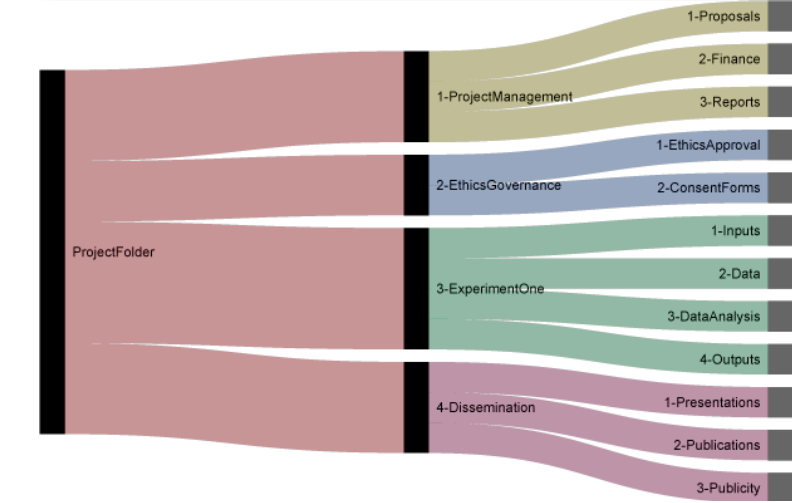
# Data organisation and Documentation

## Folder structure example



<http://nikola.me/folder_structure.html> (folder structure download available on the website)

[See here](https://doi.org/10.5281/zenodo.4410128) for another example

* Make sure you have enough (sub)folders so that files can be stored in the right folder and are not scattered in folders where they don’t belong or stored in large quantities in a single folder (which makes it hard to find the files).
* Make sure the folder structure is clear: you can structure folders based on the person that has generated the data/folder, chronologically (month, year, sessions), per project (as done in the example above), or based on analysis method/equipment/type of data.

See also: [Find Files Faster: How to Organize Files and Folders](https://zapier.com/blog/organize-files-folders/)

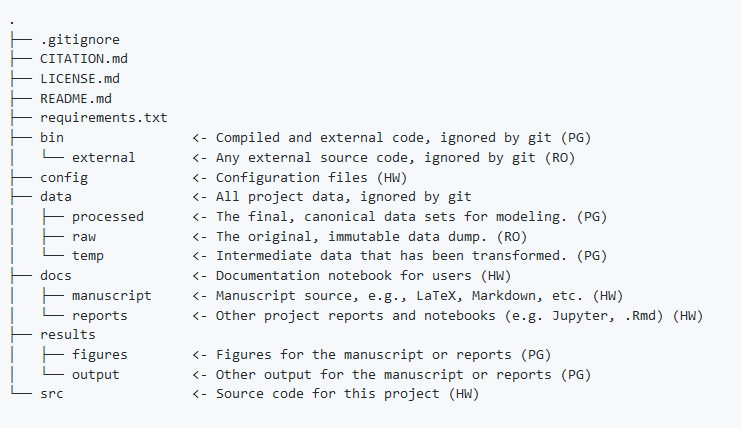
## Cookie Cutter templates

Use the [cookiecutter](https://github.com/cookiecutter/cookiecutter) templates to download folder structures through GitHub. The template by Barbara Vreede follows the template proposed by [Wilson *et al.*](https://doi.org/10.1371/journal.pcbi.1005510) (2017): <https://github.com/bvreede/good-enough-project>

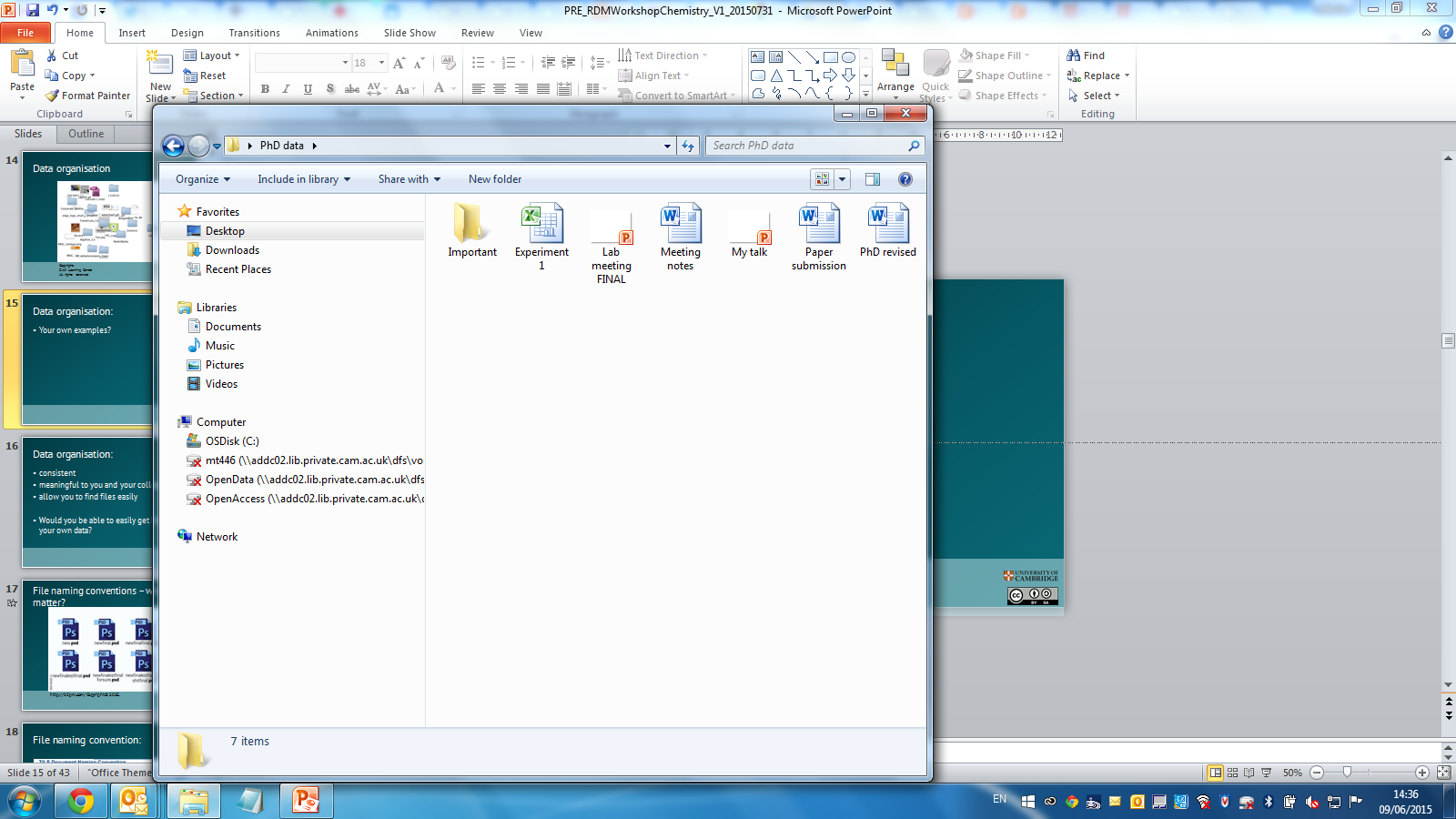
See this video for more information: <https://vimeo.com/462773031>

In general it is good practise to:

* have one folder/directory per project
* to save your raw data separately (and make it read only to ensure that it is not accidentally altered)
* to create a subdirectory for your output
  + for example: Manuscript output, figures, numerical output



## File naming convention



In 3 years’ time would you know what these are?

Structure your file names and set up a template for this. It is very useful to start with the date (when the file was generated: YYYYMMDD) which will sort your files chronologically and also creates a unique identifier for each file. (It will be immediately clear if there are multiple files generated on the same day that will have to be given a version number –or “A, B”-, because otherwise overwriting would occurs if you store these files in the proper folder). Examples:

**20190607\_ImPhys\_RDMTemplate\_v001\_eng**

• Date or date range of experiment: YYYYMMDD

• File type (or language such as the example above)

• Researcher name/initials

• Version number of file (v001, v002)

• Don’t make file names too long (30-70 characters should do the trick)

• Avoid special characters (?\!@\*%{[<>) and spaces

You can explain the file naming convention in a README.txt file, so that it will also become clear to others what the file names mean.

Check [Jenny Bryan’s ‘naming things’ presentation](https://speakerdeck.com/jennybc/how-to-name-files), this [8 step guide](https://resolver.caltech.edu/CaltechAUTHORS:20200601-161923247) on how to set up your file naming convention if you need more guidance.

See also [Danielle Navarro’s videos](https://www.youtube.com/watch?v=u6MiDFvAs9w&list=PLRPB0ZzEYegPiBteC2dRn95TX9YefYFyy&index=2) about file naming and project structure.

### Bulk renaming tools (use with care!):

* Windows ([Ant Renamer](http://www.antp.be/software/renamer), [RenameIT](sourceforge.net/prpjects/renameit), [Bulk Rename Utility](http://www.bulkrenameutility.co.uk/))
* Mac ([Renamer](https://renamer.com/), [Name Changer](https://mrrsoftware.com/namechanger/))
* Linux ([GNOME Commander](https://gcmd.github.io/), [GPRename](http://gprename.sourceforge.net/))
* Unix: Using the grep command to search for regular expressions

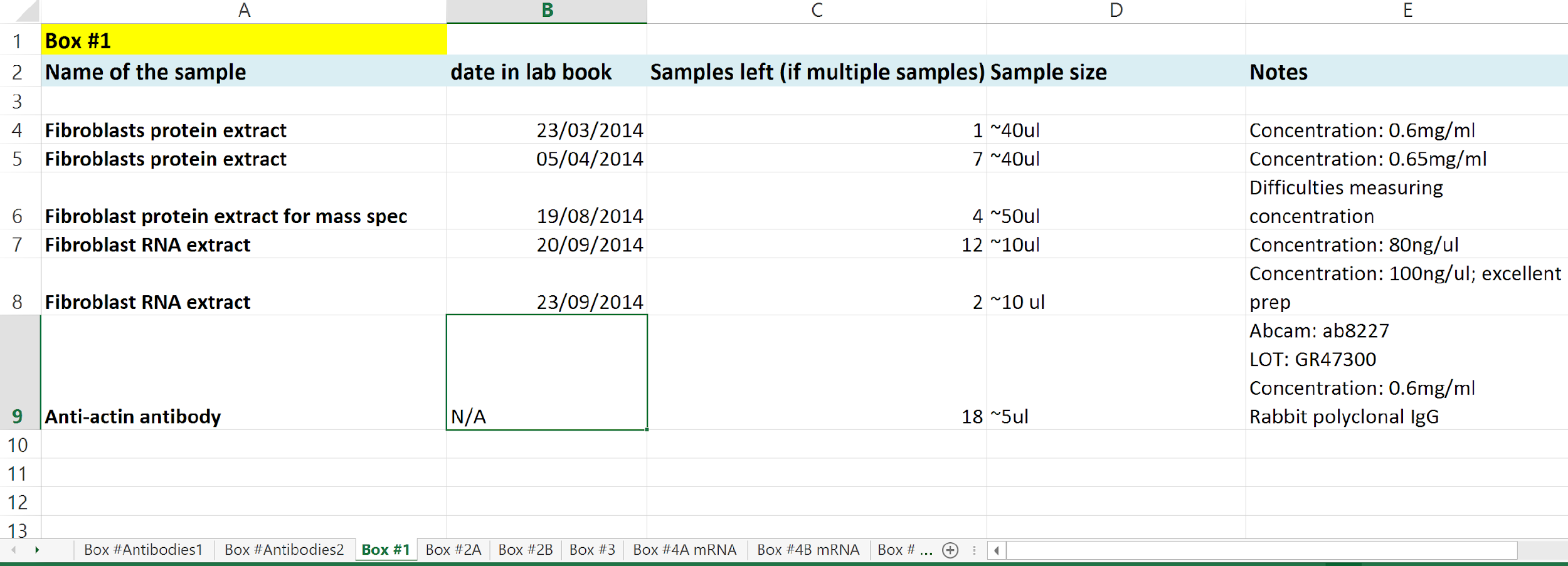
## Documentation

* [Guide for data documentation](https://doi.org/10.5281/zenodo.1914401)
* [Data organisation in spreadsheets](https://uf-repro.github.io/data-organization/slides.html) (see also the [Open Science Framework guidance](https://help.osf.io/hc/en-us/articles/360019739054-How-to-Make-a-Data-Dictionary))
* [Code Book](https://libguides.library.kent.edu/SPSS/Codebooks)
* [Guide on how to create data dictionaries](https://doi.org/10.1177/2515245920928007)

## Physical samples

Reference your samples: dates in notebooks + supplier’s name/code

Add any relevant notes:



(copyright by Marta Teperek)

## Version Control

For version control of code the use of Git (GitHub) is recommended. For an internal version of GitHub (so not shared outside of the department/TU Delft) you can use the TU Delft instance of [GitLab](https://tudelft.topdesk.net/tas/public/ssp/content/detail/service?unid=888da653ddc04d01b2d5db413fc99a42). For version control for data(sets) Subversion can be used (apply through [Top Desk](https://tudelft.topdesk.net/tas/public/ssp/content/detail/service?unid=9db38a0434fc4453a9bfcde1436d18aa)). For version control of files you can use Google Docs and OneDrive.

You can also make use of electronic lab notebooks (ELNs). TU Delft provides licenses for [eLABjournal](https://www.elabjournal.com/) and[Rspace](https://www.researchspace.com/) (please contact Esther, [e.plomp@tudelft.nl](mailto:e.plomp@tudelft.nl), to get started!).

## Metadata

Metadata is information about your data, such as the title of the dataset, the date, creator(s) and keywords that describe the data. Metadata standards with defined fields ensure machine readability. To look for disciplinary standards you can use the resources below:

* [FAIRsharing.org](https://fairsharing.org/)
* [Research Data Alliance metadata directory](https://rd-alliance.github.io/metadata-directory/standards/)
* [Digital Curation Center](http://www.dcc.ac.uk/resources/metadata-standards)

## Readme text files

Readme text files should describe the methods used for data collection and analysis and include data/software-specific information (parameters, variables, column headings, symbols used, etc.). See <https://www.makeareadme.com/> for more information on why readme files are important and how you can set up your own readme files.

Examples for data:

* <https://cornell.app.box.com/v/ReadmeTemplate>
* <https://researchdata.4tu.nl/fileadmin/user_upload/Documenten/Guidelines_for_creating_a_README_file.pdf>
* <https://datamanagement.hms.harvard.edu/collect/readme-files>

Examples for software:

* <https://github.com/Spaak/contextual-cueing-meg/blob/master/README.md>
* <https://github.com/benmarwick/rrtools>
* <https://gist.github.com/jxson/1784669>
* <https://ha0ye.github.io/CW21-README-tips/template_README.html>
* <https://github.com/othneildrew/Best-README-Template>

See this [short presentation by Carlos Martinez](https://www.youtube.com/watch?v=Jgv34Kwgga8&list=PL1CvC6Ez54KDvJbbdLn5rPvf1kInifEh9&index=7) about the importance of README files

### DMPonline: <https://dmponline.tudelft.nl/>

Digital platform to set up your data management plan (DMP), using templates by TU Delft, NWO, H2020. You can collaborate with others by inviting them to be an editor of your DMP. Support from your Faculty’s Data Steward is only one click away with the “request feedback” button.

## Mistakes

[Error Tight](https://www.carleton.edu/perception-lab/open-science/error-tight/) – Exercises for lab groups to prevent research mistakes