**Memo GIT commands : DATA +8**

|  |  |  |
| --- | --- | --- |
| **Instruction** | **Utilisation** | **Exemples** |
| status | Shows the working tree status  The *git status*command displays the state of the working directory and the staging area. It lets you see which changes have been staged, which haven’t, and which files aren’t being tracked by Git. Status output does *not* show you any information regarding the committed project history. For this, you need to use *git log*. | git status |
| diff | Shows changes between commits, commit and working tree, etc | git diff |
| add | Adds file contents to the index | git add myfile.txt |
| commit | Records changes to your local repository | git commit –m ‘’added myfile.txt ‘’  git commit # with no message after a conflict |
| log | Shows commit logs/historic | git log |
| push | Updates the server by pushing my local changes (push origin “remotename” – ex: your local repository to “branchname”: ex master on the server) | git push origin master |
| pull | Fetches from and integrate with another repository or a local branch – pulls changes down from the remote to my pc (pulls from “remotename” ex:  origin to the server to “branch name” ex: master on your local repository) | git pull origin master |
| clone | Clones a repository (from GitHub for example) into a new directory (local) | git clone <https://github.com/A-Martin-Tissier/GIT_pandas-examples> |
| init | Creates an empty git repository or reinitializes an existing one (creates .git ) | git init |
| remote | Manages a set of tracked repositories | git remote add origin https://github.com/cavallaric/my\_first\_project.git |
| merge | Incorporates changes from the commits (since the time their histories diverged from the current branch) into the current branch. It command is used by **git pull** to incorporate changes from another repository and can be used by hand to merge changes from one branch into another. | git merge --abort # delete previous merge done by git pull  git merge origin/master # merges origin and master |
| config | Allows to modify the configuration file | git config –global core.editor emacs  git config –l  git config --global user.name cavallaric  git config --global user.email cavallaric@yahoo.com |
| checkout | It allows to switch on a branch/earlier stage Local modifications to the files in the working tree are kept, so that they can be committed to the branch | git checkout sha\_of\_the\_stage # 6ed7eb9  git checkout master # go back to master |
|  |  | .gitignore # allow to add untracked files, like .pyc in python etc |

**Installation**

1. **first install homebrew**

on my mac terminal run this :

/bin/bash -c "$(curl –fsSL <https://raw.githubusercontent.com/Homebrew/install/master/install.sh>)"

Installation successful !

from <https://brew.sh/>

1. **then install git**

run this

$ brew install git

from https://git-scm.com/download/mac

https://git-scm.com/about

**Repository** :

<https://git-scm.com/book/en/v2/Git-Basics-Getting-a-Git-Repository>

**Git vs Github**

Git is a version control system that lets you manage and keep track of your source code history. GitHub is a cloud-based hosting service/plateform that lets you manage Git repositories/projects. If you have open-source projects that use Git, then GitHub is designed to help you better manage them.

**Alternatives to GitHub to host your open source projects**

**GitLab:** open source software. You can download and install it on your own server (you are not obliged to use the Microsoft server, like in GitHub), not very expensive

**[BitBucket](https://bitbucket.org/product" \t "_blank)**is a version control repository hosting service from Atlassian. Free for teams up to 5 people

**SourceForge**