Git for version control:
A short intro

Aitor Muguruza
MSc Mathematics and Finance, Imperial College London

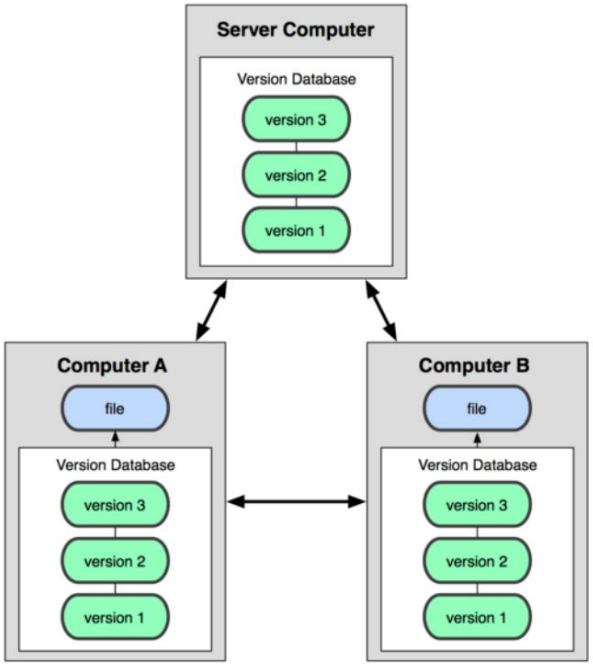
What is git?

Git is **a DevOps tool used for source code management**. It is a free and open-source version control system used to handle small to very large projects efficiently. Git is used to tracking changes in the source code, enabling multiple developers to work together on non-linear development.

In short, is the software that runs in the backend of GitHub, GitLab, Bitbucket and many other Version Control Systems (VCS)

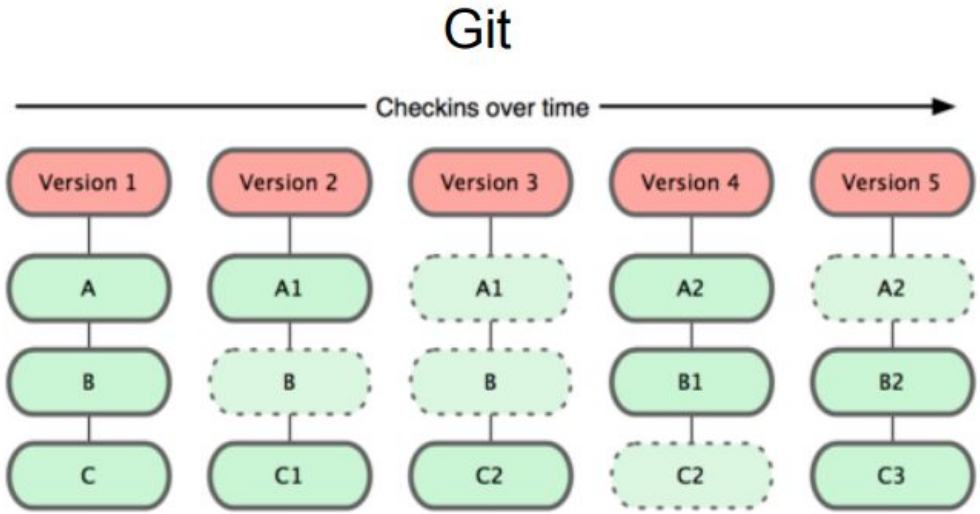
Distributed version control

- Distributed software development has become the norm in all technological companies, including banks and hedgefunds
- We call a repository (repo for short), the remote server that contains the master copy of the software we are developing
- Individual developers, clone the master repo into their local repo and make changes to it.
- Once the developer has made the desired changes he/she can deploy the changes in the master repo as were
- In this step there is usually a validation step, where an authorised developer validates the change in the master repository.



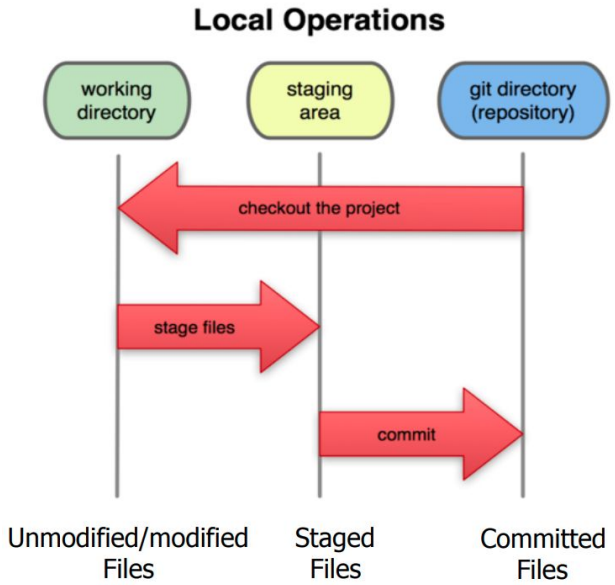
Git snapshots

- The beauty of Git is that it keeps snapshots of all the development stages of the project.
- This way, if a problem is identified it is easy to go back to a previous stage of the project



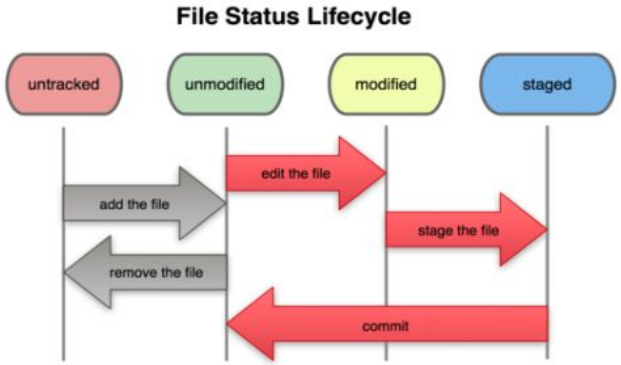
Transition of local copies to master repository

- When developing a project, we can choose which files we want to select to be staged. ***Staged*** means that these files will eventually be pushed to the master repository.
- Therefore, we will have a mix of staged and non-staged (e.g. test data or logs) files.
- Once we are happy with our changes we can ***commit*** the changes which means that we are saving a snapshot of all staged state. Careful, committing does not automatically change the master repository but it does create a snapshot in the history



Basic git workflow

- **Modify** files in your working directory.
- **Stage** files, adding snapshots of them to your staging area.
- **Commit**, which takes the files in the staging area and stores that snapshot permanently to your Git directory.



Interaction with the remote master repository

- Once changes are committed we have the option to **push** those to the master repository
- Likewise, we have the option to **pull** changes from the remote repository to get the most recent changes that other developers might have done

Branches and merges

- A **branch** is a copy of a git snapshot at a specific time. We can have multiple branched being developed at the same time in parallel and pushing changes to them does not alter the **master branch**
- A **merge** operation is the action of merging a branch with another branch (this can be the master one although not always).
- Git automatically checks that a merge is viable (in case changes in parallel might have caused issued in syntax (like some key variables not being declared)