

Versioning of data and code using Git

Introduction to Data Management Practices course

NBIS DM Team

data@nbis.se

<https://nbisweden.github.io/module-versioning-dm-practices/>



- Data has a life cycle

Raw (experiment) data – produce, collect, license, get access, ...

Processed – generate, clean, aggregate, label, transform, analyse, ...

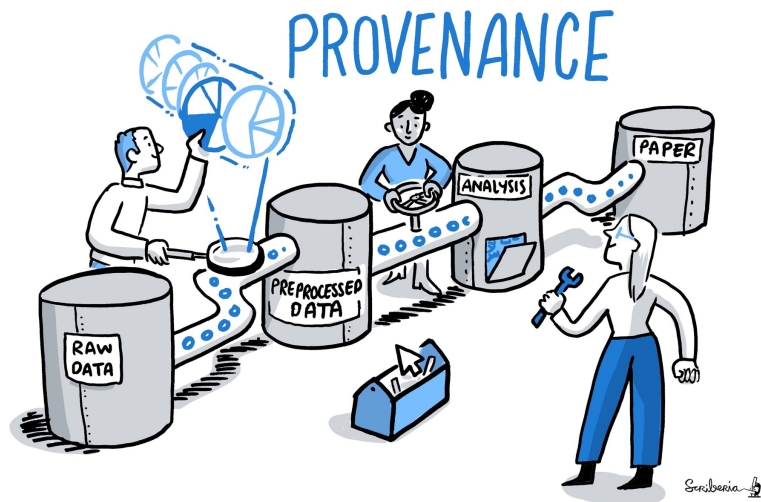
Archived – document, select, convert, package, submit, ...

Published – FAIRify, promote reuse, ...

- **Maintain data integrity and authenticity**

- **Plan a storage strategy**

- Plan a backup and disaster recovery strategy

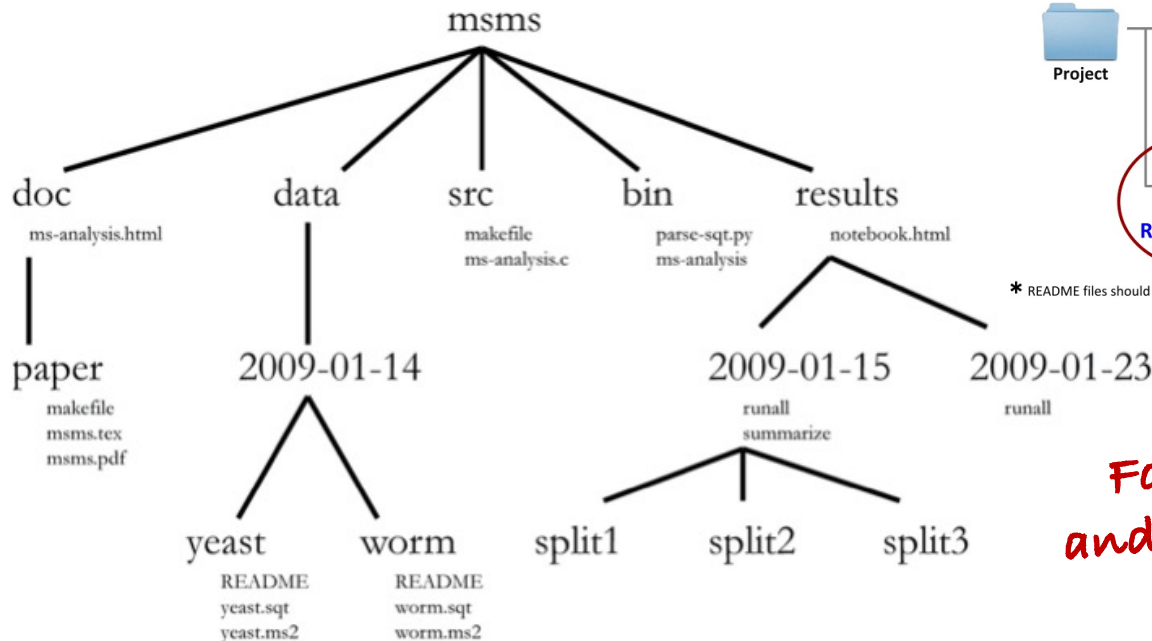


- **Keep original (raw) versions** of data files, or keep documentation that allows the reconstruction of original files
- **Track the location of files** if they are stored in a variety of locations
- Establish **terms and conditions of data use** within the project team and beyond
- Keep a 'master file' of the data and take measures to preserve its authenticity
- Decide **how many and which versions to keep** for how long
- **Document changes that were made in any version**
- **Record relationships between items** where needed, for example between code and the data file it is run against

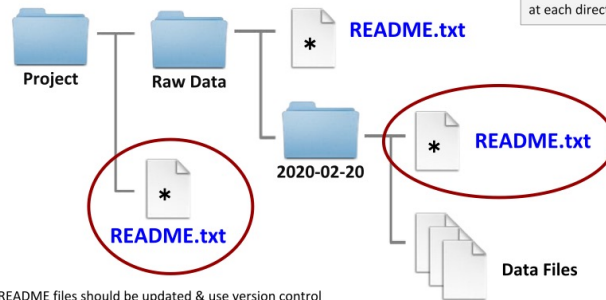
- When working on different (local) workstations, e.g. laptop at home and the desktop in the office:
 - **always make sure that you are working on the most current version, for example with the help of versioning software or guidelines**
 - make sure that the most **current version is always backed up somewhere else**
- Only suitable as a primary storage for projects involving very few people
- Avoid if data will be moved back and forth between personal computers frequently

- Granting **shared, remote and easy access to data and other files** to all involved in the project
- **Read the terms of service.**
Especially focus on rights to use content given to the service provider
- **Opt for European, national, or institutional** cloud services which store data in Europe if possible
- **Not your only storage and backup solution**
- **Not for unencrypted (sensitive) personal data**

Also be careful with passwords and other secrets!



Create Documentation Files



Example of a project folder with README documentation files at each directory level

* README files should be updated & use version control

Folder structure, docs and naming conventions are important!

Example from: Noble WS (2009) A Quick Guide to Organizing Computational Biology Projects. *PLoS Comput Biol* 5(7): e1000424.

<https://doi.org/10.1371/journal.pcbi.1000424>

- **Snapshot projects and files**
Infinite undo, traceability, reproducibility
Software, data, documents, scripts, manuscripts, ...
- **Copy–rename–describe**
Duplicate / rename files and folders
Changes.txt
- **Software assisted**
Projectplace, Google Docs, Sharepoint, Dropbox, but there is more...
- **Collaborative versioning**
On your computer, on the web, from a single line of text to a complete project



NBS
NATIONAL BIOINFORMATICS
INFRASTRUCTURE SWEDEN

Git is free and widely used SciLifeLab

git --distributed-even-if-your-workflow-isnt

Search entire site...

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.

About
The advantages of Git compared to other source control systems.

Documentation
Command reference pages, Pro Git book content, videos and other material.

Downloads
GUI clients and binary releases for all major platforms.

Community
Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release
2.31.1
Release Notes (2021-03-26)
Download 2.31.0 for Mac

<https://git-scm.com/>



Search entire site...

```

$ git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      <command> [<args>]

designed to handle everything from small to very large projects with
distributed version control. It has many client implementations,
and ClearCase with features like cheap local branching, convenient
and ClearCase with features like cheap local branching, convenient
and ClearCase with features like cheap local branching, convenient

These are common Git commands used in various situations:

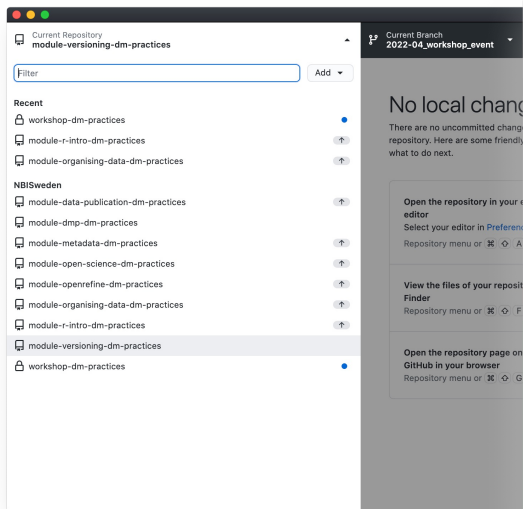
start a working area (see also: git help tutorial)
  clone Clone a repository into a new directory
  init Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add Add file contents to the index
  mv Move or rename a file, a directory, or a symlink
  restore Restore working tree files
  rm Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
  bisect Use binary search to find the commit that introduced a bug
  diff Show changes between commits, commit and working tree, etc
  grep Print lines matching a pattern
  log Show commit logs
  
```

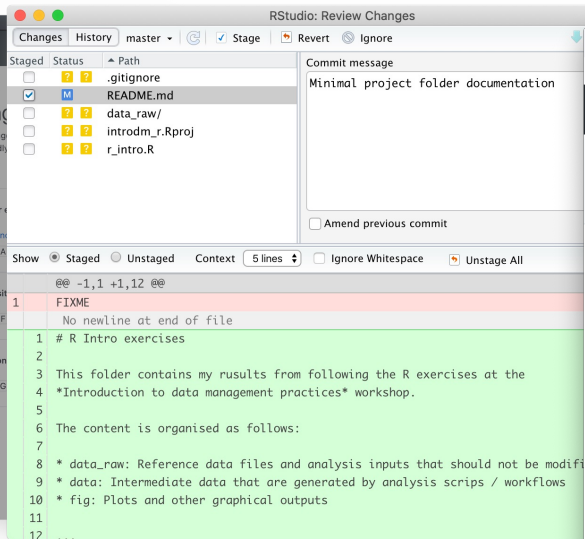


GitHub Desktop



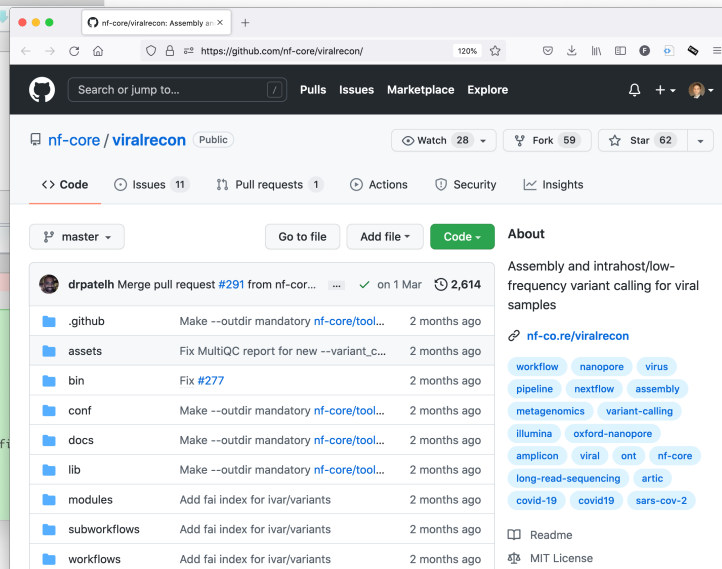
Keeping track of all your local copies in one place

RStudio



Managing changes for a specific file or project type

GitHub.com



Managing collaborations and access to shared versions of your files

- Not a replacement for back-ups
- Write informative commit messages
- Establish conventions for using and naming branches
- Best out-of-the box experience with text-based files (lines)
- Use tags / hashes to reference a specific revision
- Use data archives to preserve important revisions, e.g., Zenodo or Figshare

	COMMENT	DATE
○	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
○	ENABLED CONFIG FILE PARSING	9 HOURS AGO
○	MISC BUGFIXES	5 HOURS AGO
○	CODE ADDITIONS/EDITS	4 HOURS AGO
○	MORE CODE	4 HOURS AGO
○	HERE HAVE CODE	4 HOURS AGO
○	AAAAAAAAA	3 HOURS AGO
○	ADKFJ\$LKDFJ\$DKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

Git Commit by xkcd CC-BY-NC 2.5, <https://xkcd.com/1296/>