Git & GitHub Dictionary

- Git: versioning system that keeps track of changes, allows for collab and is locally stored (on your computer)
- GitHub: online repo for projects and keeping different, backup for git projects, cloud based (remote)

Initialize project

start your timeline: `git init`

do this in project repository

creates your git repo

NOTES:

- don't do double `git init`
- don't remove the `.git` folder where your history is kept

Commits

commiting = making a snapshot, creating a timepoint

messages should be informative: Why change, how adress issue, effects of the change, limitations

`git commit -m 'message'`

  when not adding `-m`, editor will open (set up editor: `git config --global core.editor nano`)

  add commit message to the end of this file and save

Ungit: tracks that a file is changed (saved on your computer) but it is not saved in git

without commits: file is changed but there is not anything yet in `.git`

create first timepoint: `git add Git_GitHub_Dictionary.md` `git commit -m "brief description of differences between git and github and how to get started with git"`

output:

```
[main (root-commit) 0ba4a52] brief description of differences between git and github and how to get started with git
```

```
1 file changed, 33 insertions(+)
```

```
create mode 100644 Git_GitHub_Dictionary.md
```

had to set these first:

- `git config --global user.email "julie.de.man@skynet.be"`
- `git config --global user.name "Julie DM"`

git tracks files in subfolder

GitHub: online <=> git: on your computer

Tracking

`git status` will allow me to check what files are changed, unstaged and untracked

- to be staged: you have commited it before, made new changes but git recognizes that the new changes are not yet `add`ed or `commit`ed
- to be comitted: you have commited the file before, made new changes and git recognizes that you have `add`ed but not yet `commit`ed
- untracked: is a completely new file/folder that has never been `add`ed or `commit`ted

track your messages: in `.git/logs/HEAD`

easier alternative: `git log`

Conceptual areas

01. Development area

where you develop your project, is on your computer, the folder where your project is developed => working directory

02. Staging area

place where you dump the things before sending to local `.git` repo

so first `git add` and then `git commit`

03. Local repository

the .git folder, where the snapshots are saved, where your timeline is

! is local on your computer

check if there is already an initialized `.git` repo there, so that you dont initialize another one

`git commit` sends the snapshot to the local repo

04. Repote repository

https://github.com

used as a backup: .git is your local repo so is not kept if your pc crashes

create new repo under repositories on github

github: limited upload, no data repo

project should always have:

- `README.md` : information about the project
- `.gitignore` : sometimes you want all your files in a certain folder, also fasta files etc. but you do not want to create snapshots for them. Files listed in .gitignore cannot be staged and commited

Traveling through the timeline

`git show commitID1 commitID2`

> shows the full files

alterntive: `git diff commitID1 commitID2`

> shows line by line what is happening, is really comparing