checkout – <file>..." to discard changes in working directory)

deleted: Readme.md

no changes added to commit (use "git add" and/or "git commit -a")
```

- Use the `git rm` command to remove a file and add it to index

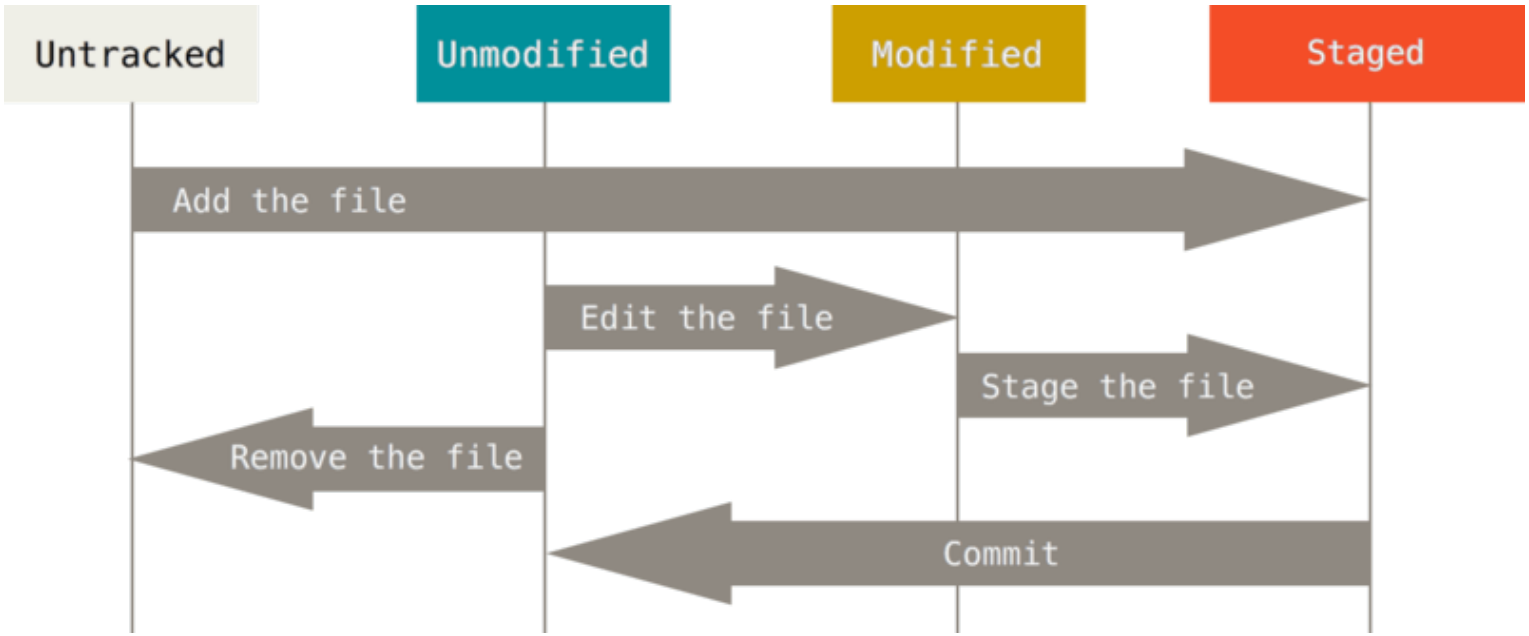
```
$ git rm Readme.md
$ git status
On branch master
Changes to be committed:
(use "git reset HEAD <file>..." to unstage)

deleted: Readme.md
```

I like repetitive humour. And you ?

The most important command with git:

```
git status
```

Git local workflow



First commands

Some other useful (random) commands

- Show a (pretty) commit history

```
$ git log –graph –decorate=full –pretty=oneline –abbrev-commit –all
```

- Show command's help, e.g for *git add*:

```
$ git help add
```

- Show diffs between last commit and current repository state:

```
$ git diff [–staged]
```