

Good practice in scientific programming

Robert Haase

April 2023

#OpenScience

#OpenSource

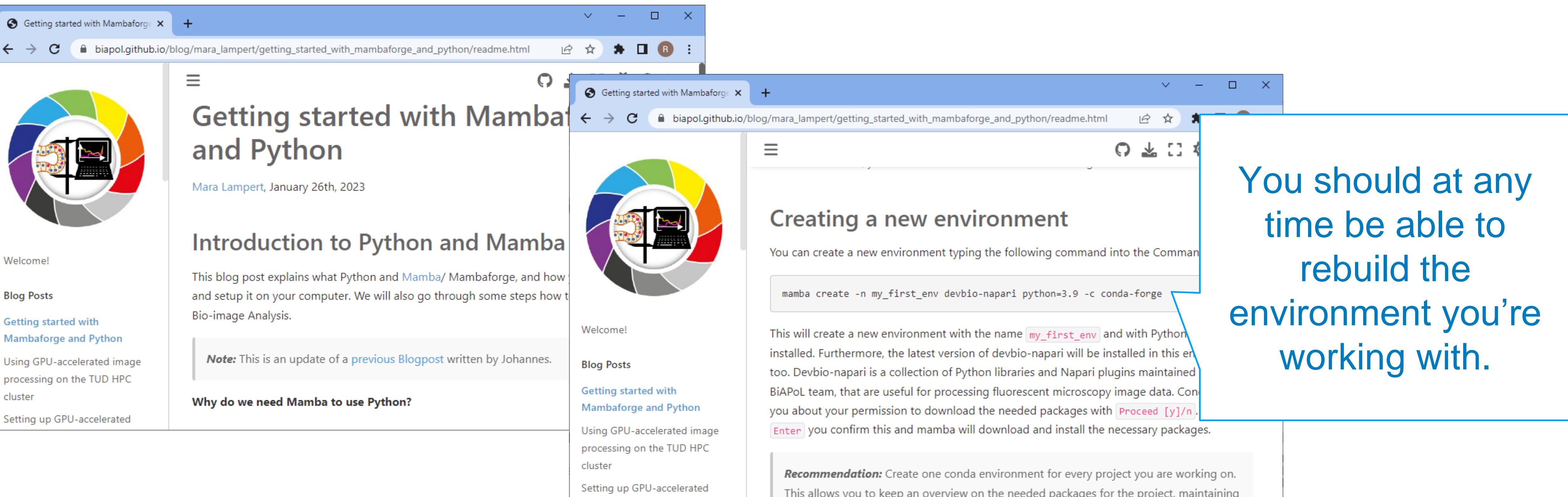
Accessible

Reusable

Sustainable

Document what you use

- Installation instructions enable reproducible science
- Not necessary as detailed as a blog-post



Getting started with Mamba and Python

Mara Lampert, January 26th, 2023

Introduction to Python and Mamba

This blog post explains what Python and [Mamba](#)/ [Mambaforge](#), and how to use them for bio-image analysis. We will go through some steps how to install and setup it on your computer. We will also go through some steps how to use it for bio-image analysis.

Note: This is an update of a [previous Blogpost](#) written by Johannes.

Why do we need Mamba to use Python?

Creating a new environment

You can create a new environment typing the following command into the Command Line:

```
mamba create -n my_first_env devbio-napari python=3.9 -c conda-forge
```

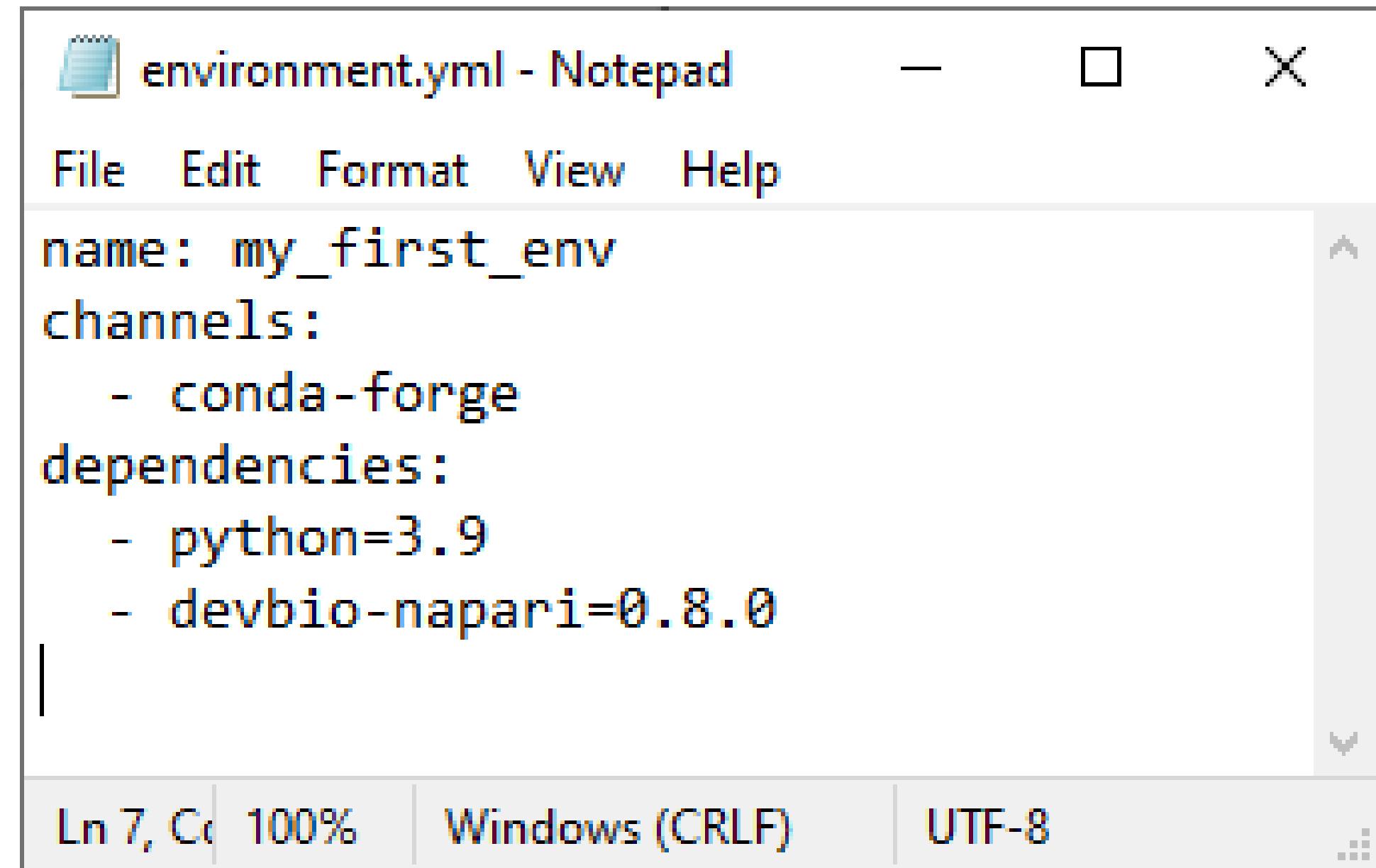
This will create a new environment with the name `my_first_env` and with Python 3.9 installed. Furthermore, the latest version of devbio-napari will be installed in this environment. Devbio-napari is a collection of Python libraries and Napari plugins maintained by the BiAPoL team, that are useful for processing fluorescent microscopy image data. Consider giving permission to download the needed packages with `Proceed [y]/n`. Press `Enter` to confirm this and mamba will download and install the necessary packages.

Recommendation: Create one conda environment for every project you are working on. This allows you to keep an overview on the needed packages for the project, maintaining consistency and reproducibility.

You should at any time be able to rebuild the environment you're working with.

Document what you use

Maintain a document with the dependencies (and versions) you need in your project!



A screenshot of a Windows Notepad window titled "environment.yml - Notepad". The window contains the following YAML code:

```
name: my_first_env
channels:
  - conda-forge
dependencies:
  - python=3.9
  - devbio-napari=0.8.0
```

The Notepad window has a standard Windows interface with a menu bar (File, Edit, Format, View, Help), status bar (Ln 7, Col 100%, Windows (CRLF), UTF-8), and scroll bars.

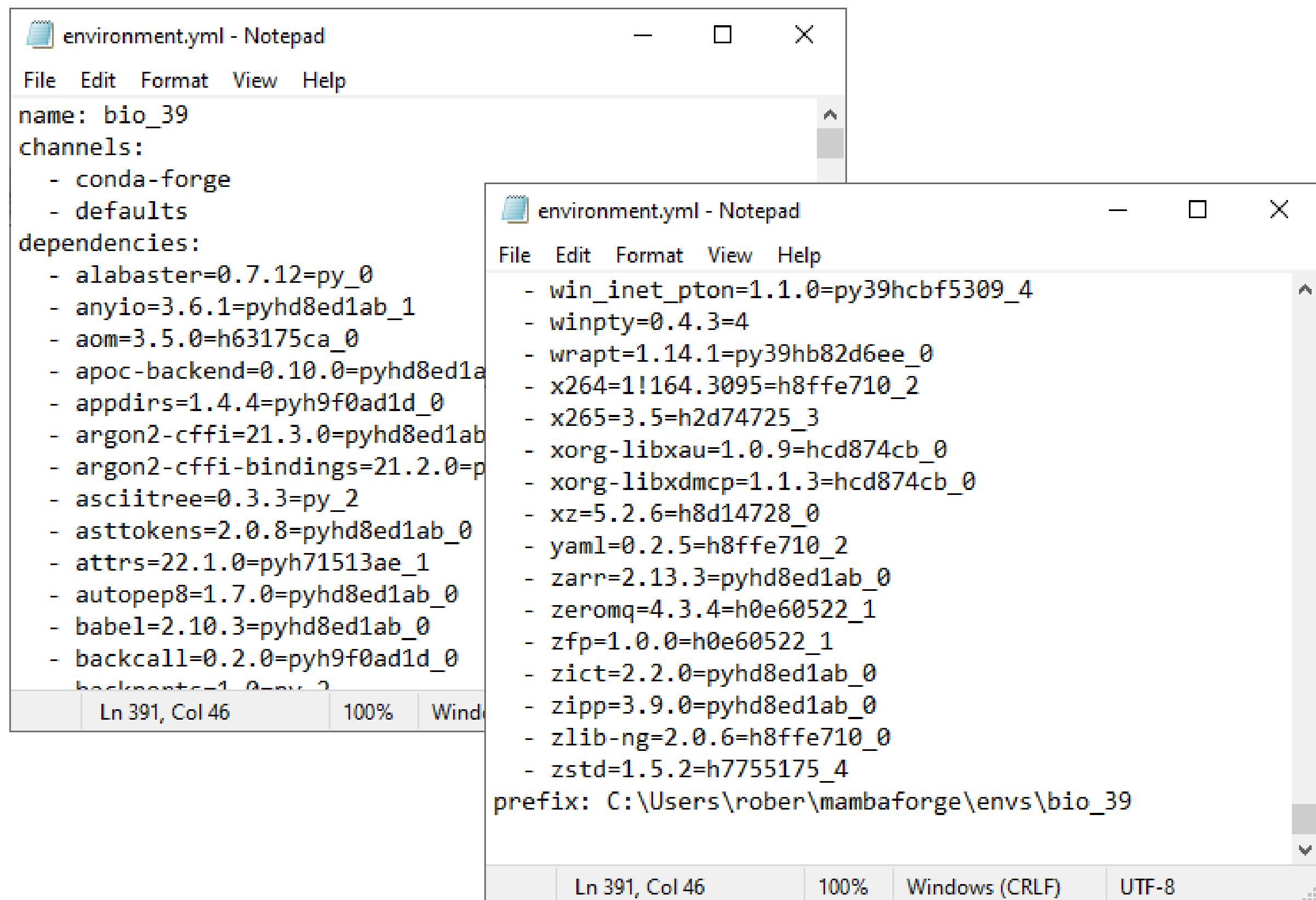
In case your
environment is screwed
up, you can rebuild it
any time.

```
conda env create -f environment.yml
```

Document what you use

... the complete way.

```
conda env export > environment.yml
```



```
environment.yml - Notepad
File Edit Format View Help
name: bio_39
channels:
- conda-forge
- defaults
dependencies:
- alabaster=0.7.12=py_0
- anyio=3.6.1=pyhd8ed1ab_1
- aom=3.5.0=h63175ca_0
- apoc-backend=0.10.0=pyhd8ed1a
- appdirs=1.4.4=pyh9f0ad1d_0
- argon2-cffi=21.3.0=pyhd8ed1ab
- argon2-cffi-bindings=21.2.0=p
- asciitree=0.3.3=py_2
- asttokens=2.0.8=pyhd8ed1ab_0
- attrs=22.1.0=pyh71513ae_1
- autopenp8=1.7.0=pyhd8ed1ab_0
- babel=2.10.3=pyhd8ed1ab_0
- backcall=0.2.0=pyh9f0ad1d_0
backrefs=1.0.0=py_0
Ln 391, Col 46 100% Wind...
```



```
environment.yml - Notepad
File Edit Format View Help
- win_inet_pton=1.1.0=py39hcbf5309_4
- winpty=0.4.3=4
- wrapt=1.14.1=py39hb82d6ee_0
- x264=1!164.3095=h8ffe710_2
- x265=3.5=h2d74725_3
- xorg-libxau=1.0.9=hcd874cb_0
- xorg-libxdmcp=1.1.3=hcd874cb_0
- xz=5.2.6=h8d14728_0
- yaml=0.2.5=h8ffe710_2
- zarr=2.13.3=pyhd8ed1ab_0
- zeromq=4.3.4=h0e60522_1
- zfp=1.0.0=h0e60522_1
- zict=2.2.0=pyhd8ed1ab_0
- zipp=3.9.0=pyhd8ed1ab_0
- zlib-ng=2.0.6=h8ffe710_0
- zstd=1.5.2=h7755175_4
prefix: C:\Users\rober\mambaforge\envs\bio_39
Ln 391, Col 46 100% Windows (CRLF) UTF-8
```

Excellent way to document all dependencies were *actually* used...

It is *questionable* if re-creating an environment from this yml file works.

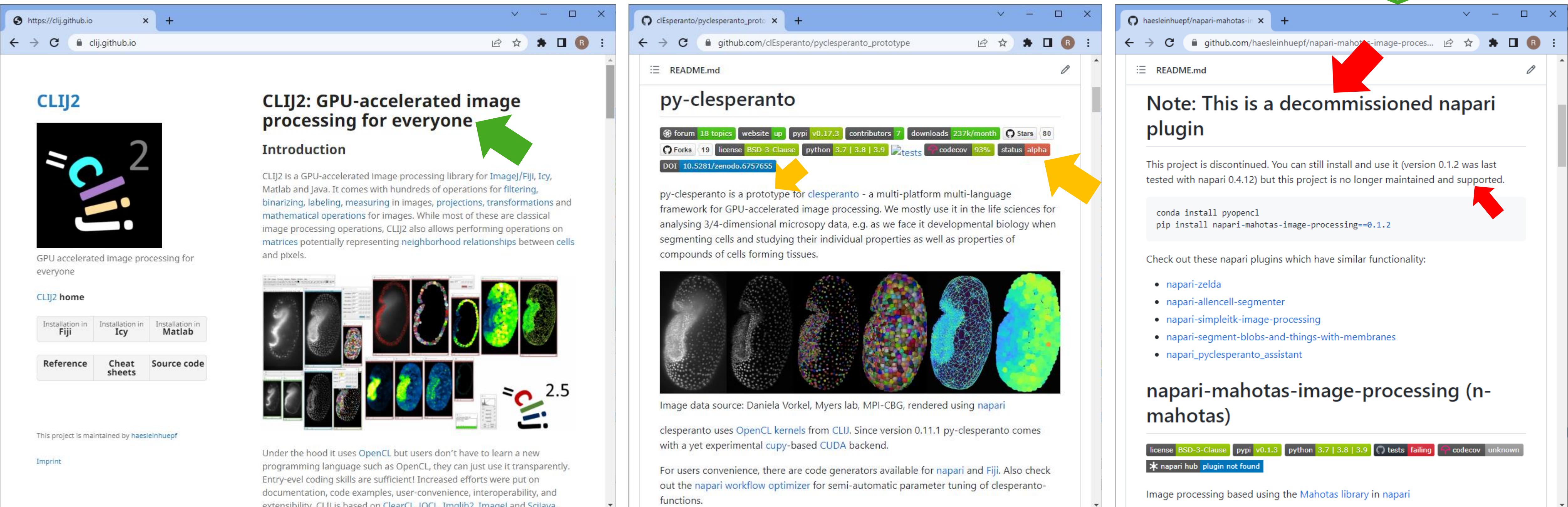
Software quality indicators

... to differentiate the good stuff from the bad.

Target audience

- Documentation should tell who is the target audience and how far it is developed

Communication
is key!



The figure displays three side-by-side browser windows showing GitHub project README pages:

- CLIJ2:** GPU-accelerated image processing for everyone. The page includes a logo, a screenshot of a microscopy image, and links for installation in Fiji, Icy, and Matlab.
- py-clesperanto_prototype:** A prototype for clesperanto - a multi-platform multi-language framework for GPU-accelerated image processing. It features a screenshot of a segmented brain image and a list of similar napari plugins.
- napari-mahotas-image-processing (n-mahotas):** An image processing plugin based on Mahotas library in napari. It includes a note about being decommissioned and a list of dependencies.

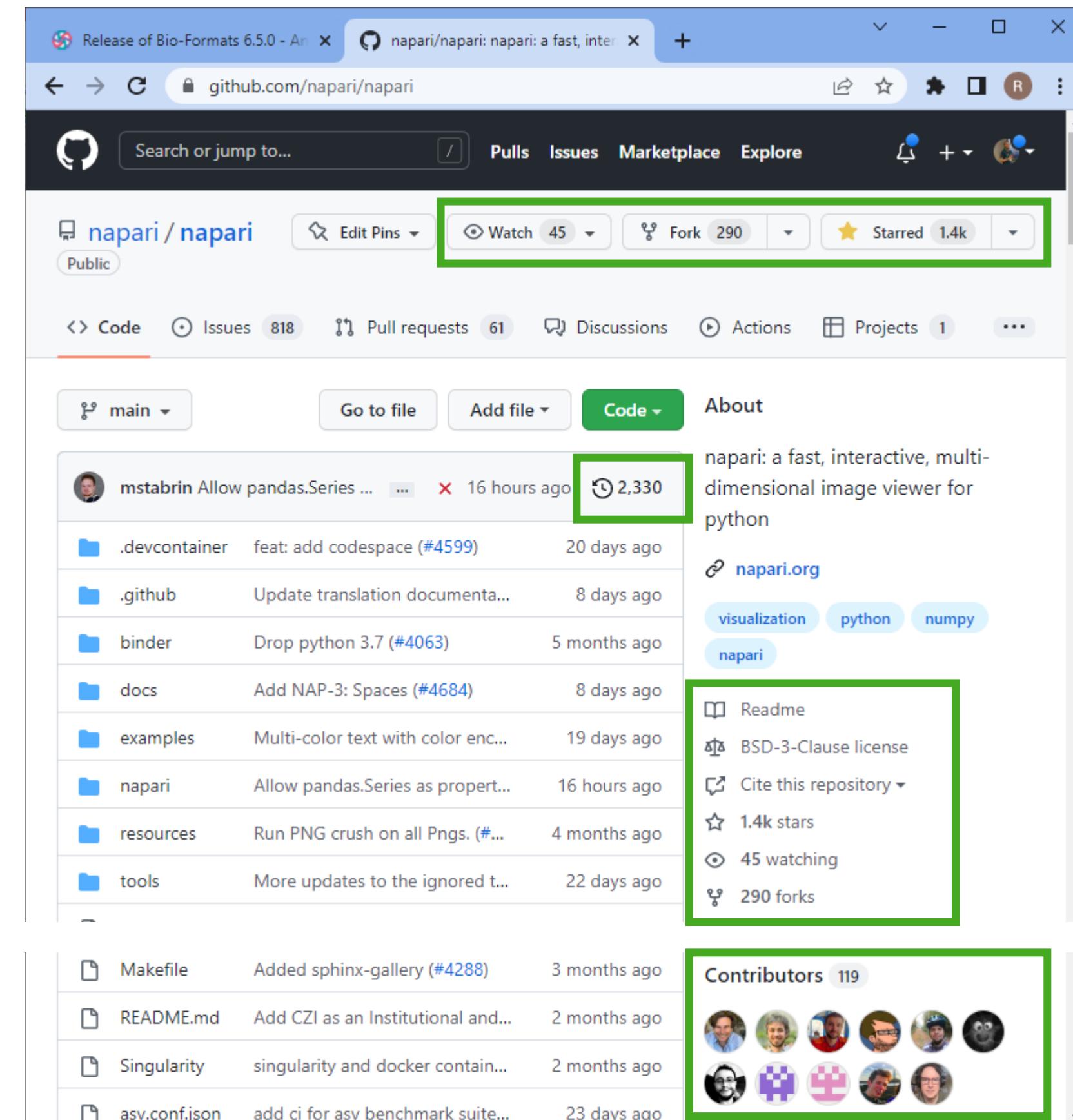
Annotations with arrows highlight specific sections in each README:

- A green arrow points to the title "GPU-accelerated image processing for everyone" in the CLIJ2 README.
- Yellow arrows point to the GitHub stats (stars, forks, license, etc.) in the py-clesperanto_prototype README.
- A red arrow points to the note "Note: This is a decommissioned napari plugin" in the n-mahotas README.
- A red arrow points to the "Check out these napari plugins which have similar functionality:" section in the n-mahotas README.

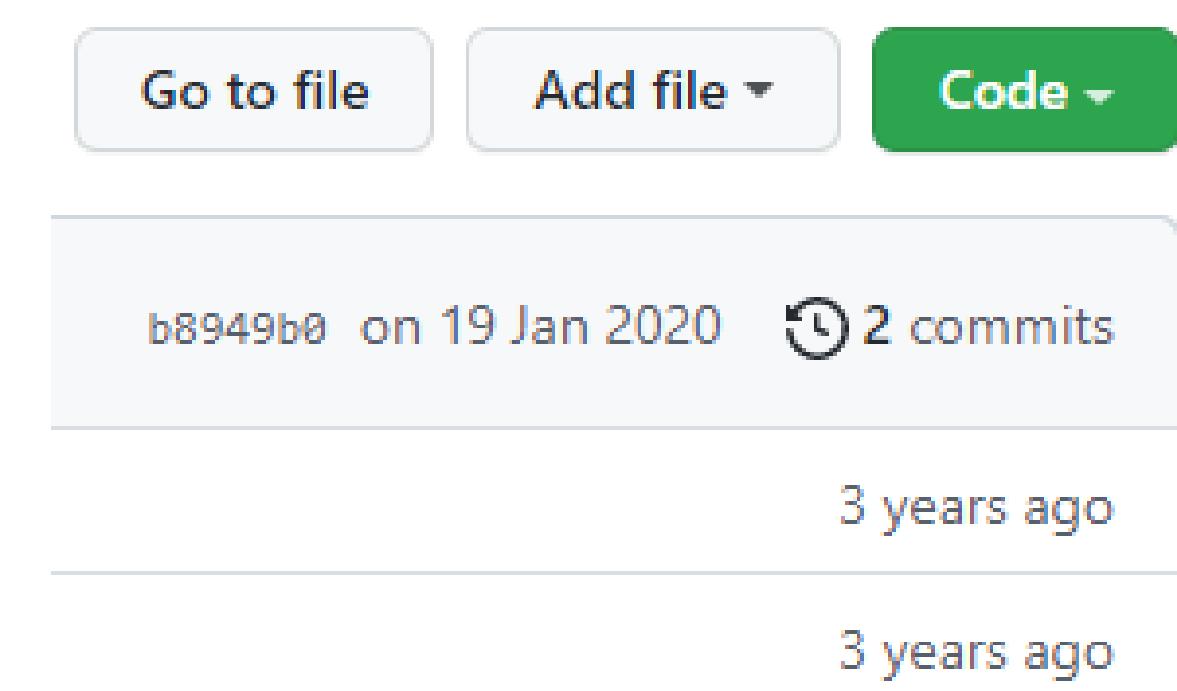
Software quality indicators

Visit the project's github or gitlab page and review indicators.

- **Stars:** People like software, similarly to tweets on Twitter
- **Watching:** People receive updates for new releases
- **Forks:** People made a copy of the code, e.g. to contribute to the project
- **Contributors:** People who contributed to the code
- **Commits:** Changes to the code



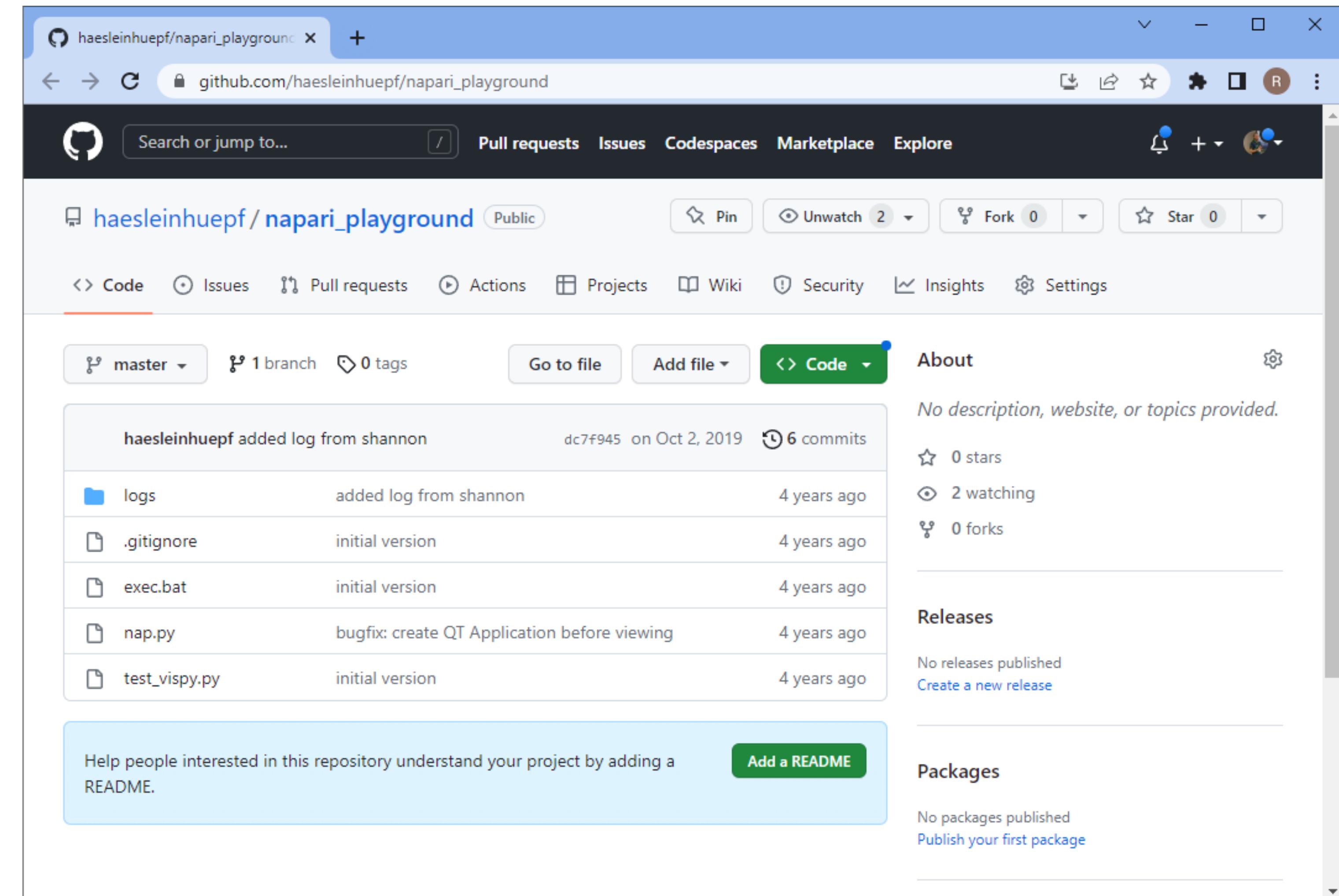
The screenshot shows the GitHub repository for napari. At the top, there are metrics: Watch (45), Fork (290), and Starred (1.4k). Below this is a list of recent commits. A green box highlights the commit count '2,330' next to a timestamp '16 hours ago'. Another green box highlights the 'Code' tab and the commit details for 'mstabin'. A third green box highlights the 'Contributors' section at the bottom, which shows 119 contributors with small profile pictures.



This screenshot shows a GitHub repository with very little information. The 'About' section only contains the text 'No description, website, or topics provided.' Below it, there are metrics: 1 star, 2 watching, and 0 forks. The commit history shows a single commit 'b8949b0' from 'on 19 Jan 2020' with '2 commits'.

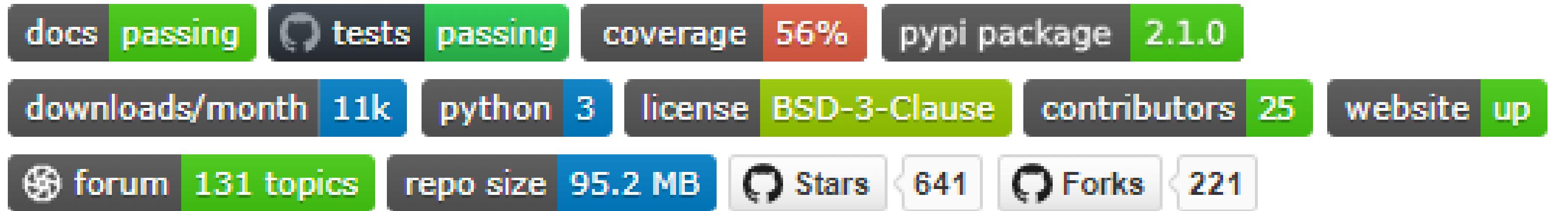
Bad example

- No readme / documentation
- No license / copyright statement
- No stars / users (?)
- Not maintained (last update 4 years ago)
- bus factor = 1



Software quality indicators

Visit the project's github or gitlab page and review indicators.



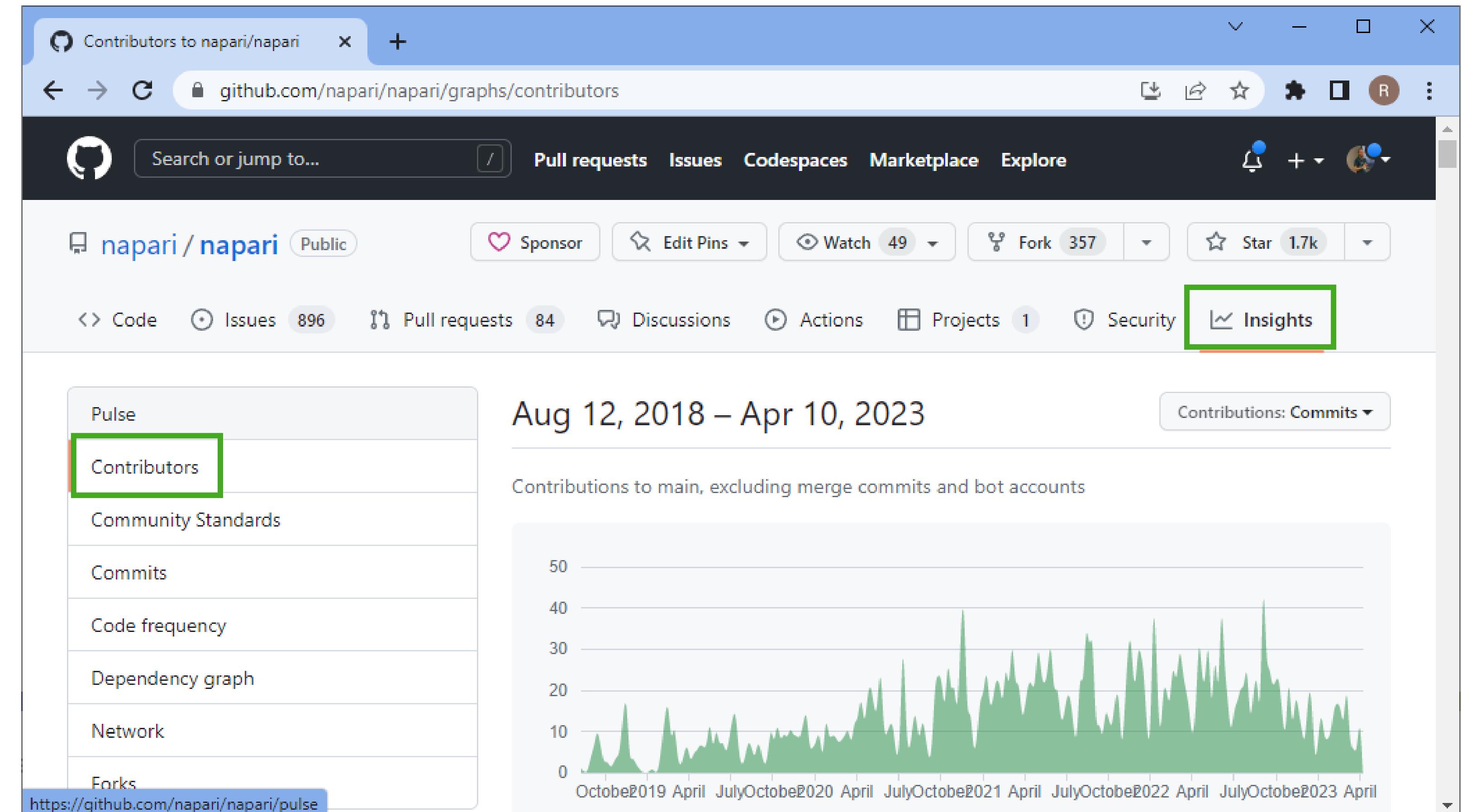
Note, github badges
cannot be *deserved*.
Developers put them
there



Image source: Adapted from <https://www.pexels.com/photo/shallow-focus-photo-of-two-persons-wearing-military-uniform-2859046/>

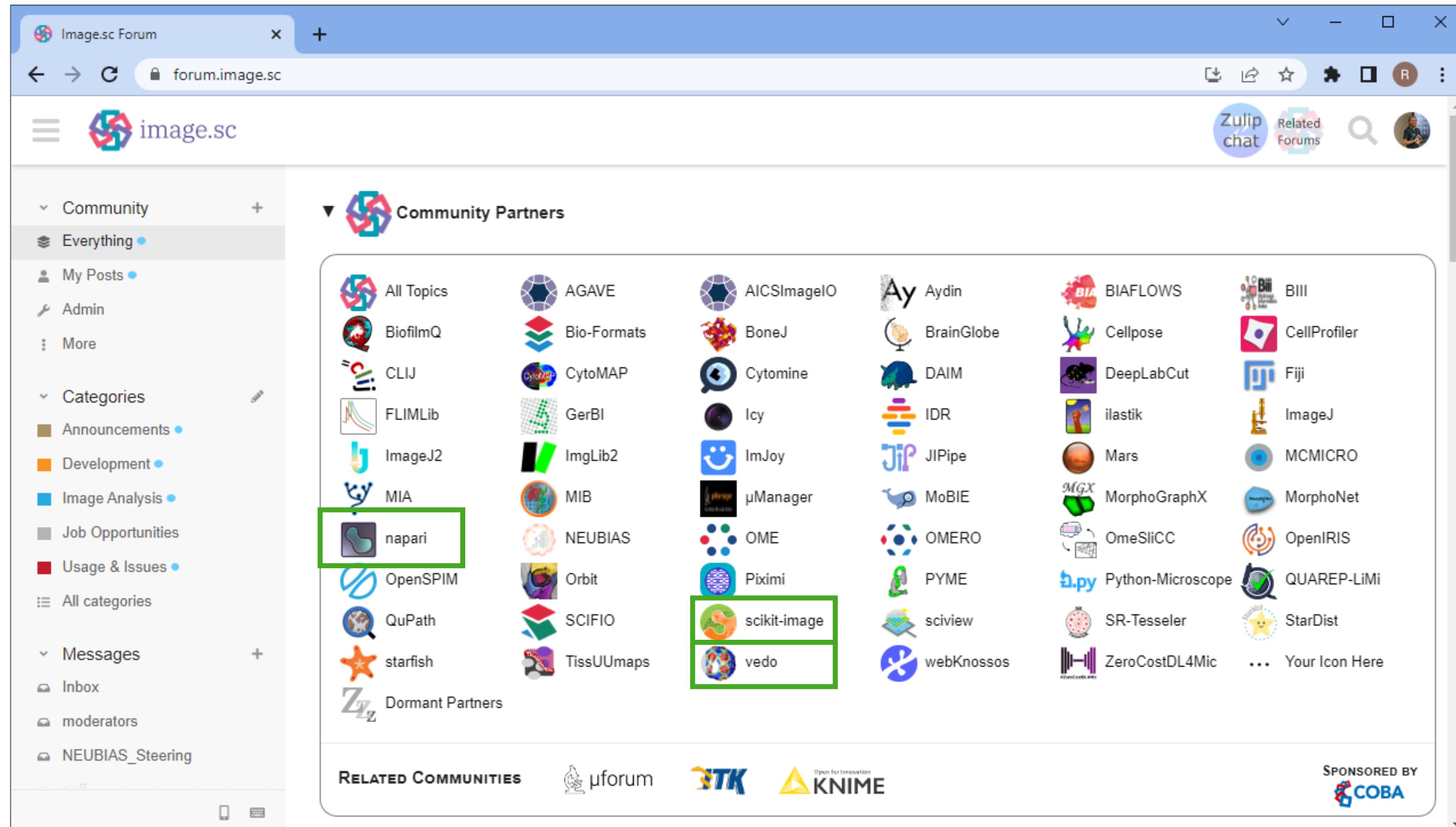
Software quality indicators

Visit the project's github or gitlab page and review indicators.



Software quality indicators

- Community actively involved

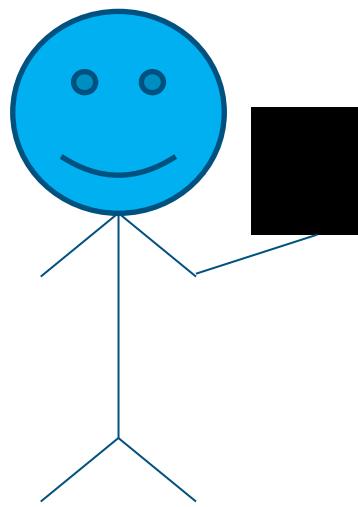


The screenshot shows the Image.sc Forum interface. On the left, there's a sidebar with navigation links like 'Community', 'Everything', 'My Posts', 'Admin', 'Categories', 'Announcements', 'Development', 'Image Analysis', 'Job Opportunities', 'Usage & Issues', 'All categories', 'Messages', 'Inbox', 'moderators', and 'NEUBIAS_Steering'. The main area is titled 'Community Partners' and lists various software projects with their logos and names. Some projects are highlighted with green boxes: 'napari' and 'scikit-image'. Other visible projects include AGAVE, Bio-Formats, AICSImageIO, Aydin, BIAFLOWS, BIII, Cellpose, CellProfiler, Fiji, ImageJ, DeepLabCut, ilastik, MARS, MCMICRO, MorphoGraphX, MorphoNet, OmeSliCC, OpenIRIS, QUAREP-LiMi, Python-Microscope, SR-Tesseler, StarDist, webKnossos, and ZeroCostDL4Mic. At the bottom, there are links for 'RELATED COMMUNITIES' to μforum, STK, and KNIME, and a 'SPONSORED BY COBA' logo.

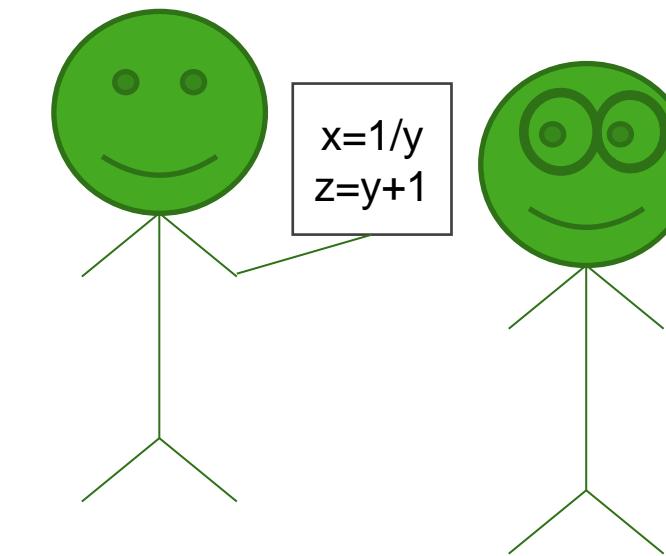
Openness of software / projects

Choose your project's level wisely, and communicate it clearly

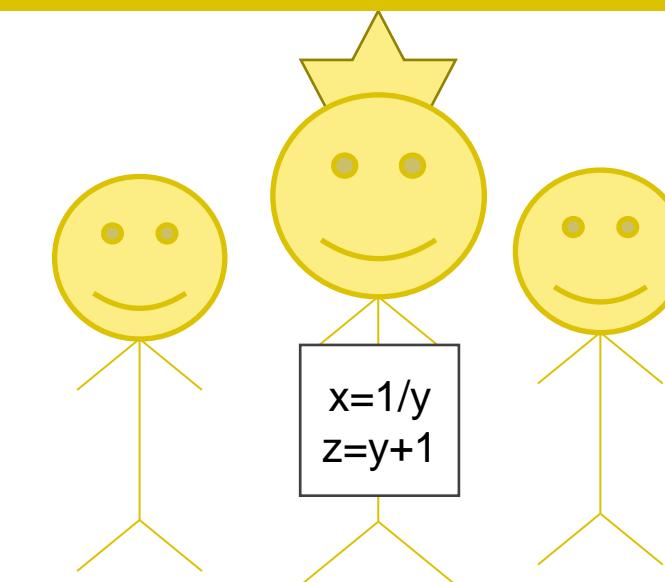
Closed source



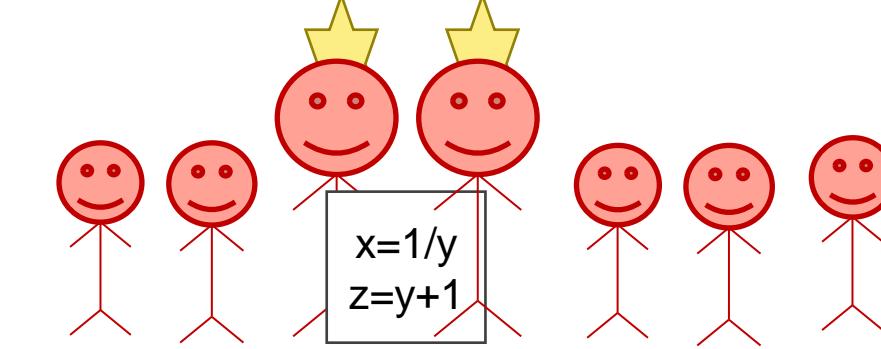
Open source



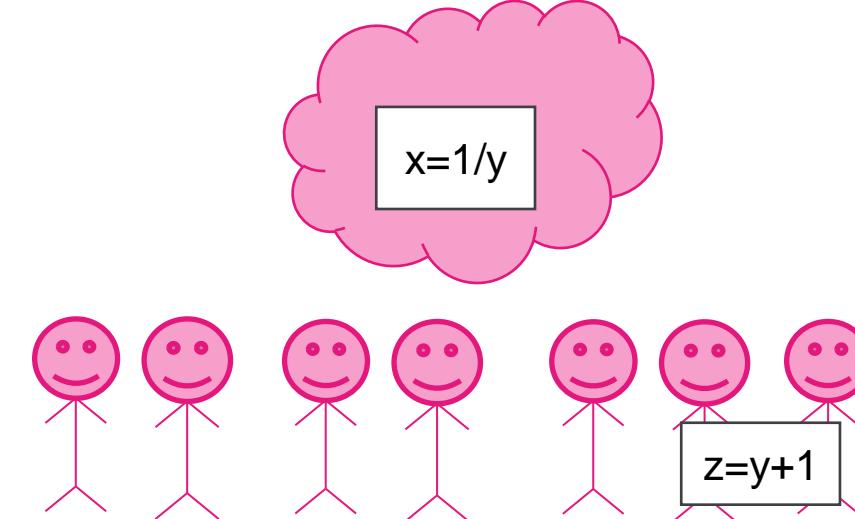
Benevolent
dictatorship



Community driven



Openly extensible



- Open to collaborations
- “Black box”
- Compiled code (e.g. C/C++)
- Good for protecting intellectual properties (\$\$\$)

Hardware device drivers

- Code available to read
- Not necessarily executable code
- No maintenance / support efforts

Custom image analysis scripts

- Open to contributions
- Single maintainer, often overwhelmed
- Efficient decision making
- Bus factor ≈ 1

TrackMate, SNT, MorpholibJ, CLIJ

- Open to contributions
- Partially democratic
- Board of maintainers (core developers)
- Long-winded decision making

scikit-image, scipy, OpenCL

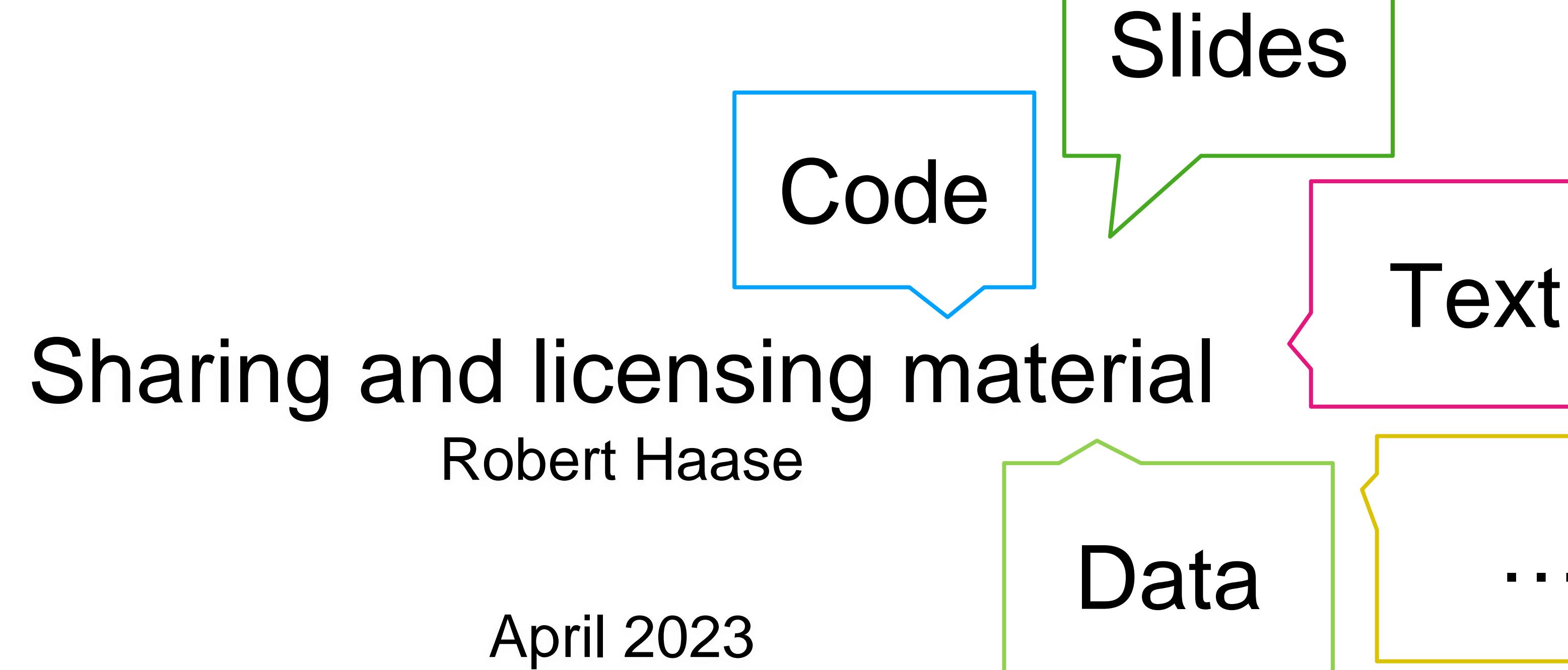
- Openly extensible; without maintainers involved
- Partially community driven

ImageJ, Python, numpy

Take home message

When using [open-source] software, make sure

- it's maintained
- used by others
- supported by an active community
- well-documented



Use cases: Data

The screenshot shows a web browser displaying a RODARE dataset page. The title is "Slice2Volume: Fusion of multimodal medical imaging and light microscopy data of irradiation-injured brain tissue in 3D". The page includes author information (Müller, Johannes; Suckert, Theresa; Beyreuther, Elke; Schneider, Moritz; Boucsein, Marc; Bodenstein, Elisabeth; Stolz-Kieslich, Liane; Krause, Mechthild; von Neubeck, Cläre; Haase, Robert; Lühr, Armin; Dietrich, Antje), a description of the dataset (comprehensive image data for nine mice after proton irradiation), and technical details (CBCT, MRI, histology, immunofluorescence). A preview section shows a grid of 53 images. A large blue arrow points from the RODARE page to the Twitter post.

Unique datasets

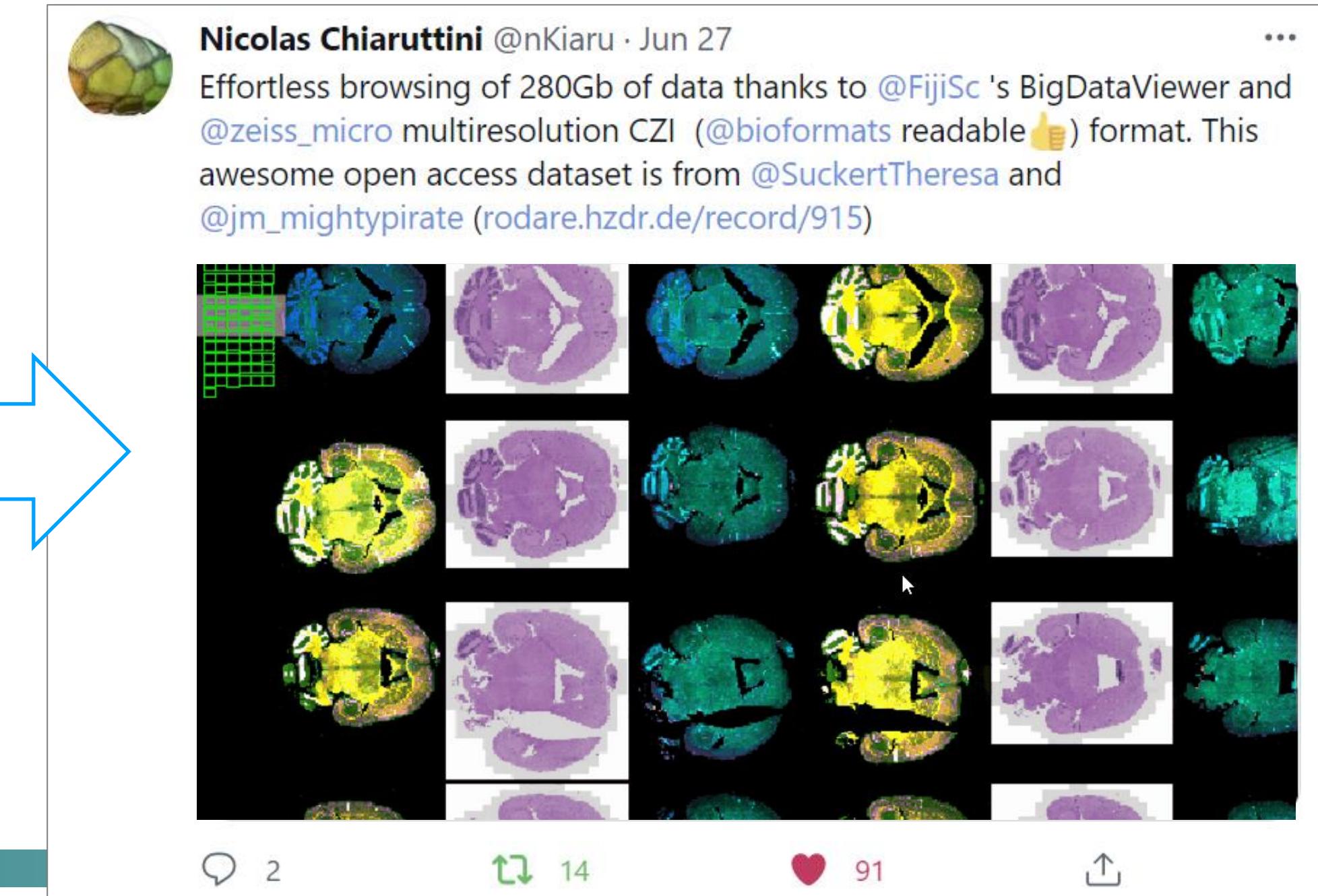
Valuable for biologists

Valuable for software developers

Institutional servers / services

<https://idr.openmicroscopy.org/>

<https://zenodo.org>

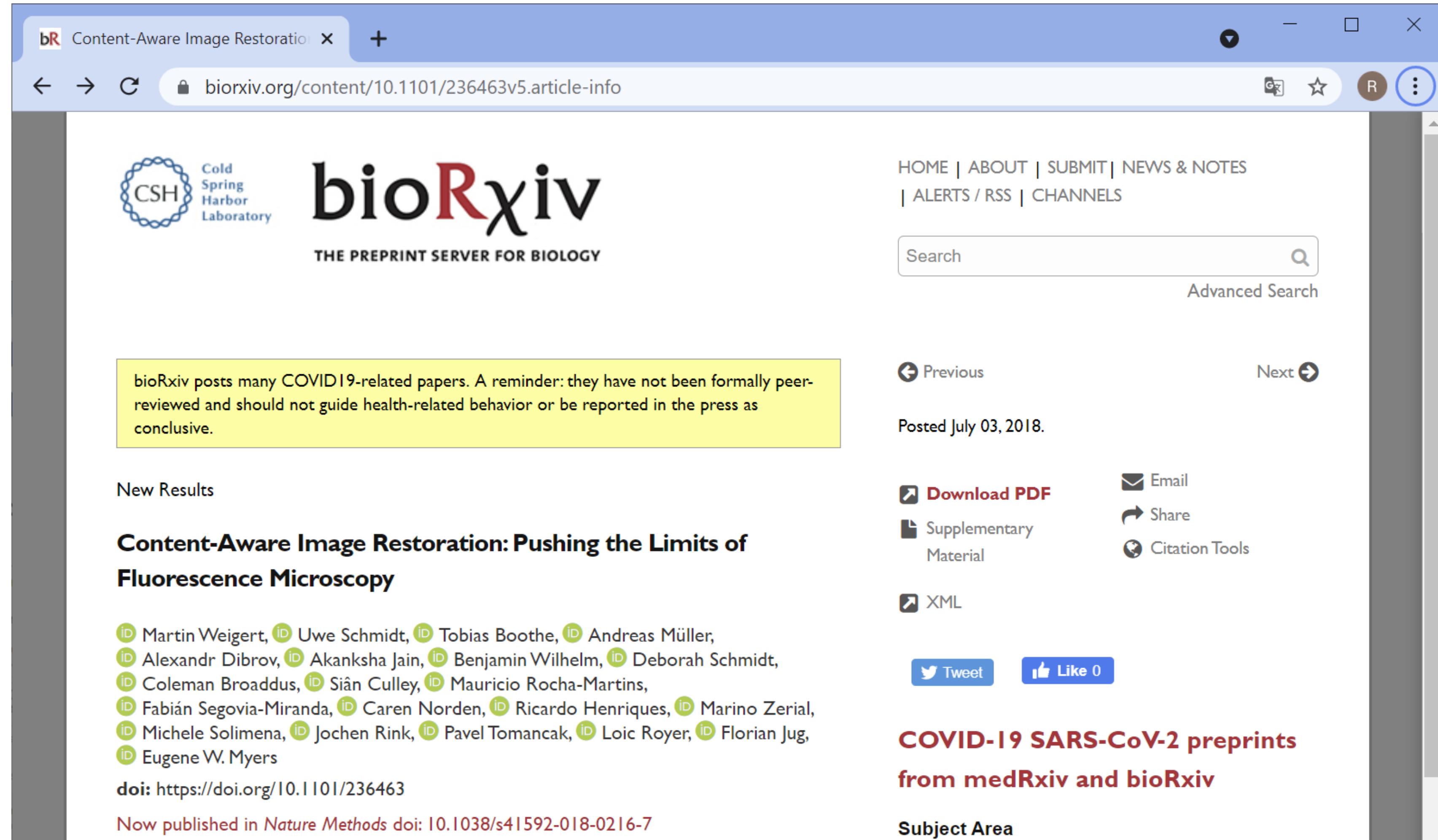


Use cases: Manuscripts

Preprints

- Accessible / reusable
- <https://arxiv.org/>
- <https://biorxiv.org/>
- <https://medrxiv.org/>

Journals



The screenshot shows a web browser window displaying a bioRxiv preprint page. The URL in the address bar is biorxiv.org/content/10.1101/236463v5.article-info. The page header includes the bioRxiv logo and navigation links for HOME, ABOUT, SUBMIT, NEWS & NOTES, ALERTS / RSS, and CHANNELS. A search bar is also present. The main content area features a yellow box containing a COVID-19-related reminder about the status of the paper. Below this, the title of the preprint is shown: "Content-Aware Image Restoration: Pushing the Limits of Fluorescence Microscopy". The authors listed are Martin Weigert, Uwe Schmidt, Tobias Boothe, Andreas Müller, Alexandre Dibrov, Akanksha Jain, Benjamin Wilhelm, Deborah Schmidt, Coleman Broaddus, Siân Culley, Mauricio Rocha-Martins, Fabián Segovia-Miranda, Caren Norden, Ricardo Henriques, Marino Zerial, Michele Solimena, Jochen Rink, Pavel Tomancak, Loic Royer, Florian Jug, and Eugene W. Myers. The DOI provided is <https://doi.org/10.1101/236463>. A note at the bottom indicates that the paper has been published in *Nature Methods* with a different DOI: [10.1038/s41592-018-0216-7](https://doi.org/10.1038/s41592-018-0216-7). On the right side of the page, there are download options (PDF, XML, Supplementary Material, Citation Tools), social sharing links (Email, Share, Like 0), and a section for COVID-19 SARS-CoV-2 preprints from medRxiv and bioRxiv.

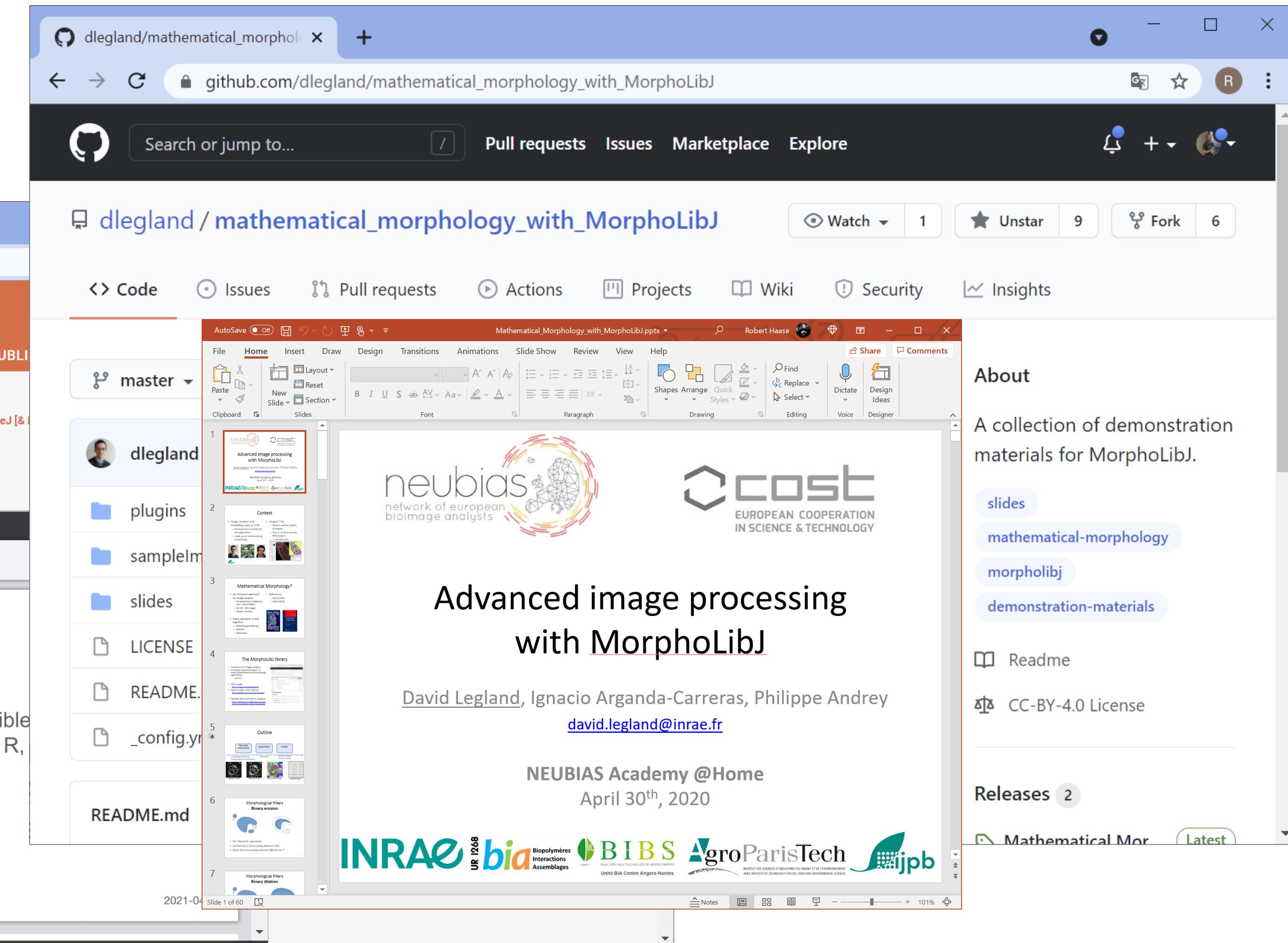
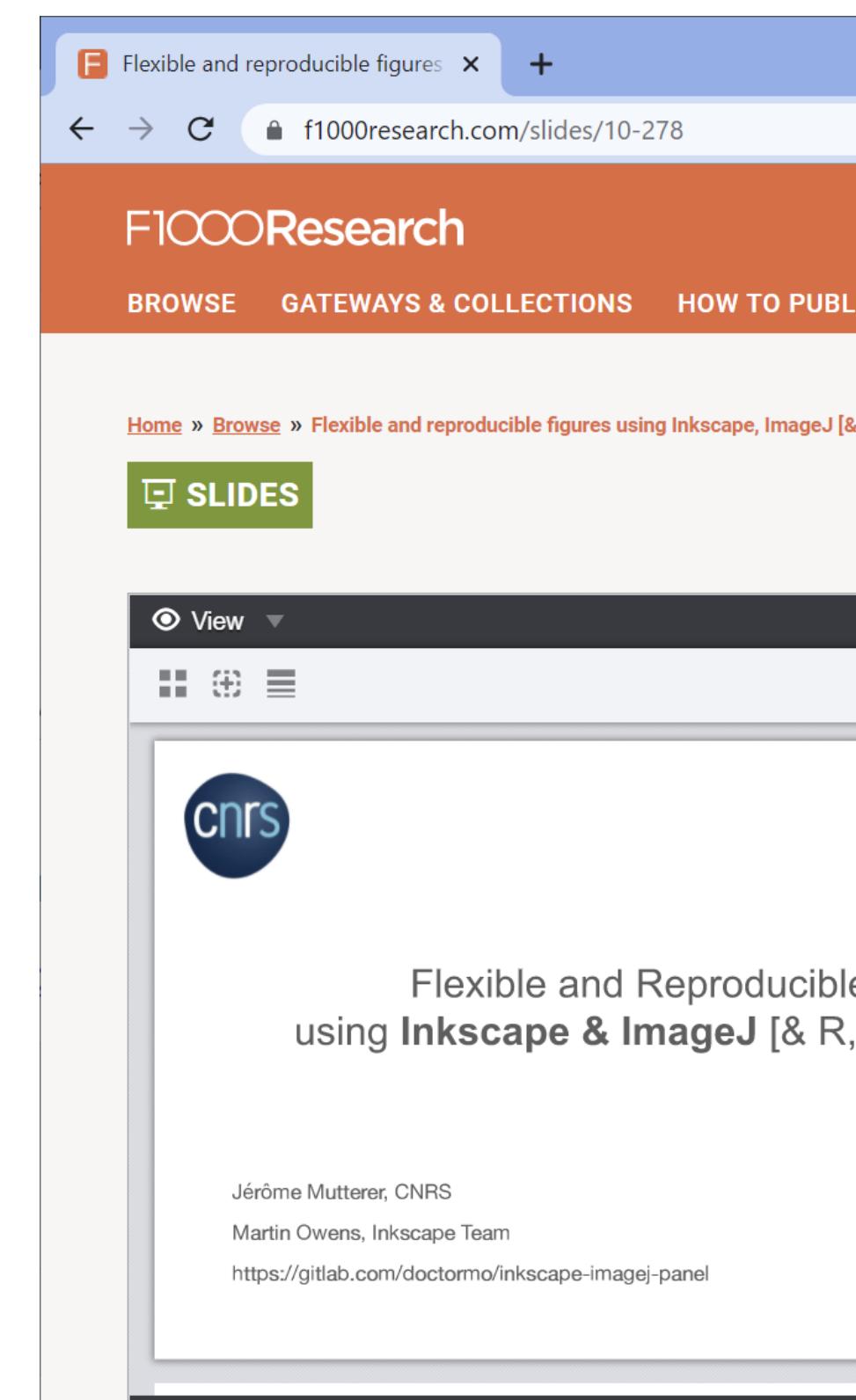
Use cases: Teaching material

Re-using, advertising your work.

<https://f1000research.com/neubias>

<https://figshare.com>

<https://github.com>



The screenshot shows a GitHub repository page for 'dlegland/mathematical_morphology_with_MorphoLibJ'. The repository has 1 watch, 9 stars, 6 forks, and 2 releases. The main page features a presentation slide titled 'Advanced image processing with MorphoLibJ' by David Legland, Ignacio Arganda-Carreras, and Philippe Andrey. The slide includes logos for neubias, cost, INRAE, bia, BIBS, AgroParisTech, and ijb. The repository contains files like 'master', 'plugins', 'samplem', 'slides', 'LICENSE', 'README.', '_config.yaml', and 'README.md'. The presentation slide is visible in the background of the GitHub interface.

<https://f1000research.com/slides/10-278> https://github.com/dlegland/mathematical_morphology_with_MorphoLibJ

Use cases: Figures

Share efforts

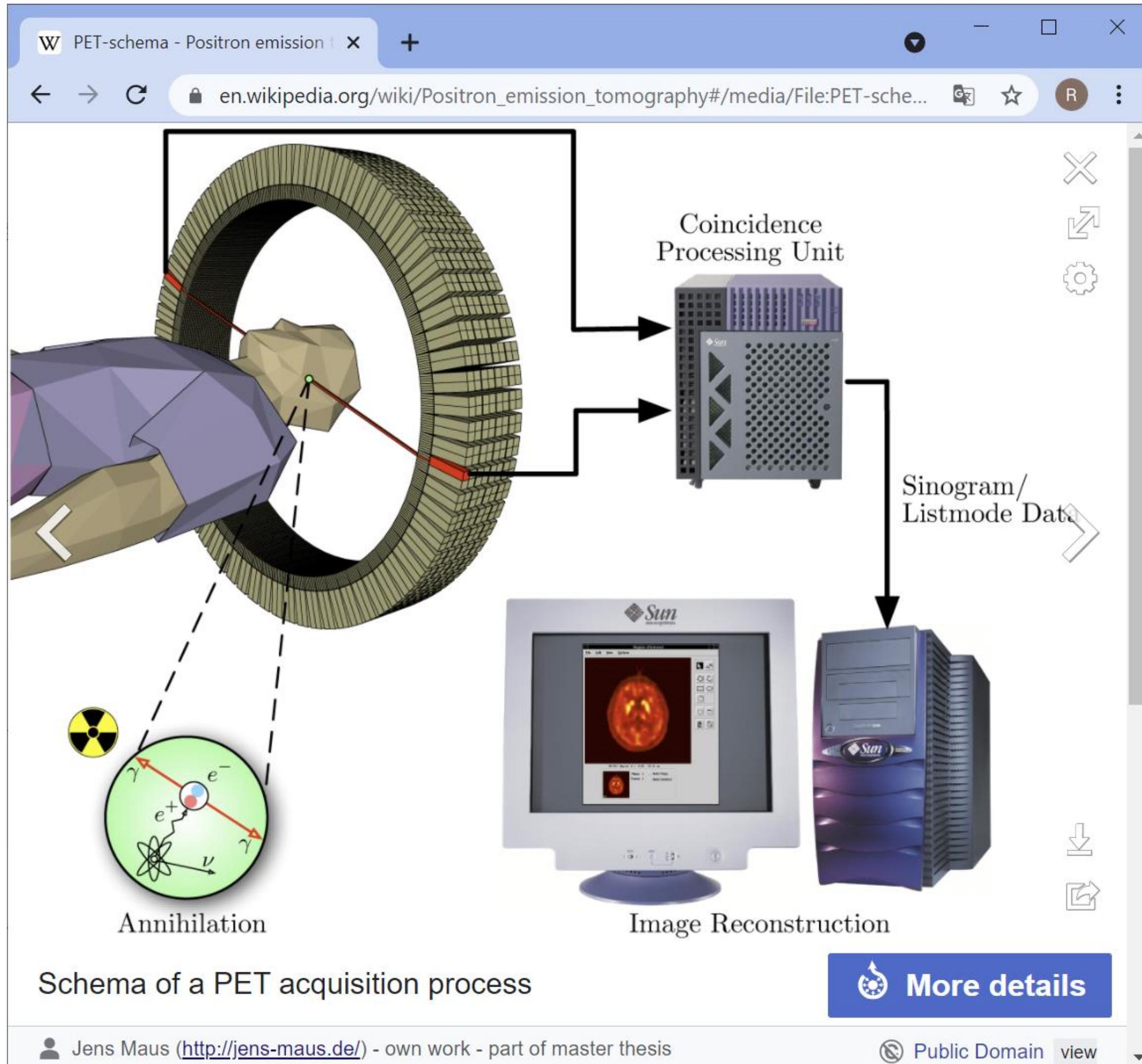
—Talk about each others' work

Advertise your work

... because our work is often publicly funded

https://commons.wikimedia.org/wiki/Main_Page

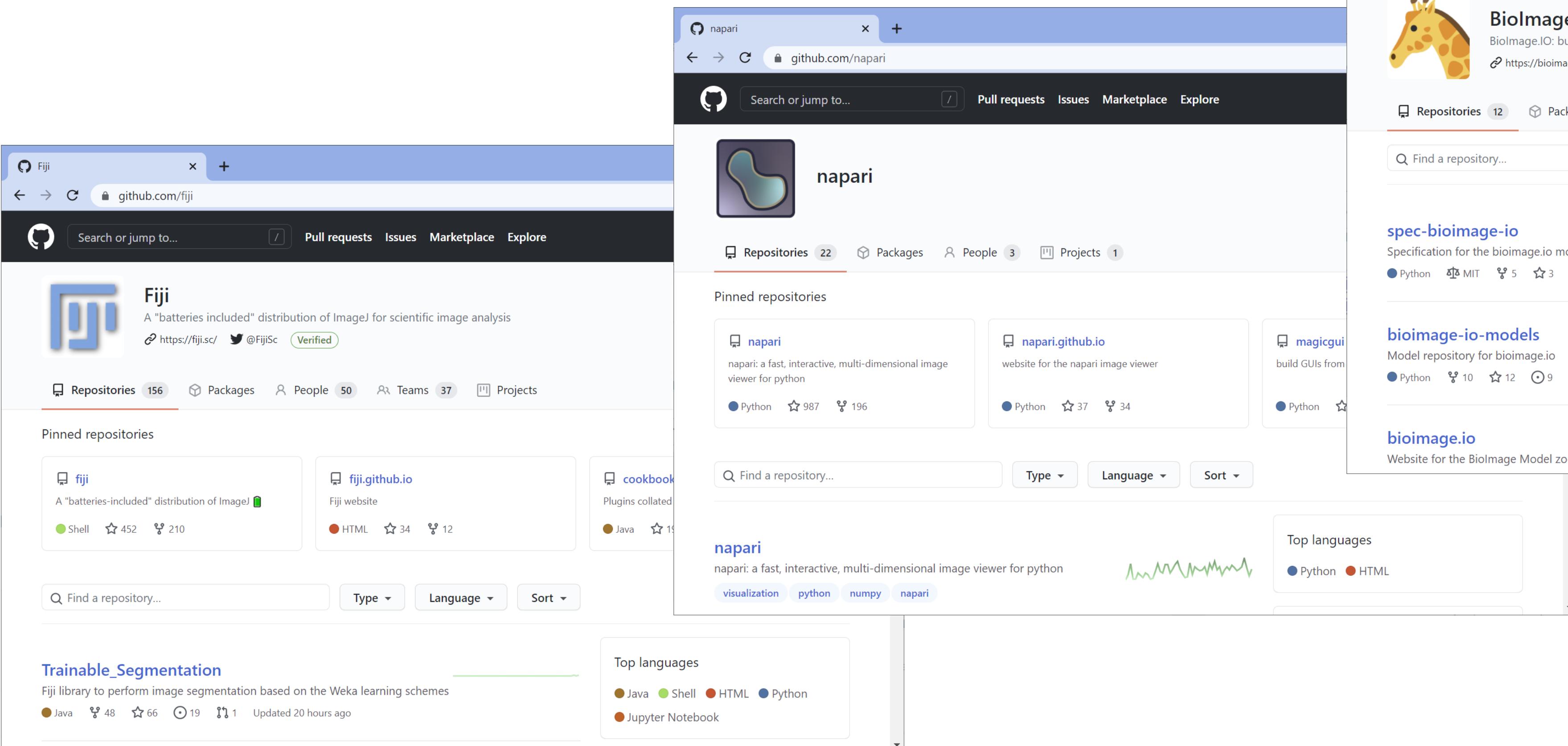
<https://figshare.com>



https://en.wikipedia.org/wiki/Positron_emission_tomography#/media/File:PET-schema.png

Use cases: Code

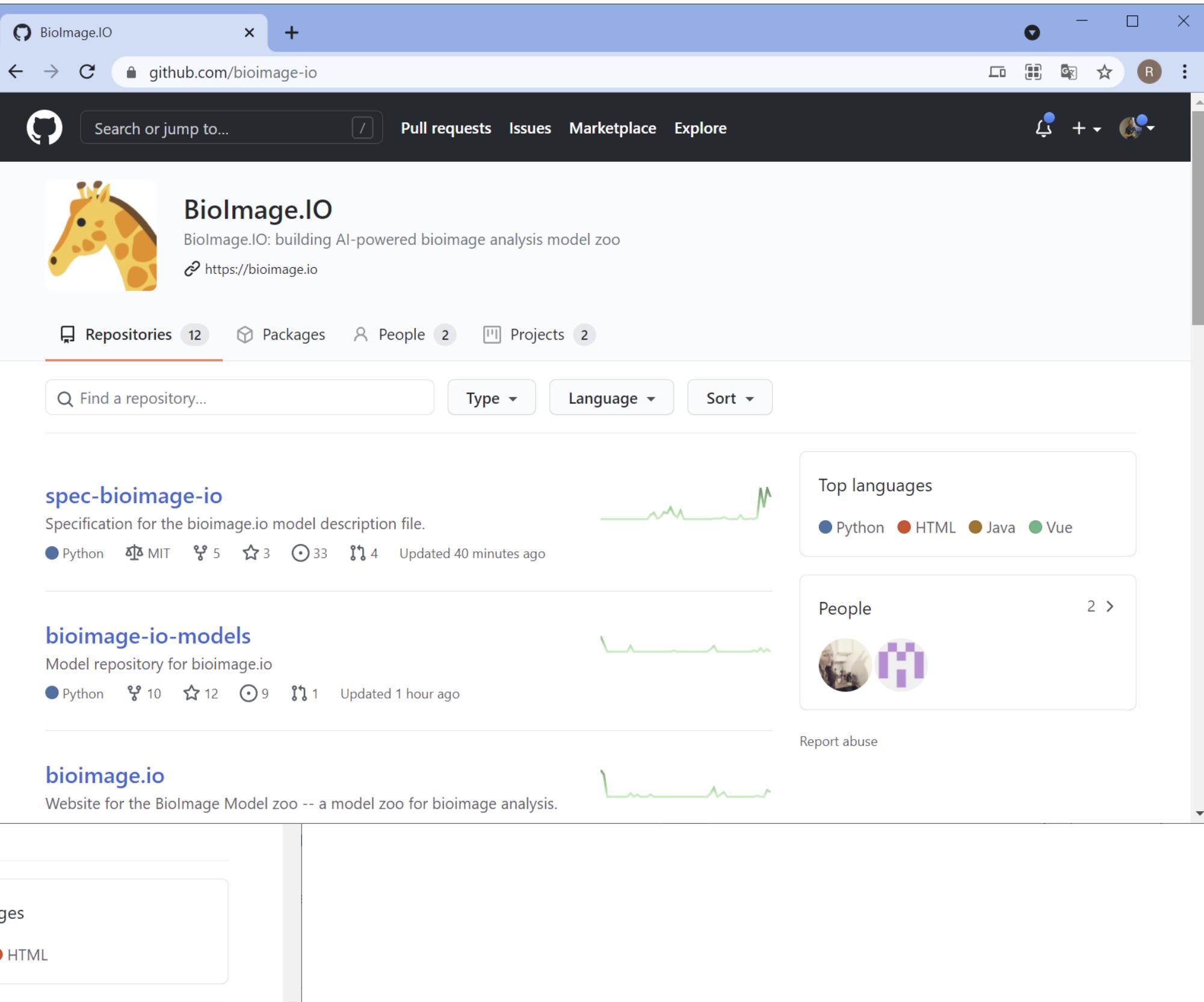
Collaboration in open-source projects *unthinkable* without openly sharing and transparent licensing



The screenshot shows three GitHub repository pages side-by-side:

- Fiji**: A "batteries included" distribution of ImageJ for scientific image analysis. It has 156 repositories, 452 stars, and 210 forks. It uses Java, Shell, and HTML.
- napari**: A fast, interactive, multi-dimensional image viewer for Python. It has 22 repositories, 987 stars, and 196 forks. It uses Python and HTML.
- BioImage.IO**: Building AI-powered bioimage analysis model zoo. It has 12 repositories, 37 stars, and 34 forks. It uses Python, MIT license, and Jupyter Notebook.

All three repositories are pinned to the BioImage.IO page, demonstrating their interconnectedness and shared licensing.

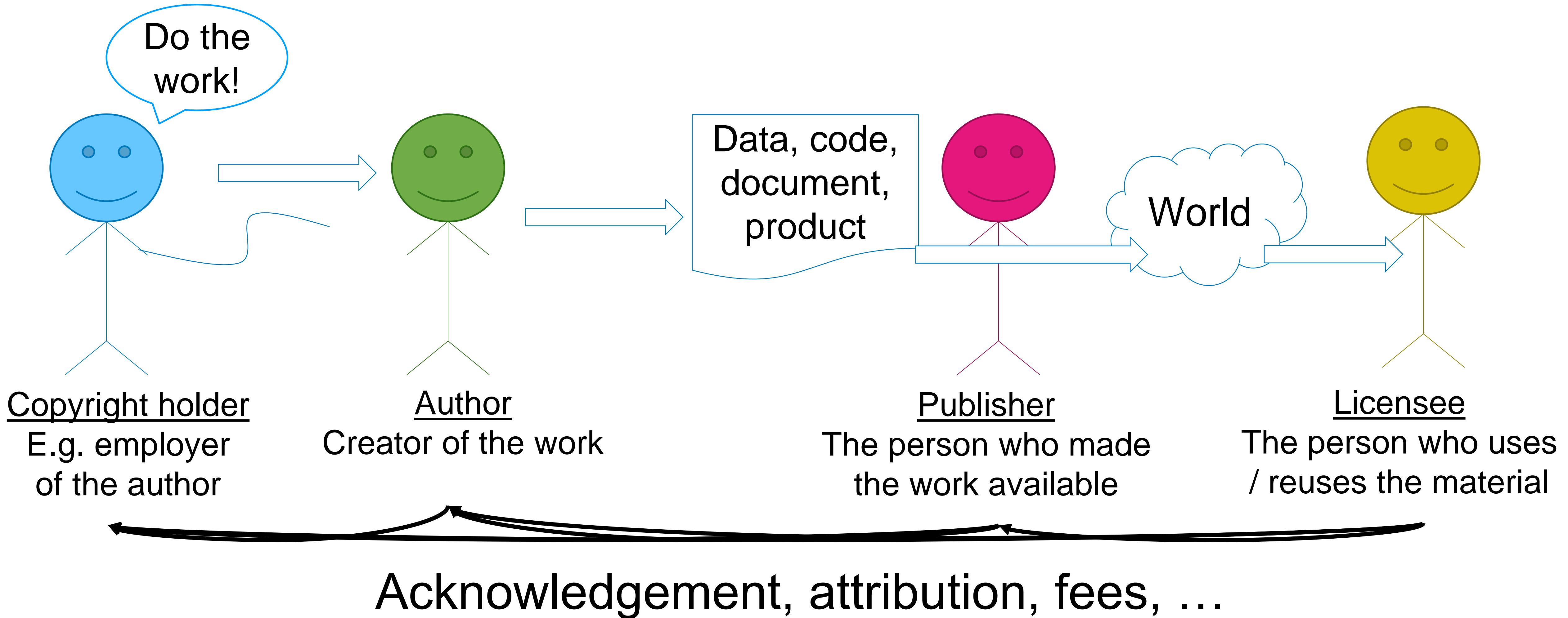


The screenshot shows the GitHub organization page for **BioImage.IO**, which contains the following repositories:

- spec-bioimage-io**: Specification for the bioimage.io model description file. (Python, MIT, 5 stars, 33 forks, 4 issues)
- bioimage-io-models**: Model repository for bioimage.io. (Python, 10 stars, 12 forks, 9 issues)
- bioimage.io**: Website for the BioImage Model zoo -- a model zoo for bioimage analysis. (Python, 1 star, 1 fork, 1 issue)

The page also displays a top languages chart showing Python as the primary language, followed by HTML, Java, and Vue. It includes sections for people and report abuse.

Terminology



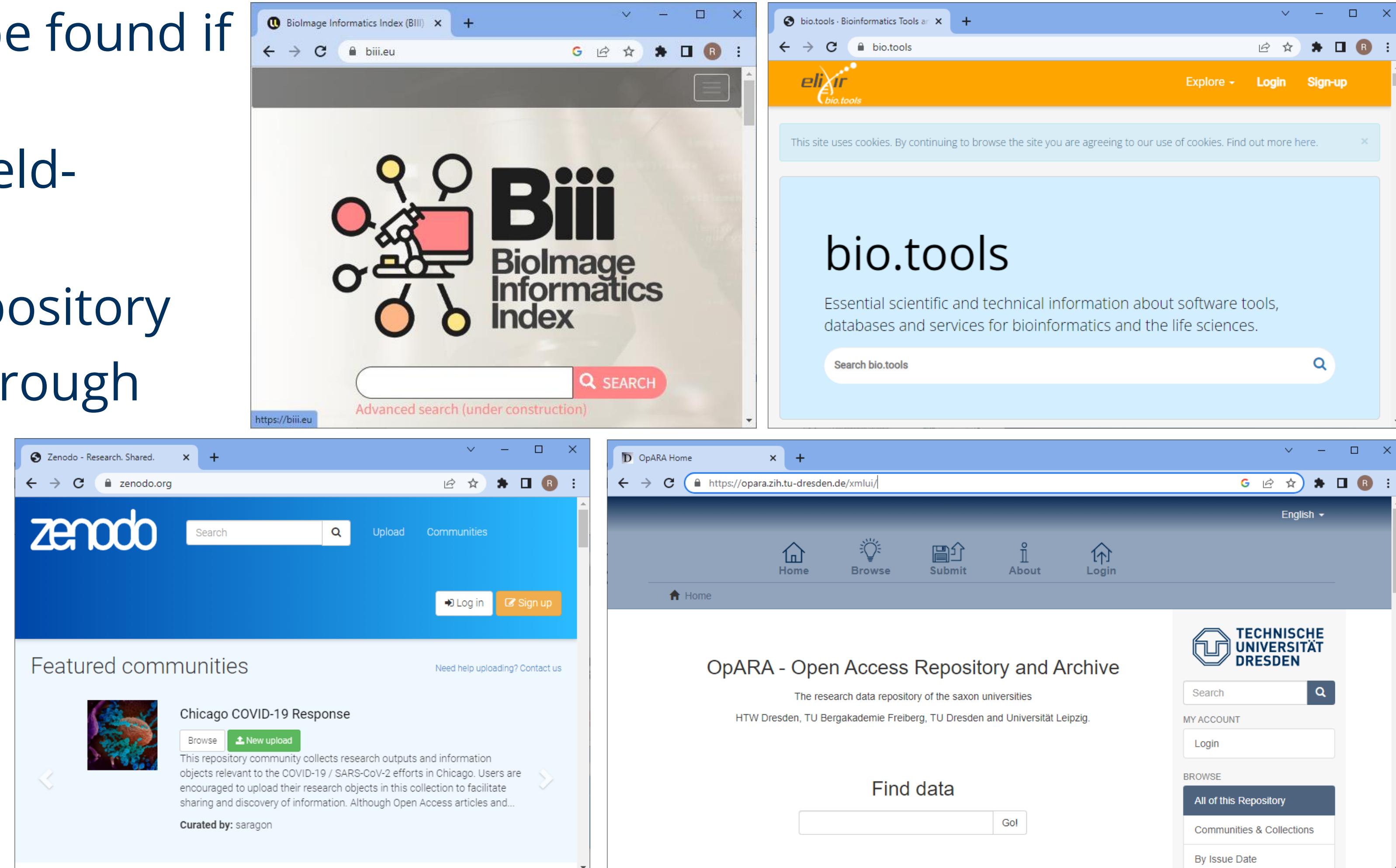
FAIR principles

- Findable
- Accessible
- Interoperable
- Reusable

=> State-of-the-art Research Data Management (RDM)

FAIR principles: Findable

- Research data / code / ... can be found if it's listed in *repositories*
 - Preferably: global, public, field-specific repository
 - Alternative: institutional repository
- Findability can be improved through attaching
 - *meta data*
 - *unique digital object identifiers (DOI)*



Quiz: Digital object identifiers

Which of these is a *unique* digital object identifier?

<https://twitter.com/haesleinhuepf/status/891596662782779392>

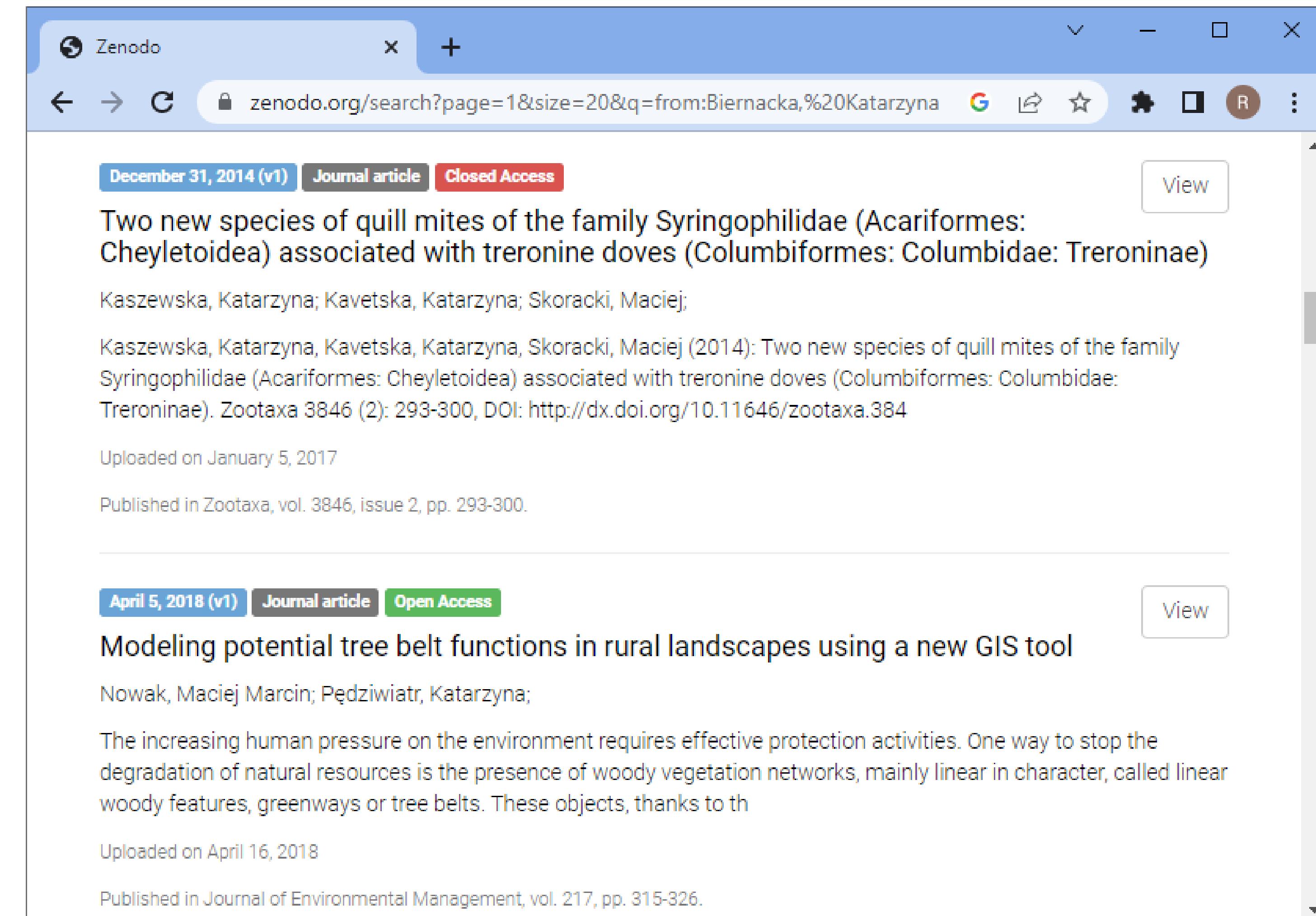
<https://doi.org/10.5281/zenodo.28325>

<https://github.com/haesleinhuepf/devbio-napari>

<https://napari.org/>

FAIR principles: Accessible

- Research data can be made accessible (after it was found by potential users)
 - Open Access is just one form of accessibility
 - Authentication enables other forms



The screenshot shows a web browser window with two entries from the Zenodo search results.

Top Record (Closed Access):

- Published: December 31, 2014 (v1)
- Type: Journal article
- Status: Closed Access
- Title: Two new species of quill mites of the family Syringophilidae (Acariformes: Cheyletoidea) associated with treronine doves (Columbiformes: Columbidae: Treroninae)
- Authors: Kaszewska, Katarzyna; Kavetska, Katarzyna; Skoracki, Maciej;
- Summary: Kaszewska, Katarzyna, Kavetska, Katarzyna, Skoracki, Maciej (2014): Two new species of quill mites of the family Syringophilidae (Acariformes: Cheyletoidea) associated with treronine doves (Columbiformes: Columbidae: Treroninae). Zootaxa 3846 (2): 293-300, DOI: <http://dx.doi.org/10.11646/zootaxa.384>
- Uploaded: January 5, 2017
- Published in: Zootaxa, vol. 3846, issue 2, pp. 293-300.

Bottom Record (Open Access):

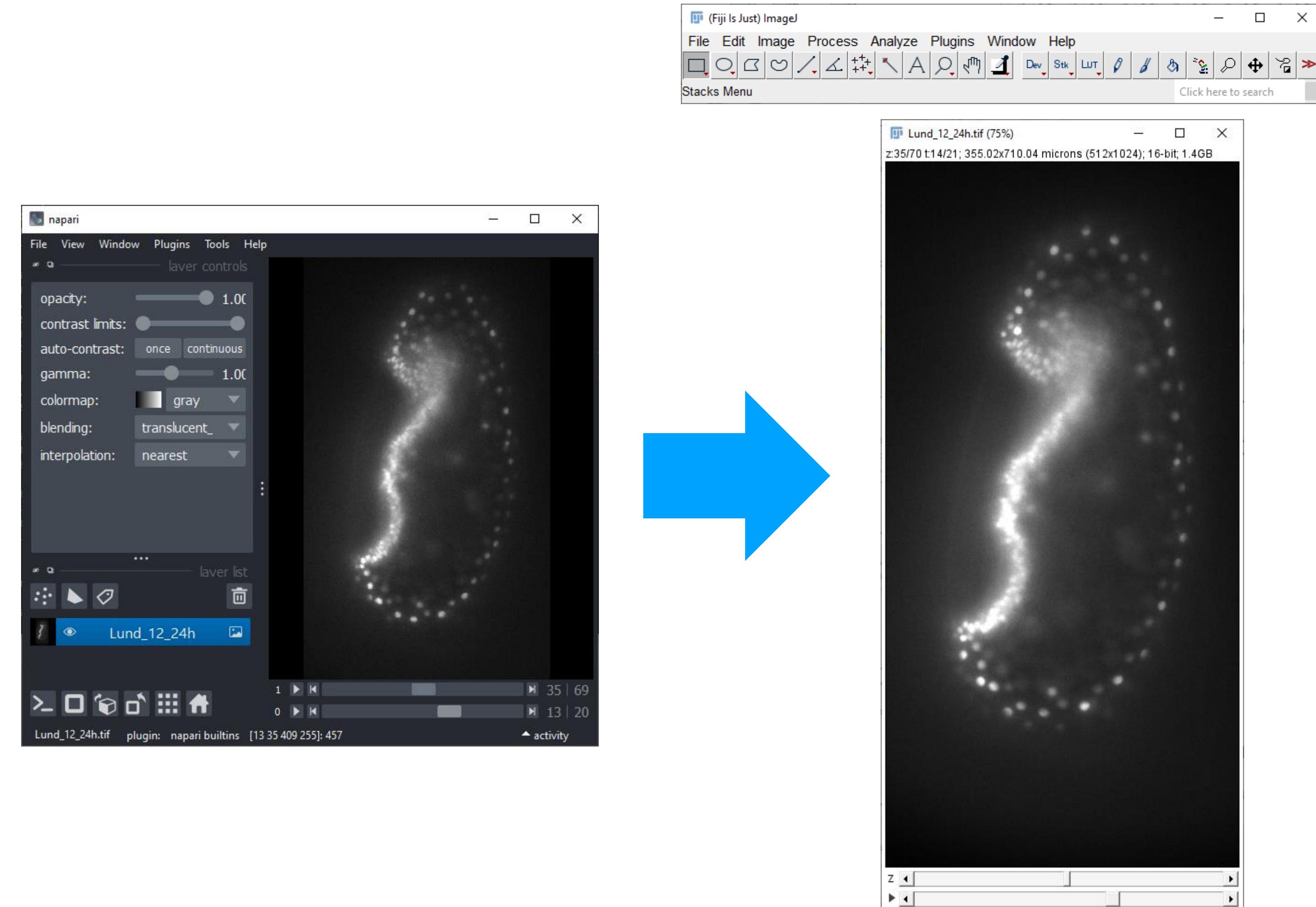
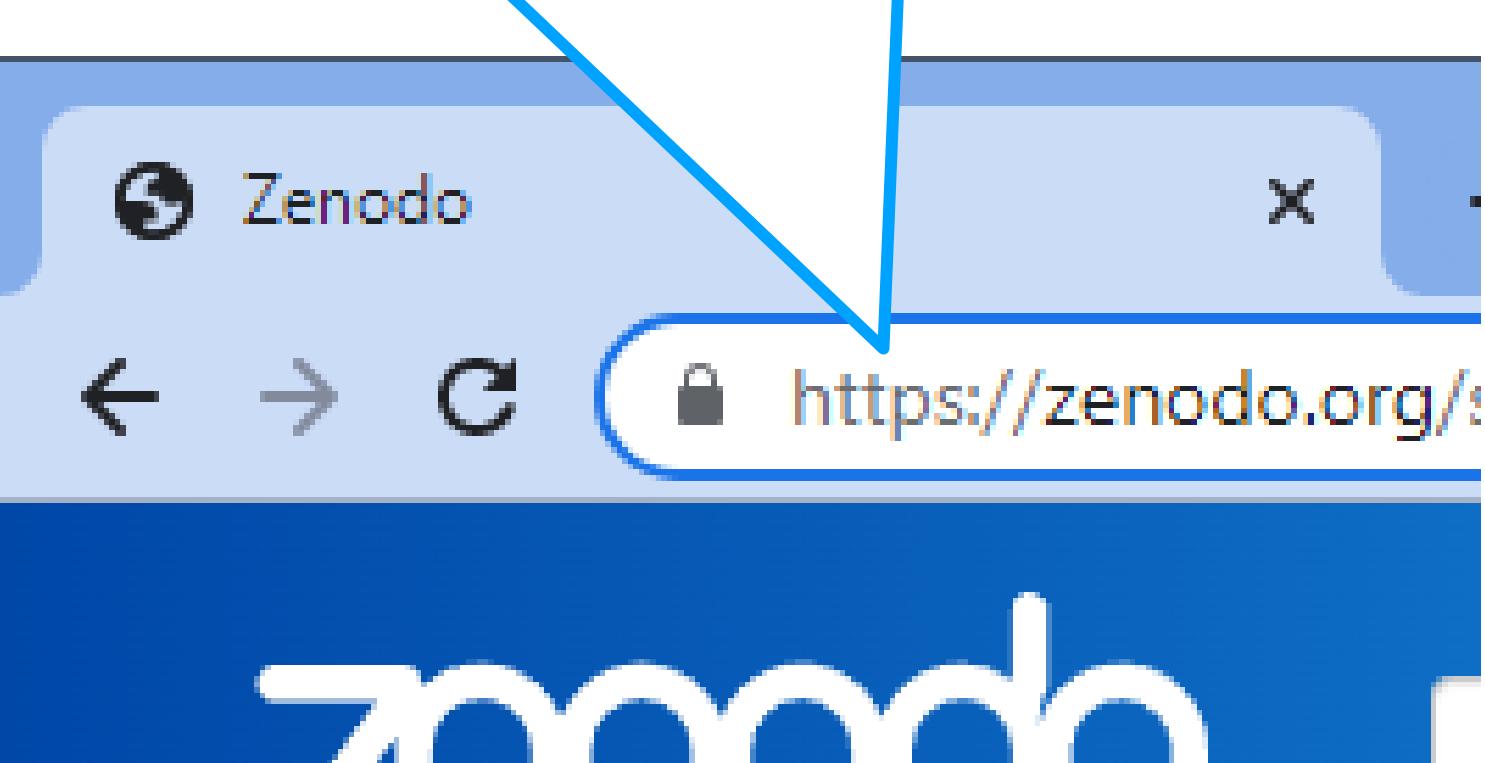
- Published: April 5, 2018 (v1)
- Type: Journal article
- Status: Open Access
- Title: Modeling potential tree belt functions in rural landscapes using a new GIS tool
- Authors: Nowak, Maciej Marcin; Pędziwiatr, Katarzyna;
- Summary: The increasing human pressure on the environment requires effective protection activities. One way to stop the degradation of natural resources is the presence of woody vegetation networks, mainly linear in character, called linear woody features, greenways or tree belts. These objects, thanks to th
- Uploaded: April 16, 2018
- Published in: Journal of Environmental Management, vol. 217, pp. 315-326.

FAIR principles: Interoperable

Data can be opened in multiple software through open and documented...

- file-formats
- protocols

hypertext transfer protocol (secure)

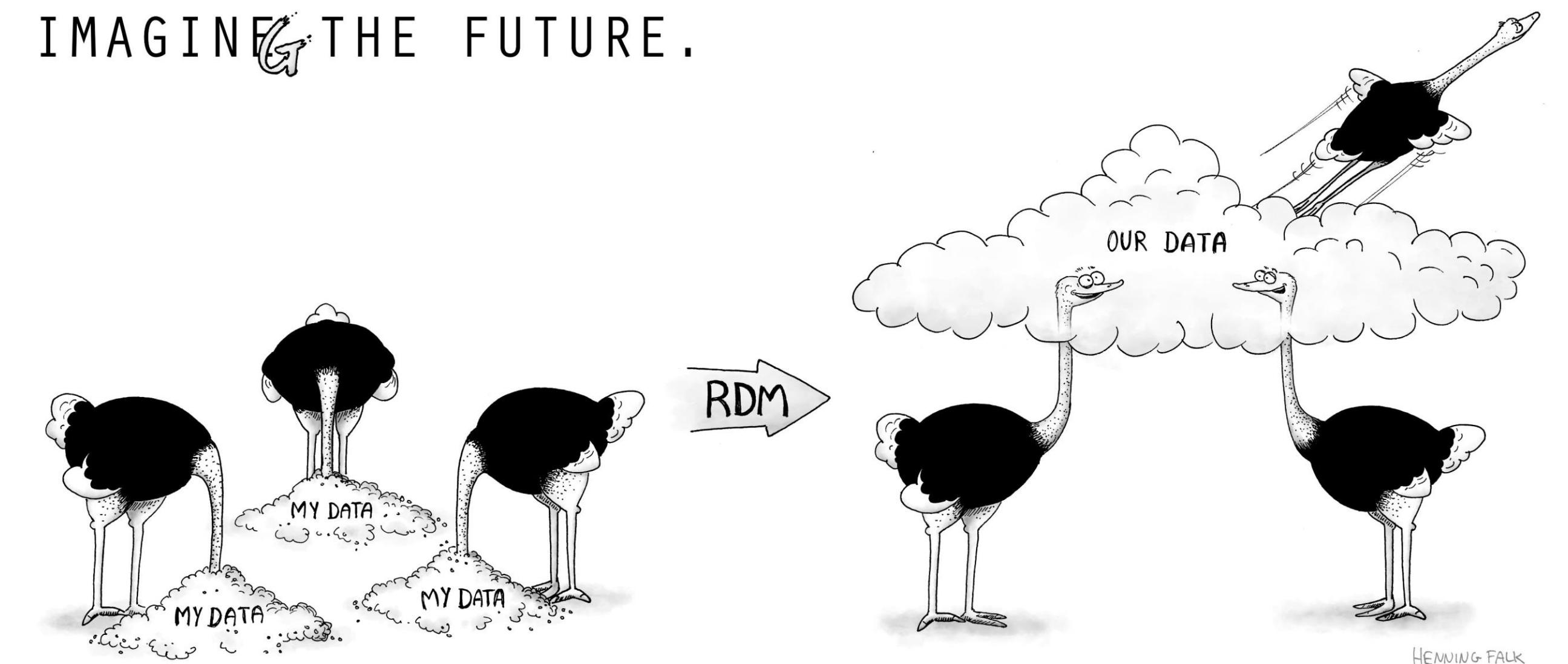


FAIR principles: Reusable

Data can be reused if

- Other FAIR principles are fulfilled
- Data is properly licensed
 - copyright statement is given
 - No copyright statement means: You do not have the right to copy

IMAGINE THE FUTURE.



Licensing: Creative Commons (CC)

Public domain (CC0)

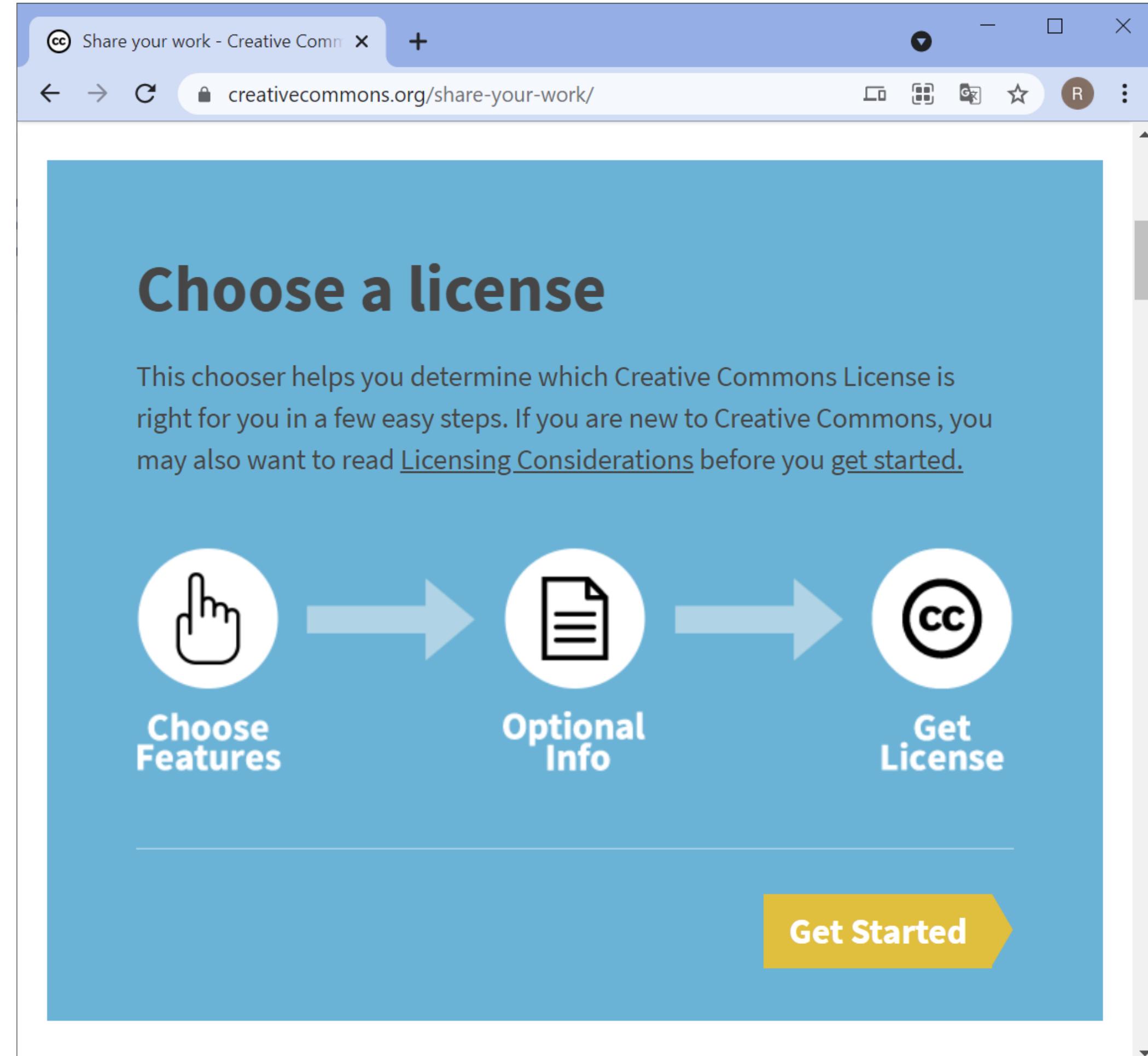
Attribution International (CC-BY)

Attribution ShareAlike Int. (CC-BY-SA)

Attribution Non-Commercial Int. (CC-BY-NC)

Attribution NoDerivatives Int. (CC-BY-ND)

+ Combinations, e.g. CC-BY-NC-ND



Content on this site is licensed under a [Creative Commons Attribution 4.0 International license](#). Icons by The Noun Project

Licensing: Creative Commons (CC)



Public domain (CC0)

Everyone can reuse without mentioning the source or author of the shared resource.

Public domain licenses cannot be revoked.

The author must own the right to copy (copyright) the resource.

—If you authored work as part of your job, you may not be the copyright holder. (check your employers' guidelines)

Employers don't like this one because you give away the rights to *exploit* your work.

Content on this slide was adapted from <https://creativecommons.org/about/cclicenses/> which is licensed under a [Creative Commons Attribution 4.0 International license](#). [Icons](#) by The Noun Project.

Licensing: Creative Commons (CC)



“By attribution”

CC BY: This license allows reusers to distribute, remix, adapt, and build upon the material in any medium or format, so long as attribution is given to the creator. The license allows for commercial use.

CC BY includes the following elements:

BY  - Credit must be given to the creator

Content on this slide was adapted from <https://creativecommons.org/about/cclicenses/> which is licensed under a [Creative Commons Attribution 4.0 International license](#). Icons by The Noun Project.

Licensing: Creative Commons (CC)



Example



You must put such a sentence and keep the link to CC-BY

Figure adapted from <https://www.openmicroscopy.org/> licensed by University of Dundee & Open Microscopy Environment under [Creative Commons Attribution 4.0 International License](#)

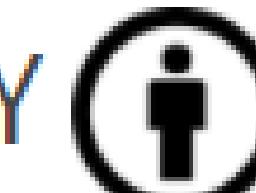
Licensing: Creative Commons (CC)



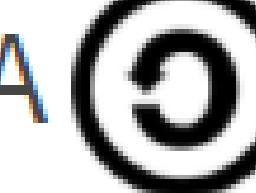
CC BY-SA: This license allows reusers to distribute, remix, adapt, and build upon the material in any medium or format, so long as attribution is given to the creator. The license allows for commercial use. If you remix, adapt, or build upon the material, you must license the modified material under identical terms.

“Share alike”

CC BY-SA includes the following elements:

BY 

Credit must be given to the creator

SA 

Adaptations must be shared under the same terms

“Restrictive”
licensing

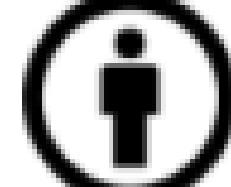
Content on this slide was adapted from <https://creativecommons.org/about/cclicenses/> which is licensed under a [Creative Commons Attribution 4.0 International license](#). Icons by The Noun Project.

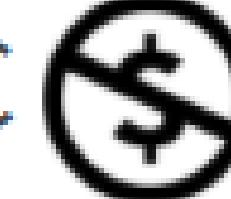
Licensing: Creative Commons (CC)



CC BY-NC: This license allows reusers to distribute, remix, adapt, and build upon the material in any medium or format for noncommercial purposes only, and only so long as attribution is given to the creator.

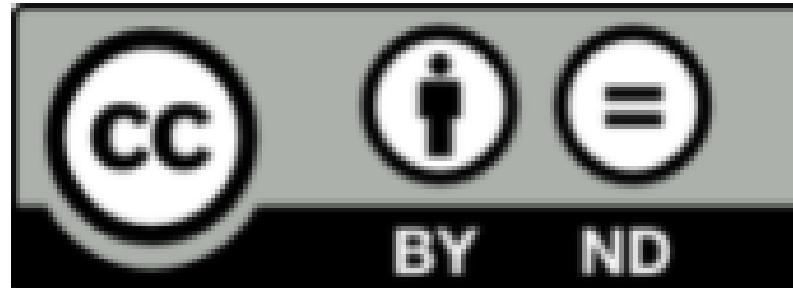
It includes the following elements:

BY  – Credit must be given to the creator

NC  – Only noncommercial uses of the work are permitted

Content on this slide was adapted from <https://creativecommons.org/about/cclicenses/> which is licensed under a [Creative Commons Attribution 4.0 International license](#). [Icons](#) by The Noun Project.

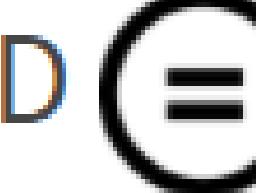
Licensing: Creative Commons (CC)



CC BY-ND: This license allows reusers to copy and distribute the material in any medium or format in unadapted form only, and only so long as attribution is given to the creator. The license allows for commercial use.

CC BY-ND includes the following elements:

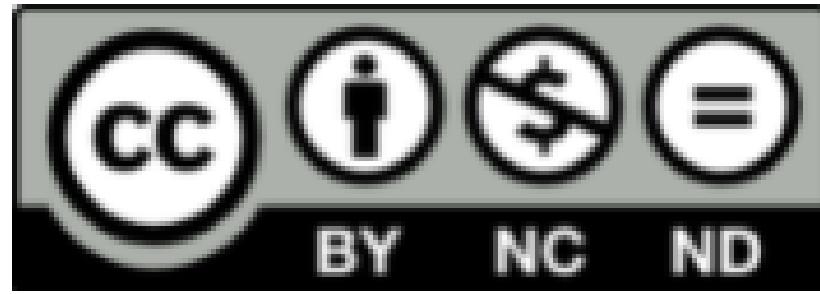
BY  – Credit must be given to the creator

ND  – No derivatives or adaptations of the work are permitted

“Restrictive”
licensing

Content on this slide was adapted from <https://creativecommons.org/about/cclicenses/> which is licensed under a [Creative Commons Attribution 4.0 International license](#). [Icons](#) by The Noun Project.

Licensing: Creative Commons (CC)



CC BY-NC-ND: This license allows reusers to copy and distribute the material in any medium or format in unadapted form only, for noncommercial purposes only, and only so long as attribution is given to the creator.

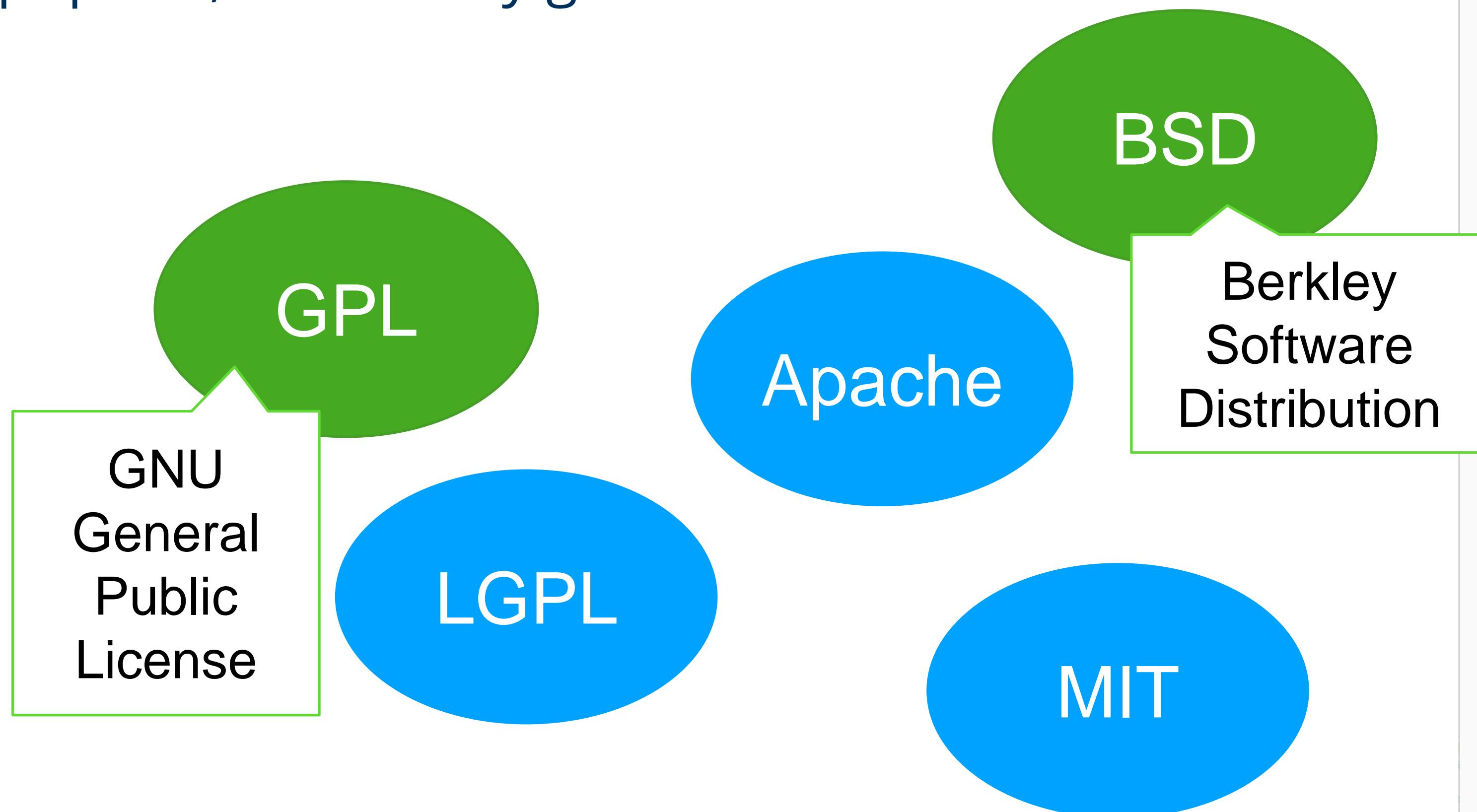
CC BY-NC-ND includes the following elements:

- BY – Credit must be given to the creator
- NC – Only noncommercial uses of the work are permitted
- ND – No derivatives or adaptations of the work are permitted

Content on this slide was adapted from <https://creativecommons.org/about/cclicenses/> which is licensed under a [Creative Commons Attribution 4.0 International license](#). [Icons](#) by The Noun Project.

Licensing software

In the software world, other licenses are more popular, historically grown.



Choose an open source license

An open source license protects contributors and users. Businesses and savvy developers won't touch a project without this protection.

Which of the following best describes your situation?

- I need to work in a community.** (Icon: three people) Use the [license preferred by the community](#) you're contributing to or depending on. Your project will fit right in.
- I want it simple and permissive.** (Icon: anchor) The [MIT License](#) is short and to the point. It lets people do almost anything they want with your project, like making and distributing closed source versions.
- I care about sharing improvements.** (Icon: double arrow) The [GNU GPLv3](#) also lets people do almost anything they want with your project, except distributing closed source versions.

What if none of these work for me?

- My project isn't software.** (Icon: document) There are [licenses for that](#).
- I want more choices.** (Icon: gear) [More licenses are available](#).
- I don't want to choose a license.** (Icon: question mark) [Here's what happens if you don't](#).

The content of this site is licensed under the Creative Commons Attribution 3.0 Unported License.

About Terms of Service Help improve this page Curated with ❤ by GitHub, Inc. and You!

Licensing Software: GPL

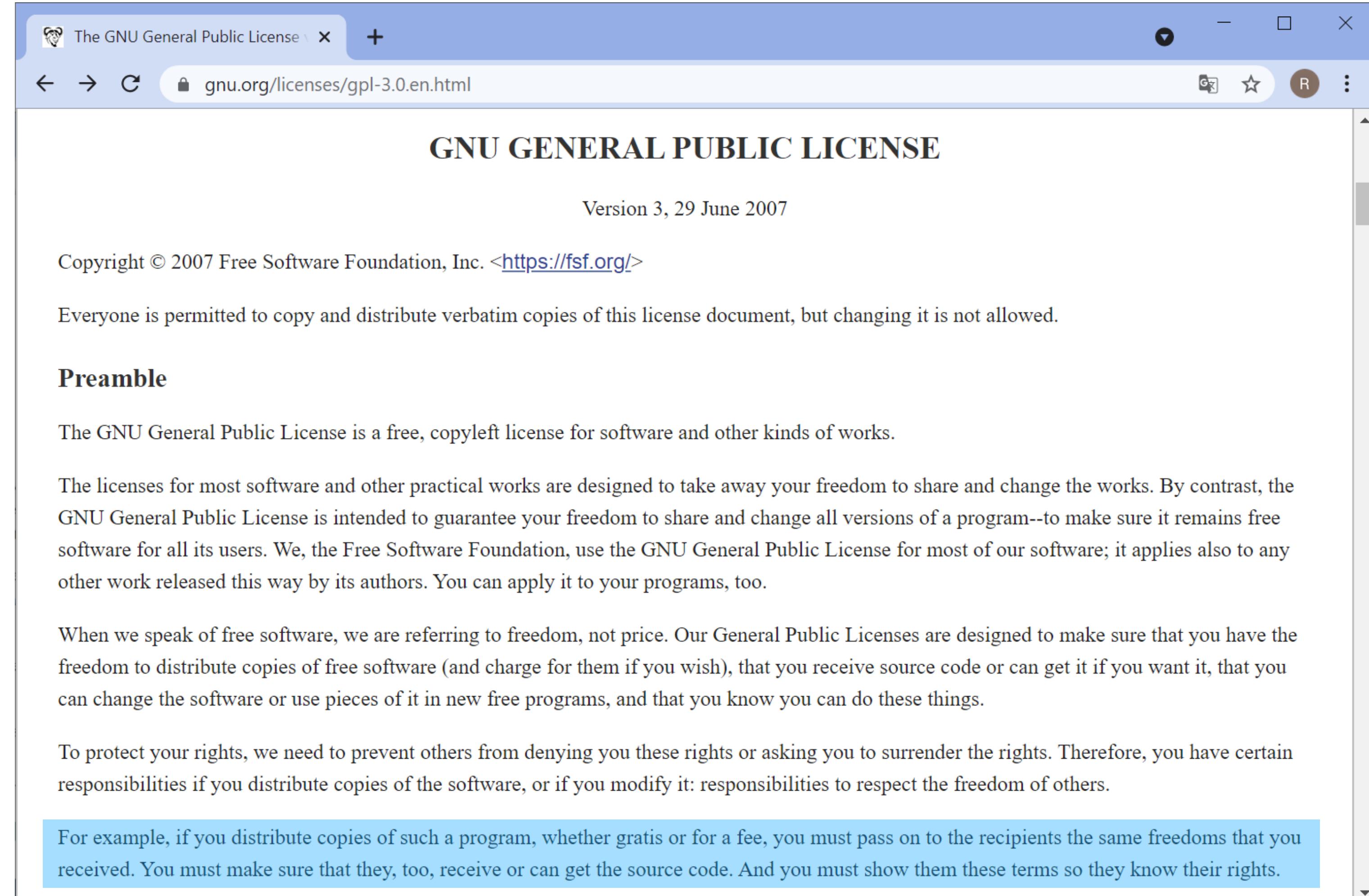
GPL

- Derivatives must also be GPL-licensed

“Restrictive”
licensing

See also:

- Lesser General Public License (LGPL)
 - Integrate LGPL-licensed code into not-LGPL-licensed code



The screenshot shows a web browser window displaying the GNU General Public License version 3. The title "GNU GENERAL PUBLIC LICENSE" is at the top, followed by the date "Version 3, 29 June 2007". Below this, it says "Copyright © 2007 Free Software Foundation, Inc. <<https://fsf.org/>>". The text explains that everyone is permitted to copy and distribute verbatim copies of the license document, but changing it is not allowed. A "Preamble" section follows, stating that the GNU General Public License is a free, copyleft license for software and other kinds of works. It contrasts this with other licenses and emphasizes the freedom to share and change the works. The text continues to explain the purpose of the license, mentioning the Free Software Foundation and its application to various software projects. It also discusses the concept of free software and the rights it grants. A blue callout box highlights a specific point about distributing copies of the program while maintaining the same freedoms for recipients.

Licensing Software: BSD0

Copyright (C) [year] by [copyright holder] <[email]>

Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted.

Similar to CC0
(public domain)

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE

Licensing Software: BSD0

Copyright (C) [year] by [copyright holder] <[email]>

Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE

Disclaimer

Whatever you do with it,
we [the authors] are not
liable

Relevant for [image]
data analysis script-
authors!

Licensing Software: BSD2

Copyright (c) <year>, <copyright holder>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

} Similar to CC-BY

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

Licensing Software: BSD3

Copyright <year> <copyright holder>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

Similar to CC-BY

You must not use the copyright holder's name to endorse your derivative of the work.

That's also part of all CC-BY licenses

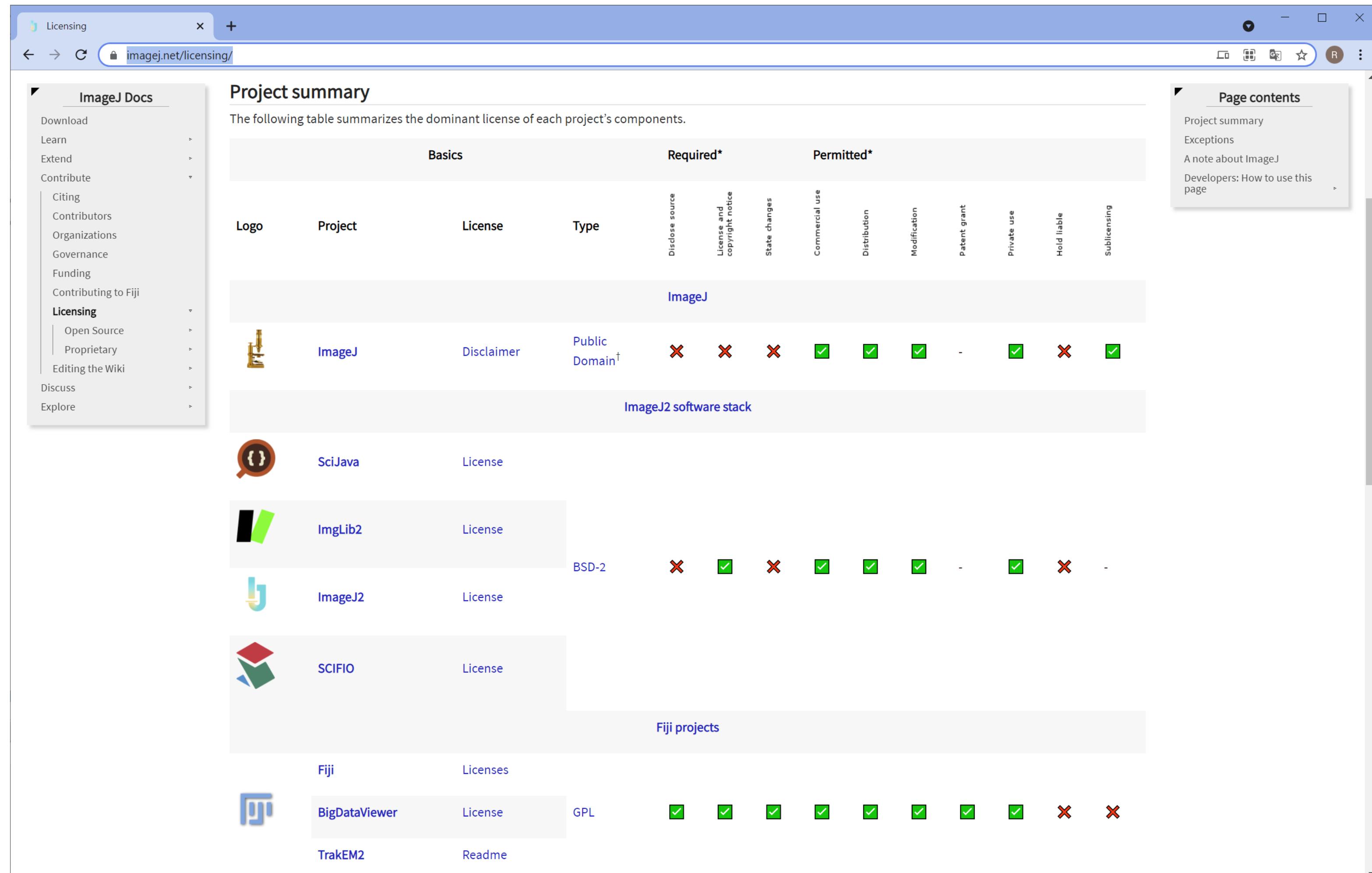
How/when to decide for a license

How?

- Choose a license compatible with your ecosystem

When?

- When the project starts
- As early as possible
- Changing the license later may be hard.
- Beware: Your employer might be the copyright holder. They have the final word on how to license, publish and make accessible your work!

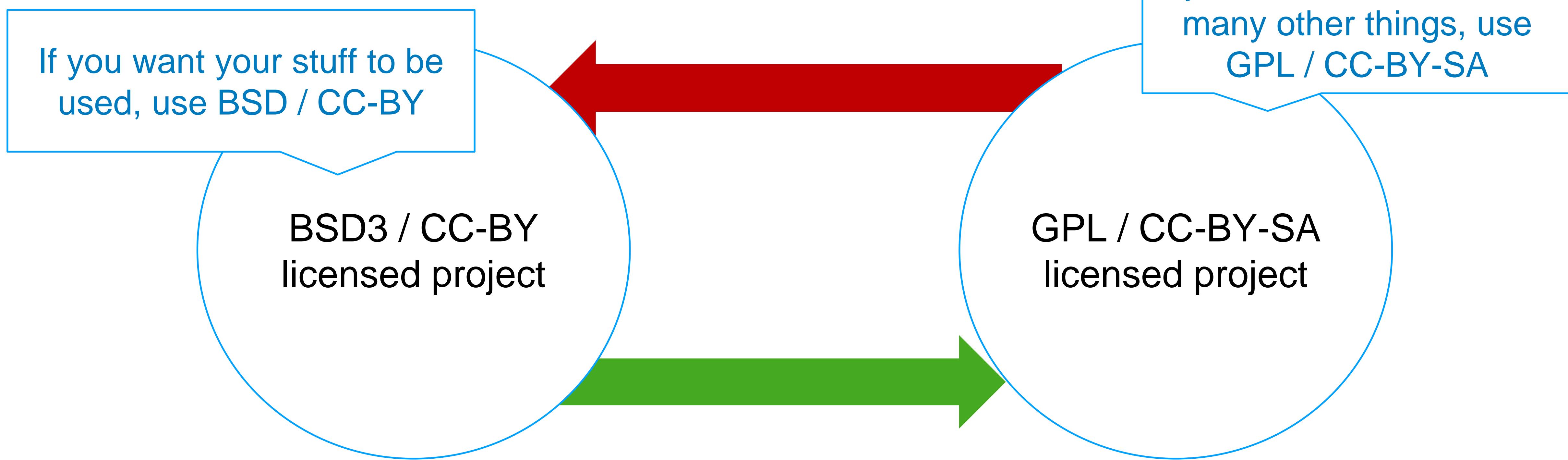


The screenshot shows a web browser displaying the 'Licensing' page from imagej.net/licensing/. The page title is 'Licensing'. On the left, there's a sidebar with 'ImageJ Docs' navigation links: Download, Learn, Extend, Contribute, Citing, Contributors, Organizations, Governance, Funding, Contributing to Fiji, Licensing (selected), Open Source, Proprietary, Editing the Wiki, Discuss, and Explore. The main content area is titled 'Project summary' with the sub-instruction: 'The following table summarizes the dominant license of each project's components.' A table follows, with columns: Basics, Required*, Permitted*, Logo, Project, License, Type, Disclose source, License and copyright notice, State changes, Commercial use, Distribution, Modification, Patent grant, Private use, Hold liable, and Sublicensing. The table lists several projects:

- ImageJ**: Disclaimer, Public Domain[†]. Status: No (red X) for Disclose source, License and copyright notice, State changes, and Distribution; Yes (green checkmark) for Commercial use, Modification, Patent grant, Private use, Hold liable, and Sublicensing.
- ImageJ2 software stack**: License. Status: No (red X) for Disclose source, License and copyright notice, State changes, and Distribution; Yes (green checkmark) for Commercial use, Modification, Patent grant, Private use, Hold liable, and Sublicensing.
- SciJava**: License. Status: No (red X) for Disclose source, License and copyright notice, State changes, and Distribution; Yes (green checkmark) for Commercial use, Modification, Patent grant, Private use, Hold liable, and Sublicensing.
- ImgLib2**: License. Status: No (red X) for Disclose source, License and copyright notice, State changes, and Distribution; Yes (green checkmark) for Commercial use, Modification, Patent grant, Private use, Hold liable, and Sublicensing.
- ImageJ2**: License. Status: No (red X) for Disclose source, License and copyright notice, State changes, and Distribution; Yes (green checkmark) for Commercial use, Modification, Patent grant, Private use, Hold liable, and Sublicensing.
- SCIFIO**: License. Status: No (red X) for Disclose source, License and copyright notice, State changes, and Distribution; Yes (green checkmark) for Commercial use, Modification, Patent grant, Private use, Hold liable, and Sublicensing.
- Fiji projects**: Fiji: Licenses. Status: Yes (green checkmark) for all columns except Hold liable and Sublicensing. BigDataViewer: License. Status: Yes (green checkmark) for all columns except Hold liable and Sublicensing. TrakEM2: Readme. Status: No (red X) for Hold liable and Sublicensing.

Permissive versus restrictive

Use a license which
is compatible to other projects you're collaborating with and
fits to your needs / role.

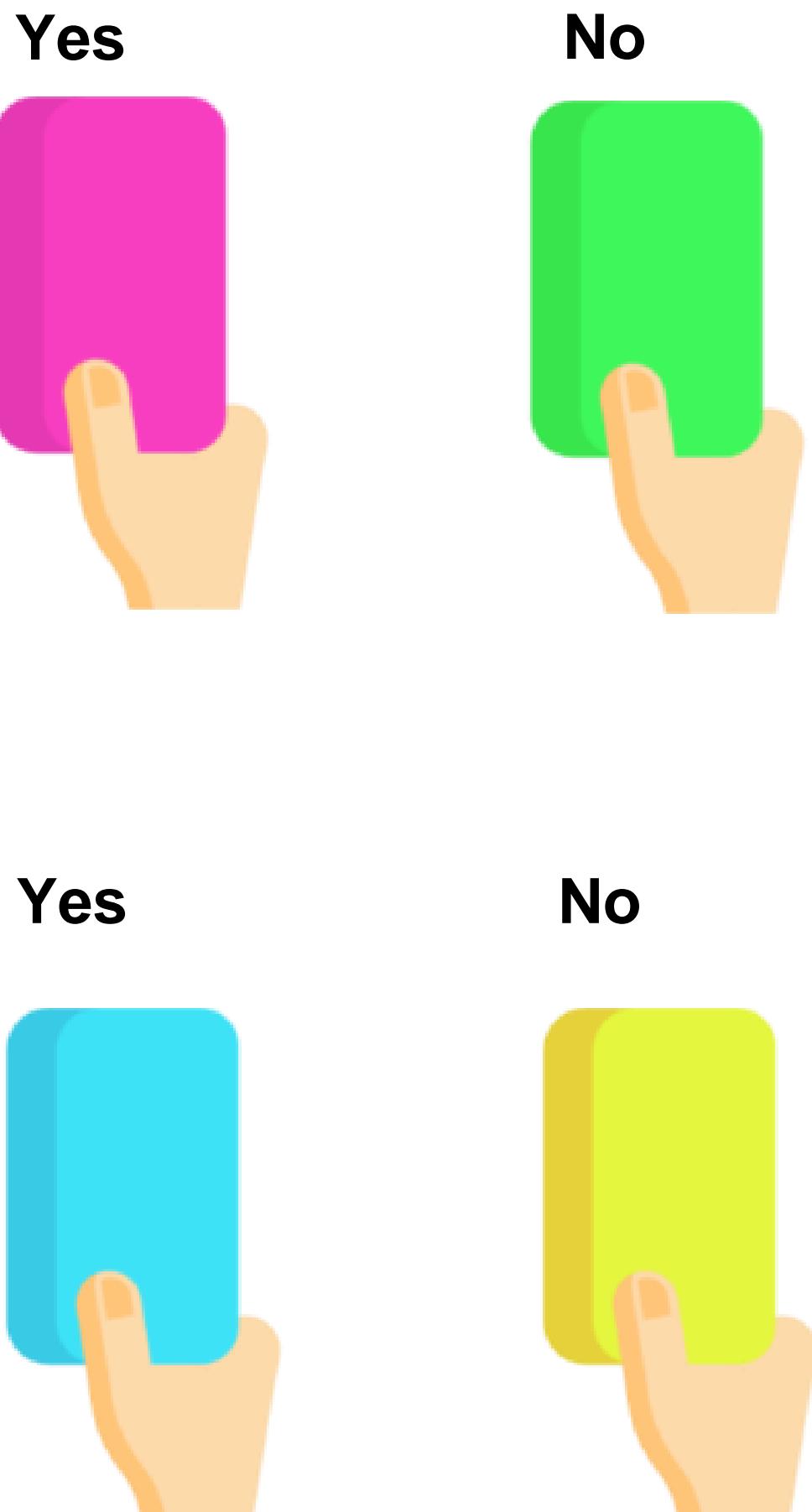


Quiz

May I use one of the Figures from this preprint?

The image shows a screenshot of a bioRxiv preprint page. At the top left is the CSHL logo and the text "Cold Spring Harbor Laboratory". The bioRxiv logo is prominently displayed with the tagline "THE PREPRINT SERVER FOR BIOLOGY". A yellow box contains a reminder: "bioRxiv posts many COVID19-related papers. A reminder: they have not been formally peer-reviewed and should not guide health-related behavior or be reported in the press as conclusive." Below this, there are "New Results" and a "Follow this preprint" button. The title of the preprint is "Content-Aware Image Restoration: Pushing the Limits of Fluorescence Microscopy". The authors listed are Martin Weigert, Uwe Schmidt, Tobias Boothe, Andreas Müller, Alexandr Dibrov, Akanksha Jain, Benjamin Wilhelm, Deborah Schmidt, Coleman Broaddus, Siân Culley, Mauricio Rocha-Martins, Fabián Segovia-Miranda, Caren Norden, Ricardo Henriques, Marino Zerial, Michele Solimena, Jochen Rink, Pavel Tomancak, Loic Royer, Florian Jug, and Eugene W. Myers. The DOI is provided as <https://doi.org/10.1101/236463>. It is noted that the article is now published in *Nature Methods* with DOI [10.1038/s41592-018-0216-7](https://doi.org/10.1038/s41592-018-0216-7). Below the title, there are social media sharing icons: 1 comment, 0 likes, 2 reviews, 0 citations, 2 full-text downloads, 0 versions, and 618 tweets. Navigation tabs include Abstract, Full Text, Info/History (which is selected), Metrics, and Preview PDF. The "ARTICLE INFORMATION" section includes the DOI and a History entry for July 3, 2018. The "ARTICLE VERSIONS" section lists five versions, with Version 5 being the current one viewed. A copyright notice at the bottom states: "Copyright The copyright holder for this preprint is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under a CC-BY-NC-ND 4.0 International license."

May I download and redistribute this preprint to students of a course for free?



Quiz

May I reuse code from this repository in my own BSD-licensed work?

Yes



No



cnr-isti-vclab/meshlab: The open source mesh processing system

github.com/cnr-isti-vclab/meshlab

cnr-isti-vclab / meshlab Public

Watch 150 Fork 686 Star 3.3k

Code Issues 124 Pull requests 2 Discussions Actions Projects Wiki Security Insights

main 3 branches 40 tags Go to file Add file Code About

The open source mesh processing system

[www.meshlab.net](#)

point-cloud mesh mesh-generation
3d-printing 3d-scanning 3d
3d-reconstruction 3d-models
mesh-processing mesh-editing
mesh-simplification triangle-mesh

Readme
GPL-3.0 license
3.3k stars
150 watching
686 forks

alemuntoni bugfix while loading exif info 2672c14 22 days ago 10,254 commits

.github fix missing windeployqt in windows workflows 5 months ago

docs Fix various typos 8 months ago

resources icon on windows folder 5 months ago

sample add gltf samples 16 months ago

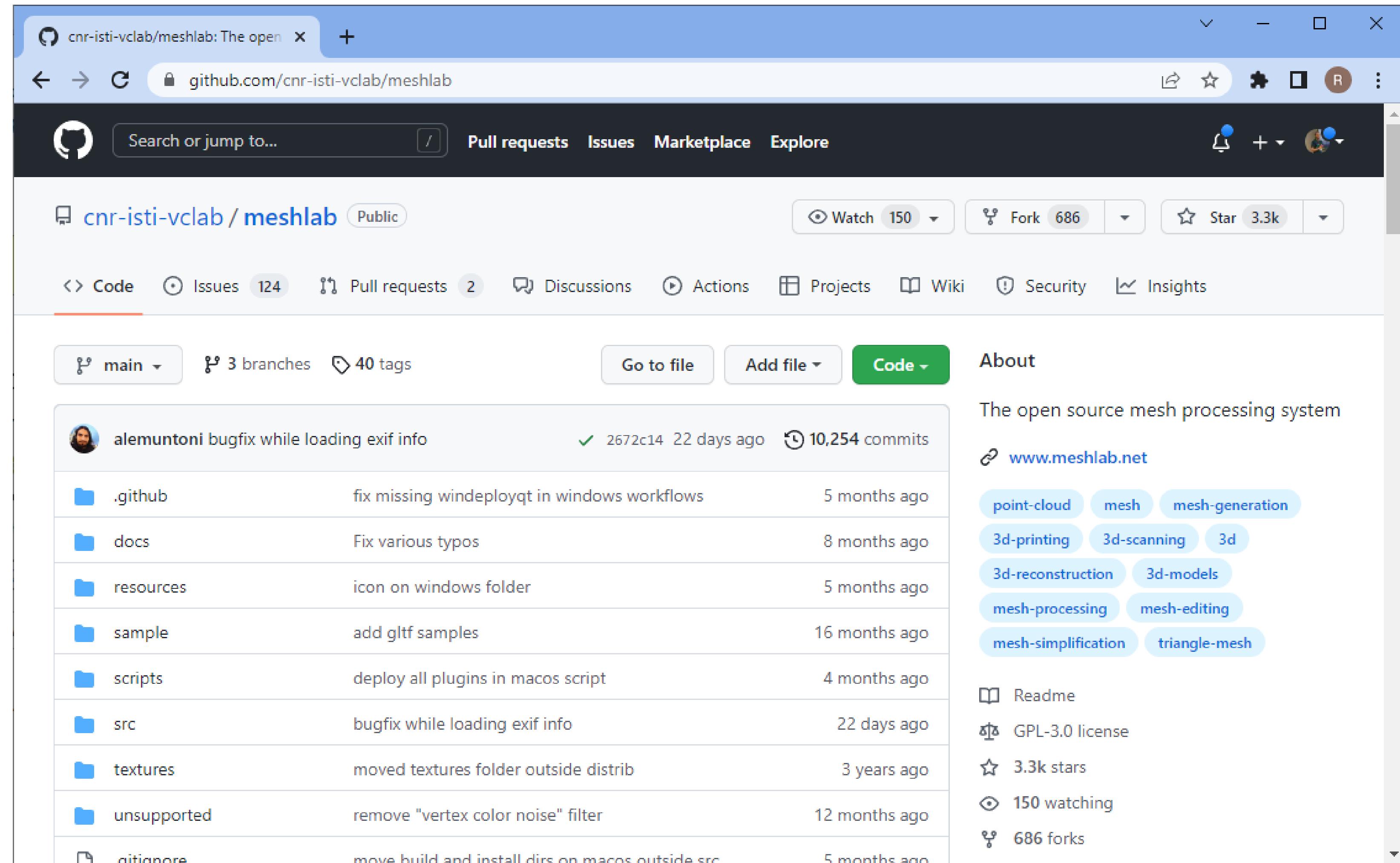
scripts deploy all plugins in macos script 4 months ago

src bugfix while loading exif info 22 days ago

textures moved textures folder outside distrib 3 years ago

unsupported remove "vertex color noise" filter 12 months ago

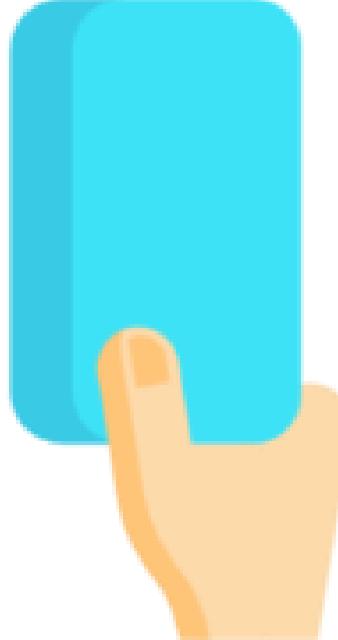
.aitianore move build and install dirs on macos outside src 5 months ago



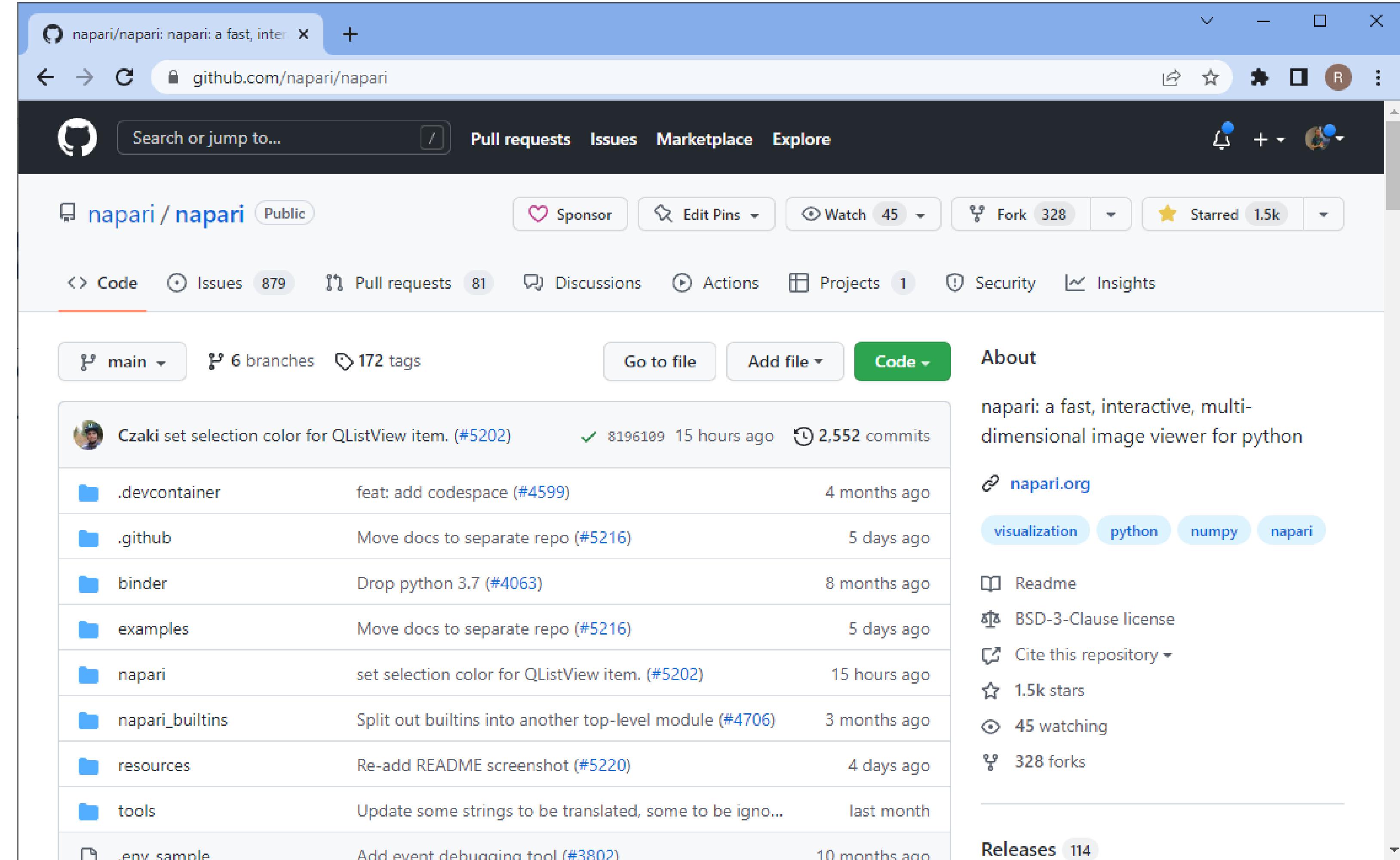
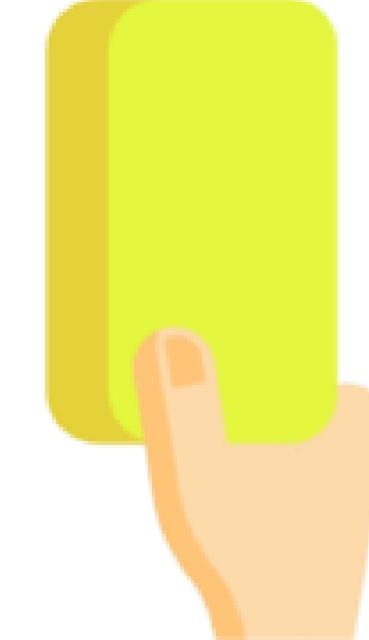
Quiz

May I reuse code from this repository in my own GPL-licensed work?

Yes



No



napari/napari: napari: a fast, interactive, multi-dimensional image viewer for python

Code

Issues 879 Pull requests 81 Discussions Actions Projects 1 Security Insights

main 6 branches 172 tags Go to file Add file Code

Czaki set selection color for QListView item. (#5202) 15 hours ago 2,552 commits

.devcontainer feat: add codespace (#4599) 4 months ago

.github Move docs to separate repo (#5216) 5 days ago

binder Drop python 3.7 (#4063) 8 months ago

examples Move docs to separate repo (#5216) 5 days ago

napari set selection color for QListView item. (#5202) 15 hours ago

napari_builtins Split out builtins into another top-level module (#4706) 3 months ago

resources Re-add README screenshot (#5220) 4 days ago

tools Update some strings to be translated, some to be igno... last month

.env_sample Add event debugging tool (#3802) 10 months ago

About napari.org visualization python numpy napari

Readme BSD-3-Clause license Cite this repository 1.5k stars 45 watching 328 forks

Releases 114

Take home message

If you share material (openly or not)

license it

and it'll be harder to steal it

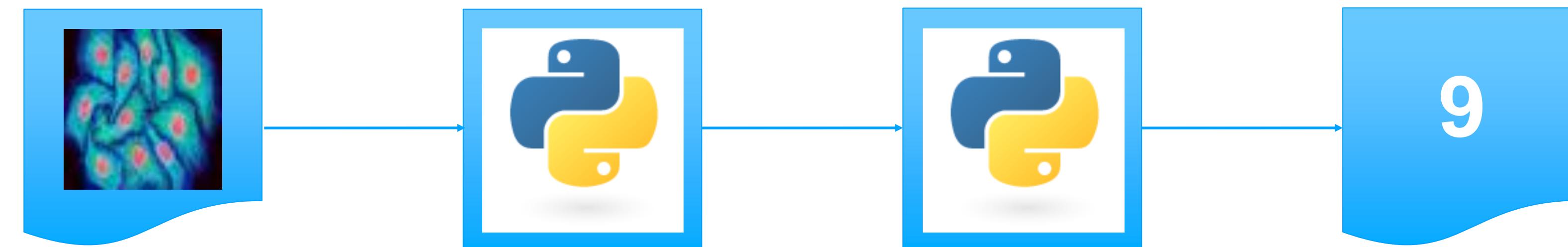
Python Algorithms: Conditions, loops, functions and custom libraries

Robert Haase

April 2023

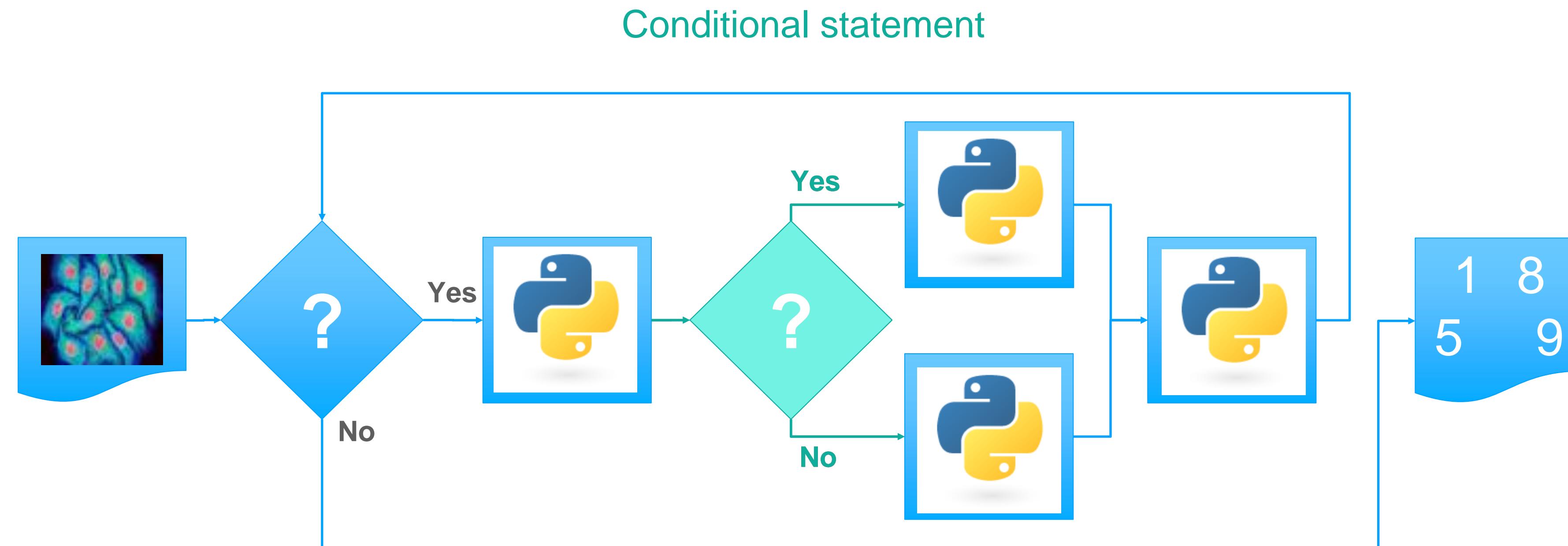
Conditions

Data science workflows *rarely* look like this



Conditions

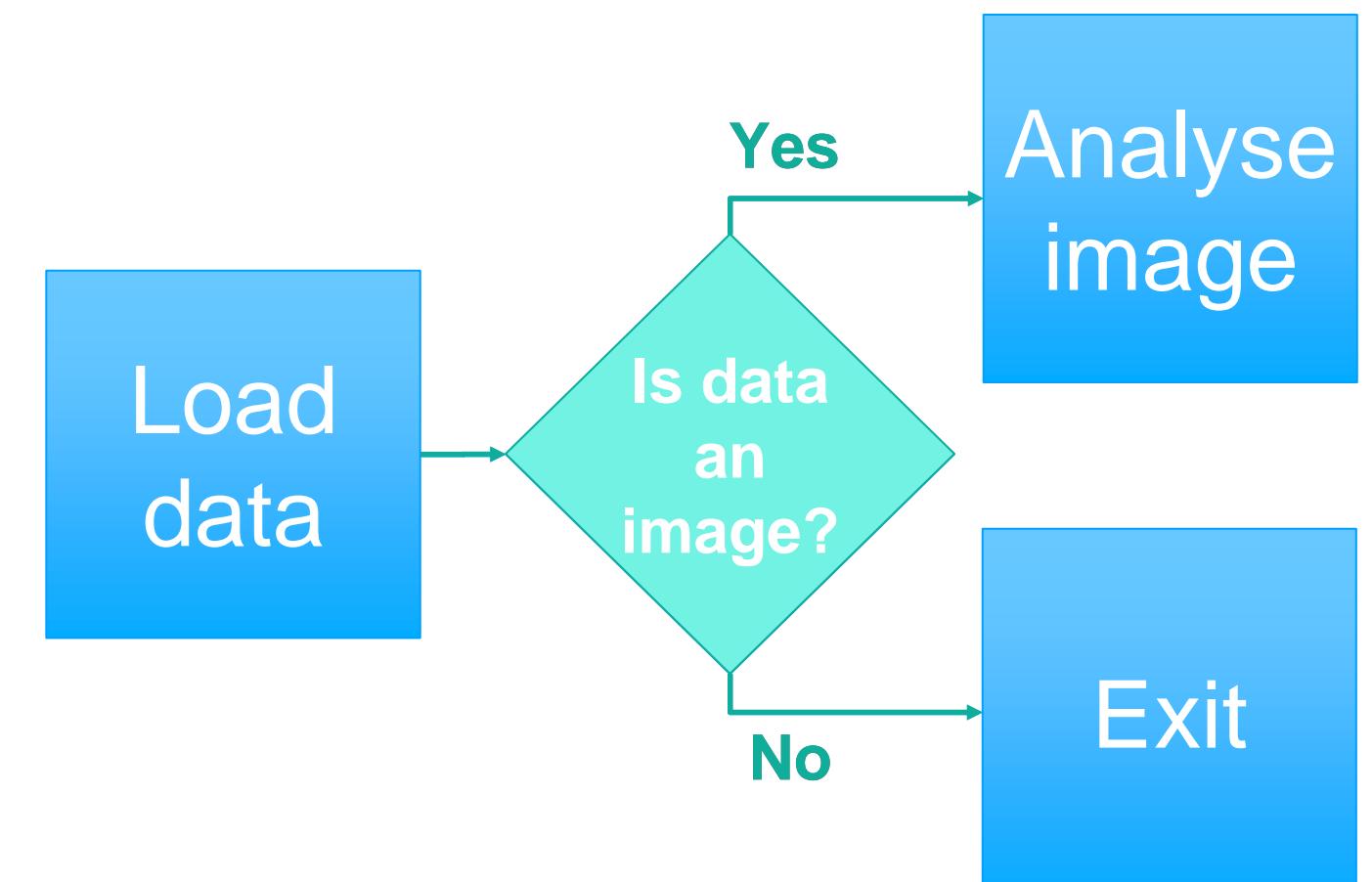
Data science workflows *rather* look like this



Conditions

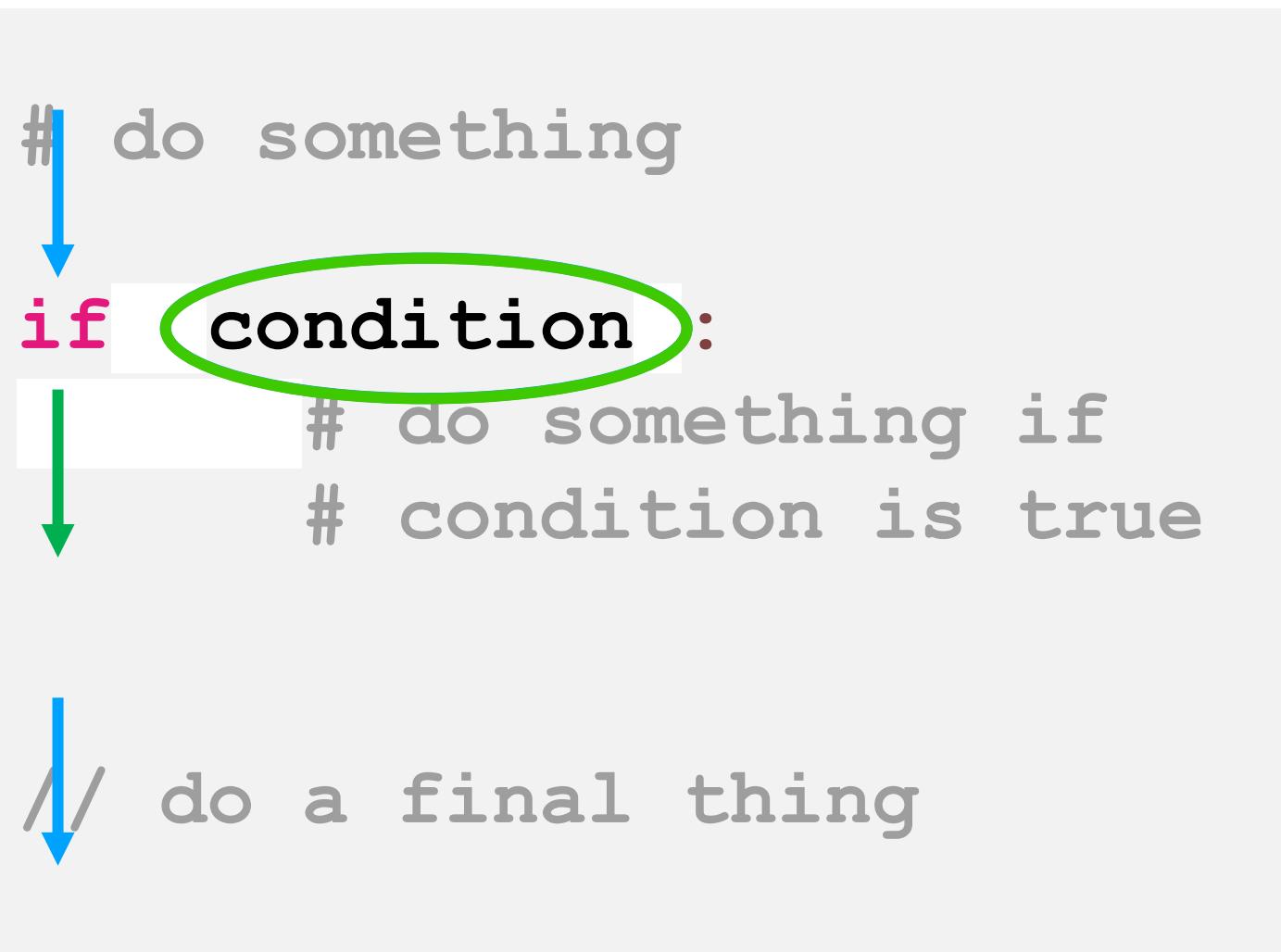
Conditional statements can be used to

- Check if pre-requisites are met
- Check if data has the right format
- Check if processing results are within an expected range
- Check for errors



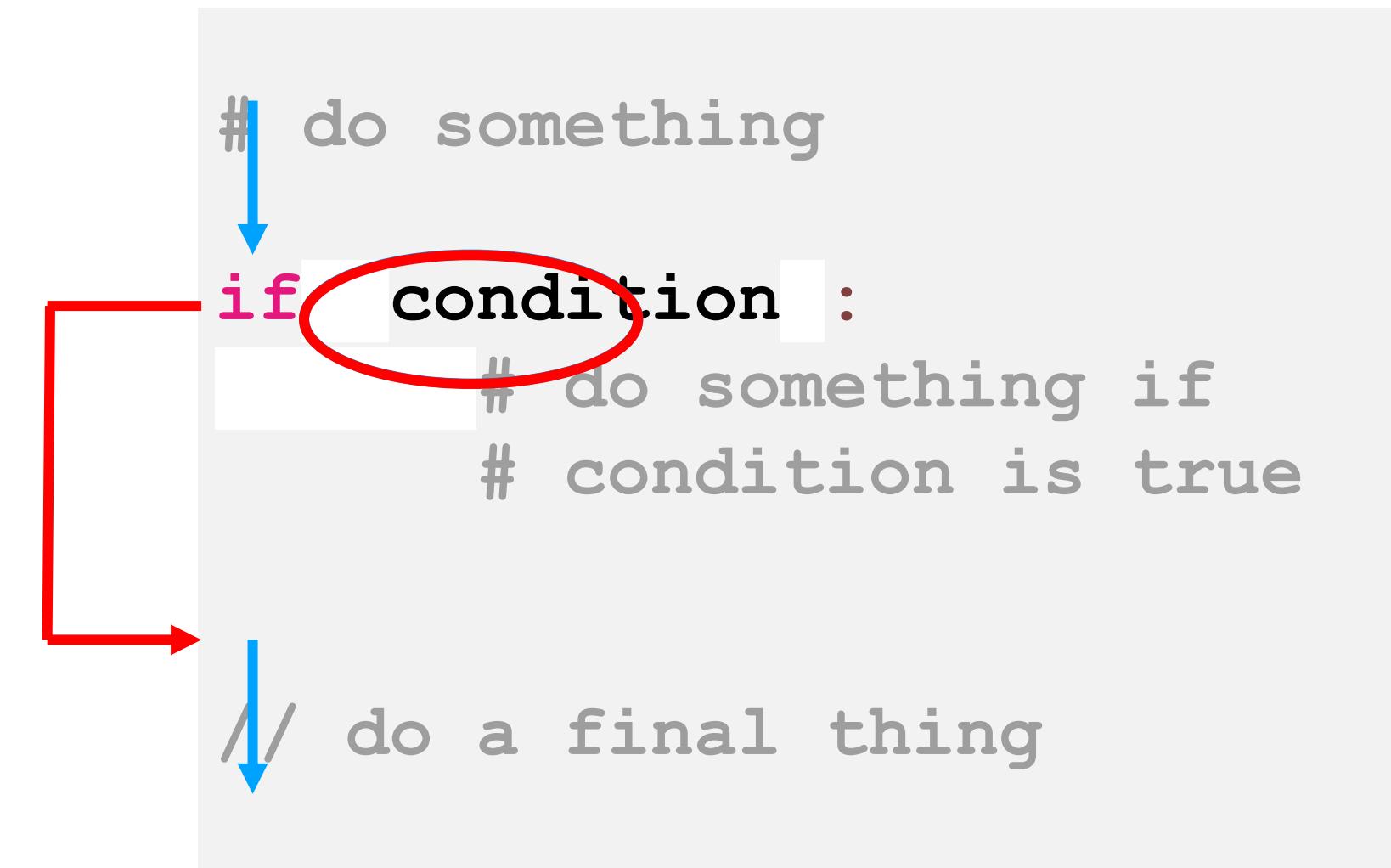
If-statement

Depending on a condition, some lines of code are executed or not.



If-statement

Depending on a condition, some lines of code are executed or not.



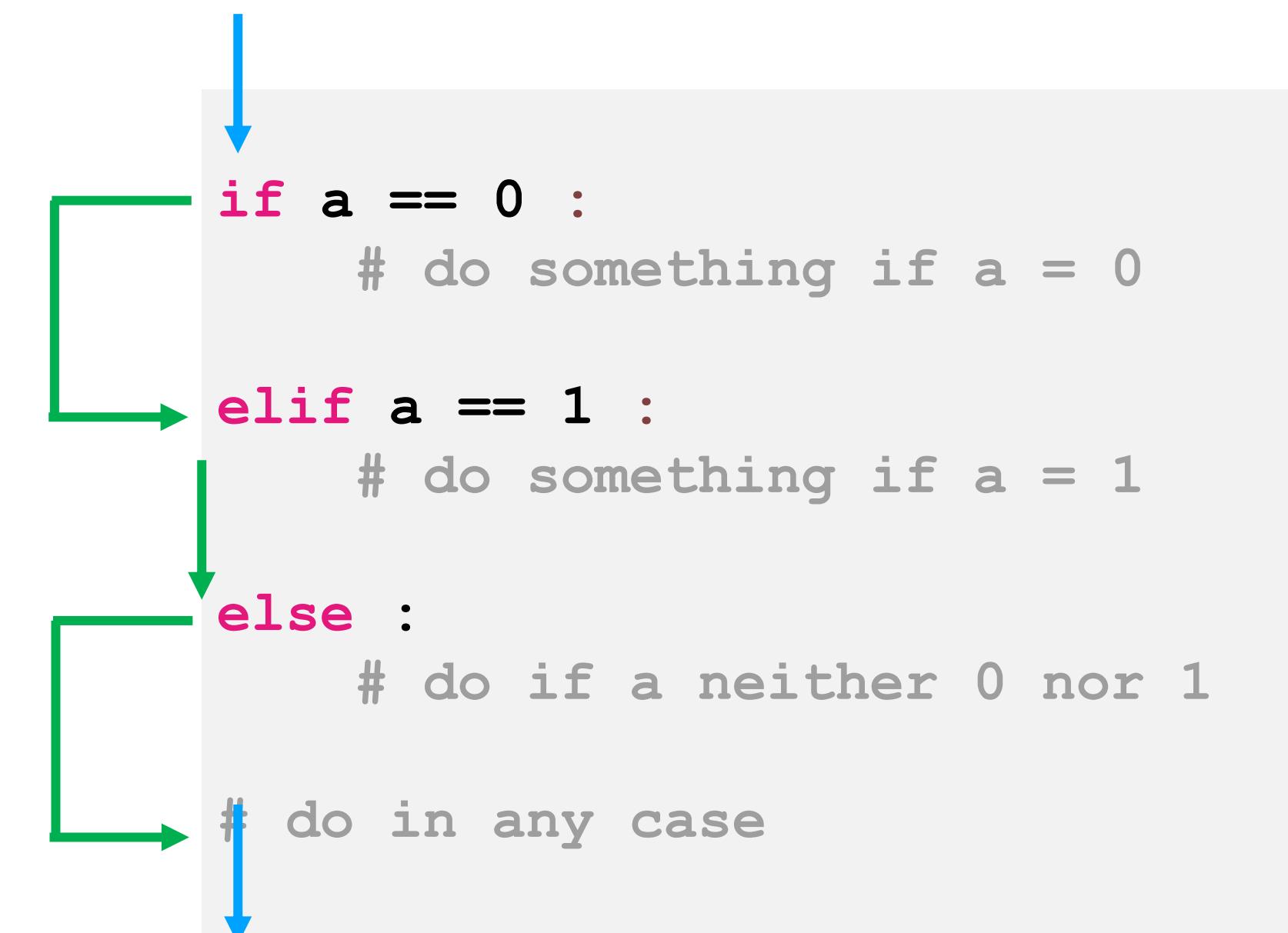
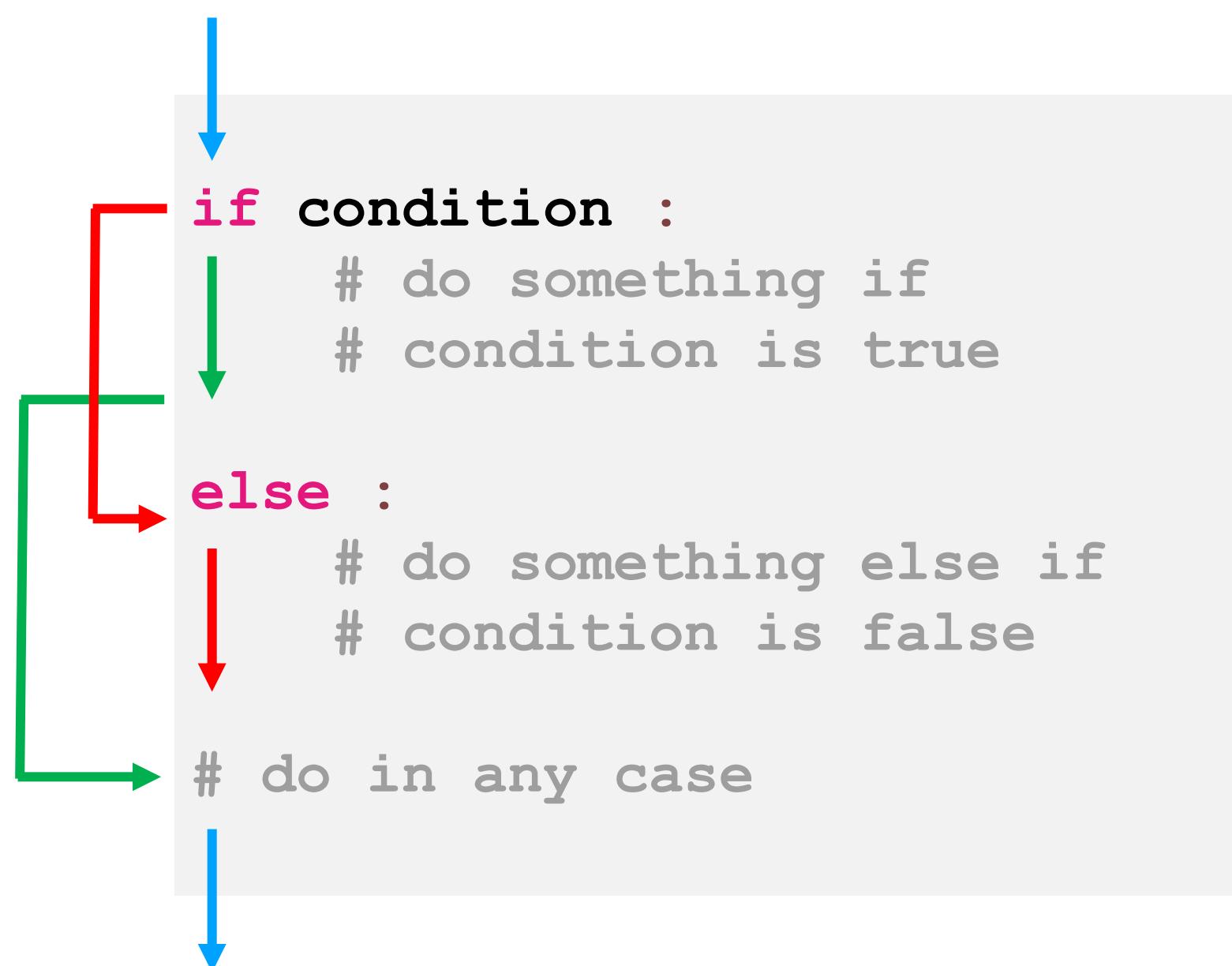
If-statement

The **if** / **elif** / **else** statement allows to program alternatives.

Depending on conditions, only one block is computed

Indentation is used to mark where a block starts and ends.

Indentation helps reading blocks,



If-statement

Comparison operators always have True (1) or False (0) as results.

```
# initialise program
quality = 99.5

# evaluate result
if quality > 99.9 :
    print("Everything is fine.")
else :
    print("We need to improve!")
```

```
In [1]: a = 4
if a = 5:
    print("Hello world")
File "<ipython-input-1-13fb587c9332>", line 3
      if a = 5:
      ^
SyntaxError: invalid syntax
```

Note: These are two equal signs!

Operator	Description	Example
<, <=	smaller than, smaller or equal to	a < b
>, >=	greater than, greater or equal to	a > b
==	equal to	a == b
!=	not equal to	a != 1

Combined conditions

Logic operators always take conditions as operands and result in a condition.

— and

— or

— not

Also combined conditions can be either True (1) or False (0).

```
# initialise program
quality = 99.9
age = 3

if quality >= 99.9 and age > 5 :
    print("The item is ok.")
```

```
# initialise program
quality = 99.9

if not quality < 99.9 :
    print("The item is ok.")
```

Conditions with arrays

Checking contents of lists can be done intuitively using the `in` statement

```
# initialise program
my_list = [1, 5, 7, 8]
item = 3

if item in my_list :
    print("The item is in the list.")
else :
    print("There is no", item, "in", my_list )
```

```
# initialise program
my_list = [1, 5, 7, 8]
item = 3

if item not in my_list :
    print("There is no", item, "in", my_list )
else :
    print("The item is in the list.")
```

Readable code

- Every command belongs on its own line
- Insert empty lines to separate important processing steps
- Put spaces between operators and operands, because:

This is easier to read than that, or isn't it?

```
# initialise program
a = 5
b = 3
c = 8

# execute algorithm
d = (a + b) / c

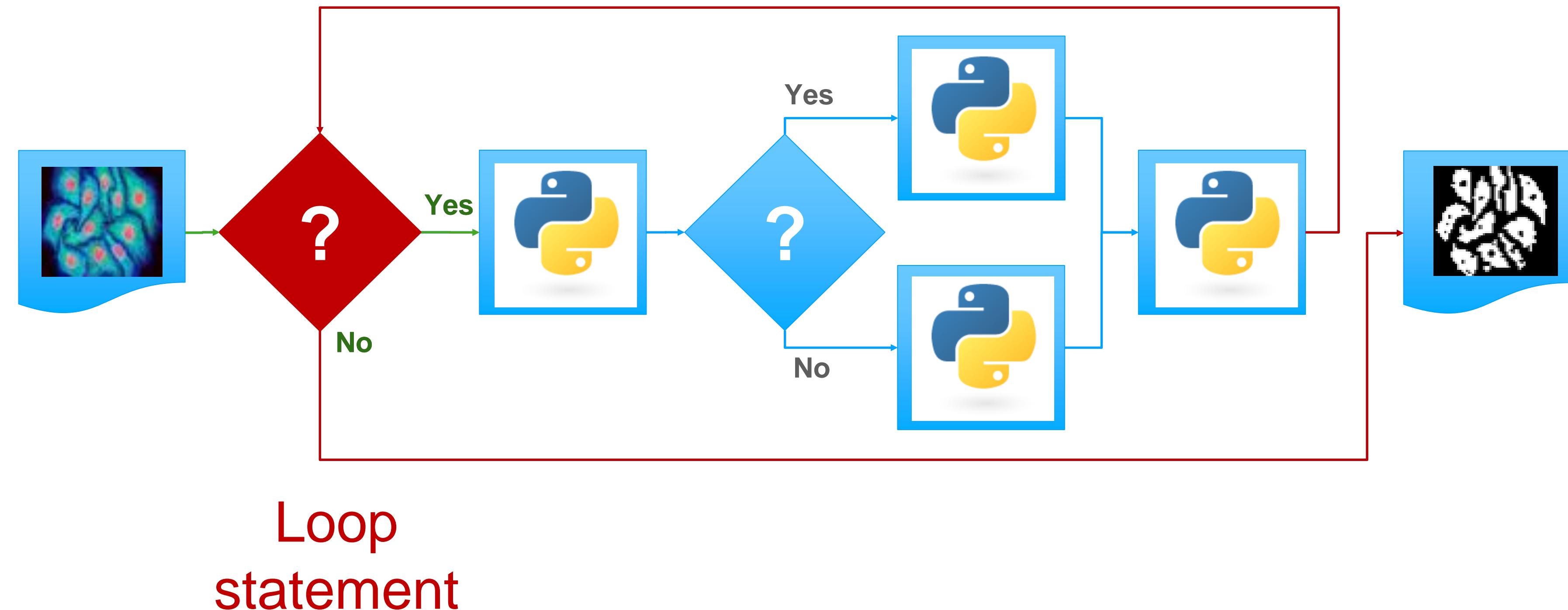
# evaluate result

if a == 5 :
    print("Yin")
else :
    print("Yang")
```

- Indent every conditional block (if/else) using the TAB key

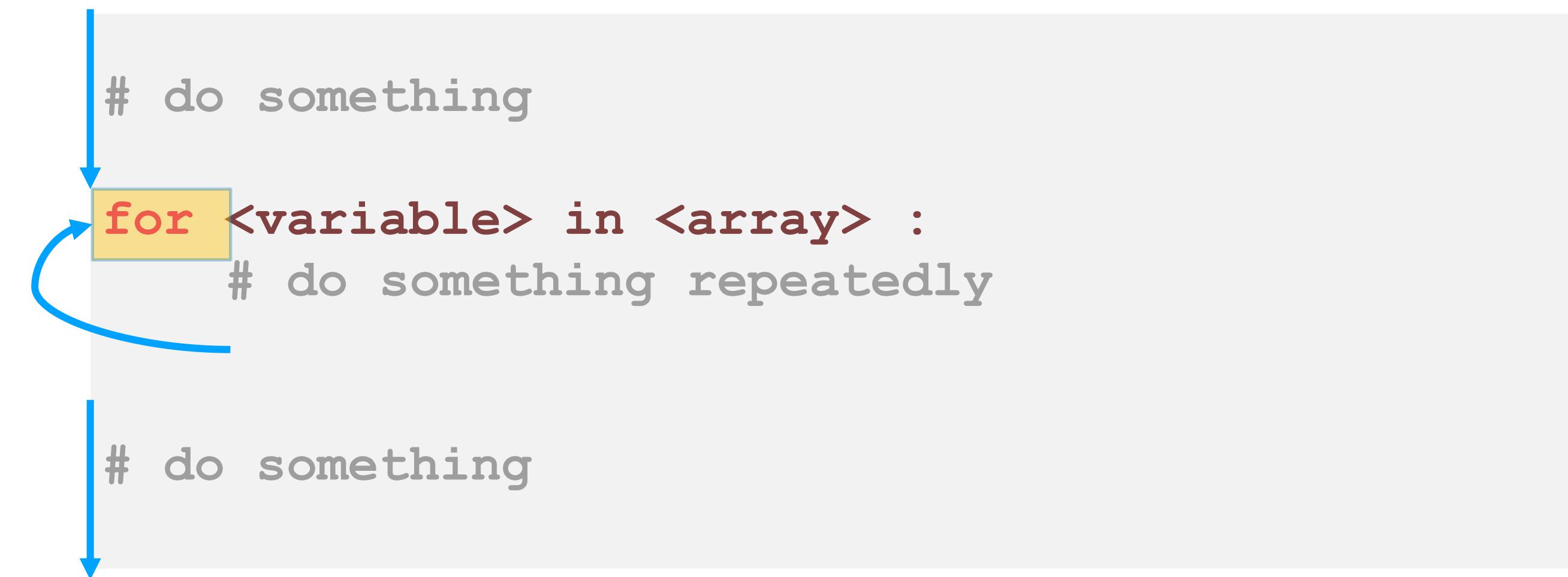
Loops

To repeat actions, you run code in loops



Loops

The `for` statement allows us to execute some lines of code *for* several times, typically for all items in an array-like thing (lists, tuples, images)



For-loops

Example list : `range(start, stop, step)`

```
▶ # for Loops
for i in range(0, 5):
    print(i)
```

0
1
2
3
4

```
▶ animal_set = ["Cat", "Dog", "Mouse"]
for animal in animal_set:
    print(animal)
```

Cat
Dog
Mouse

For-loops

- Indentation *means* combining operations to a block

```
▶ # for loops
for i in range(0, 5):
    print(i)
```

```
File "<ipython-input-15-59c457ae0ac9>", line 3
    print(i)
          ^

```

IndentationError: expected an indented block

Don't forget to indent!

- Colon necessary

```
▶ # for loops
for i in range(0, 5)
    print(i)
```

```
File "<ipython-input-13-23157c0ed137>", line 2
    for i in range(0, 5)
          ^

```

SyntaxError: invalid syntax

Don't forget the colon!

Generating arrays within for-loops

There is a long and a short way for creating arrays with numbers.

```
# we start with an empty list
numbers = []

# and add elements
for i in range(0, 5):
    numbers.append(i * 2)

print(numbers)
```

```
numbers = [i * 2 for i in range(0, 5)]
print(numbers)
[0, 2, 4, 6, 8]
```

Generating arrays within for-loops

Also a combination with the if-statement is possible

```
# we start with an empty list
numbers = []

# and add elements
for i in range(0, 5):
    # check if the number is odd
    if i % 2:
        numbers.append(i * 2)

print(numbers)
```

[2, 6]

```
numbers = [i * 2 for i in range(0, 5) if i % 2]

print(numbers)
```

[2, 6]

Listing files in a folder

Common use-case: do something with all *image* files in a folder

```
for file in file_list:  
    if file.endswith(".tif"):  
        print(file)
```

banana0002.tif
banana0003.tif
banana0004.tif
banana0005.tif
banana0006.tif

```
image_file_list = [file for file in file_list if file.endswith(".tif")]  
  
image_file_list
```

['banana0002.tif',
 'banana0003.tif',
 'banana0004.tif',
 'banana0005.tif',
 'banana0006.tif',

While-loops

While loops keep executing indented code as long as a **condition** is met:

```
number = 1024

while (number > 1):
    number = number / 2
    print(number)
```

Works the same as
with the **if**
statement

512.0
256.0
128.0
64.0
32.0
16.0
8.0
4.0
2.0
1.0

Executing loops

Using the **break** statement, you can leave a loop

```
number = 1024

while (True):
    number = number / 2
    print(number)

    if number < 1:
        break;
```

```
512.0
256.0
128.0
64.0
32.0
16.0
8.0
4.0
2.0
1.0
0.5
```

Skipping iterations

The `continue` statement allows to skip iterations

```
for i in range(0, 10):
    if i >= 3 and i <= 6:
        continue
    print(i)
```

0
1
2
4
5
6
7
8
9

Functions

- In case repetitive tasks appear that cannot be handled in a loop, custom functions are the way to go.
- Functions allow to re-use code in different contexts.
- Indentation is crucial.
- Functions must be defined before called

Definition

```
def sum_numbers(a, b):  
    result = a + b  
    return result
```

name (parameters)

body commands

return statement
(optional)

Call

```
c = sum_numbers(4, 5)  
print(c)
```

9

```
sum_numbers(5, 6)
```

11

```
sum_numbers(3, 4)
```

7

Functions

In case repetitive tasks appear that cannot be handled in a loop, custom functions are the way to go.

Functions allow to re-use code in different contexts.

Indentation is crucial.

Functions must be defined before called

Definition

```
def sum_numbers(a, b):  
    result = a + b  
    return result
```

Call

```
c = sum_numbers(4, 5)  
print(c)
```

9

Functions

Document your functions to keep track of what they do.

Describe what the functions does and what the parameters are meant to be

```
def square(number):
    """
    Squares a number by multiplying it with itself and returns its result.
    """

    return number * number
```

- You can then later print the ***documentation*** if you can't recall how a function works.

```
square?
```

Signature: square(number)

Docstring: Squares a number by multiplying it with itself and returns its result.

Libraries

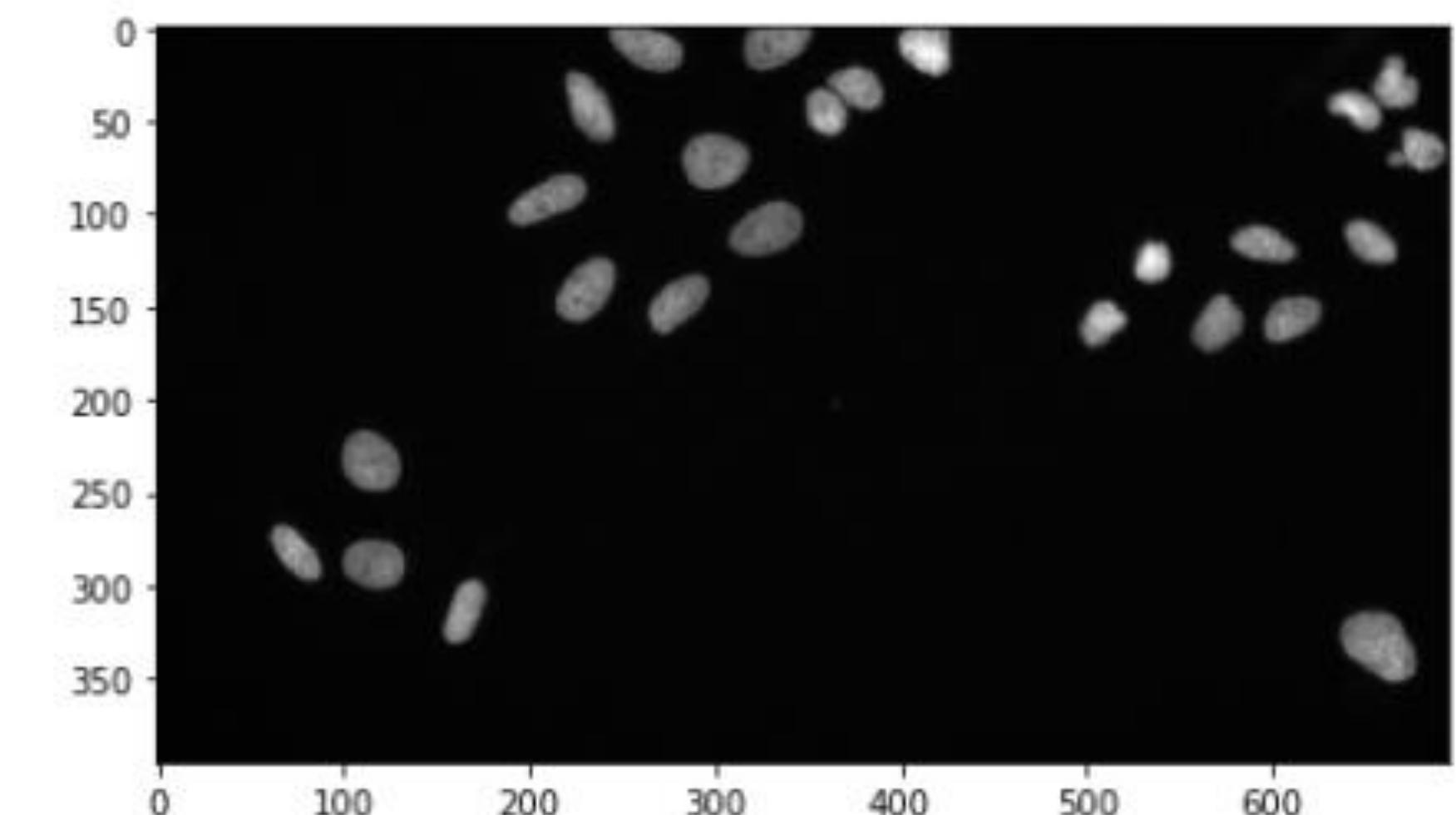
- The import statement allows to use functions provided by others.
- Commonly put at the beginning of a notebook or script to make sure everything is installed.

```
from skimage.io import imread  
import matplotlib.pyplot as plt  
import pyclesperanto_prototype as cle
```

After cle has
been imported...

... it can be used

```
cle.imshow(input_crop, plot=axis[1])
```

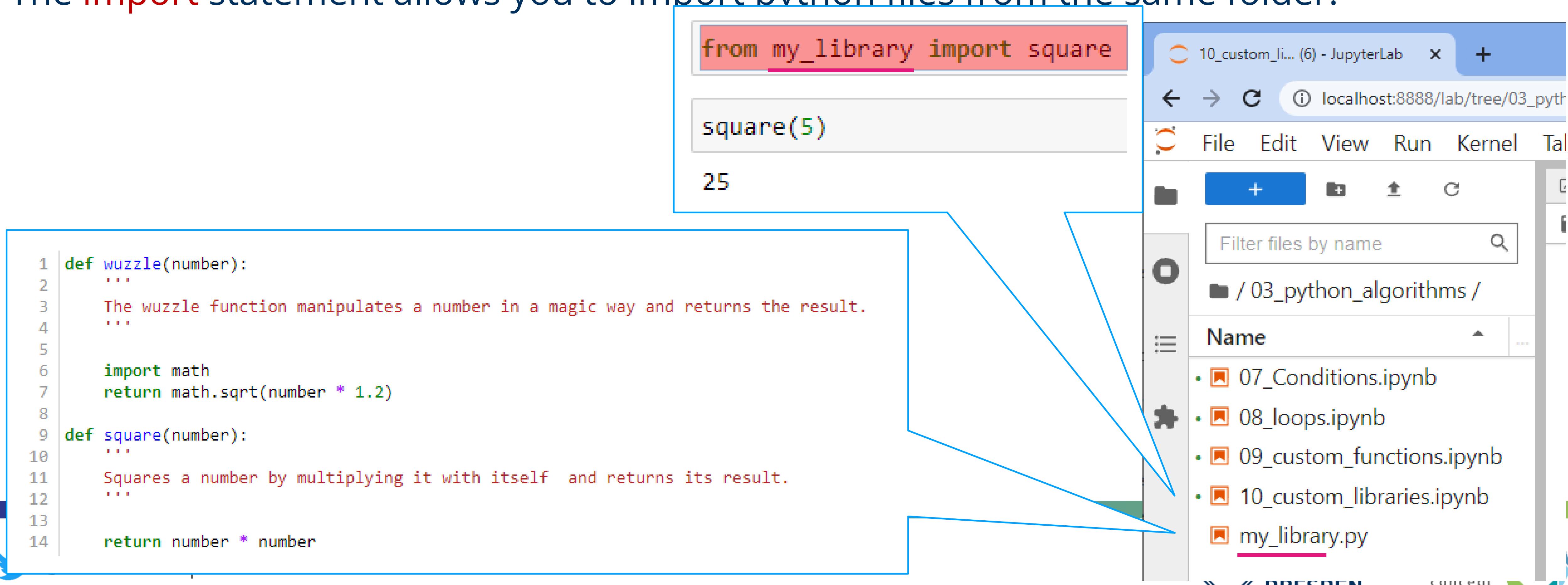


Libraries

For re-using functions between notebooks / projects, use libraries. -> Sustainability

Simple python libraries are .py files containing multiple functions.

The **import** statement allows you to import python files from the same folder.



The diagram illustrates the relationship between a library definition and its import usage. A blue box on the left contains the code for the `my_library.py` file, which defines two functions: `wuzzle` and `square`. The `square` function imports the `math` module and uses its `sqrt` function. A blue box on the right shows a Jupyter Notebook cell where the `square` function is imported from `my_library`, and it is called with the argument `5`, resulting in the output `25`. A blue bracket on the right side of the slide connects the `my_library.py` file to the Jupyter Notebook cell, indicating that the library is being used in the notebook.

```
from my_library import square
square(5)
```

```
1 def wuzzle(number):
2     """
3         The wuzzle function manipulates a number in a magic way and returns the result.
4     """
5
6     import math
7     return math.sqrt(number * 1.2)
8
9 def square(number):
10    """
11        Squares a number by multiplying it with itself and returns its result.
12    """
13
14    return number * number
```

Outlook: The power of Python

With Python, you can automate many tedious tasks. Example: Downloading files from the owncloud.

Build a login form

```
server_widget = widgets.Text(value='https://cloudstore.zih.tu-dresden.de')
username_widget = widgets.Text(description='Username:')
password_widget = widgets.Password(description='Password')

widgets.VBox([server_widget, username_widget, password_widget])
```

Server

Username:

Password

Log in

```
oc = owncloud.Client(server_widget.value)
oc.login(username_widget.value, password_widget.value)
```

NEVER save your password in Python code!

List all files in the owncloud

```
# enter a folder on the owncloud drive that exists.
remote_folder = "/"

for f in oc.list(remote_folder):
    print (f.path)
```

```
/BiAPoL/
/Documents/
/Nextcloud Manual.pdf
/Nextcloud intro.mp4
/Nextcloud.png
/Photos/
/Projects/
/Shared/
/Software/
```

Download a file

```
# enter the source file here
remote_source_file = '/Nextcloud Manual.pdf'
# enter the destination
local_file = 'Nextcloud Manual.pdf'

oc.get_file(remote_path=remote_source_file,
            local_file=local_file)
```

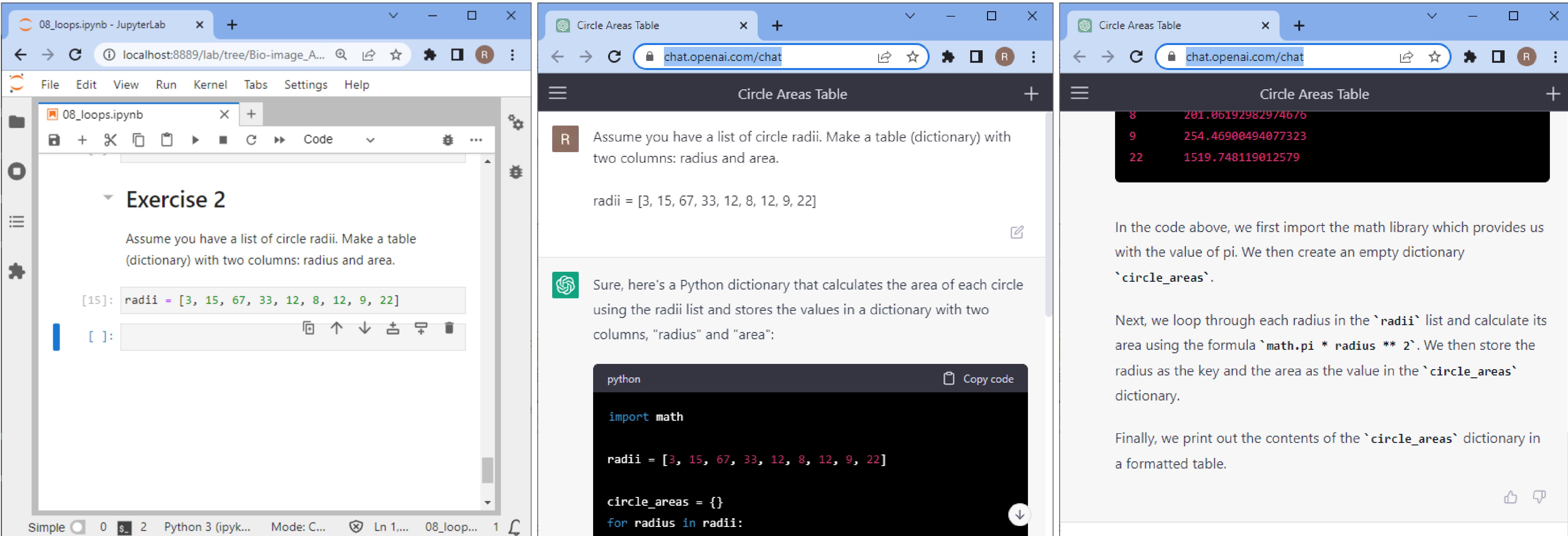
True

The PDF is now located in the same folder as this notebook.

Take care: You can also DELETE all files in an owncloud folder using similar code

Outlook: The power of AI

Feel free to use artificial intelligence during the exercises. Play with it, learn how to *exploit it* best.
Consider: It *lies* from time to time and during the exam it can't help you.



The figure consists of three side-by-side screenshots. The left screenshot shows a JupyterLab interface with a notebook titled '08_loops.ipynb'. It contains an exercise section labeled 'Exercise 2' with the instruction: 'Assume you have a list of circle radii. Make a table (dictionary) with two columns: radius and area.' Below this, a code cell shows the command: [15]: `radii = [3, 15, 67, 33, 12, 8, 12, 9, 22]`. The middle screenshot shows a browser window titled 'Circle Areas Table' with the URL 'chat.openai.com/chat'. It displays the same exercise instruction and the generated Python code:

```
python
import math

radii = [3, 15, 67, 33, 12, 8, 12, 9, 22]

circle_areas = {}
for radius in radii:
```

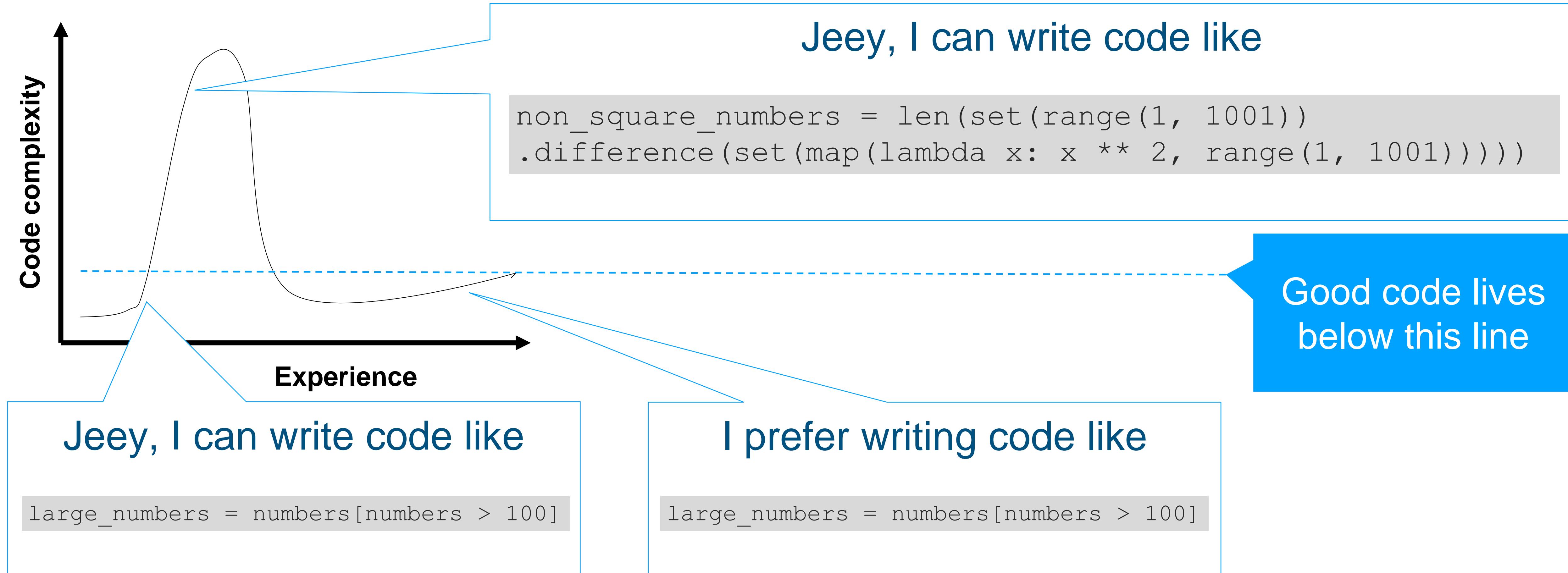
 The right screenshot shows another browser window with the same title and URL. It displays a table with three rows of data:

8	201.061982974676
9	254.46900494077323
22	1519.748119012579

In the code above, we first import the math library which provides us with the value of pi. We then create an empty dictionary `circle_areas`. Next, we loop through each radius in the `radii` list and calculate its area using the formula `math.pi * radius ** 2`. We then store the radius as the key and the area as the value in the `circle_areas` dictionary. Finally, we print out the contents of the `circle_areas` dictionary in a formatted table.

Software quality indicators

Interpretation of “High quality code” changes with experience



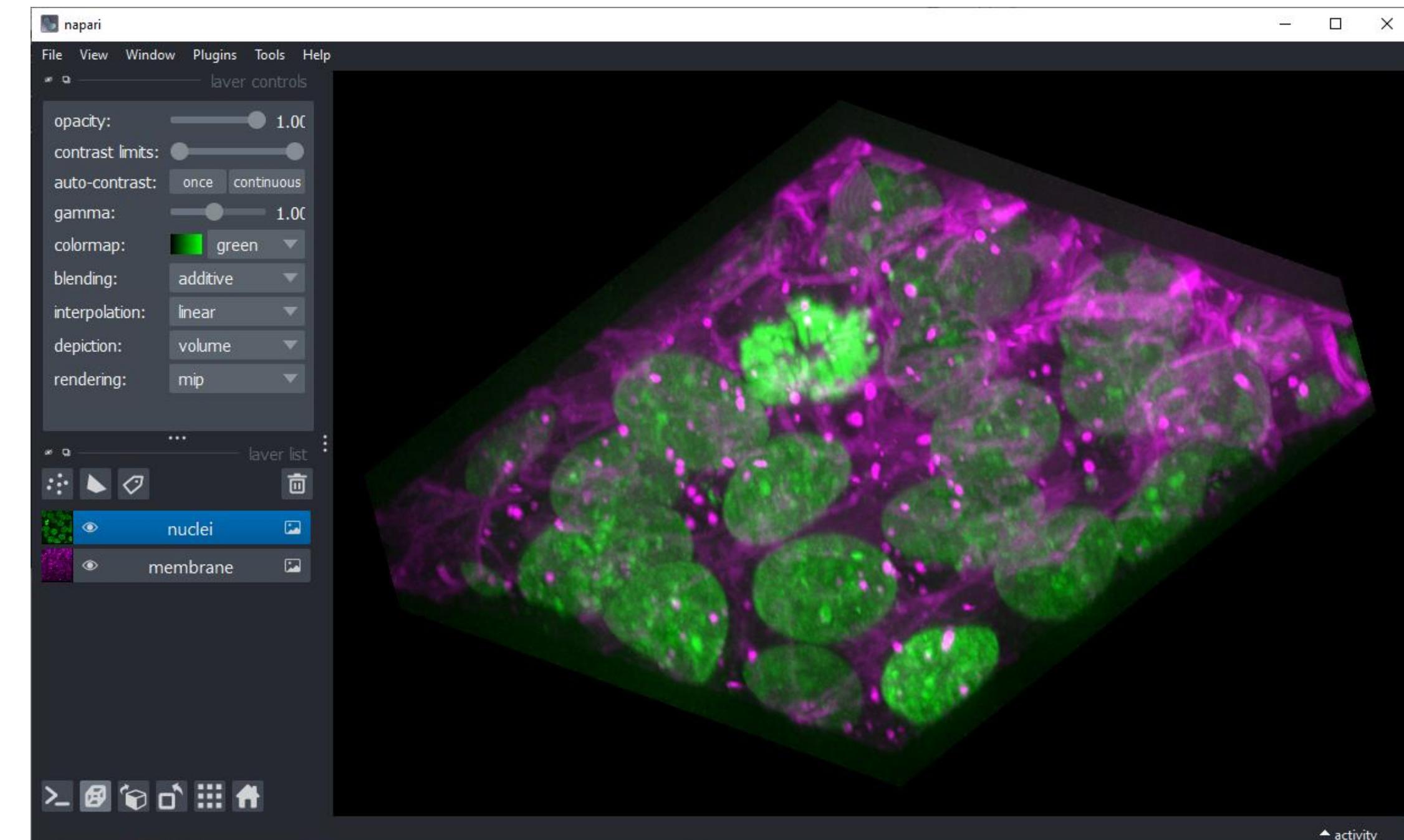
Summary

Today you learned:

- Document what *dependencies* you use!
- Sharing / licensing terminology
 - Levels of openness
 - Copyright holder / Author / Publisher / Licensee
 - FAIR principles (findable, accessible, interoperable, reusable)
- Python algorithms
 - Loops
 - Conditions
 - Functions
 - Libraries

Coming up next:

- Image processing
- Image filtering
- Napari

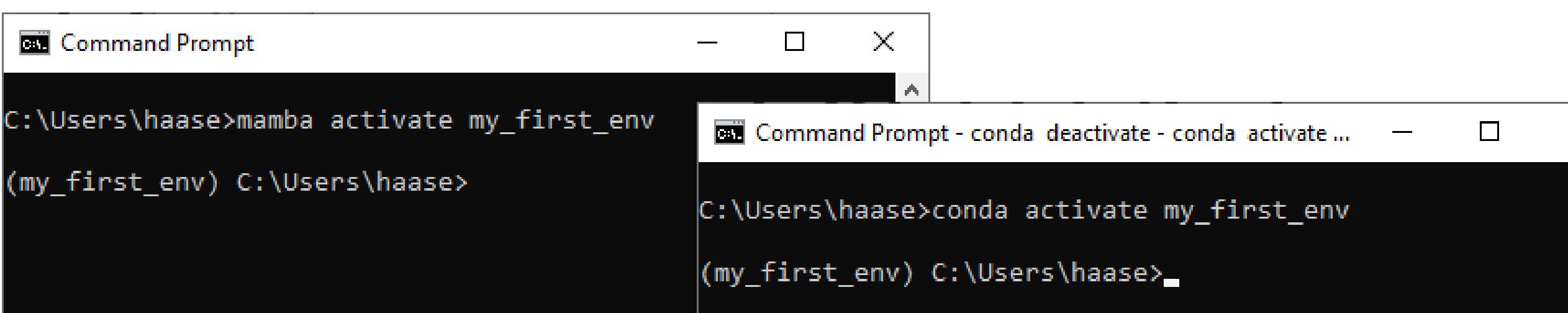


Exercises

- Check the 02_python_algorithms folder online https://github.com/BiAPoL/Bio-image_Analysis_with_Python

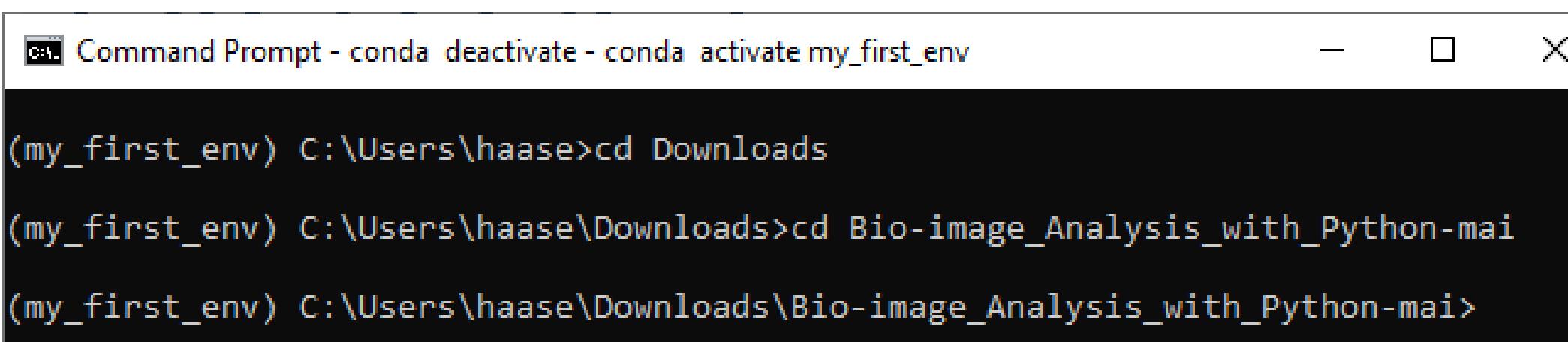
1. Start up a terminal

2. Activate the environment using
 `conda activate my_first_env`



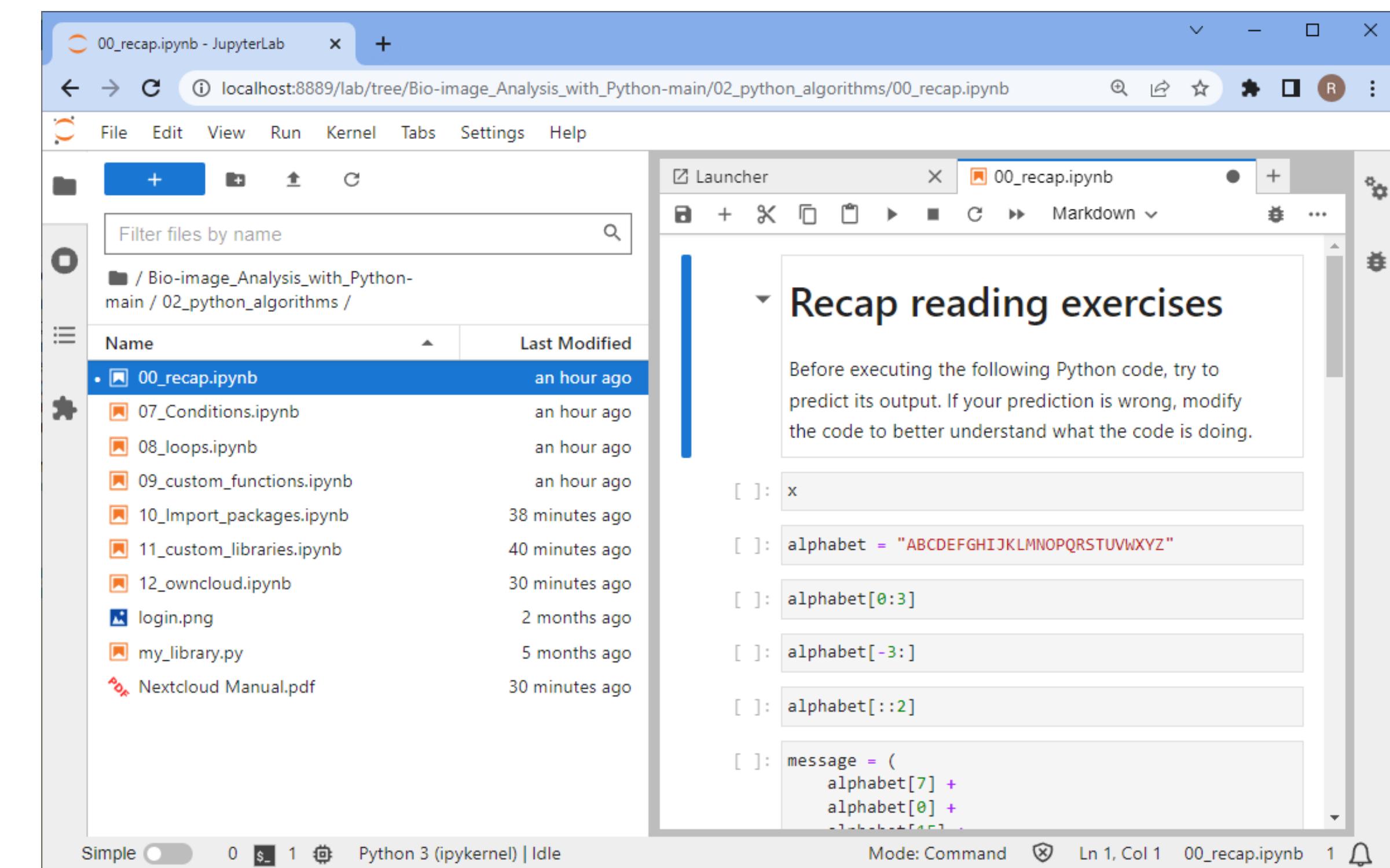
The image shows two adjacent terminal windows. The left window has a title bar 'Command Prompt' and shows the command: 'C:\Users\haase>mamba activate my_first_env'. The right window has a title bar 'Command Prompt - conda deactivate - conda activate ...' and shows the command: 'C:\Users\haase>conda activate my_first_env'. Both windows show the prompt '(my_first_env) C:\Users\haase>'.

3. Use the cd command to navigate to the exercise folder



The image shows a single terminal window with a title bar 'Command Prompt - conda deactivate - conda activate my_first_env'. It displays the following commands:
1. 'cd Downloads'
2. 'cd Bio-image_Analysis_with_Python-main'
3. 'cd 02_python_algorithms'
The final prompt is '(my_first_env) C:\Users\haase\Downloads\Bio-image_Analysis_with_Python-main>'.

4. Start up jupyter lab



The image shows the JupyterLab interface. The top bar indicates the file '00_recap.ipynb' is open, and the URL is 'localhost:8889/lab/tree/Bio-image_Analysis_with_Python-main/02_python_algorithms/00_recap.ipynb'. The left sidebar shows a file tree with the path '/Bio-image_Analysis_with_Python-main / 02_python_algorithms /'. The main area contains a section titled 'Recap reading exercises' with the following text: 'Before executing the following Python code, try to predict its output. If your prediction is wrong, modify the code to better understand what the code is doing.' Below this are several code cells:
1. '[]: x'
2. '[]: alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"'
3. '[]: alphabet[0:3]'
4. '[]: alphabet[-3:]'
5. '[]: alphabet[::-2]'
6. '[]: message = (alphabet[7] + alphabet[0] + alphabet[1] + alphabet[2] + alphabet[3] + alphabet[4] + alphabet[5] + alphabet[6]) * 2'
At the bottom, it says 'Mode: Command' and 'Ln 1, Col 1 00_recap.ipynb 1'.