A Short Introduction to git

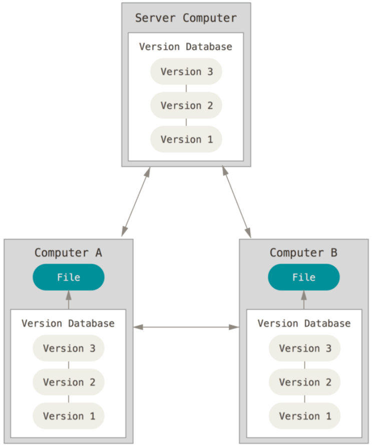
Projet programmation 2 – ENS Cachan

KOLČÁK Juraj
`juraj.kolcak@lsv.fr`

February 2, 2018

What is git?

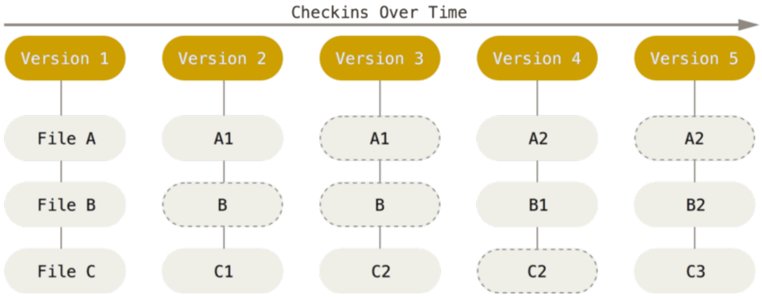
Git is a distributed version control system.



Basics

Versioning

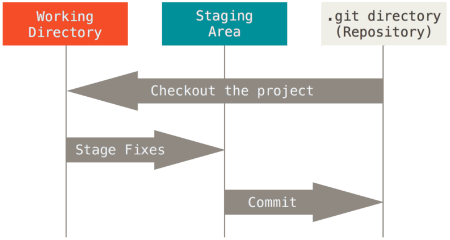
Git records your data in a stream of snapshots.



Basics

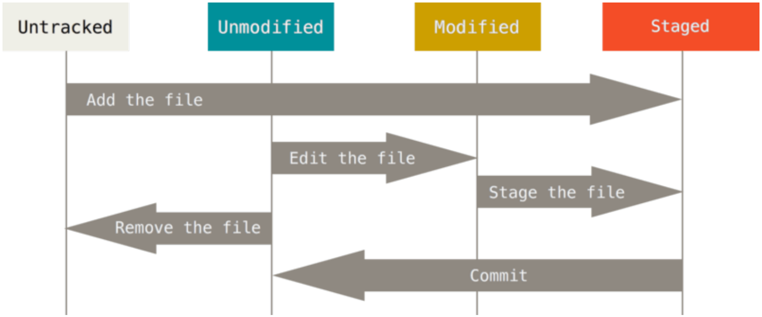
The three sections of git project

- .git directory, contains metadata and the object database (snapshot chain...).
- working tree, this is where the local copy of project files is located.
- staging area, file specifying which changes are going into the next version (snapshot).



Basics

The file lifecycle



Each file in your working tree is in one of the following four states. The transitions between the states also illustrate the basic git workflow.

Basic Commands

- `$ git init` : initialise new repository.
- `$ git status` : prints status information.
- `$ git add <file>` : stage a file.
- `$ git rm <file>` : remove file and stage the removal.
- `$ git reset HEAD <file>` : unstage file.
- `$ git checkout [-] <file>` : revert file to last committed version (snapshot).
- `$ git diff` : prints differences between working tree and index (stage area).
- `$ git log` : show commit history.
- `$ git commit` : commit currently staged files (create new snapshot).

Working with Remotes

Collaboration

All of the the commands we have listed so far work locally. To collaborate it is necessary to communicate with remote repositories.

Commands

- `$ git clone <URL>` : Clone into an existing repository.
- `$ git remote` : prints associated remote repositories.
- `$ git remote add <name> <URL>` : add new remote repository.
- `$ git fetch` : get the newest information from the repository (synchronise).
- `$ git pull` : Download the newest version (snapshot) from the repository.
- `$ git push` : Push your local commits to the repository.

Branches

Git allows to create several branches thus maintaining two or more different version at the same time. It is subsequently possible to marge the branches together, or abandon them altogether. This is particularly useful for experimental features or benchmarking different versions of algorithms.

Commands

- `$ git branch` : prints current branch.
- `$ git branch <name>` : create new branch
- `$ git checkout <branch>` : switch to a different branch.
- `$ git remote add <name> <URL>` : add new remote repository.
- `$ git merge <branch>` : merge branch into current.
- `$ git rebase <branch>` : rebase current branch to the newest commit of the branch specified.