Initialize Manual

Table of contents

# Project Handbook

This manual summarizes information for the following:

* the project resources
* recommended good practices for project organization
* introduces the key project tools

## Using Github

### Working with Git

### Basic workflow

## Containers: Docker & Singularity

### How to setup a container

### How to run a container

## CSC resources

## Organizing a project

This is a generic project template including good practices for code organization. The aim is to enable painless internal reproduction of a project and to ease communication about a project’s structure.

**Most of the features here are recommendations, and can be varied on as needed.**

### Repository name

A proposal for unified naming scheme for publication related repositories is as follows: document type\_date\_project name. An example would be: article\_2023\_kickoff. Date should follow the format YYYYMMDD, with month and day optional, and would probably refer to the projected or actual end date of the project. The date -element can also be optional, to be included only if relevant, eg. article\_kickoff would be equally valid.

### Practices

* **Project overview documentation:**
  + Should reside in the project root in a README.md (this file).
  + Should list people involved and their roles in the project.
* **Naming files and folders:**
  + Use all lowercase (except for established standards such as README.md and the .R filename extension).
  + Separate words in file and directory names by underscore: \_. eg. my\_project.R instead of my-project.R or MyProject.R.
* **Structure:**
  + Follow the directory structure laid out below.
  + Include README.md in each directory documenting the contents of that directory.
    - This is especially important in data and final code directories.
  + If feasible, to avoid confusion only use single .gitattributes and single .gitignore file residing in the project root.

### Directory structure

The project repository structured is variation of formats laid out in a few data science project organization articles (see the end of this README). code and output -directories include work/ and final/ -subdirectories. The work/ -subdirectory is optional, but helps to keep development material separate from the polished and clean end products that should reside in the final/ directory.

project\_name/  
├── README.md # project overview  
├── documentation/ # project documentation  
├── input/  
│ ├── data\_raw/ # immutable raw input data  
│ ├── data\_work/ # intermediate data  
│ └── data\_processed/ # processed data for final analysis tasks  
├── code/  
│ ├── work/  
│ │ ├── person1/ # use first name or github user name  
│ │ ├── person2/ # a directory for each person or task  
│ │ └── task1/ # etc ...  
│ └── final/  
│ ├── task1/ # a directory for each analysis task  
│ └── another\_task/ # etc ...  
└── output/  
 ├── figures/  
 │ ├── work/  
 │ └── final/  
 └── publications/  
 ├── work/  
 └── final/

### Logic

* **[documentation/]:** Project meta documentation. Links to all relevant planning papers, interim notes, google drive folders, etc.
* **[input/]:** Input data. Either a whole dataset or if that is impractical, a link pointing to the data source (likely another repository). *[data\_raw/]* subdirectory should have immutable original input data and/or references to the repositories where it can be retrieved from. *[data\_processed/]* holds data that has been processed to analysis ready format and should include README.md pointing to the code that is used to produce the data. *[data\_work/]* is a development directory for work-in-progress datasets. Ideally, all datasets should be producible by scripts from the raw data.
* **[code/]:** Data processing code. Finished code used for publication should be moved to *[final/]* subdirectory. Organization of the development directory *[work/]* can vary and the breakdown by person or task is just a suggestion. All directories, but especially *[final/]* should include a README.md clearly documenting what each script does.
* **[output/]:** Both figures and publication texts/files. Divided to work and final subdirectories.

### Articles on data science project git repo organization

* [PLoS Comput Biol. 2016 Jul; 12(7): **Ten Simple Rules for Taking Advantage of Git and GitHub**. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4945047/](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4945047/)
* [Human in a Machine World. May 25, 2016: **Folder Structure for Data Analysis**. https://medium.com/human-in-a-machine-world/folder-structure-for-data-analysis-62a84949a6ce](https://medium.com/human-in-a-machine-world/folder-structure-for-data-analysis-62a84949a6ce)
* [**Cookiecutter Data Science** - A logical, reasonably standardized, but flexible project structure for doing and sharing data science work. https://drivendata.github.io/cookiecutter-data-science/](https://drivendata.github.io/cookiecutter-data-science/)
* [Thinking on Data. December 9, 2018: **Best practices organizing data science projects**. https://www.thinkingondata.com/how-to-organize-data-science-projects/](https://www.thinkingondata.com/how-to-organize-data-science-projects/)