OHBM BRAINHACK TRAINTRACK

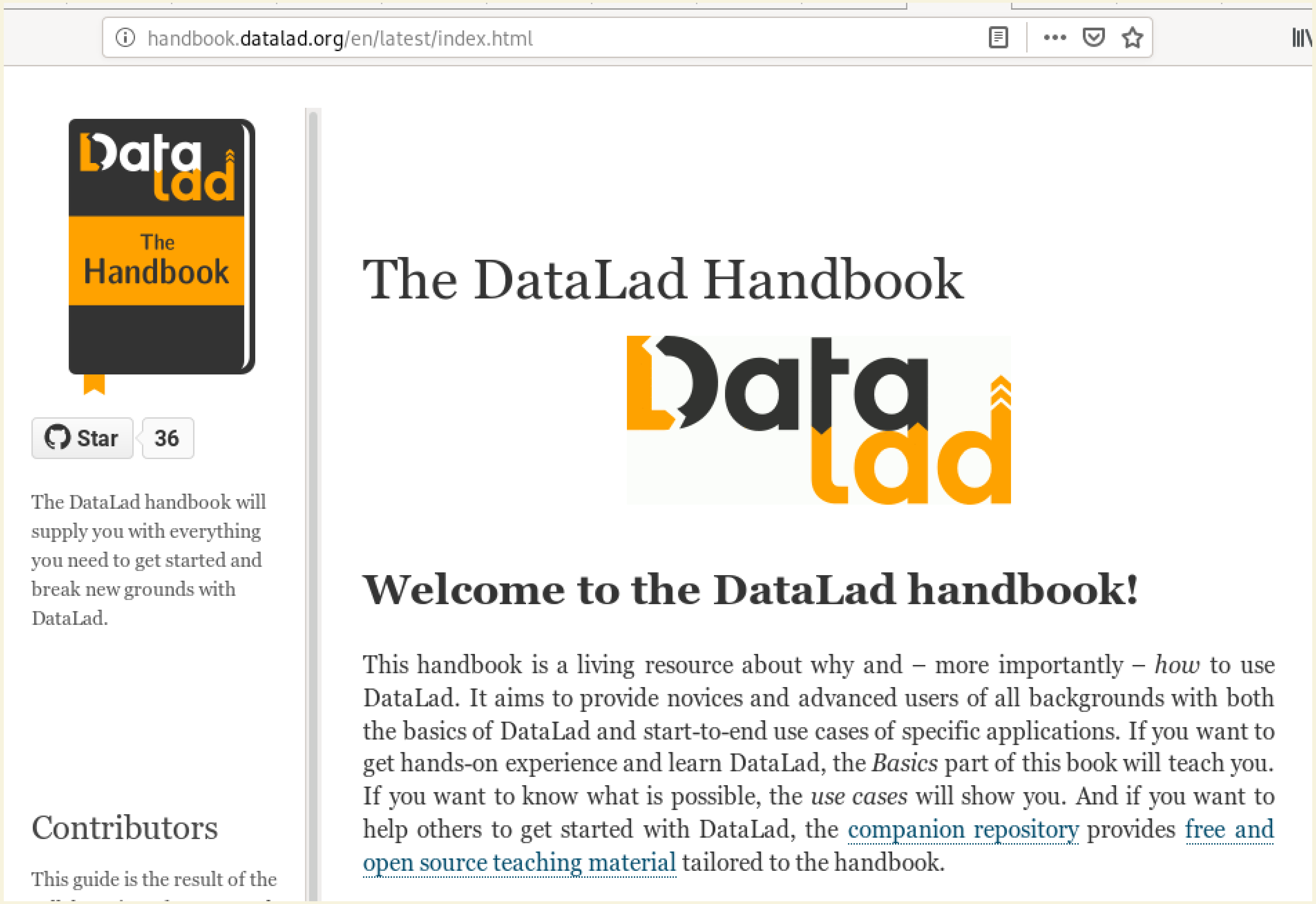
AN INTRODUCTION TO DATALAD

Adina Wagner
@AdinaKrik

Psychoinformatics lab,
Institute of Neuroscience and Medicine, Brain & Behavior (INM-7)
Research Center Jülich

Slides: https://github.com/datalad-handbook/course/

LEARN ALL ABOUT DATALAD AT HANDBOOK.DATALAD.ORG



The DataLad handbook will supply you with everything you need to get started and break new grounds with DataLad.

Contributors

This guide is the result of the

The DataLad Handbook

Welcome to the DataLad handbook!

This handbook is a living resource about why and – more importantly – *how* to use DataLad. It aims to provide novices and advanced users of all backgrounds with both the basics of DataLad and start-to-end use cases of specific applications. If you want to get hands-on experience and learn DataLad, the *Basics* part of this book will teach you. If you want to know what is possible, the *use cases* will show you. And if you want to help others to get started with DataLad, the companion repository provides free and open source teaching material tailored to the handbook.

DataLad IN BRIEF

- A command-line tool with Python API
- Build on top of Git and Git-annex
- **Allows...**

  ... version-controlling arbitrarily large content,

  ... easily sharing and obtaining data (note: no data hosting!),

  ... (computationally) reproducible data analysis,

  ... and *much* more
- Completely domain-agnostic
- available for all major operating systems (Linux, macOS/OSX, Windows)

STEP 1: INSTALL DATALAD

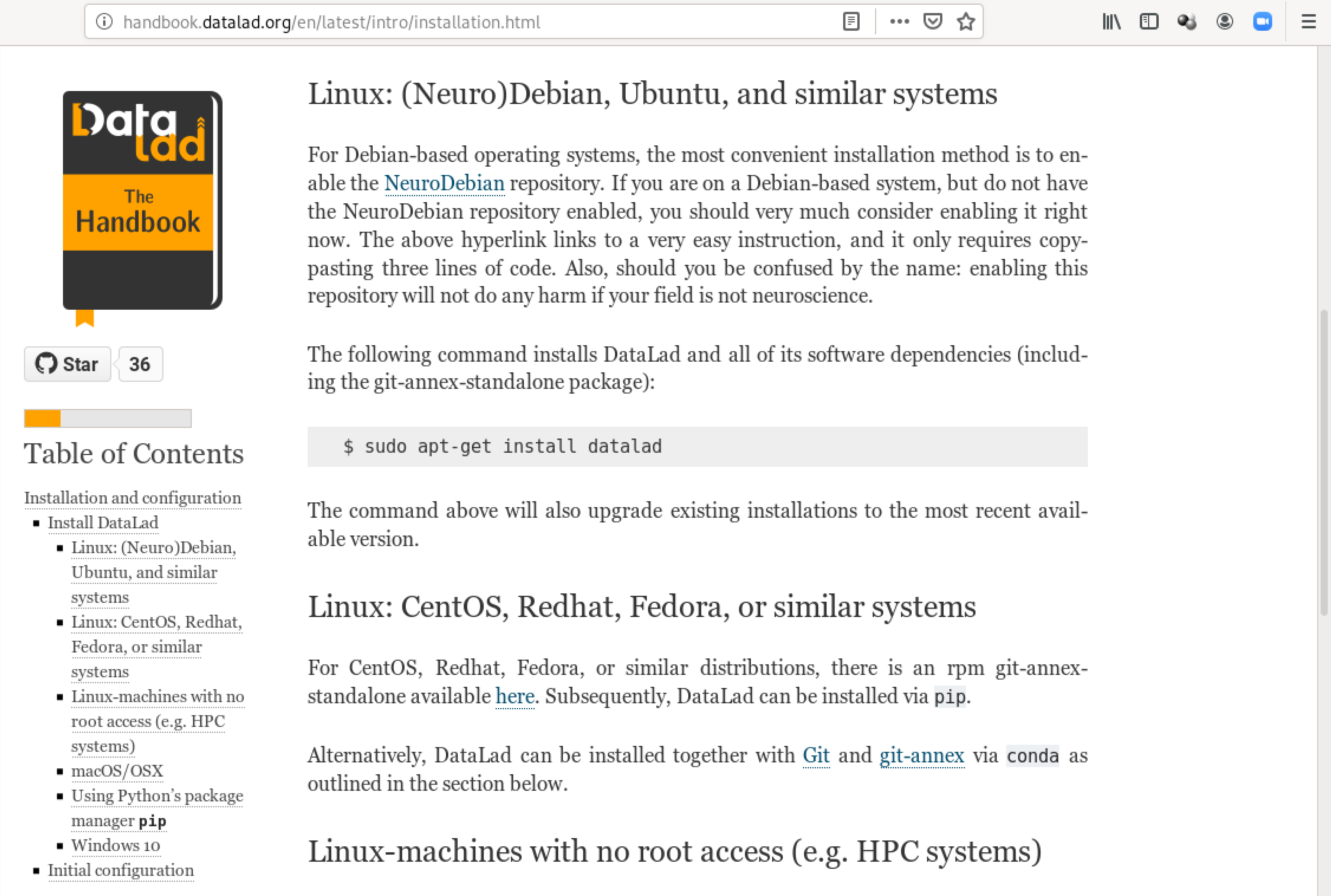


Table of Contents

Linux: (Neuro)Debian, Ubuntu, and similar systems

For Debian-based operating systems, the most convenient installation method is to enable the NeuroDebian repository. If you are on a Debian-based system, but do not have the NeuroDebian repository enabled, you should very much consider enabling it right now. The above hyperlink links to a very easy instruction, and it only requires copy-pasting three lines of code. Also, should you be confused by the name: enabling this repository will not do any harm if your field is not neuroscience.

The following command installs DataLad and all of its software dependencies (including the git-annex-standalone package):

```
$ sudo apt-get install datalad
```

The command above will also upgrade existing installations to the most recent available version.

Linux: CentOS, Redhat, Fedora, or similar systems

For CentOS, Redhat, Fedora, or similar distributions, there is an rpm git-annex-standalone available here. Subsequently, DataLad can be installed via `pip`.

Alternatively, DataLad can be installed together with Git and git-annex via `conda` as outlined in the section below.

Linux-machines with no root access (e.g. HPC systems)

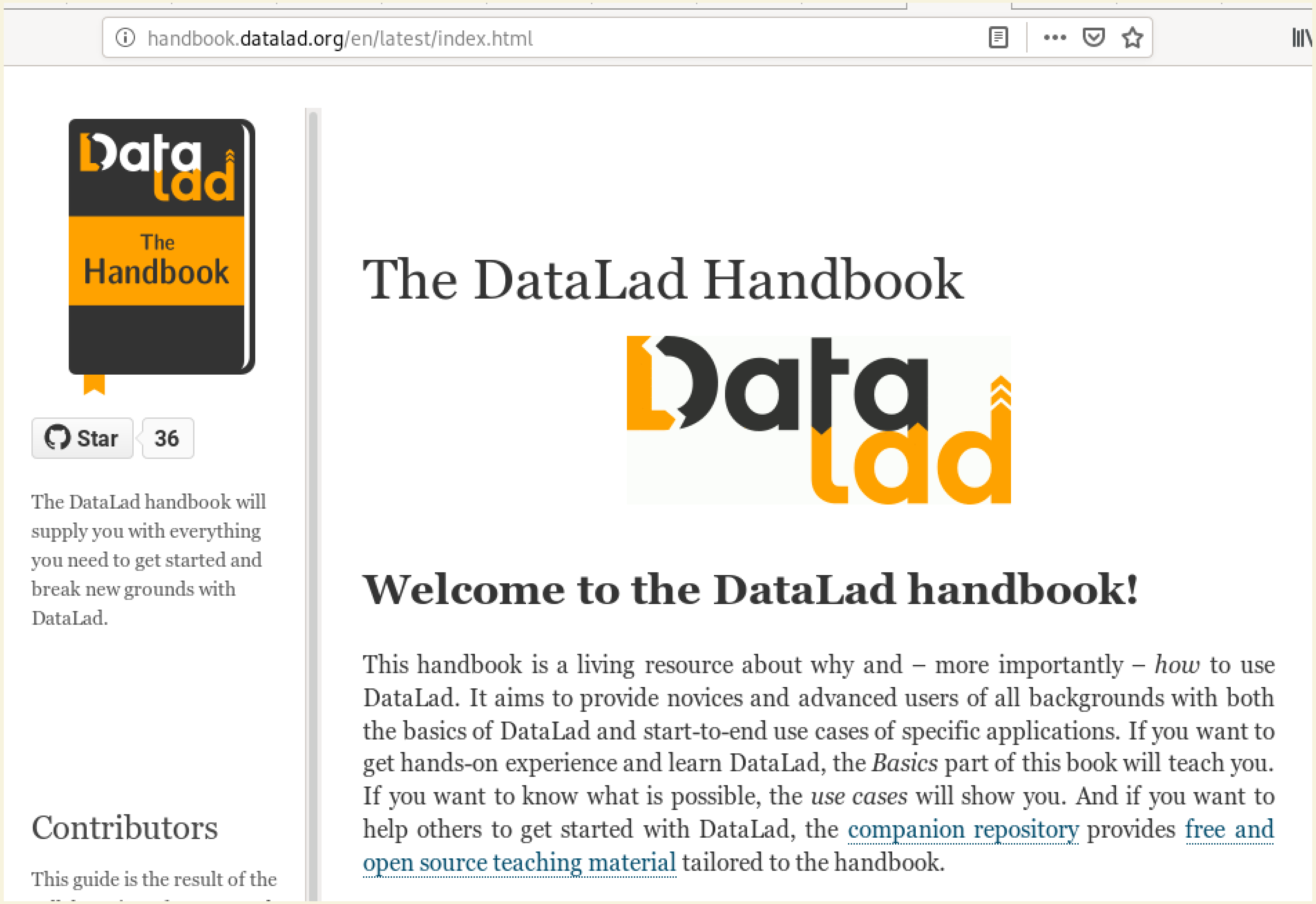
STEP 2: CONFIGURE YOUR GIT IDENTITY

>

```
git config --global --add user.name "Firstname Lastname"
git config --global --add user.email "some@email.com"
```

LET'S START!

FOLLOW ALONG!



The DataLad handbook will supply you with everything you need to get started and break new grounds with DataLad.

Contributors

This guide is the result of the

The DataLad Handbook

Welcome to the DataLad handbook!

This handbook is a living resource about why and – more importantly – *how* to use DataLad. It aims to provide novices and advanced users of all backgrounds with both the basics of DataLad and start-to-end use cases of specific applications. If you want to get hands-on experience and learn DataLad, the *Basics* part of this book will teach you. If you want to know what is possible, the *use cases* will show you. And if you want to help others to get started with DataLad, the companion repository provides free and open source teaching material tailored to the handbook.

Code to follow along:
http://handbook.datalad.org/en/latest/code_from_chapters/OHBM.html

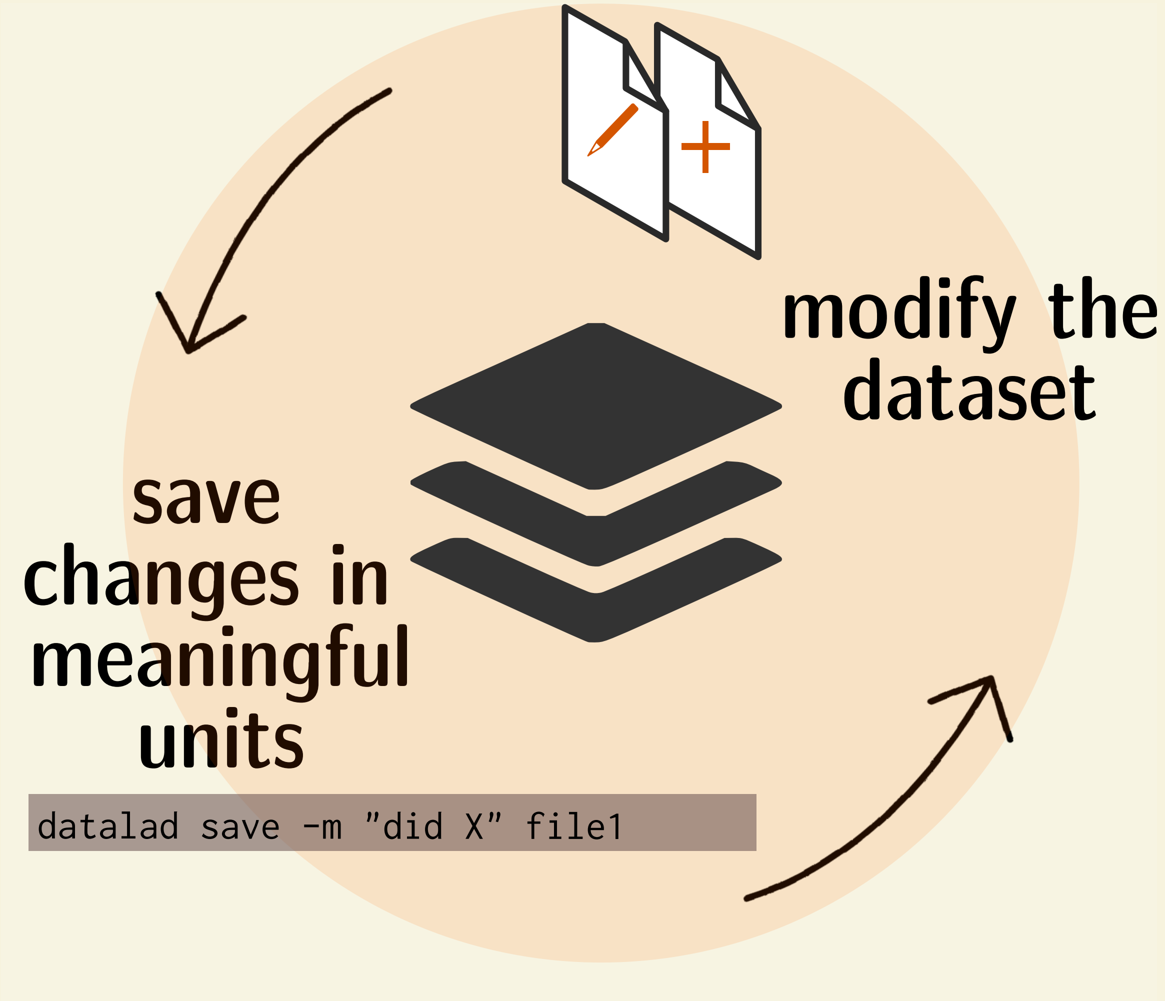
DATALAD DATASETS

- DataLad's core data structure
  - Dataset = A directory managed by DataLad
  - Any directory of your computer can be managed by DataLad.
  - Datasets can be *created* (from scratch) or *installed*
  - Datasets can be nested: *linked subdirectories*

LOCAL VERSION CONTROL

LOCAL VERSION CONTROL

Procedurally, version control is easy with DataLad!



```
datalad save -m "did X" file1
```

Advice:

- Save *meaningful* units of change
- Attach helpful commit messages

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful.

A dataset has a *history* to track files and their modifications.

Explore it with Git (**git log**) or external tools (e.g., **tig**).

`datalad save` records the dataset or file state to the history.

Concise **commit messages** should summarize the change for future you and others.

`datalad status` reports the current state of the dataset.

A clean dataset status is good practice.

FROM HERE

TO THIS:

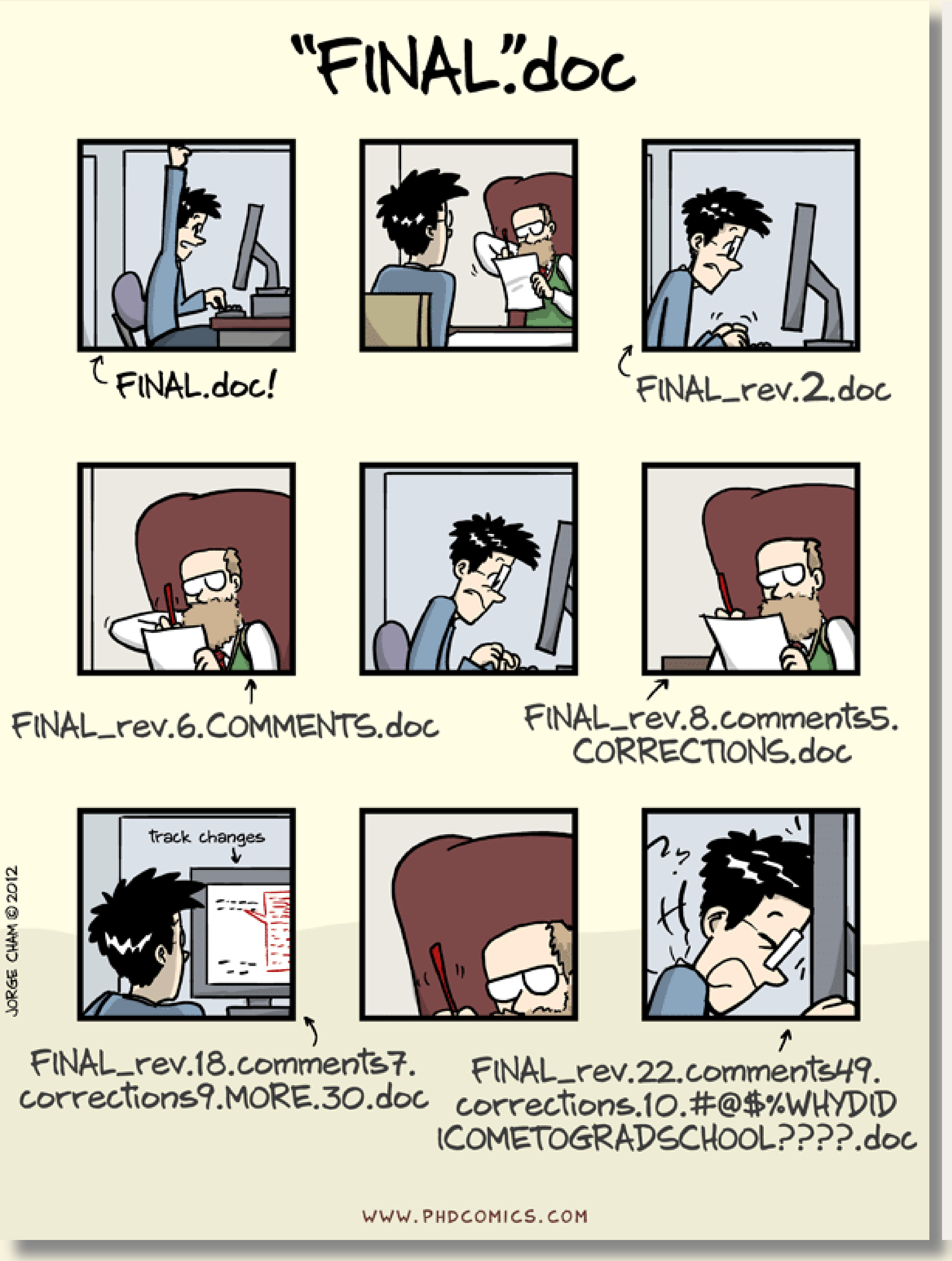
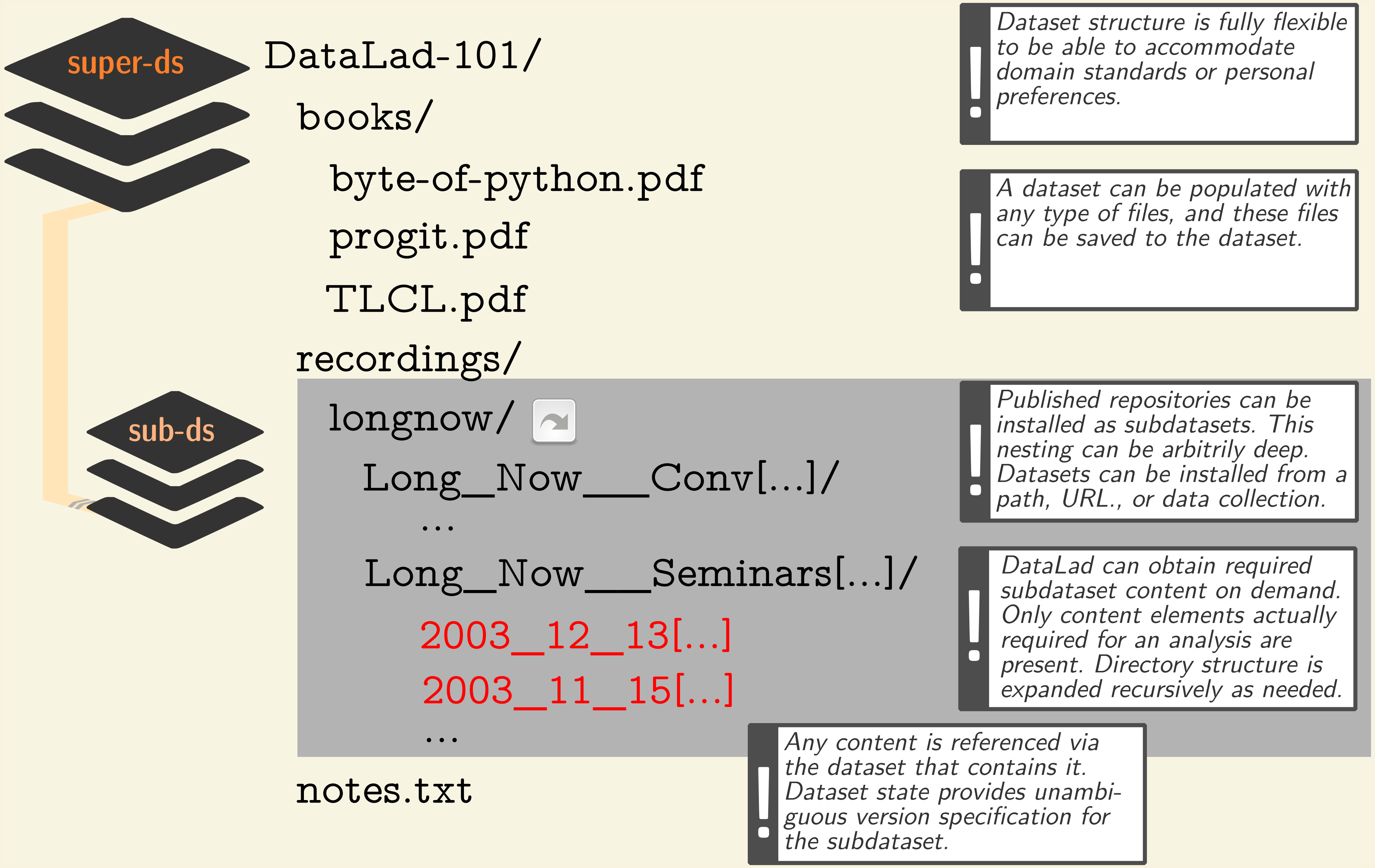


Image credit: www.phdcomics.com; www.linode.com

CONSUMING DATASETS AND DATASET NESTING

CONSUMING DATASETS



```
DataLad-101/
  books/
    byte-of-python.pdf
    progit.pdf
    TLCL.pdf
  recordings/
    longnow/
      Long_Now__Conv[...]/
        ...
      Long_Now__Seminars[...]/
        2003_12_13[...]
        2003_11_15[...]
        ...
  notes.txt
```

! *Dataset structure is fully flexible to be able to accommodate domain standards or personal preferences.*

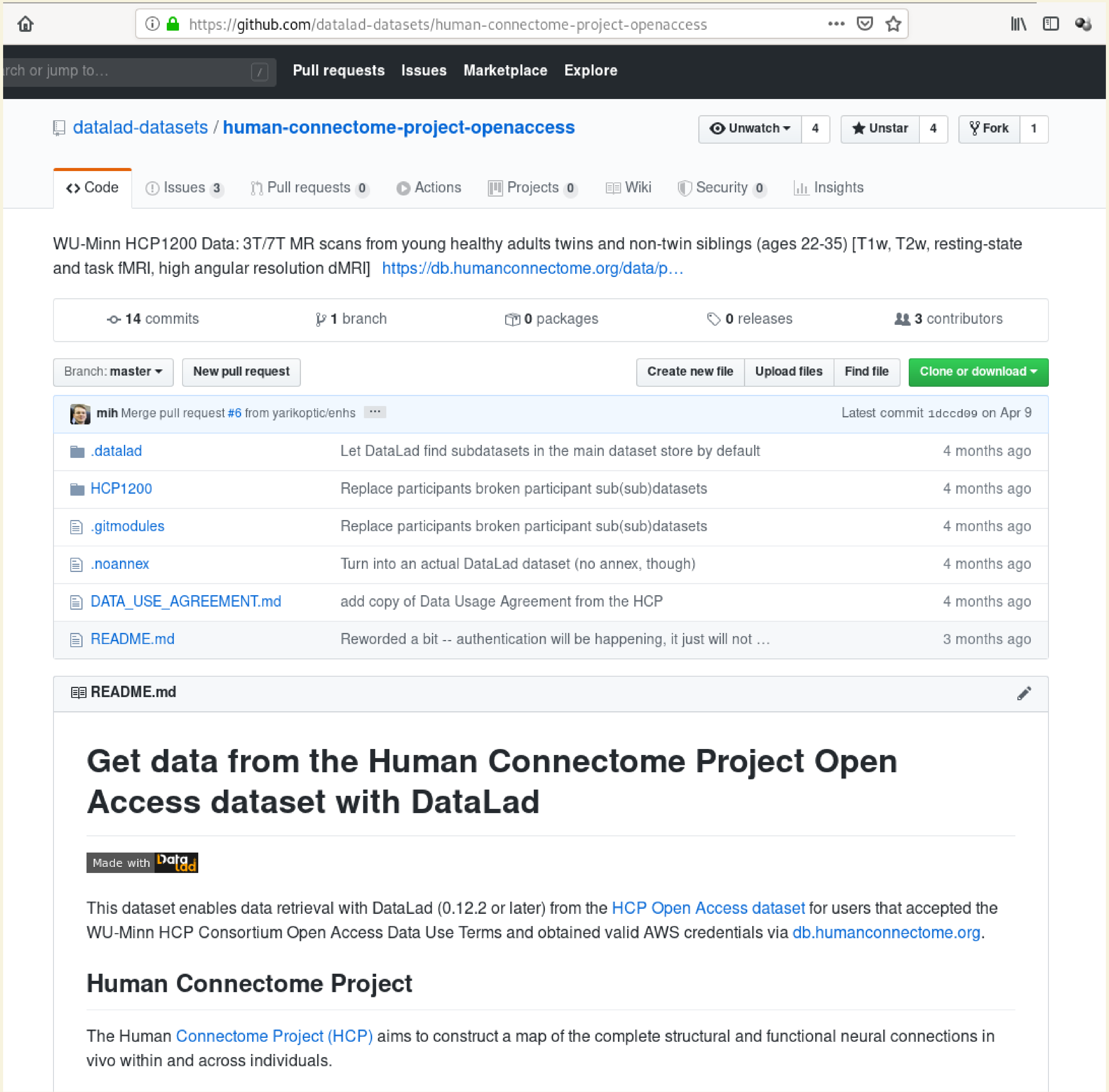
! *A dataset can be populated with any type of files, and these files can be saved to the dataset.*

! *Published repositories can be installed as subdatasets. This nesting can be arbitrily deep. Datasets can be installed from a path, URL., or data collection.*

! *DataLad can obtain required subdataset content on demand. Only content elements actually required for an analysis are present. Directory structure is expanded recursively as needed.*

! *Any content is referenced via the dataset that contains it. Dataset state provides unambiguous version specification for the subdataset.*

- Datasets are light-weight: Upon installation, only small files and meta data about file availability are retrieved.
- Content can be obtained on demand via `datalad get`.



WU-Minn HCP1200 Data: 3T/7T MR scans from young healthy adults twins and non-twin siblings (ages 22-35) [T1w, T2w, resting-state and task fMRI, high angular resolution dMRI] https://db.humanconnectome.org/data/p…

mih Merge pull request #6 from yarikoptic/enhs … Latest commit 1dccd09 on Apr 9

.datalad	Let DataLad find subdatasets in the main dataset store by default	4 months ago
HCP1200	Replace participants broken participant sub(sub)datasets	4 months ago
.gitmodules	Replace participants broken participant sub(sub)datasets	4 months ago
.noannex	Turn into an actual DataLad dataset (no annex, though)	4 months ago
DATA_USE_AGREEMENT.md	add copy of Data Usage Agreement from the HCP	4 months ago
README.md	Reworded a bit -- authentication will be happening, it just will not …	3 months ago

README.md

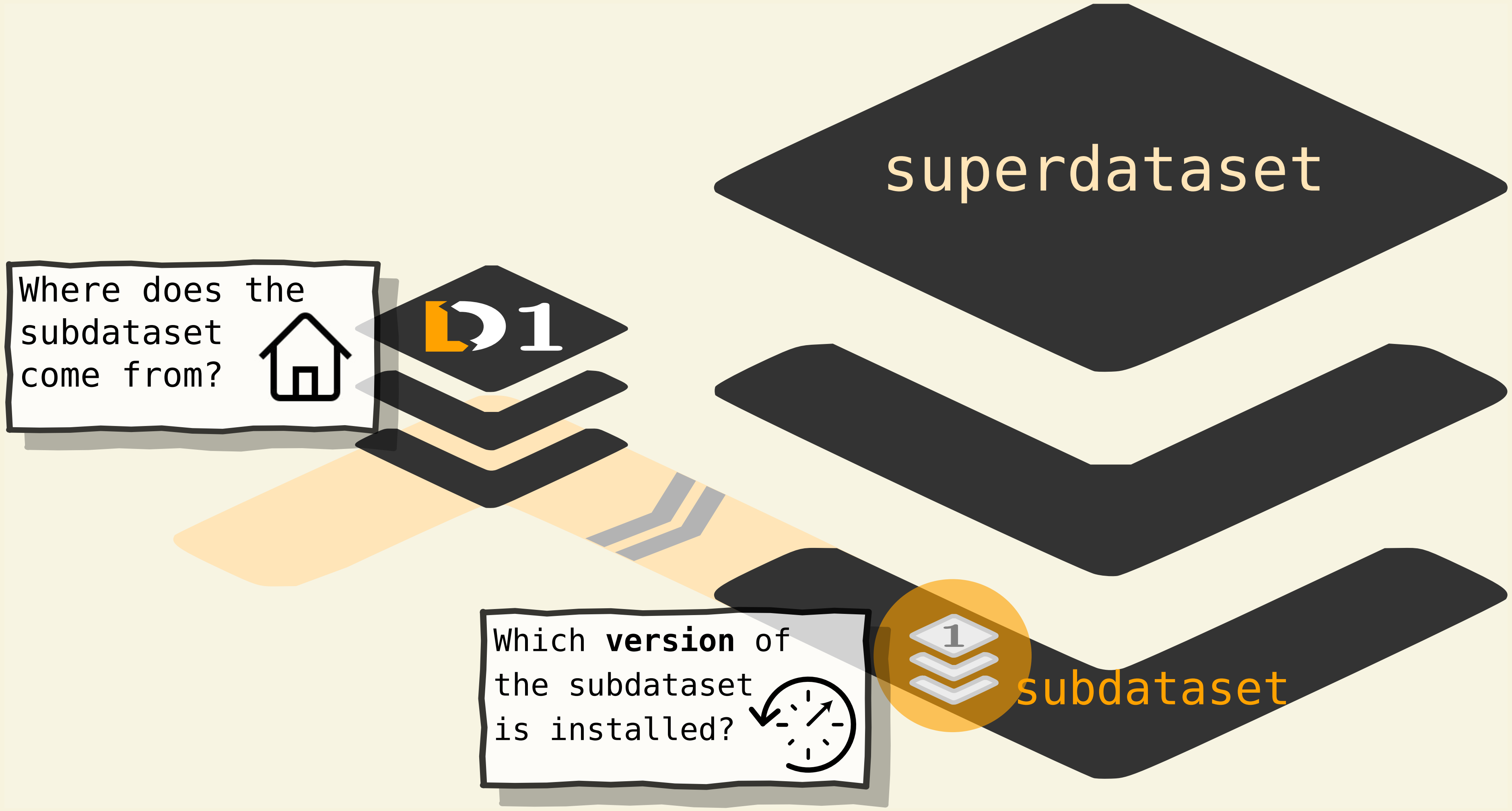
Get data from the Human Connectome Project Open Access dataset with DataLad

This dataset enables data retrieval with DataLad (0.12.2 or later) from the HCP Open Access dataset for users that accepted the WU-Minn HCP Consortium Open Access Data Use Terms and obtained valid AWS credentials via db.humanconnectome.org.

Human Connectome Project

The Human Connectome Project (HCP) aims to construct a map of the complete structural and functional neural connections in vivo within and across individuals.

DATASET NESTING



superdataset
Where does the subdataset come from?
1
Which **version** of the subdataset is installed?
1
subdataset

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

It can be installed "on its own": Specify the source (url, path, ...) of the dataset, and an optional **path** for it to be installed to.

Datasets can be installed as subdatasets within an existing dataset.

The **--dataset/-d** option needs a path to the root of the superdataset.

Only small files and metadata about file availability are present locally after an install.

To retrieve actual file content of larger files, `datalad get` downloads large file content on demand.

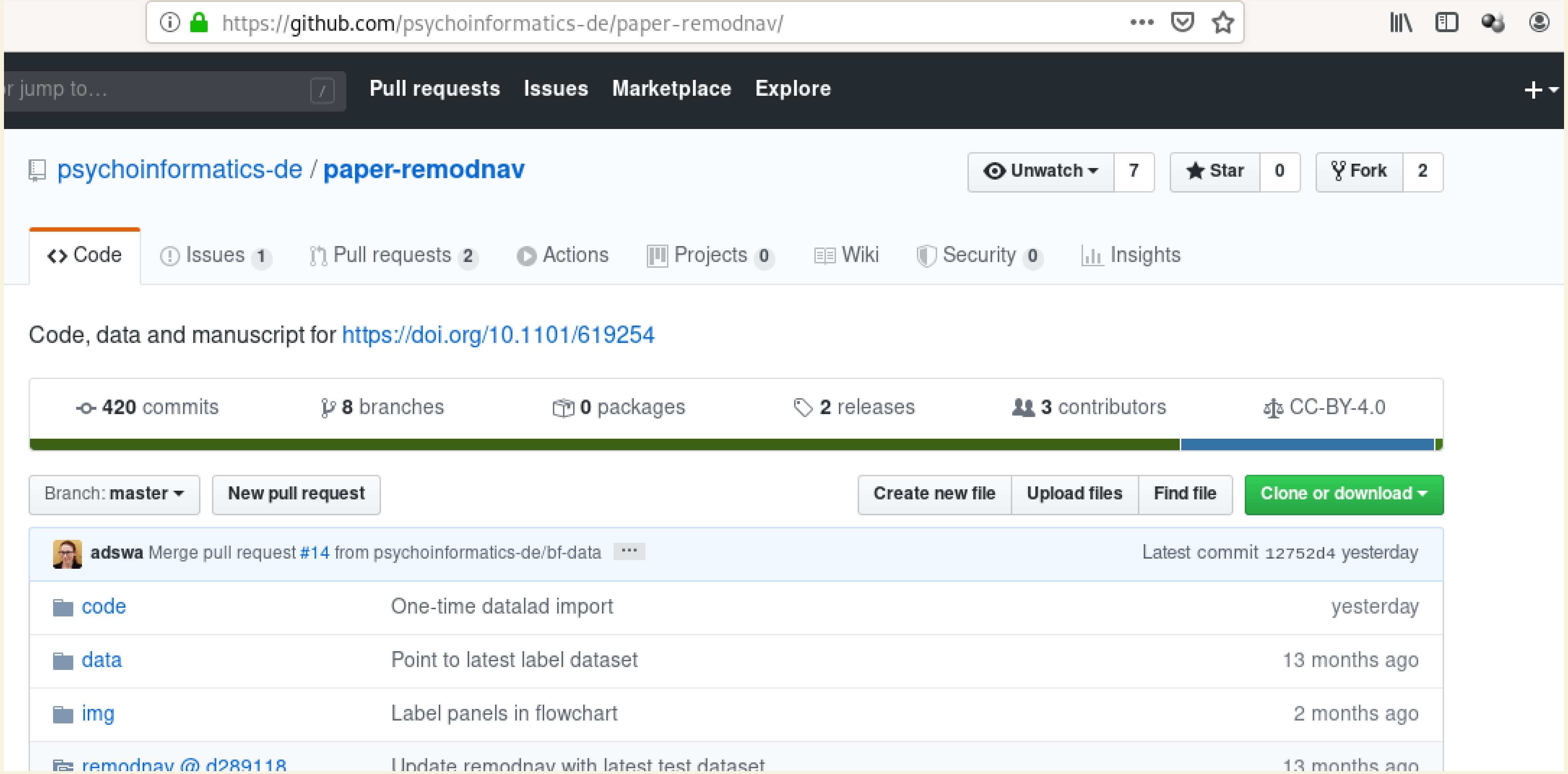
- Content can be dropped to save disk space with `datalad drop.`

  Do this only if content can be easily reobtained.

Datasets preserve their history.

In nested datasets, the superdataset records only the *version state* of the subdataset.

EXAMPLE: REPRODUCIBLE RESEARCH OBJECTS



Code, data and manuscript for https://doi.org/10.1101/619254

420 commits · 8 branches · 0 packages · 2 releases · 3 contributors · CC-BY-4.0

adswa Merge pull request #14 from psychoinformatics-de/bf-data … Latest commit 12752d4 yesterday

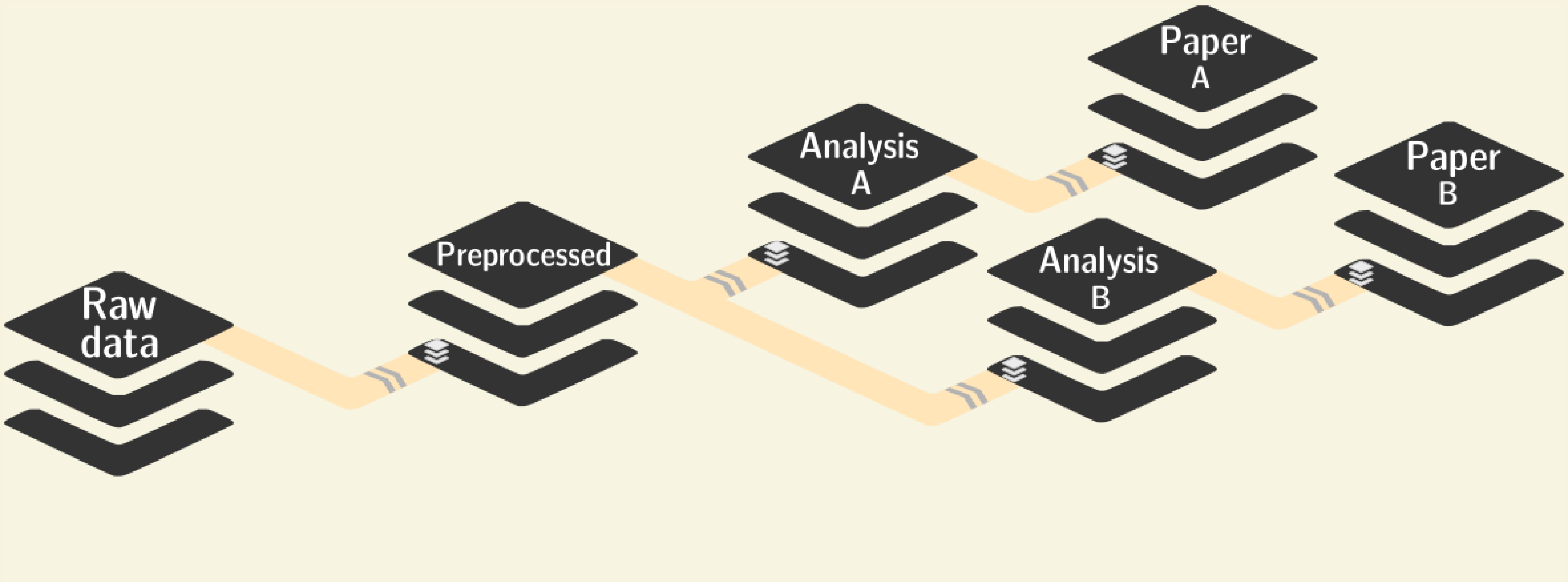
code	One-time datalad import	yesterday
data	Point to latest label dataset	13 months ago
img	Label panels in flowchart	2 months ago
remodnav @ d289118	Update remodnav with latest test dataset	13 months ago

Find this repo at github.com/psychoinformatics-de/paper-remodnav

Read all about it at handbook.datalad.org/en/latest/usecases/reproducible-paper.html

ADVANTAGES OF NESTING

- A modular structure makes individual components (with their respective provenance) reusable.
- Nesting can flexibly link all components and allows recursive operations across dataset boundaries
- Read all about this in the chapter on YODA principles



REPRODUCIBLE DATA ANALYSIS

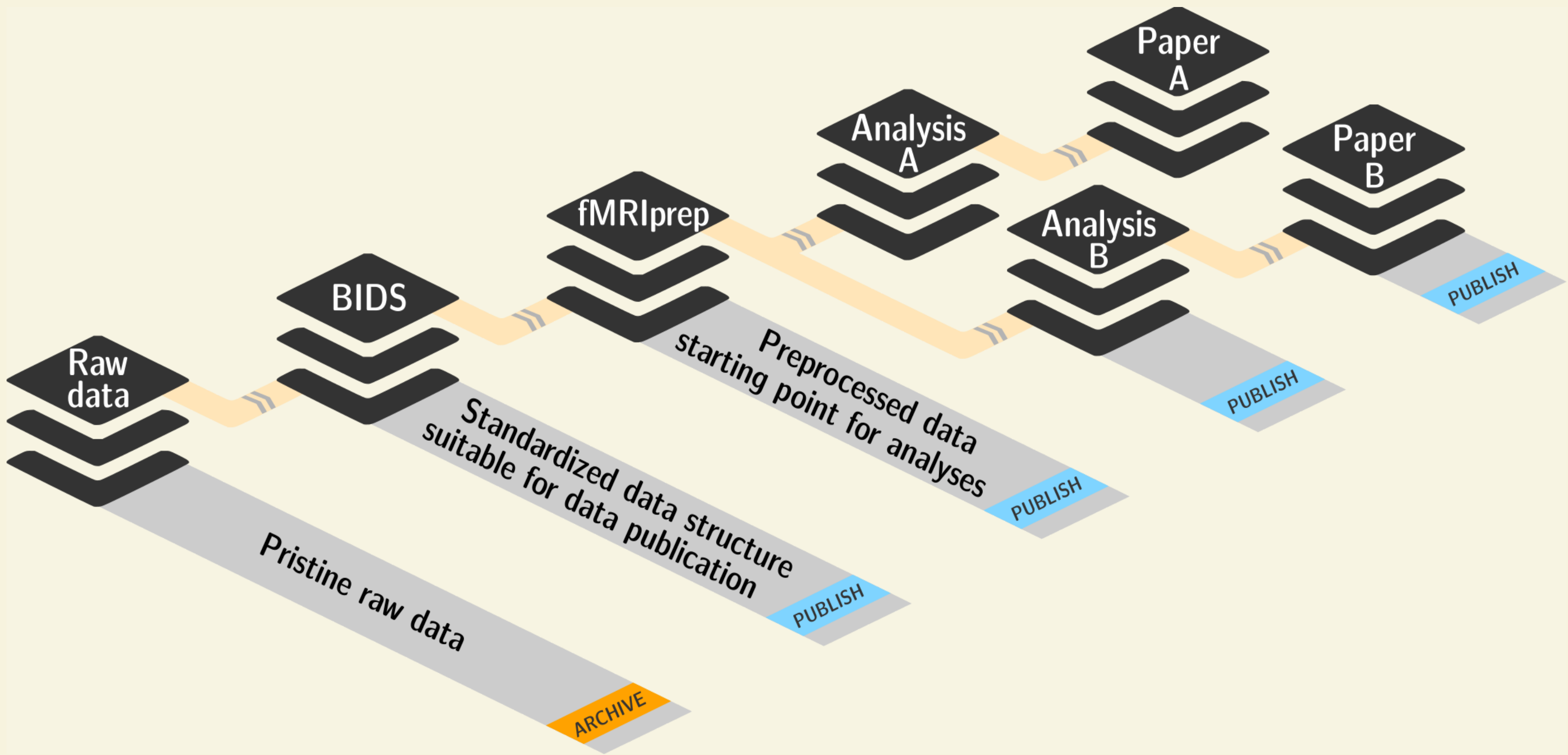
DON'T WORRY, YOU DON'T HAVE TO START YOUR CODE FROM SCRATCH.
YOU CAN RE-USE THE SOFTWARE THAT you wrote yourself SEVERAL YEARS AGO.
ARE THERE INSTRUCTIONS FOR HOW TO USE IT?
I DOUBT IT.
IS THE CODE COMMENTED?
NOT LIKELY.
WHERE ARE THE FILES?
WHO KNOWS.
JORGE CHAM © 2014
THIS IS GOING TO BE PAINFUL, ISN'T IT?
JUST A SCRATCH.
WWW.PHDCOMICS.COM

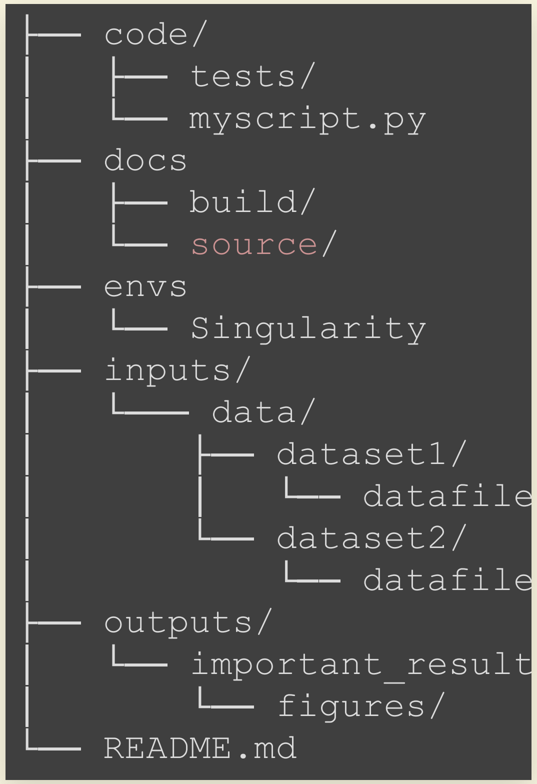
Image credit: Full comic at http://phdcomics.com/comics.php?f=1979

BASIC ORGANIZATIONAL PRINCIPLES FOR DATASETS

Read all about this in the chapter on YODA principles

- Keep everything clean and modular





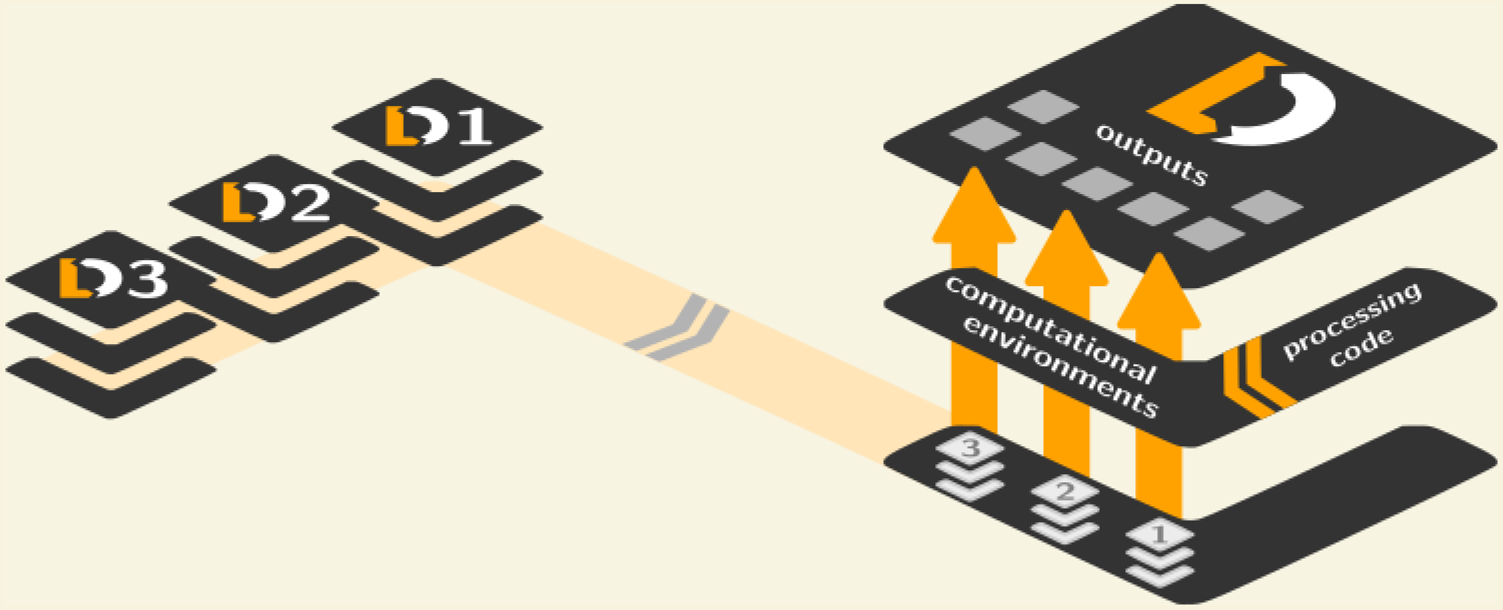
```
├── code/
│   ├── tests/
│   └── myscript.py
├── docs
│   ├── build/
│   └── source/
├── envs
│   └── Singularity
├── inputs/
│   └─── data/
│        ├── dataset1/
│        │   └── datafile
│        └── dataset2/
│            └── datafile
├── outputs/
│   └── important_result
│       └── figures/
└── README.md
```

- do not touch/modify raw data: save any results/computations *outside* of input datasets
- Keep a superdataset self-contained: Scripts reference subdatasets or files with *relative paths*

BASIC ORGANIZATIONAL PRINCIPLES FOR DATASETS

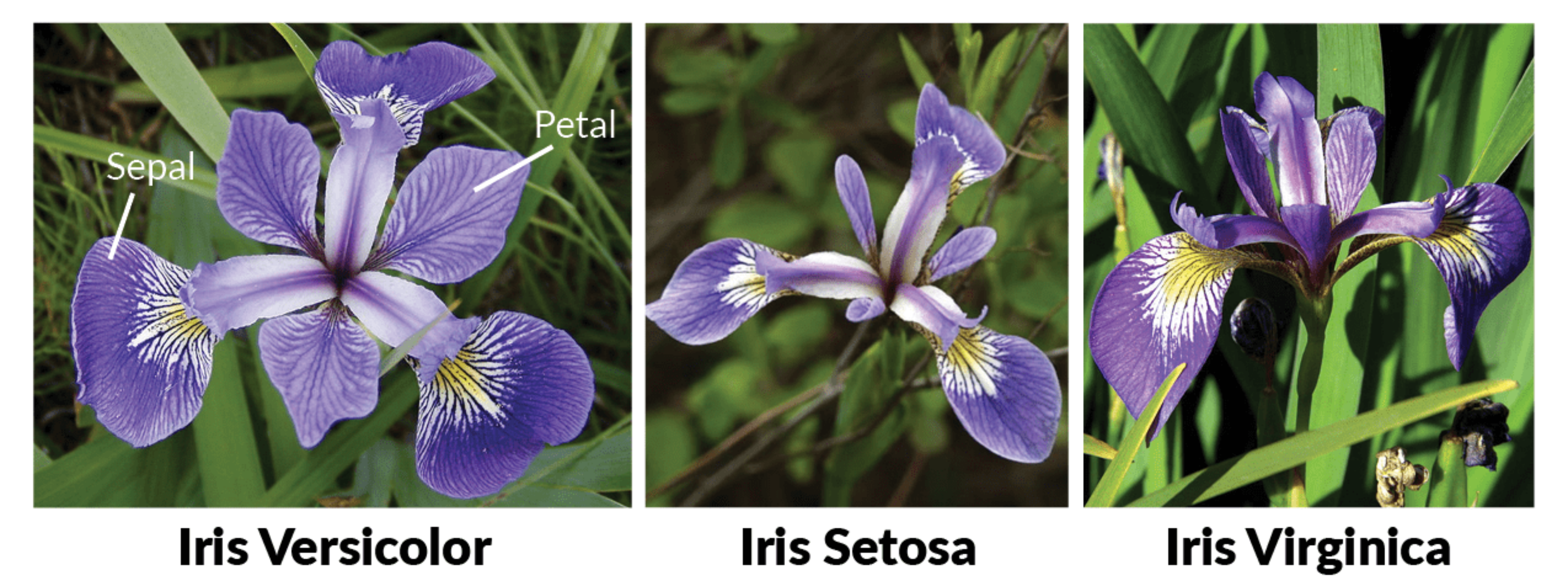
Record where you got it from, where it is now, and what you do to it

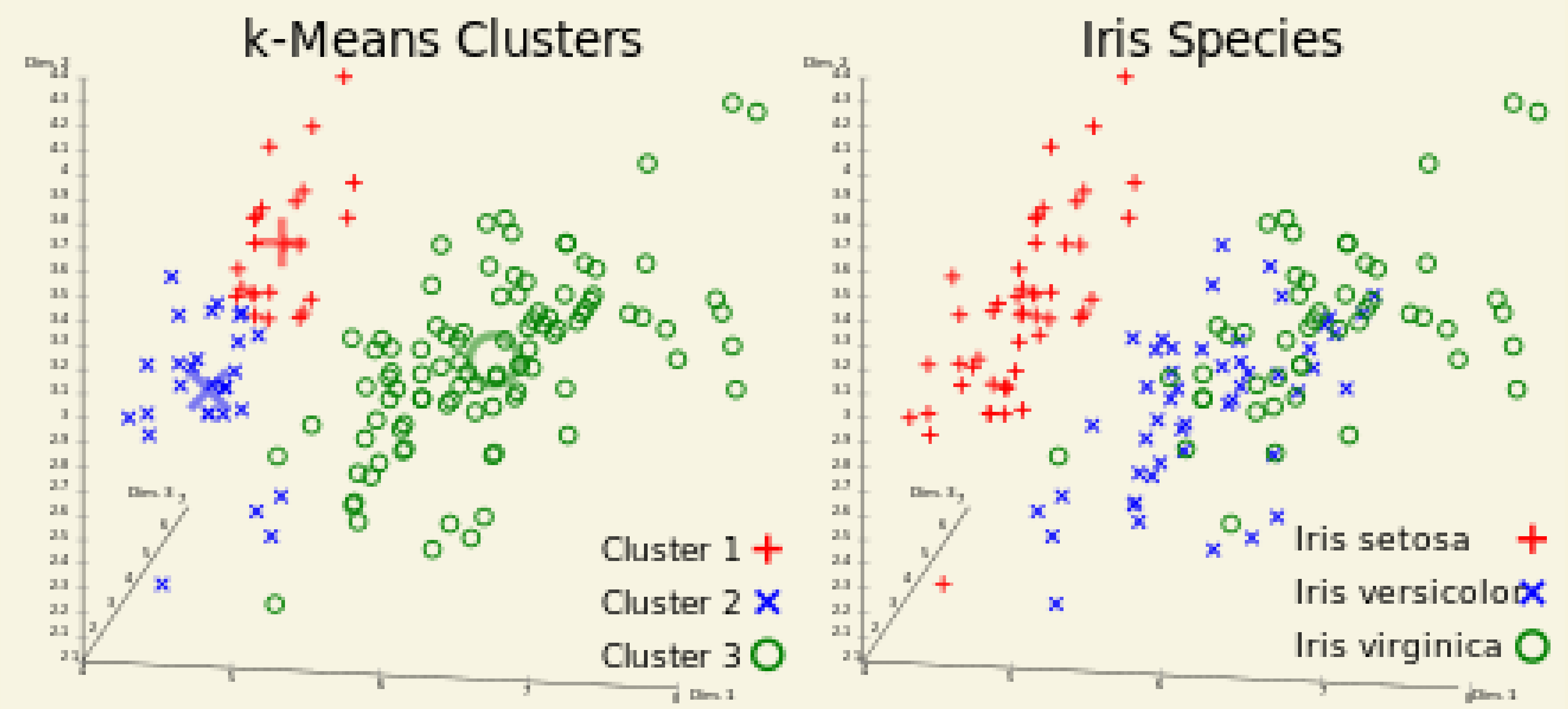
- Link datasets (as subdatasets), record data origin
- Collect and store provenance of all contents of a dataset that you create



- Record command execution: Which script produced which output? From which data? In which software environment? …

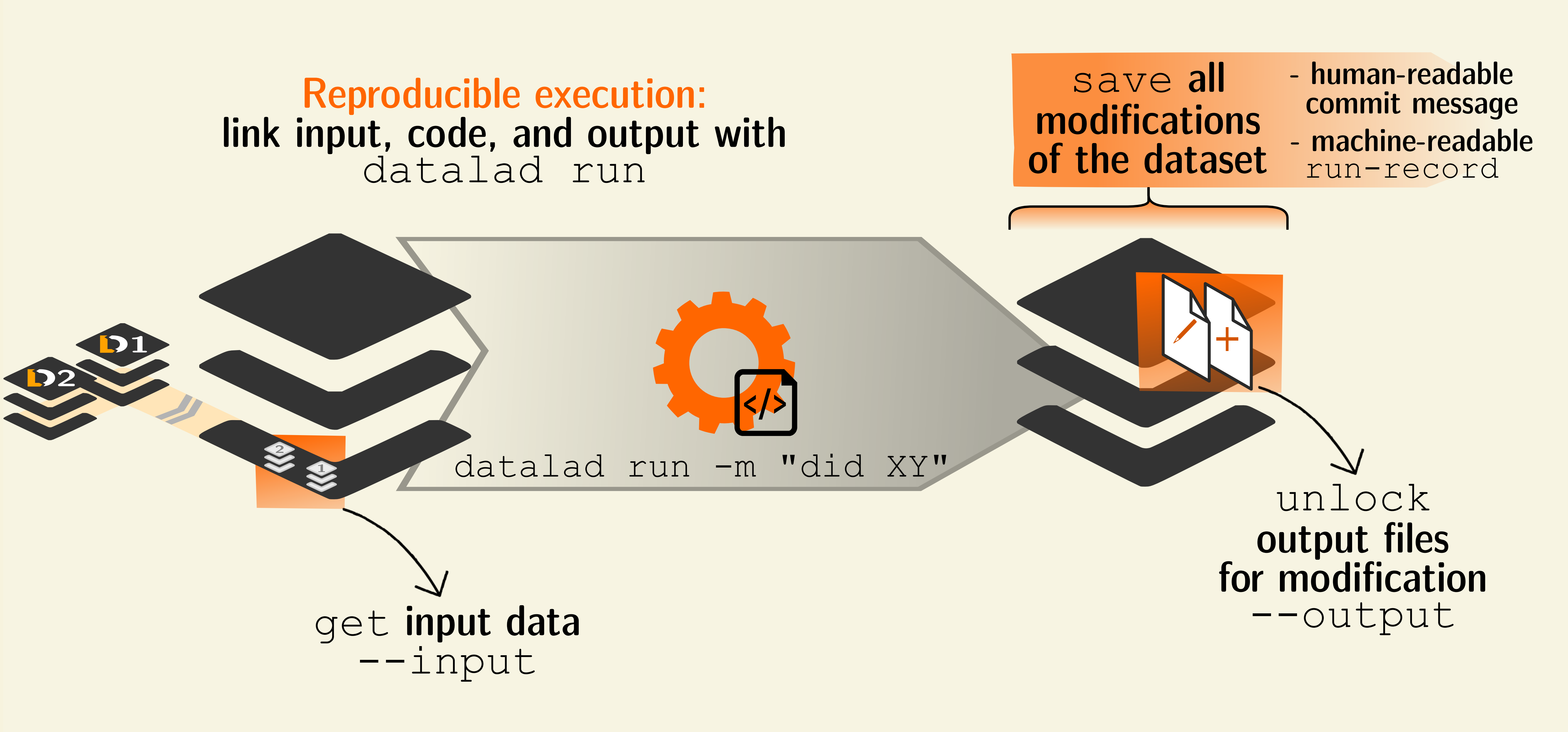
A CLASSIFICATION ANALYSIS ON THE IRIS FLOWER DATASET





REPRODUCIBLE EXECUTION & PROVENANCE CAPTURE

datalad run



Reproducible execution:
link input, code, and output with
`datalad run`

`save` **all modifications of the dataset**

- human-readable commit message
- machine-readable `run-record`

```
datalad run -m "did XY"
```

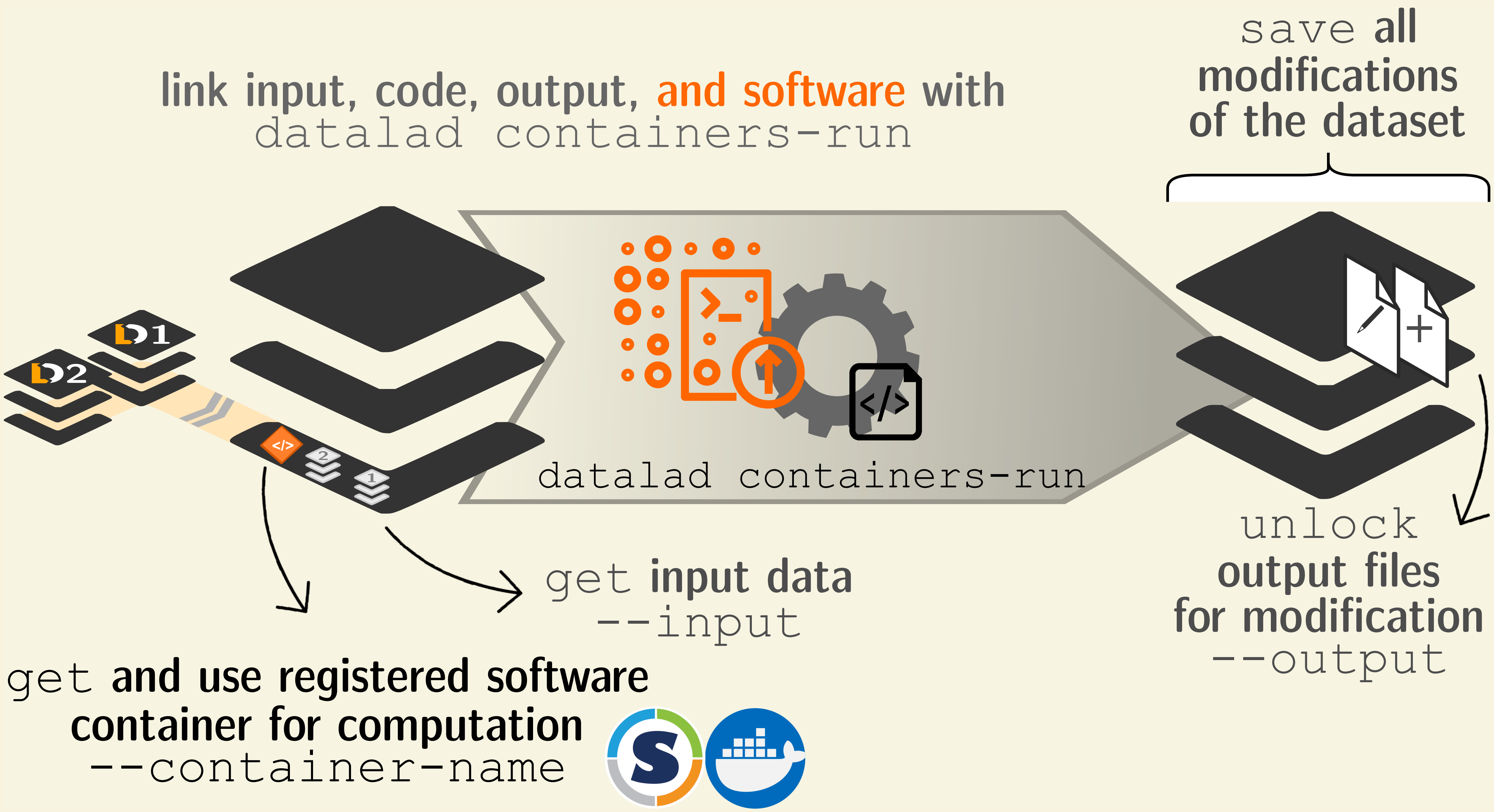
`get` **input data**
`--input`

`unlock` **output files for modification**
`--output`

COMPUTATIONAL REPRODUCIBILITY

- Code may produce different results or fail with different software
- Datasets can store & share software environments and execute code inside of the software container
- DataLad extension: `datalad-container`

datalad-containers run

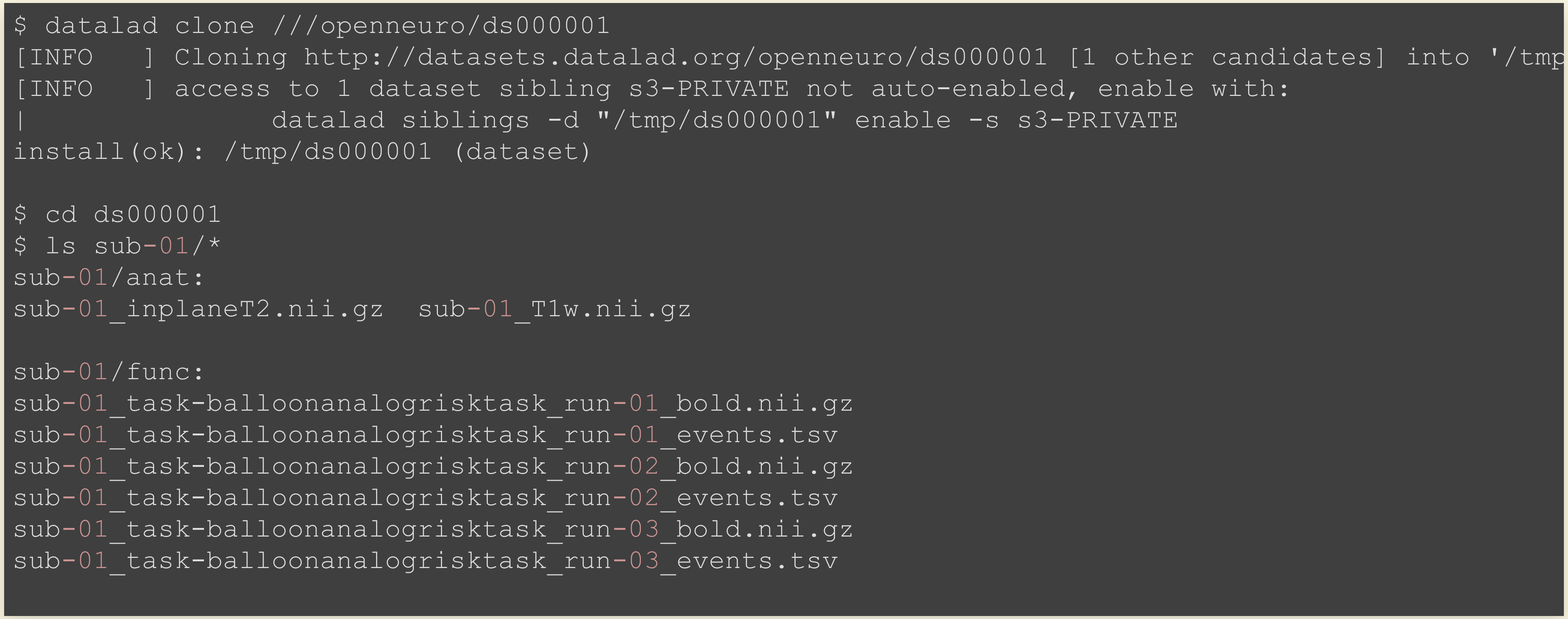


HOW TO GET STARTED WITH DATALAD

Read the DataLad handbook

An interactive, hands-on crash-course (free and open source)

Check out or used public DataLad datasets, e.g., from OpenNeuro



```
$ datalad clone ///openneuro/ds000001
[INFO   ] Cloning http://datasets.datalad.org/openneuro/ds000001 [1 other candidates] into '/tmp
[INFO   ] access to 1 dataset sibling s3-PRIVATE not auto-enabled, enable with:
|               datalad siblings -d "/tmp/ds000001" enable -s s3-PRIVATE
install(ok): /tmp/ds000001 (dataset)

$ cd ds000001
$ ls sub-01/*
sub-01/anat:
sub-01_inplaneT2.nii.gz  sub-01_T1w.nii.gz

sub-01/func:
sub-01_task-balloonanalogrisktask_run-01_bold.nii.gz
sub-01_task-balloonanalogrisktask_run-01_events.tsv
sub-01_task-balloonanalogrisktask_run-02_bold.nii.gz
sub-01_task-balloonanalogrisktask_run-02_events.tsv
sub-01_task-balloonanalogrisktask_run-03_bold.nii.gz
sub-01_task-balloonanalogrisktask_run-03_events.tsv
```

ACKNOWLEDGEMENTS

- Michael Hanke
- Yaroslav Halchenko
- Joey Hess (git-annex)
- Benjamin Poldrack
- Kyle Meyer
- 22+ additional contributors

The DataLad Handbook

- Laura Waite
- Michael Hanke
- 17+ additional contributors

Reach out, get to know the team, contribute:
DataLad on Riot,
DataLad Handbook @ Github

THANK YOU!

QUESTIONS?