# Version control with Git for scientists

March 28th, 2018

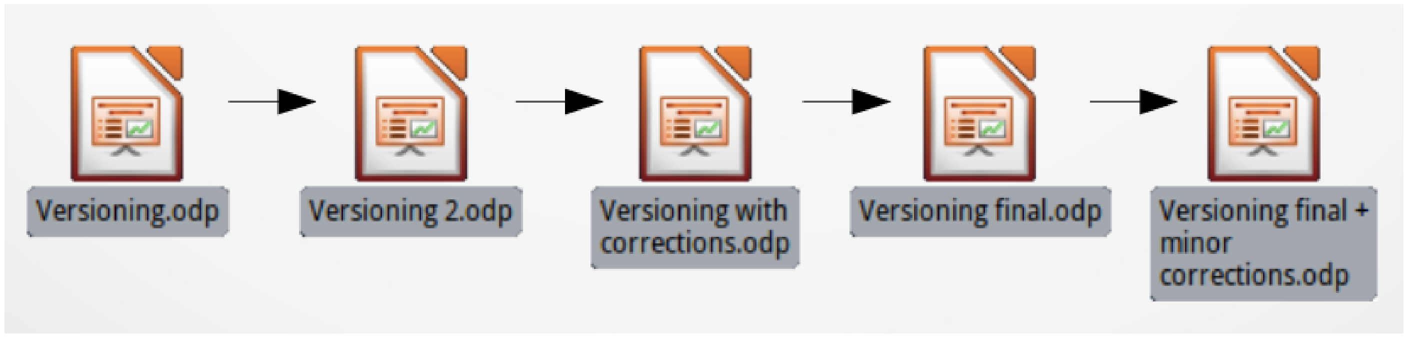
PY Barriat & F. Massonnet

https://github.com/fmassonn/Git_Training

## Discuss

How do you manage different file versions ?

How do you work with collaborators on the same files ?


Versioning.odp
Versioning 2.odp
Versioning with corrections.odp
Versioning final.odp
Versioning final + minor corrections.odp

## Notions of code versioning

Track the history and evolution of the project

- think of it as a series of snapshots (commits) of your code

Benefits:

- team work
- tracking bugs
- recovering from mistakes

Different usage:

- local
- client-server

## What is Git ?

Version control system

- Manage different versions of files
- Collaborate with yourself
- Collaborate with other people

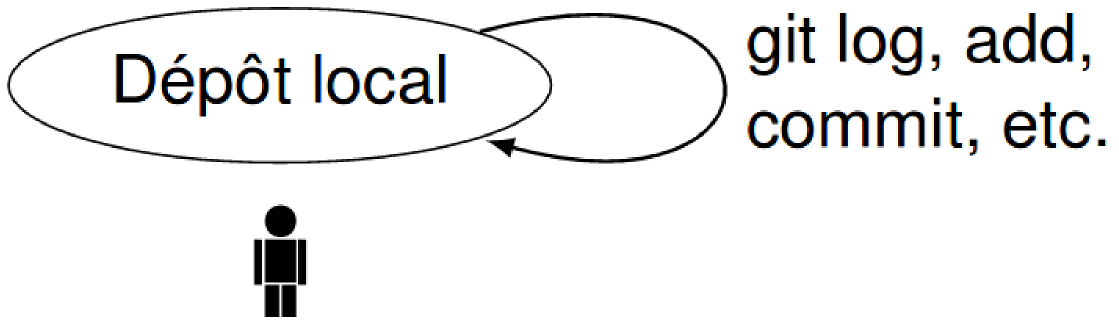
Why use Git

- “Always remember your first collaborator is your future self, and your past self doesn’t answer emails”
- → Christie Balhai

## What is Git good for ?

Local

Backup, reproducibility

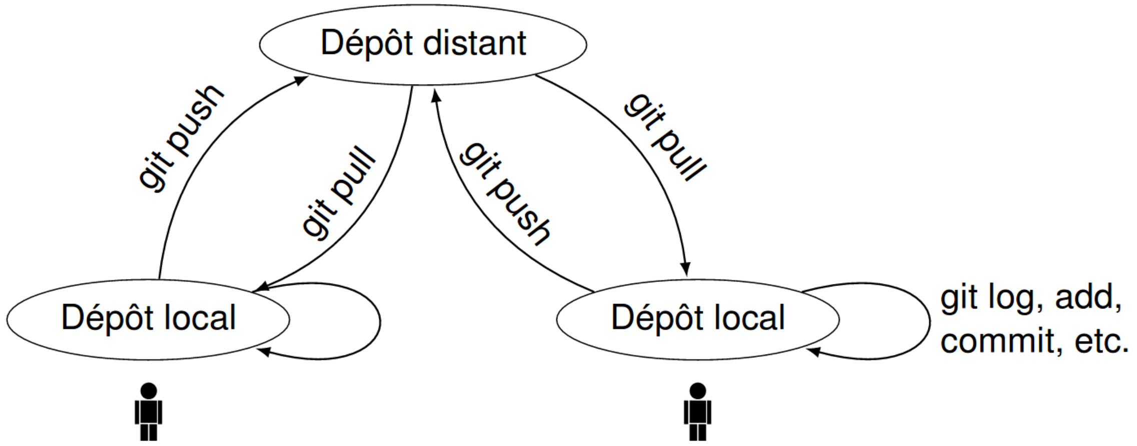


Utilisation locale

## What is Git good for ?

Client-Server

Backup, reproducibility, collaboration



Dépôt commun distant
(Gitolite, Redmine, FusionForge, *GitHub*)

## Difference between Git & GitHub?

Git is the version control system service

Git runs local if you don't use GitHub

GitHub is the hosting service, a website

on which you can publish (push) your Git repositories and collaborate with other people

- It provides a backup of your files
- It gives you a visual interface for navigating your repos
- It gives other people a way to navigate your repos
- It makes repo collaboration easy (e.g., multiple people contributing to the same project)
- **It provides a lightweight issue tracking system**

## … and GitLab vs GitHub vs others

GitLab is an alternative to GitHub

- GitLab is free for unlimited private projects. GitHub doesn't provide private projects for free

And for ELIC, **Gogs** does the job

- shares the same features (Dashboard, File browser, Issue tracking, Groups support, Webhooks, etc)
- easy to install, cross-platform friendly,
- uses little memory, uses little CPU power
- … and 100% free

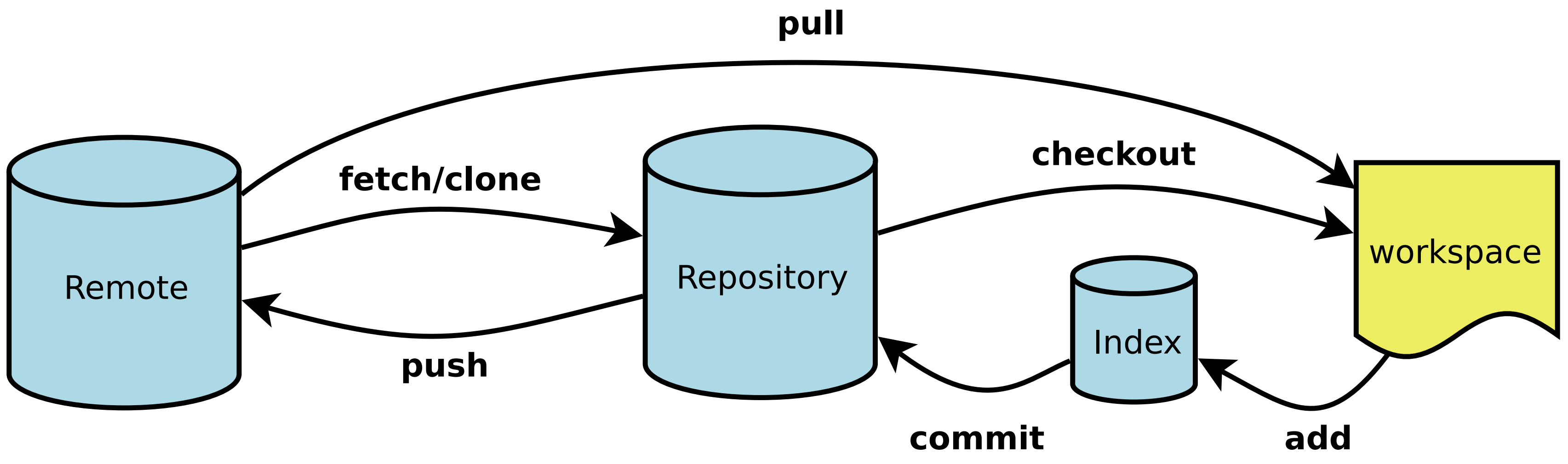
# Git workflow

**Workspace** which holds the actual files

**Index** which acts as a staging area

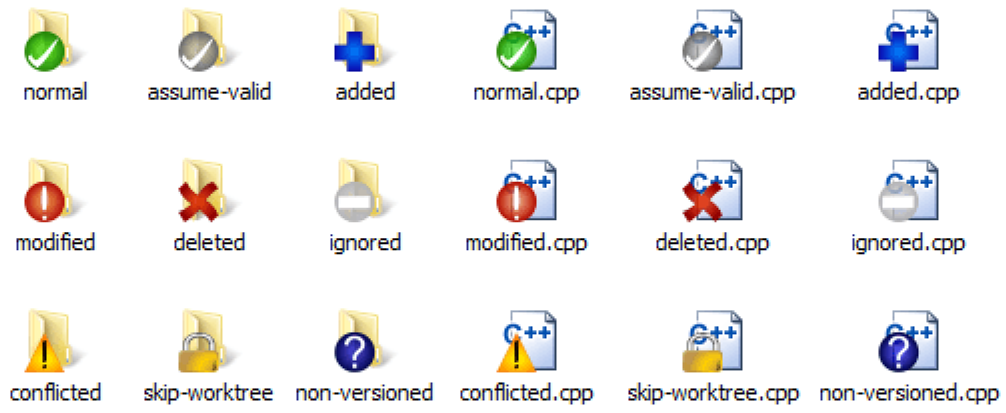
**HEAD** of Repository (local) or Remote

which points to the last commit you've made



# Windows users

How commonly do programmers use TortoiseGit instead of the command line?



But, to be familiar with Git, try the command line (clone, push/pull, merge, log, tag, etc).