FAIR_bioinfo for bioinformaticians

Introduction to the tools of reproducibility in bioinformatics

C. Hernandez[1] T. Denecker[1] J.Sellier[2] C. Toffano-Nioche[1]

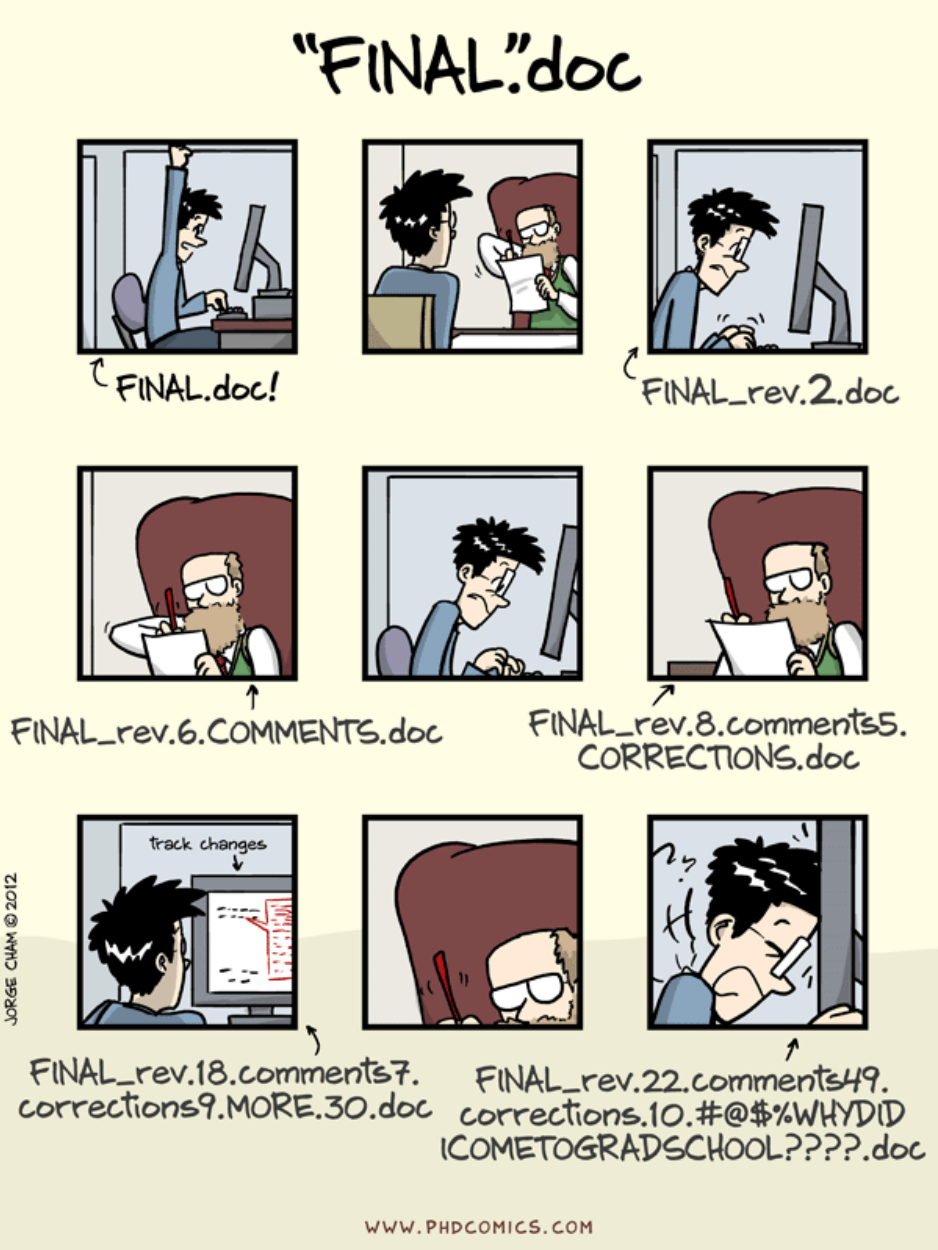
[1]Institute for Integrative Biology of the Cell (I2BC)
UMR 9198, Université Paris-Sud, CNRS, CEA
91190 - Gif-sur-Yvette, France

[2]Institut Français de Bioinformatique
à compléter

Sept. 2020

Introduction to code versioning

Really need of a files history?



"Most researchers are primarily collaborating with themselves," [Tracy] Teal explains. "So, we teach it from the perspective of being helpful to a 'future you'."

Files history = good practice for reproducible research

"Rule 4: Version Control All Custom Scripts"

OPEN ACCESS Freely available online

PLOS | COMPUTATIONAL BIOLOGY

Editorial

Ten Simple Rules for Reproducible Computational Research

Geir Kjetil Sandve[1,2]*, Anton Nekrutenko[3], James Taylor[4], Eivind Hovig[1,5,6]

1 Department of Informatics, University of Oslo, Blindern, Oslo, Norway, **2** Centre for Cancer Biomedicine, University of Oslo, Blindern, Oslo, Norway, **3** Department of Biochemistry and Molecular Biology and The Huck Institutes for the Life Sciences, Penn State University, University Park, Pennsylvania, United States of America, **4** Department of Biology and Department of Mathematics and Computer Science, Emory University, Atlanta, Georgia, United States of America, **5** Department of Tumor Biology, Institute for Cancer Research, The Norwegian Radium Hospital, Oslo University Hospital, Montebello, Oslo, Norway, **6** Institute for Medical Informatics, The Norwegian Radium Hospital, Oslo University Hospital, Montebello, Oslo, Norway

Replication is the cornerstone of a cumulative science [1]. However, new tools and technologies, massive amounts of data, interdisciplinary approaches, and

We further note that reproducibility is just as much about the habits that ensure reproducible research as the technologies that can make these processes efficient and

than to do it while underway). We believe that the rewards of reproducibility will compensate for the risk of having spent valuable time developing an annotated

Version control

Definition

version control, revision control, source control, or source code management: class of systems responsible for managing changes to files.

Feature

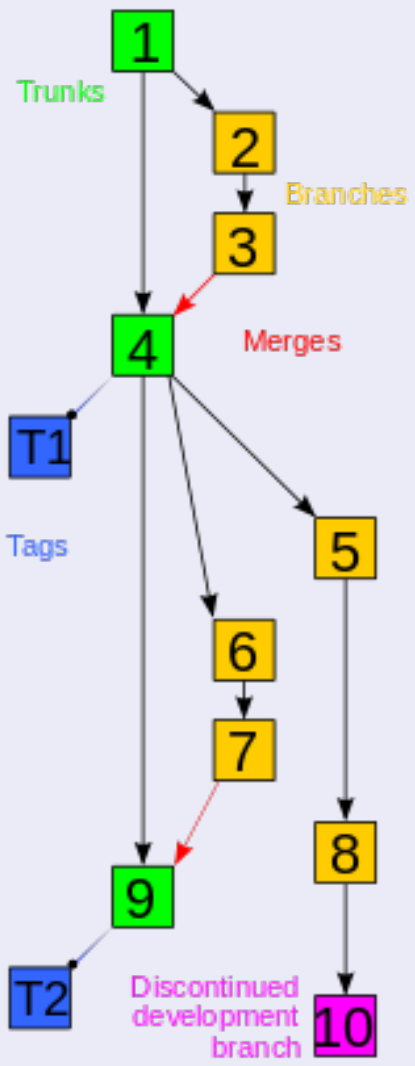
Each revision is associated with a timestamp and the person making the change.
Revisions can be compared, restored, and merged.

Software

SVN, Git, Mercurial, GNU arch, etc

wikipedia source

Revisions graph



Git and GitHub

Git

- will track and version your files
- enables you to collaborate with ... yourself
- open source license GPL (GNU General Public License)
- created in 2005 by Linus Torvalds for the development of the Linux kernel

GitHub

- stores your git repositories online
- enables you to collaborate with others (and yourself)
- first commit in 2007 by Chris Wanstrath, founded in feb. 2008, Microsoft Corporation still 2018

Git

Concepts, objects

- working directory: a user private copy of a whole repository of interest
- staging area: list of files of the working directory that will be considered for next commit (ie. could be not all the modified files)
- clone: a local copy of a repository (include all commits and branches), the original repository can be local, or remote (http access)
- commit: a git object, the snapshot of your entire repository compressed into a SHA (also the command the saves changes by creating the snapshot)
- HEAD: pointer representing your current working directory. Can be moved (git checkout) to different branches, tags, or commits
- branch: a lightweight movable pointer to a commit
- merge: combines remote tracking branches into current local branch

https://www.tutorialspoint.com/git/git_quick_guide.htm
https://www.powershellmagazine.com/2015/07/13/git-for-it-professionals-getting-started-2/

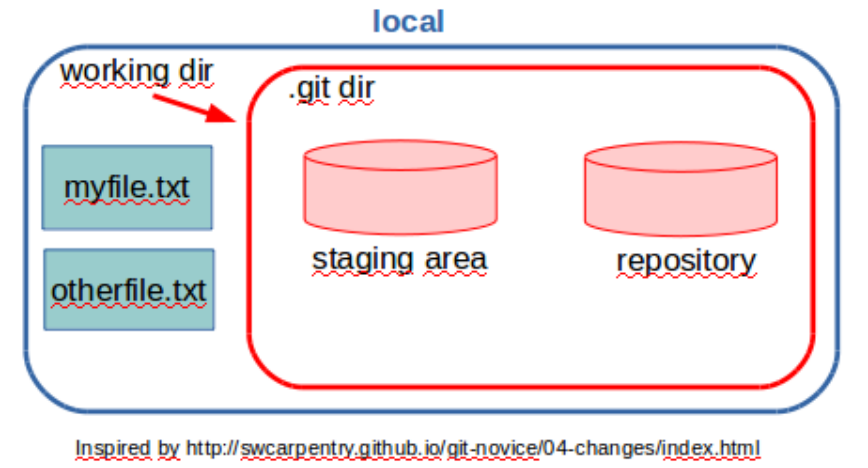
Setup

Git configuration: if not yet done, tell git our identity

```
1 git config --global user.name 'Your Name'
2 git config --global user.email 'Your Email'
```

Git repository initialisation

The initialisation (red arrow) is the creation of a .git repository:

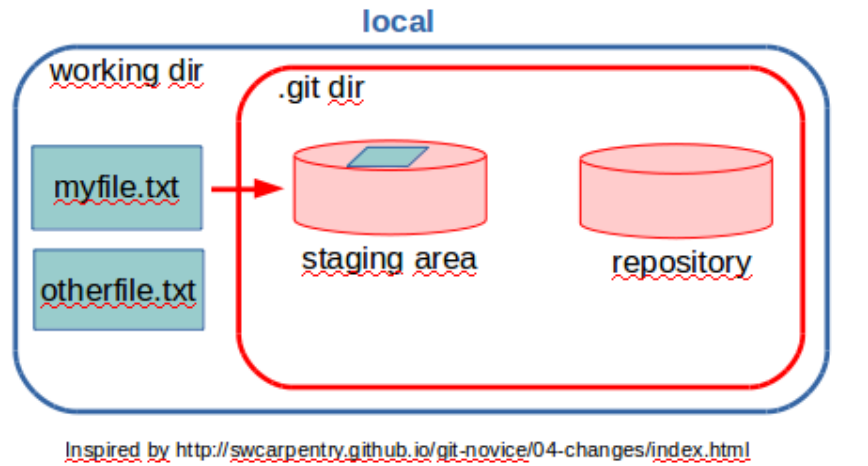


3 ways to initialize a .git repository:

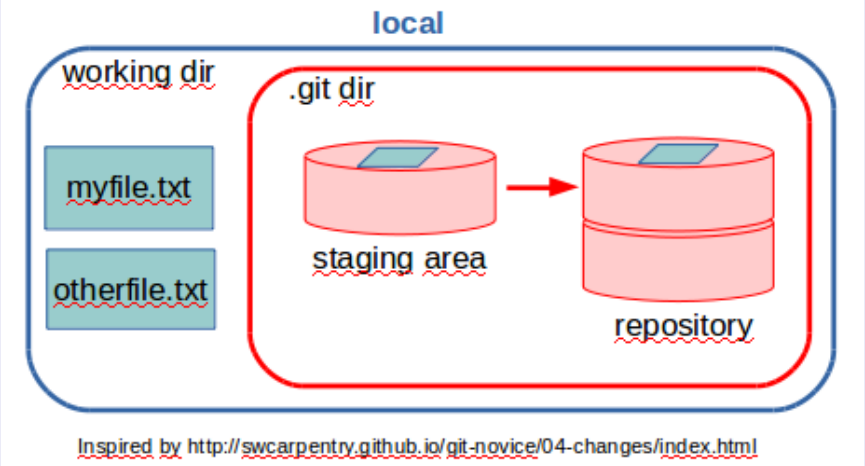
- git init: inside an existing folder (possibly containing files)
- git init project: create the folder "project" + initializes the .git subfolder inside it
- git clone /gitfolder/path /new/path copy the existing git repository to a new one

Tracking file

git add command for myfile.txt:



git commit -m "my reason":



`http://swcarpentry.github.io/git-novice/fig/git-staging-area.svg`

Git file states

Checking the file status: `git status`

File goes from untracked to tracked state (init), unstaged to staged state (add) and finally, to a committed state (commit).

Git Exercise

Objective

- install git
- initialize git
- create a git repository
- use the basic git commands for tracking changes
- copy another repository from github
- use branching and merging to manage code change

git setup

Git access by doker

```
docker run -i -t -v ${PWD}:/data continuumio/miniconda3
```

Git configuration

Global configuration (checking `user.name` with: `git config --list`):

```
git config --global user.name 'Your Name'
git config --global user.email 'Your Email'
```

Git repository initialization

On a new dedicated folder run:

```
git init # observe the .git folder (ls -la)
git status # find the current branch, "nothing to commit"
```

git adding file

create 2 files, check their git status

```
for i in 1 2 ; do echo "file"${i}" text" > file${i}.txt ;
    done
git status # observe list of untracked files
```

add file1 to stating area

```
git add file1.txt
git status # observe the changing status of file1: untracked
     => staged
```

change file1 text

```
sed 's/text/text change/' file1.txt > tmp ; mv tmp file1.txt
git status # observe the 3 states, why file1 appears in "to
    be commited" and also in "not staged for commit"?
```

git commit

stage all files

```
1 git add file1.txt file2.txt # all files
2 git status
```

commit

```
1 git commit -m "1st commit + file1 change" # always add a
    message
2 git status # all ok
```

So far, we have initiated a new project whose code is versioned by git: we have created files and all their successive changes were saved thanks to git.

We will now create a 2nd project by copying an already existing one. We're going to bring this project from an online git project site, *e.g.* github.

git cloning

copy of a project: clone

To download a project from github, we use the git `clone` command:

```
1 git clone https://github.com/clairetn/FAIR_bioinfo_github.
    git
```

observe result

- a new folder has been created (check with the shell `ls` command)
- its name is directly deduced from the url used
- as our previous git project, this `FAIR_bioinfo_github` folder contains a `.git` repository and also a `README.md` file (see with `ls -la FAIR_bioinfo_github/`)
- it is a minimal project!

git branching

We plan to change the README file by adding our firstname at the authors list. With a git versioning system, a good practice is to create a branch to reserve the initial code until we validate our change.

create a branch nammed "branch1"

```
cd FAIR_bioinfo_github
git branch branch1
```

list all branches

```
git branch # find the star
```

git branching

go into the new "branch1"

```
1 git checkout branch1
2 git branch # find the star
3 git status # find the branch
```

work inot branch: change a file and keep change

Edit the `README.md` file and add your firstname to the "Authors list"

```
1 git status # file README.md is modified
2 git add README.md ; git commit -m "add my firstname in
    branch1"
```

return to master branch

```
1 git checkout master
2 more README.md # Is README.md modified or initial version?
```

git merging branch

We have check that our change is valid, so we now plan to move it into the `master` branch.

merge branch, then delete branch

```
1 git merge branch1
2 more README.md # what README.md version?
3 git branch -d branch1 # -d for delete
```

GitHub

GitHub "Who" ?

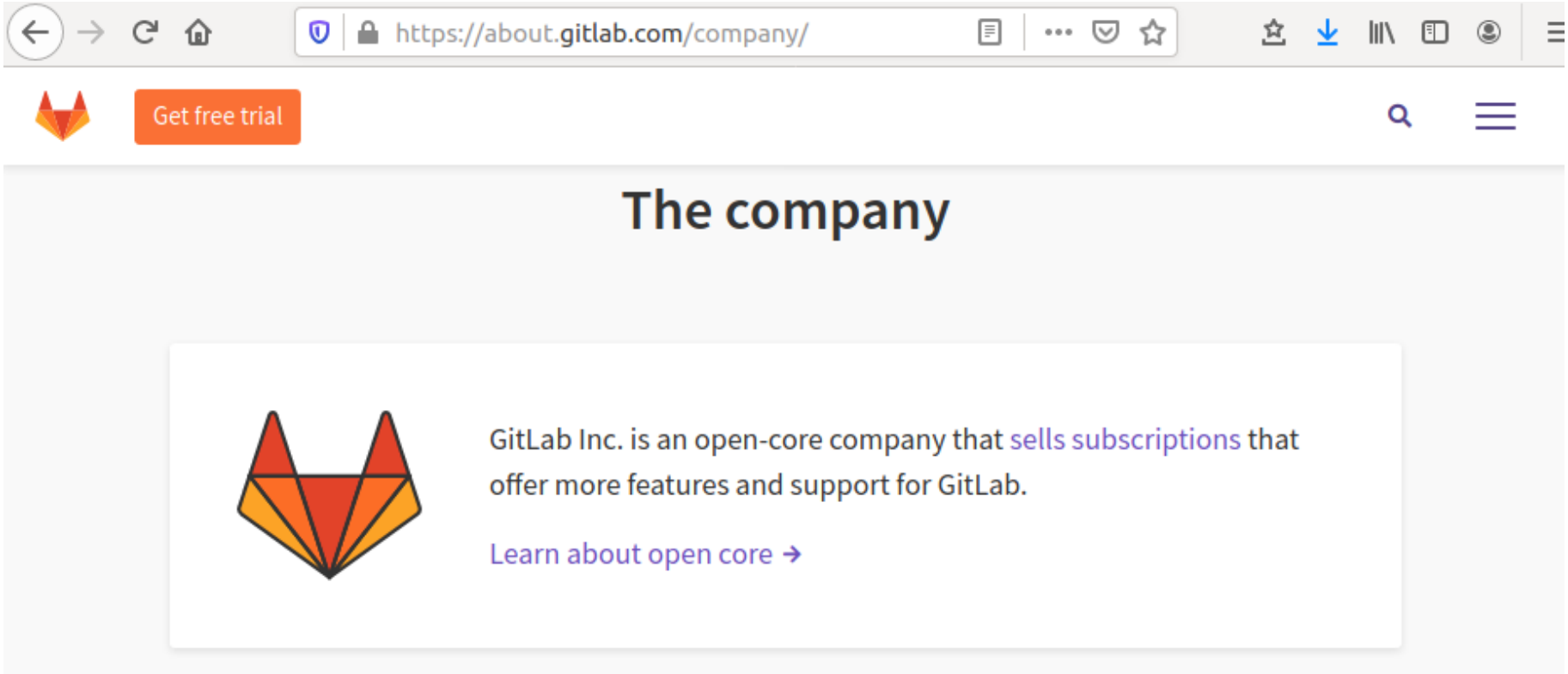
Quizz

1. public institute (governmental)?
2. semi-public institute?
3. not-for-profit organisation?
4. private company?

Response

See `https://github.com/about`: Careers' paragraph, you'll see a "company" word

GitLab, a GitHub alternative?



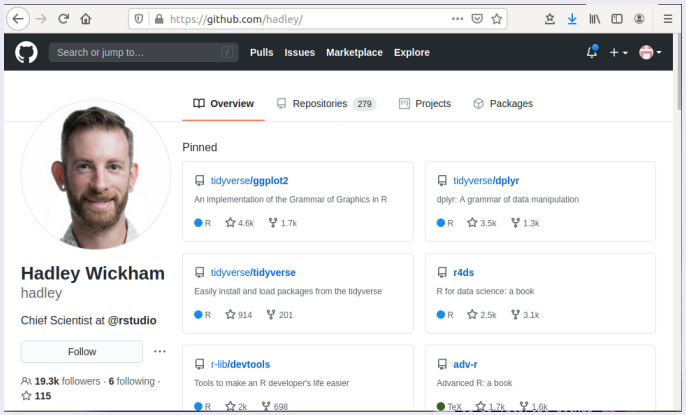
GitHub "What" ?

Quizz

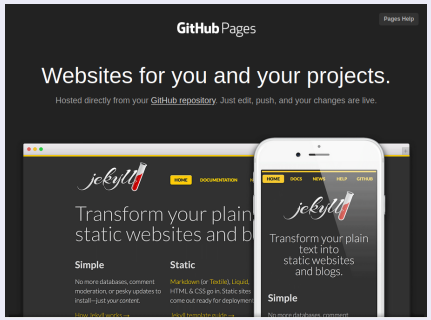
1. social network?
2. desktop application?
3. tool to create websites?
4. stable repository to publish any file?

GitHub

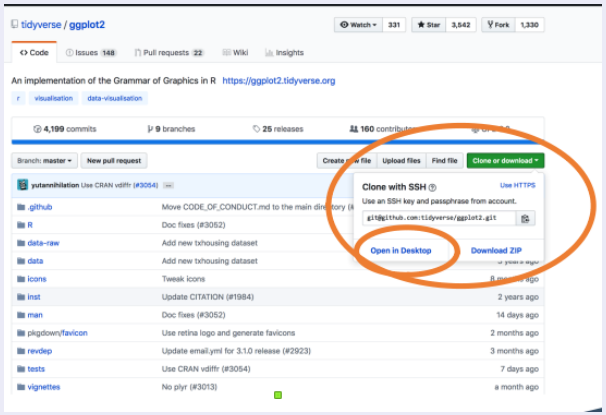
a social network ✔



a tool to create websites ✔



a desktop application ✔



a stable repository ...

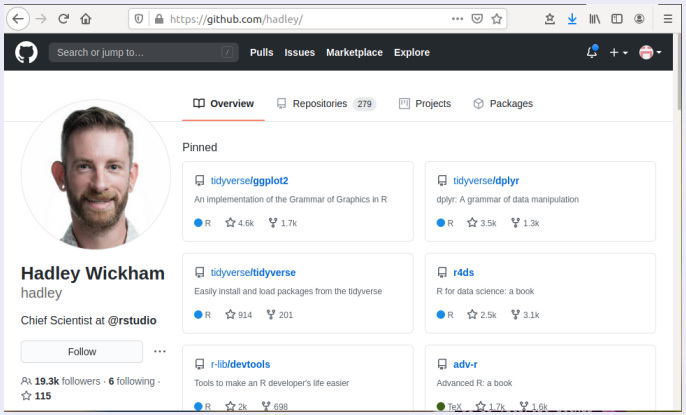
Popularity [edit]

Name	Users	Projects	Alexa rank (lower = more popular)
Assembla	Unknown	526,581+[46]	35,454 as of 28 July 2020[47]
Bitbucket	5,000,000[48]	Unknown	1,341 as of 28 July 2020[49]
Buddy	Unknown	Unknown	39,857 as of 28 July 2020[50]
CloudForge	Unknown	Unknown	402,884 as of 28 July 2020[51]
Gitea	Unknown	Unknown	236,332 as of 28 July 2020[52]
GitHub	31,000,000[53]	100,000,000[53]	78 as of 28 July 2020[54]
GitLab	100,000[55]	546,000[56][k]	2,710 as of 28 July 2020[57]
GNU Savannah	93,346[58]	3,848[58]	162,054 as of 28 July 2020[59]
Launchpad	3,965,288[60]	40,881[61]	11,533 as of 28 July 2020[62]
OSDN	54,826[63]	6,294[63]	8,708 as of 28 July 2020[64]
Ourproject.org	6,353[65]	1,846[65]	1,083,012 as of 28 July 2020[66]
OW2 Consortium	Unknown	Unknown	1,509,677 as of 28 July 2020[67]
Rosetta code	Unknown	Unknown	66,029 as of 28 July 2020[68]
SEUL	Unknown	Unknown	1,602,812 as of 28 July 2020[69]
SourceForge	3,700,000[70]	500,000[70]	470 as of 28 July 2020[71]
Name	Users	Projects	Alexa rank (lower = more popular)

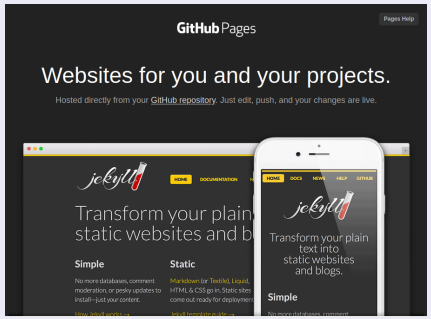
en.wikipedia, comparison of source-code-hosting facilities

GitHub

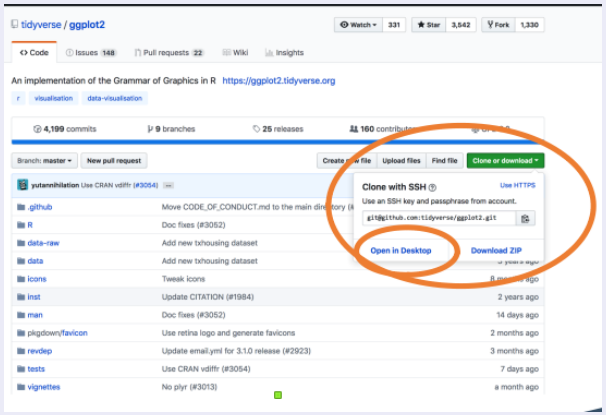
a social network ✔



a tool to create websites ✔



a desktop application ✔



... to publish any file ✔ ✘

Files for which git can calculate the difference between versions.

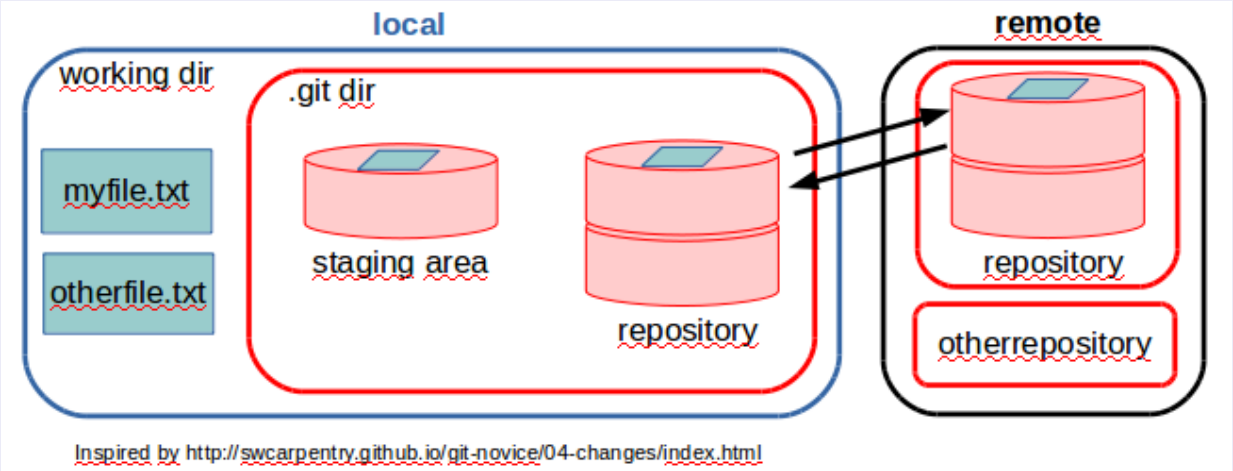
Usually txt files of reasonable size:

- R script: ✔
- Python script: ✔
- pdf file: ✘
- fastq file: ✘

GitHub main usage: sharing code with others

GitHub:

- so used that Microsoft was interested in it (bought in june 2018)
- web-based: graphical interface + many added functionalities
- git-based: so all git concepts and commands are retained
- commands for the "sharing step": `git push origin master` (from local to remote) and `git pull origin master` (from remote to local):



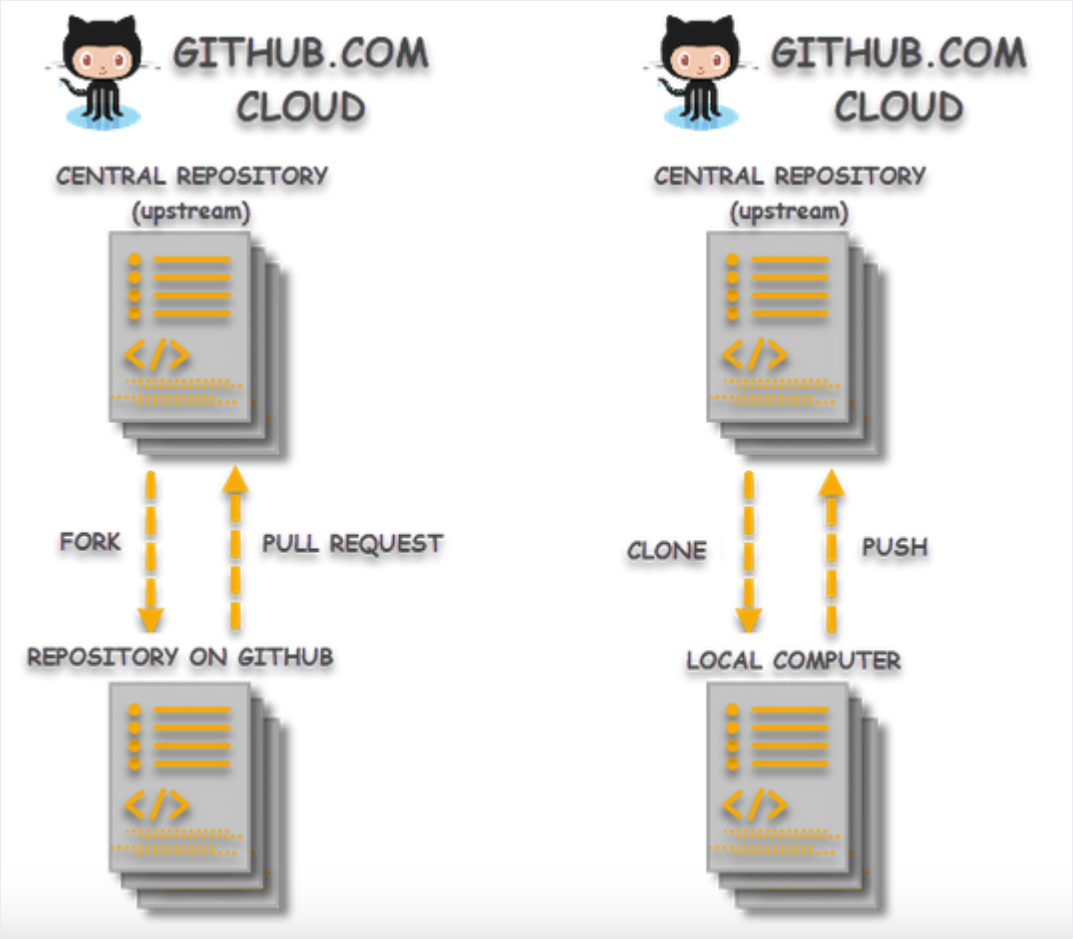
GitHub

Concepts, objects

- user: your account on GitHub (unlimited for academics)
- organization: account for one or more user (e.g., swcarpentry)
- local GitHub: copies of GitHub files located your computer
- remote GitHub: your GitHub files located on https://github.com
- fork: a copy of a GitHub repository to your own GitHub account
- push: send changes on the working repository to your remote GitHub repository
- pull: copy changes on the remote GitHub repository to your local GitHub repository (useful when multiple people make changes)
- pull request: propose your changes to the initial forked GitHub repository. Like a place to compare and discuss the differences introduced on a branch with reviews, comments, integrated tests, etc

GitHub Copy a project

Clone *vs.* Fork?



See here an historical point of view of those 2 words.

GitHub Exercise 1

extrait du programme IFB

Github collaboratif :
Clone du projet principal
Création d'une branche
Ajout de son nom dans le README en local
Demande de révision (Pull Request)
Merge de la branche
Récupération du README avec tous les noms (toutes les branches ont été mergées)

GitHub: Account

If not already yet, sign up, otherwise sign in

https://github.com

Why GitHub? Team Enterprise Explore Marketplace Pricing Search GitHub Sign in Sign up

Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 50 million developers.

Username

Email

Password

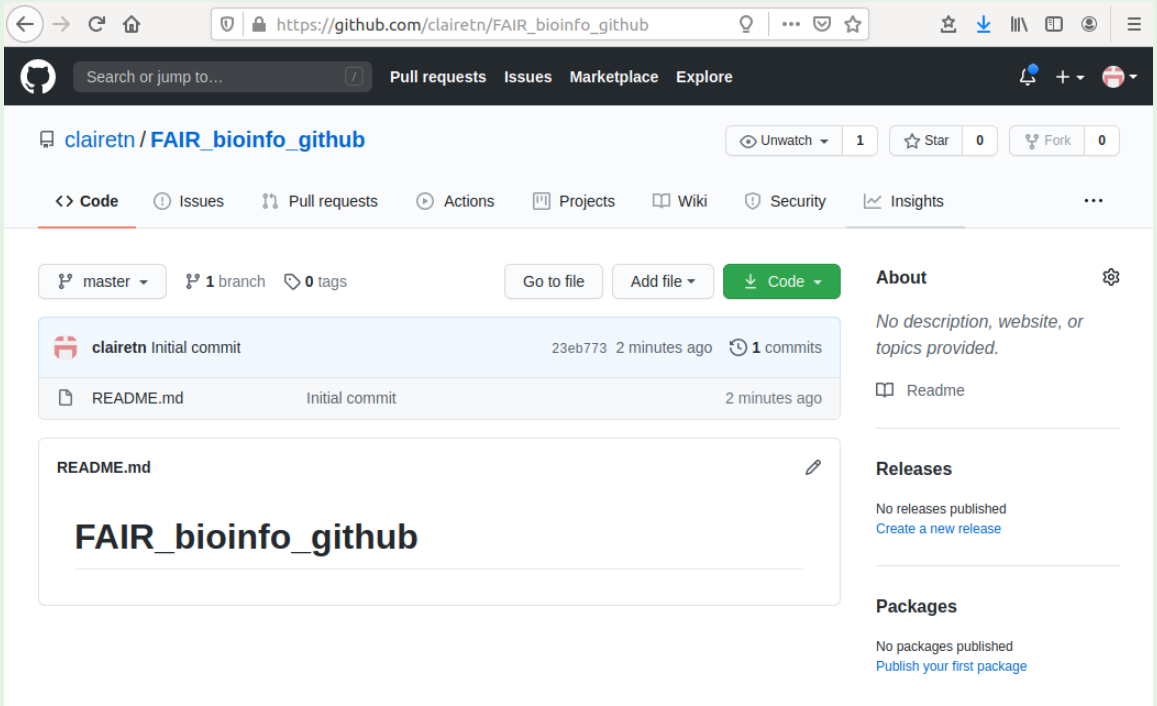
Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. Learn more.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our Terms of Service and Privacy Statement. We'll occasionally send you account related emails.

GitHub: clone a project

For this exercise, we will replay the addition of our first name, but by using the user interface proposed by github.
With a browser, go to the url of the original project, `https://github.com/clairetn/FAIR_bioinfo_github.git` and click on "clone" green button:



GitHub Exercise 2

extrait du programme IFB

Github/Gitlab (en linéaire) :
FORK d'un projet existant
Invitation collaborateur des intervenants
Ajouter son nom dans le README
Commit en local puis push sur Github
Pull request (à valider par un collaborateur)

Github collaboratif :
Clone du projet principal
Création d'une branche
Ajout de son nom dans le README en local
Demande de révision (Pull Request)
Merge de la branche
Récupération du README avec tous les noms (toutes les branches ont été mergées)

Objective

The objective of this exercise is to propose change to an existing project. We will:

- fork an existing project to a local folder
- made a change by adding our name in the README file (local)
- save the change (local) and github (personal remote)
- create a pull request and waiting until its validation (remote)

Web interface

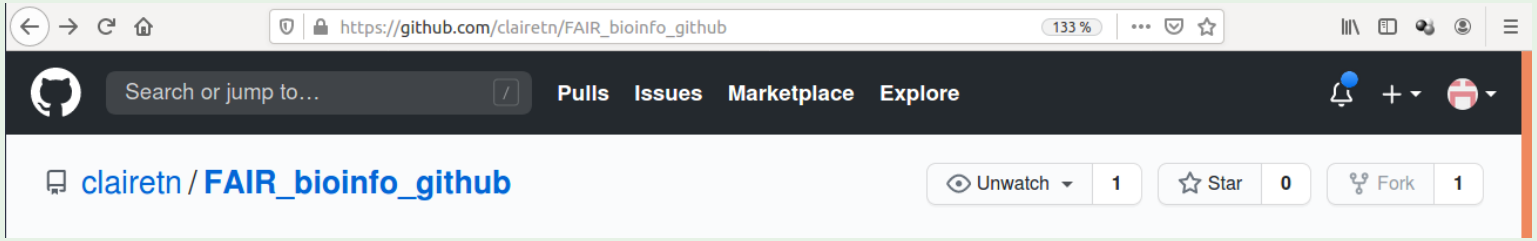
During this exercise, most of the actions that will be performed will be done through github's web interface, i.e. a lot of "click-buttons".

GitHub: Fork a repository

Repository to fork:

https://github.com/clairetn/FAIR_bioinfo_github

Click on the fork button:



GitHub: Fork a repository

Result:

You can see the result in your Github Overview: you have a new repository, named `FAIR_bioinfo_github` and entitled "forked from clairetn/FAIR_bioinfo_github".

GitHub: Add a change to the project

Clone our fork:

Make a copy of the forked repository in our local computer to be able to work on the project.
By command line with `git clone` or by the "clone" button on the GitHub interface.

Work on the project:

Edit the README file and add your name at the end of the file.

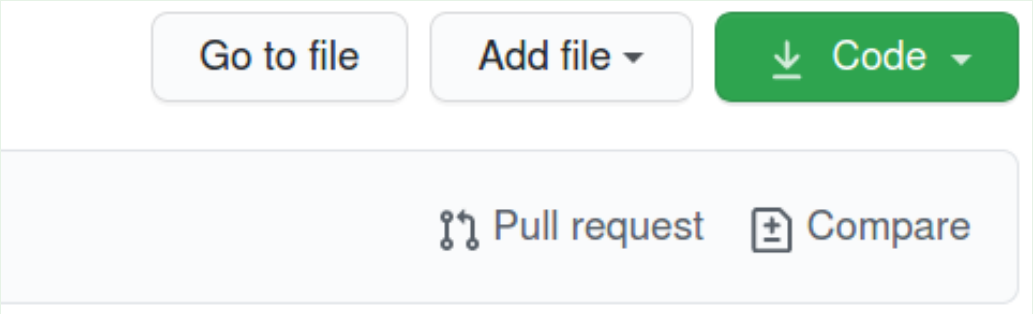
Git add, commit and push by GUI or command lines:

```
1 git add README.md
2 git commit -m "add name"
3 git push origin master
```

GitHub: Propose your change into the initial project

In your forked repository

"Compare" and then "Pull request" your issue (explain your proposals as much as possible):



The pull request asks the maintainer(s) to review your work, provide comments, request edits, etc. If your change will be approved, the maintainer(s) add your change into the code.

Wait for validation from the initial repository

...

Bonus

Challenge

- make a (voluntary today) "error" by suppressing the new dedicated repository created for this git exercise
- retrieve your code with the git clone command on your github repository

Frame Title

ajouter éditeur intégrés avec git

Ressources

- https://nbis-reproducible-research.readthedocs.io/en/latest/git/
- https://swcarpentry.github.io/git-novice/: Learning Git by Software Carpentry
- https://services.github.com/on-demand/resources/cheatsheets/: git Cheat Sheets
- https://jules32.github.io/2016-07-12-Oxford/git/: step-by-step progression to link RStudio and GitHub
- https://cupnet.net/git-github/: Pierre Poulain fr ressources