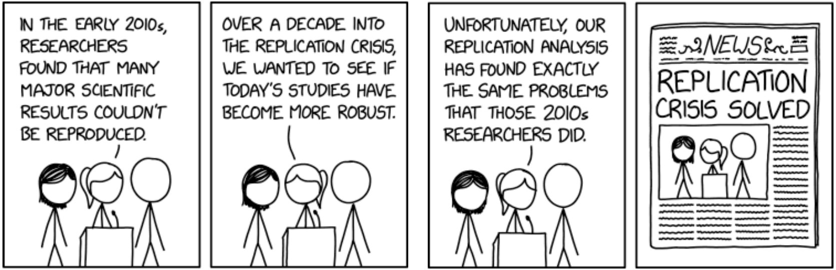
Practical tools for reproducible experimental analyses

Charlie J. Paxman (GANIL)

EURO-LABS ATS: Collaborative Software, Feb 2026



xkcd #3117

What is Reproducibility?

What is Reproducibility?

- Not a trivial question! Different people define reproducibility differently

What is Reproducibility?

- Not a trivial question! Different people define reproducibility differently
    - Same/different data?
    - Same/different set-up?
    - Same/different team?

What is Reproducibility?

- Not a trivial question! Different people define reproducibility differently
  - Same/different data?
  - Same/different set-up?
  - Same/different team?

L. A. Barba (2018)[1]

Repeatable	Same team, same setup
Replicable	Diff. team, same setup
Reproducible	Diff. team, diff. setup

---

[1] *Terminologies for Reproducible Research*, arXiv, `https://doi.org/10.48550/arXiv.1802.03311`

What is Reproducibility?

- Not a trivial question! Different people define reproducibility differently
  - Same/different data?
  - Same/different set-up?
  - Same/different team?

X. Chen *et al.* (2019)[2]

Repeat	Same team, setup, lab
Reproduce	Diff. team, setup, lab
Rerun	Variation, same lab
Replicate	Same setup, different lab
Reuse	New independent discovery

---

[2] *Open is not enough*, Nature Phys. **15**, 113–119 (2019). `https://doi.org/10.1038/s41567-018-0342-2`

What is Reproducibility?

- Not a trivial question! Different people define reproducibility differently
  - Same/different data?
  - Same/different set-up?
  - Same/different team?
- Driven largely by biomedical science

X. Chen *et al.* (2019)[2]

Repeat	Same team, setup, lab
Reproduce	Diff. team, setup, lab
Rerun	Variation, same lab
Replicate	Same setup, different lab
Reuse	New independent discovery

---

[2] *Open is not enough*, Nature Phys. **15**, 113–119 (2019). `https://doi.org/10.1038/s41567-018-0342-2`

What is Reproducibility?

- Not a trivial question! Different people define reproducibility differently
  - Same/different data?
  - Same/different set-up?
  - Same/different team?
- Driven largely by biomedical science
- Many of these definitions not suitable for accelerator-based science!

X. Chen *et al.* (2019)[2]

Repeat	Same team, setup, lab
Reproduce	Diff. team, setup, lab
Rerun	Variation, same lab
Replicate	Same setup, different lab
Reuse	New independent discovery

---

[2] *Open is not enough*, Nature Phys. **15**, 113–119 (2019). `https://doi.org/10.1038/s41567-018-0342-2`

What is Reproducibility?

- Not a trivial question! Different people define reproducibility differently
  - Same/different data?
  - Same/different set-up?
  - Same/different team?
- Driven largely by biomedical science
- Many of these definitions not suitable for accelerator-based science!
  - Cannot "transpose" setup
  - Beam time is competitive
  - Experiments are expensive

X. Chen *et al.* (2019)[2]	
Repeat	Same team, setup, lab
Reproduce	Diff. team, setup, lab
Rerun	Variation, same lab
Replicate	Same setup, different lab
Reuse	New independent discovery

---

[2] *Open is not enough*, Nature Phys. **15**, 113–119 (2019). `https://doi.org/10.1038/s41567-018-0342-2`

What is Reproducibility?

- Not a trivial question! Different people define reproducibility differently
  - Same/different data?
  - Same/different set-up?
  - Same/different team?
- Driven largely by biomedical science
- Many of these definitions not suitable for accelerator-based science!
  - Cannot "transpose" setup
  - Beam time is competitive
  - Experiments are expensive
- What we're talking about...

‘The Turing Way’[2]	
Reproducible	Same analysis, same data
Replicable	Same analysis, diff data
Robust	Diff analysis, same data
Generalisable	Diff analysis, diff data

Finally ran out of "R" words...

---

[2] `https://book.the-turing-way.org/reproducible-research/overview/overview-definitions/` (Accessed 28/01/2026)

What is Reproducibility?

For our purposes, we care about **computational reproducibility**:

Computational Reproducibility

"[computational] reproducibility is obtaining consistent results using the **same input data; computational steps, methods, and code;** and conditions of analysis." [3]

- Can be performed by a different person, different machine, at a different time

⇒ Without relying on personal knowledge of the process!

- Collaboration: share your analysis with a colleague without version conflicts
- Future-proofing: store your analysis for yourself in 1 month, 6 months, 5 years...

---

[3] Nation. Acad. Sci. Eng. Med. (2019). *Reproducibility and Replicability in Science.* `https://doi.org/10.17226/3531.`

Why bother with reproducible analyses?

Every physics researcher is an amateur software engineer → mistakes are inevitable!

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

- Not a criticism, however, becomes a problem if code is private, bespoke & unique

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

- Not a criticism, however, becomes a problem if code is private, bespoke & unique
- Shared code is stress-tested! More people to spot mistakes

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

- Not a criticism, however, becomes a problem if code is private, bespoke & unique
- Shared code is stress-tested! More people to spot mistakes

Experiments are getting more and more complex, analysis time is stretching

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

- Not a criticism, however, becomes a problem if code is private, bespoke & unique
- Shared code is stress-tested! More people to spot mistakes

Experiments are getting more and more complex, analysis time is stretching

- Multiple heterogenous detector systems coupled together, 100s-1000s of channels...

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

- Not a criticism, however, becomes a problem if code is private, bespoke & unique
- Shared code is stress-tested! More people to spot mistakes

Experiments are getting more and more complex, analysis time is stretching

- Multiple heterogenous detector systems coupled together, 100s-1000s of channels...
- Students and postdocs move on, loss of institutional memory of analysis

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

- Not a criticism, however, becomes a problem if code is private, bespoke & unique
- Shared code is stress-tested! More people to spot mistakes

Experiments are getting more and more complex, analysis time is stretching

- Multiple heterogenous detector systems coupled together, 100s-1000s of channels...
- Students and postdocs move on, loss of institutional memory of analysis
- Difficult to verify or extend past results, potential science trapped in inaccessible files.

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

- Not a criticism, however, becomes a problem if code is private, bespoke & unique
- Shared code is stress-tested! More people to spot mistakes

Experiments are getting more and more complex, analysis time is stretching

- Multiple heterogenous detector systems coupled together, 100s-1000s of channels...
- Students and postdocs move on, loss of institutional memory of analysis
- Difficult to verify or extend past results, potential science trapped in inaccessible files.

Simplest (and most convincing?) reason: **laptops break!**

Why bother with reproducible analyses?

Every physics researcher is an **amateur** software engineer → mistakes are inevitable!

- Not a criticism, however, becomes a problem if code is private, bespoke & unique
- Shared code is stress-tested! More people to spot mistakes

Experiments are getting more and more complex, analysis time is stretching

- Multiple heterogenous detector systems coupled together, 100s-1000s of channels...
- Students and postdocs move on, loss of institutional memory of analysis
- Difficult to verify or extend past results, potential science trapped in inaccessible files.

Simplest (and most convincing?) reason: **laptops break!**

If an analysis is self-contained and reproducible, then it can be easily passed from person to person and machine to machine

Why bother with reproducible analyses?

Minor programming mistakes

*"Analysis of a 2019 randomised controlled trial showed a treatment benefit for patients with chronic obstructive pulmonary disease (COPD). The intervention was a three-month self-management program delivered by COPD-specialised nurses; the control group was usual care. ... **Ten months later the paper was retracted.** The authors had detected a programming mistake in the analysis of the data set. The group assignment (treatment versus control) was **coded in the format of "1, 2" and was converted to a binary type of "0, 1"**, but the assignment was made incorrectly and flipped the two study groups. Thus, **the correct result reversed the main finding.**"* [4]

---

[4] S. Schwab and L. Held, *Statistical Programming: Small Mistakes, Big Impacts*, Significance **18** 3 (2021). `https://doi.org/10.1111/1740-9713.01522`

Why bother with reproducible analyses?

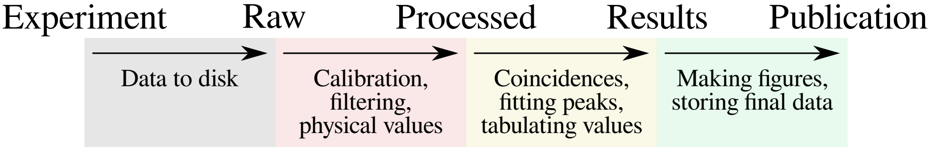
Variance with software and OS version

"Using a set of ***30 anatomical T1-weighted 3T MRI scans****, we investigated the effects of data processing variables such as* ***FreeSurfer version*** *(v4.3.1, v4.5.0, and v5.0.0), workstation (Macintosh and Hewlett-Packard), and Macintosh operating system version* ***(OSX 10.5 and OSX 10.6)****. Significant differences were revealed between FreeSurfer version v5.0.0 and the two earlier versions.* ***These differences were on average 8.8 ± 6.6% (range 1.3-64.0%) (volume) and 2.8 ± 1.3% (1.1-7.7%) (cortical thickness).*** *About a factor two smaller differences were detected between Macintosh and Hewlett-Packard workstations and between OSX 10.5 and OSX 10.6.* ***The observed differences are similar in magnitude as effect sizes reported in accuracy evaluations and neurodegenerative studies.****"* [5]

---

[5] E. H. B. M. Gronenschild *et al.*, *The Effects of FreeSurfer Version, Workstation Type, and Macintosh Operating System Version on Anatomical Volume and Cortical Thickness Measurements*, PLOS ONE **7** 6 (2012). `https://doi.org/10.1371/journal.pone.0038234`

Which stages of the research process?



Experiment Raw Processed Results Publication

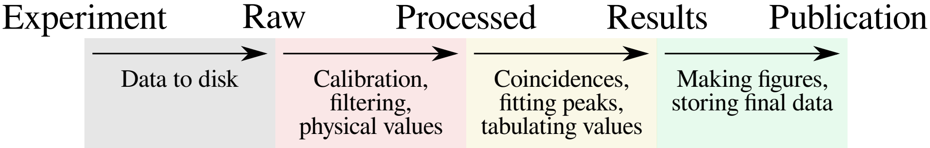
Data to disk

Calibration, filtering, physical values

Coincidences, fitting peaks, tabulating values

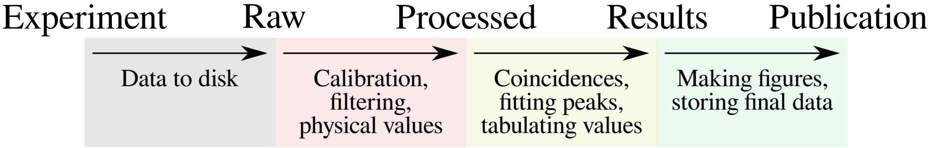
Making figures, storing final data

Which stages of the research process?



- Ideally, reproducibility should be in mind at every stage

Which stages of the research process?



Experiment → Raw → Processed → Results → Publication

Data to disk

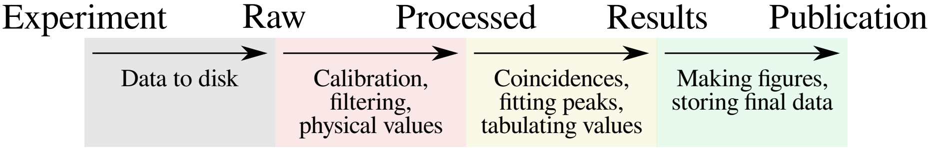
Calibration,
filtering,
physical values

Coincidences,
fitting peaks,
tabulating values

Making figures,
storing final data

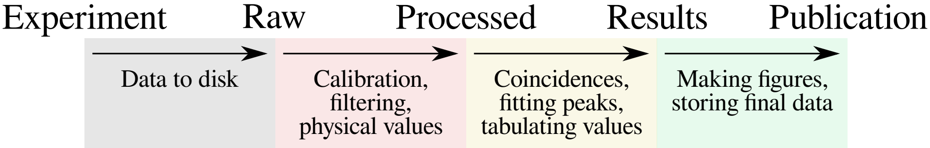
- Ideally, reproducibility should be in mind at every stage
- In practice, first step is often out of your control (except for the experimental logbook!)

Which stages of the research process?



- Ideally, reproducibility should be in mind at every stage
- In practice, first step is often out of your control (except for the experimental logbook!)
- However, the rest of the process is up to you!

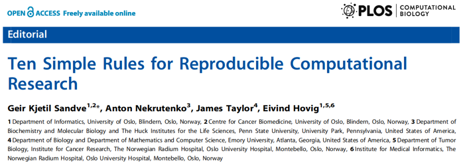
Which stages of the research process?



Experiment → Raw → Processed → Results → Publication

- Experiment → Raw: Data to disk
- Raw → Processed: Calibration, filtering, physical values
- Processed → Results: Coincidences, fitting peaks, tabulating values
- Results → Publication: Making figures, storing final data

- Ideally, reproducibility should be in mind at every stage
- In practice, first step is often out of your control (except for the experimental logbook!)
- However, the rest of the process is up to you!
- But how? Again, a few of different "5 simple steps..."



OPEN ACCESS Freely available online

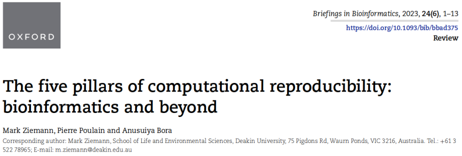
PLOS | COMPUTATIONAL BIOLOGY

Editorial

Ten Simple Rules for Reproducible Computational Research

Geir Kjetil Sandve[1,2*], Anton Nekrutenko[3], James Taylor[4], Eivind Hovig[1,5,6]

1 Department of Informatics, University of Oslo, Blindern, Oslo, Norway, **2** Centre for Cancer Biomedicine, University of Oslo, Blindern, Oslo, Norway, **3** Department of Biochemistry and Molecular Biology and The Huck Institutes for the Life Sciences, Penn State University, University Park, Pennsylvania, United States of America, **4** Department of Biology and Department of Mathematics and Computer Science, Emory University, Atlanta, Georgia, United States of America, **5** Department of Tumor Biology, Institute for Cancer Research, The Norwegian Radium Hospital, Oslo University Hospital, Montebello, Oslo, Norway, **6** Institute for Medical Informatics, The Norwegian Radium Hospital, Oslo University Hospital, Montebello, Oslo, Norway



OXFORD

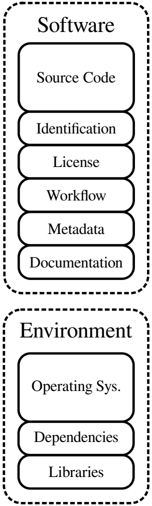
Briefings in Bioinformatics, 2023, **24(6)**, 1–13
https://doi.org/10.1093/bib/bbad375
Review

The five pillars of computational reproducibility: bioinformatics and beyond

Mark Ziemann, Pierre Poulain and Anusuiya Bora

Corresponding author: Mark Ziemann, School of Life and Environmental Sciences, Deakin University, 75 Pigdons Rd, Waurn Ponds, VIC 3216, Australia. Tel.: +61 3 522 78965; E-mail: m.ziemann@deakin.edu.au

The good news – you're already partway there


Software

Source Code

Identification

License

Workflow

Metadata

Documentation

Environment

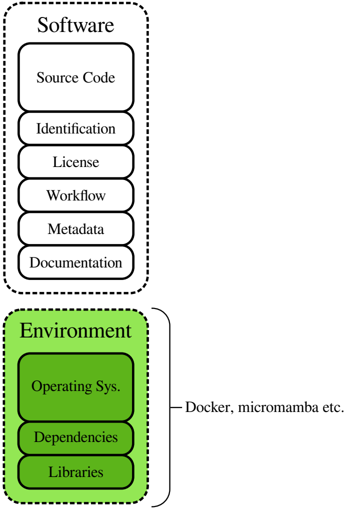
Operating Sys.

Dependencies

Libraries

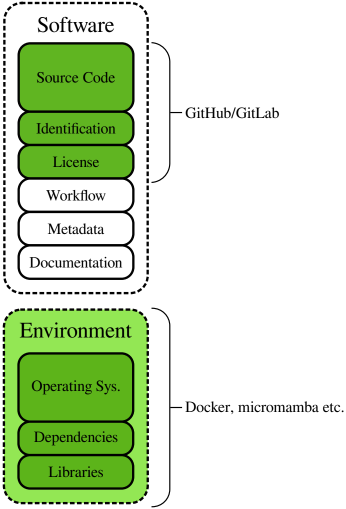
Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

The good news – you're already partway there



Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

The good news – you’re already partway there



Software

Source Code

Identification

License

Workflow

Metadata

Documentation

GitHub/GitLab

Environment

Operating Sys.

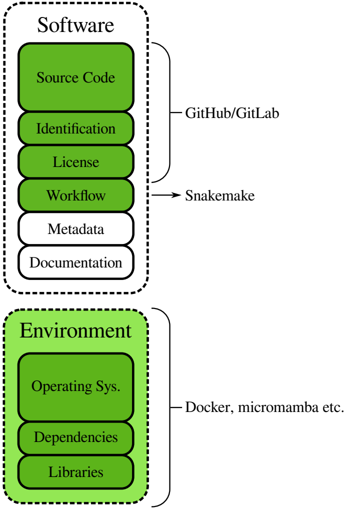
Dependencies

Libraries

Docker, micromamba etc.

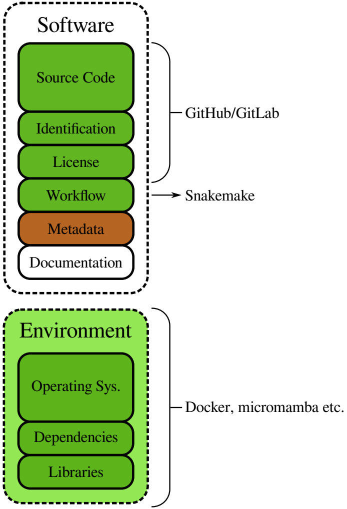
Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

The good news – you’re already partway there



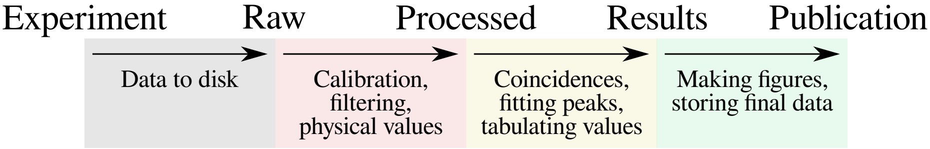
Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

The good news – you're already partway there



Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

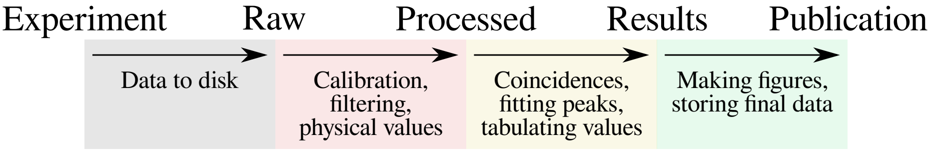
Metadata = "data about data"



Experiment → Raw → Processed → Results → Publication

- Experiment → Raw: Data to disk
- Raw → Processed: Calibration, filtering, physical values
- Processed → Results: Coincidences, fitting peaks, tabulating values
- Results → Publication: Making figures, storing final data

Metadata = "data about data"

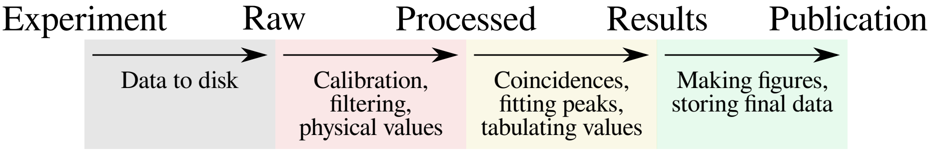


Experiment → Raw → Processed → Results → Publication

- Data to disk
- Calibration, filtering, physical values
- Coincidences, fitting peaks, tabulating values
- Making figures, storing final data

Metadata

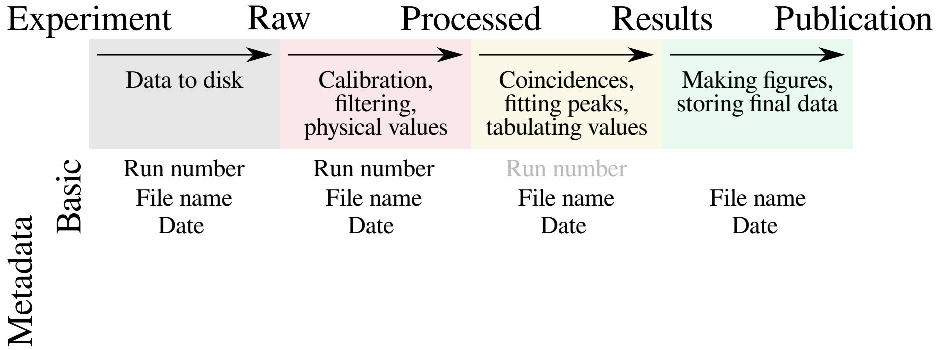
Metadata = "data about data"


Experiment → Raw → Processed → Results → Publication

Metadata	Basic

Data to disk	Calibration, filtering, physical values	Coincidences, fitting peaks, tabulating values	Making figures, storing final data
Run number	Run number	Run number	
File name	File name	File name	File name
Date	Date	Date	Date

Metadata = "data about data"



Experiment → Raw → Processed → Results → Publication

	Data to disk	Calibration, filtering, physical values	Coincidences, fitting peaks, tabulating values	Making figures, storing final data
Metadata: Basic	Run number File name Date	Run number File name Date	Run number File name Date	File name Date

Loads of context missing!

Article | Open access | Published: 14 October 2022

Introducing the FAIR Principles for research software

Michelle Barker, Neil P. Chue Hong, Daniel S. Katz, Anna-Lena Lamprecht, Carlos Martinez-Ortiz, Fotis Psomopoulos, Jennifer Harrow, Leyla Jael Castro, Morane Gruenpeter, Paula Andrea Martinez & Tom Honeyman

Scientific Data **9**, Article number: 622 (2022) | Cite this article

50k Accesses | **308** Citations | **164** Altmetric | Metrics

Findable

Accessible

Interoperable

Reusable

Findable

- F1. Software is assigned a globally unique and persistent identifier.
  - F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
  - F1.2. Different versions of the software are assigned distinct identifiers.
- F2. Software is described with rich metadata.
- F3. Metadata clearly and explicitly include the identifier of the software they describe.
- F4. Metadata are FAIR, searchable and indexable.

Interoperable

- I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
- I2. Software includes qualified references to other objects.

Accessible

- A1. Software is retrievable by its identifier using a standardised communications protocol.
  - A1.1. The protocol is open, free, and universally implementable.
  - A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
- A2. Metadata are accessible, even when the software is no longer available.

Reusable

- R1. Software is described with a plurality of accurate and relevant attributes.
  - R1.1. Software is given a clear and accessible license.
  - R1.2. Software is associated with detailed provenance.
- R2. Software includes qualified references to other software.
- R3. Software meets domain-relevant community standards.

Findable

F1. Software is assigned a globally unique and persistent identifier.
  F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
  F1.2. Different versions of the software are assigned distinct identifiers.
F2. Software is described with rich metadata.
F3. Metadata clearly and explicitly include the identifier of the software they describe.
F4. Metadata are FAIR, searchable and indexable.

Interoperable

I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. Software includes qualified references to other objects.

Accessible

A1. Software is retrievable by its identifier using a standardised communications protocol.
  A1.1. The protocol is open, free, and universally implementable.
  A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. Metadata are accessible, even when the software is no longer available.

Reusable

R1. Software is described with a plurality of accurate and relevant attributes.
  R1.1. Software is given a clear and accessible license.
  R1.2. Software is associated with detailed provenance.
R2. Software includes qualified references to other software.
R3. Software meets domain-relevant community standards.

Findable

F1. Software is assigned a globally unique and persistent identifier.
  F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
  F1.2. Different versions of the software are assigned distinct identifiers.
F2. Software is described with rich metadata.
F3. Metadata clearly and explicitly include the identifier of the software they describe.
F4. Metadata are FAIR, searchable and indexable.

Interoperable

I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. Software includes qualified references to other objects.

GitLab/Hub

Accessible

A1. Software is retrievable by its identifier using a standardised communications protocol.
  A1.1. The protocol is open, free, and universally implementable.
  A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. Metadata are accessible, even when the software is no longer available.

Reusable

R1. Software is described with a plurality of accurate and relevant attributes.
  R1.1. Software is given a clear and accessible license.
  R1.2. Software is associated with detailed provenance.
R2. Software includes qualified references to other software.
R3. Software meets domain-relevant community standards.

Findable

F1. Software is assigned a **globally** unique and persistent identifier.
  F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
  F1.2. Different versions of the software are assigned distinct identifiers.
F2. Software is described with rich metadata.
F3. Metadata clearly and explicitly include the identifier of the software they describe.
F4. Metadata are FAIR, searchable and indexable.

Interoperable

I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. Software includes qualified references to other objects.

GitLab/Hub + Zenodo DOI

Accessible

A1. Software is retrievable by its identifier using a standardised communications protocol.
  A1.1. The protocol is open, free, and universally implementable.
  A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. Metadata are accessible, even when the software is no longer available.

Reusable

R1. Software is described with a plurality of accurate and relevant attributes.
  R1.1. Software is given a clear and accessible license.
  R1.2. Software is associated with detailed provenance.
R2. Software includes qualified references to other software.
R3. Software meets domain-relevant community standards.

Findable

- F1. Software is assigned a globally unique and persistent identifier.
  - F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
  - F1.2. Different versions of the software are assigned distinct identifiers.
- F2. Software is described with rich metadata.
- F3. Metadata clearly and explicitly include the identifier of the software they describe.
- F4. Metadata are FAIR, searchable and indexable.

Interoperable

- I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
- I2. Software includes qualified references to other objects.

GitLab/Hub + Zenodo DOI + e.g. .root

Accessible

- A1. Software is retrievable by its identifier using a standardised communications protocol.
  - A1.1. The protocol is open, free, and universally implementable.
  - A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
- A2. Metadata are accessible, even when the software is no longer available.

Reusable

- R1. Software is described with a plurality of accurate and relevant attributes.
  - R1.1. Software is given a clear and accessible license.
  - R1.2. Software is associated with detailed provenance.
- R2. Software includes qualified references to other software.
- R3. Software meets domain-relevant community standards.

Findable

F1. Software is assigned a globally unique and persistent identifier.
  F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
  F1.2. Different versions of the software are assigned distinct identifiers.
F2. Software is described with **rich metadata.**
F3. **Metadata** clearly and explicitly include the identifier of the software they describe.
F4. **Metadata** are FAIR, searchable and indexable.

Interoperable

I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. Software includes qualified references to other objects. **(metadata)**

GitLab/Hub + Zenodo DOI + e.g. .root + metadata!

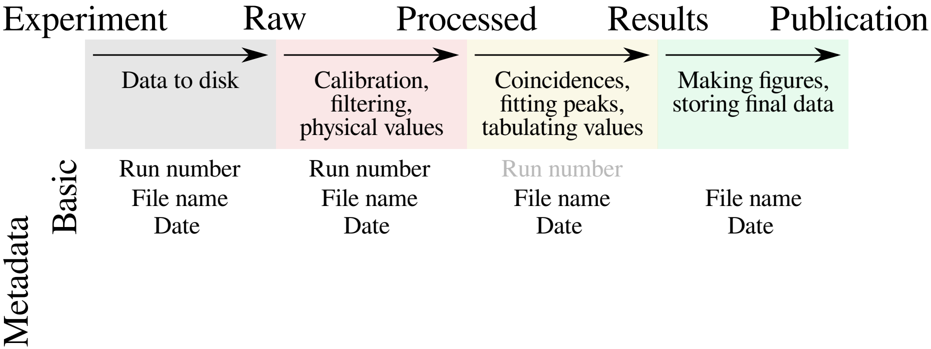
Accessible

A1. Software is retrievable by its identifier using a standardised communications protocol.
  A1.1. The protocol is open, free, and universally implementable.
  A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. **Metadata** are accessible, even when the software is no longer available.

Reusable

R1. Software is described with a plurality of accurate and relevant attributes. **(metadata)**
  R1.1. Software is given a clear and accessible license.
  R1.2. Software is associated with detailed provenance.
R2. Software includes qualified references to other software. **(metadata)**
R3. Software meets domain-relevant community standards.

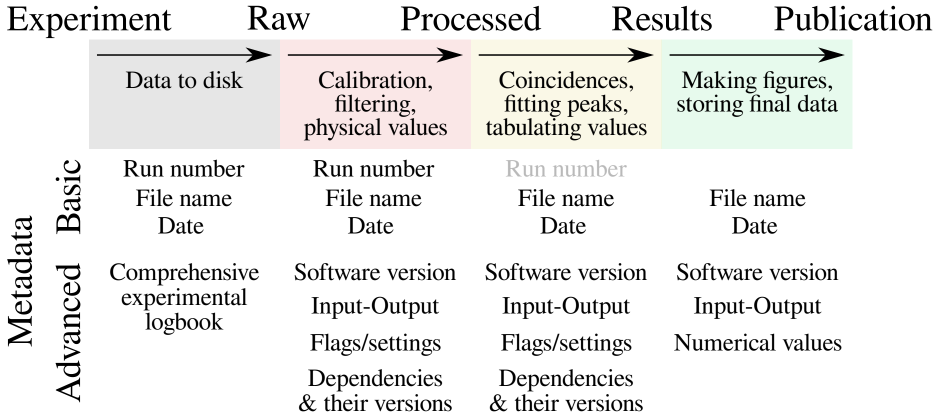
Metadata = "data about data"



Experiment → Raw → Processed → Results → Publication

	Data to disk	Calibration, filtering, physical values	Coincidences, fitting peaks, tabulating values	Making figures, storing final data
Metadata – Basic	Run number File name Date	Run number File name Date	Run number File name Date	File name Date

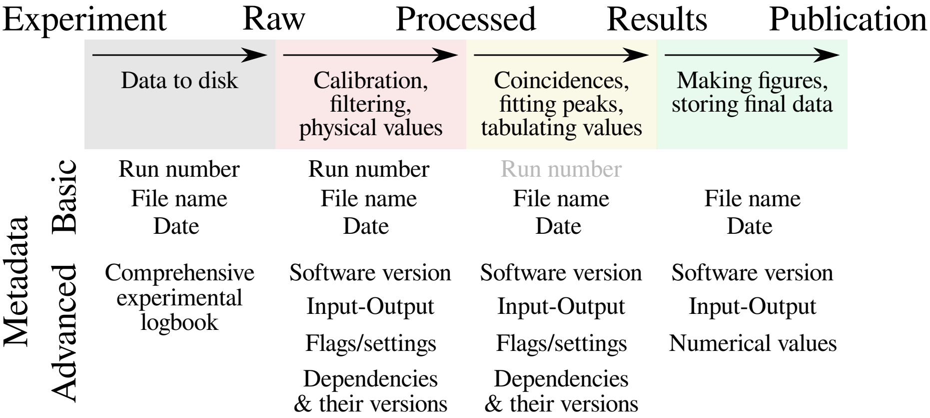
Metadata = "data about data"



Experiment → Raw → Processed → Results → Publication

Metadata		Experiment → Raw	Raw → Processed	Processed → Results	Results → Publication
		Data to disk	Calibration, filtering, physical values	Coincidences, fitting peaks, tabulating values	Making figures, storing final data
	Basic	Run number File name Date	Run number File name Date	Run number File name Date	File name Date
	Advanced	Comprehensive experimental logbook	Software version Input-Output Flags/settings Dependencies & their versions	Software version Input-Output Flags/settings Dependencies & their versions	Software version Input-Output Numerical values

Metadata = "data about data"

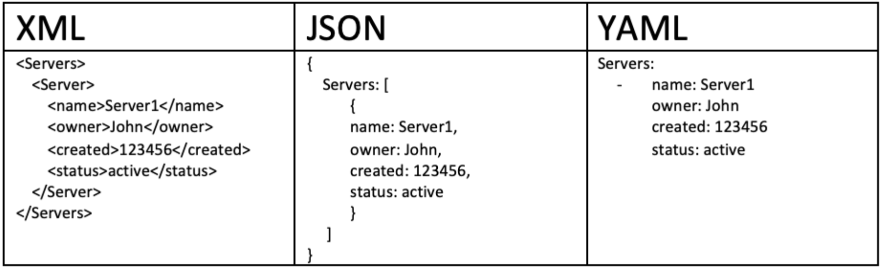


Experiment → Raw → Processed → Results → Publication

Metadata	Data to disk	Calibration, filtering, physical values	Coincidences, fitting peaks, tabulating values	Making figures, storing final data
Basic	Run number File name Date	Run number File name Date	Run number File name Date	File name Date
Advanced	Comprehensive experimental logbook	Software version Input-Output Flags/settings Dependencies & their versions	Software version Input-Output Flags/settings Dependencies & their versions	Software version Input-Output Numerical values

Goal: metadata so extensive that you can recreate Raw → Publication

Metadata format



XML

```
<Servers>
  <Server>
    <name>Server1</name>
    <owner>John</owner>
    <created>123456</created>
    <status>active</status>
  </Server>
</Servers>
```

JSON

```
{
  Servers: [
      {
      name: Server1,
      owner: John,
      created: 123456,
      status: active
      }
    ]
}
```

YAML

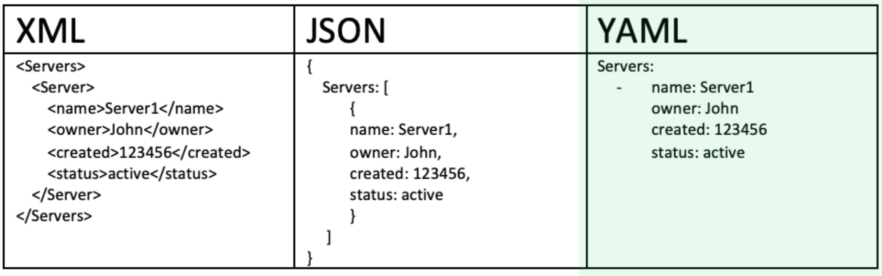
```
Servers:
  -     name: Server1
        owner: John
        created: 123456
        status: active
```

[6]

---

[6] `https://www.devopsschool.com/blog/comparison-between-xml-vs-json-vs-yaml/` (Accessed 28/01/2026)

Metadata format



XML

```
<Servers>
  <Server>
    <name>Server1</name>
    <owner>John</owner>
    <created>123456</created>
    <status>active</status>
  </Server>
</Servers>
```

JSON

```
{
  Servers: [
      {
      name: Server1,
      owner: John,
      created: 123456,
      status: active
      }
    ]
}
```

YAML

```
Servers:
  -     name: Server1
        owner: John
        created: 123456
        status: active
```

[6]

---

[6] https://www.devopsschool.com/blog/comparison-between-xml-vs-json-vs-yaml/ (Accessed 28/01/2026)

Automating metadata in python

```
import yaml

output_yaml = """
# Sequences/lists
sequence:
  - value1
  - value2

# Mapping
key: value

# Nested mapping
toplevel_key:
  attribute1: valueA
  attribute2: valueB
"""

output = yaml.safe_load(output_yaml)

with open(’output.yaml’, ’w’) as file:
    yaml.dump(output, file)
```

Extremely simple to use in python:

Automating metadata in python

```python
import yaml

output_yaml = """
# Sequences/lists
sequence:
  - value1
  - value2

# Mapping
key: value

# Nested mapping
toplevel_key:
  attribute1: valueA
  attribute2: valueB
"""

output = yaml.safe_load(output_yaml)

with open('output.yaml', 'w') as file:
    yaml.dump(output, file)
```

Extremely simple to use in python:

- `pip install pyyaml`

Automating metadata in python

```python
import yaml

output_yaml = """
# Sequences/lists
sequence:
  - value1
  - value2

# Mapping
key: value

# Nested mapping
toplevel_key:
  attribute1: valueA
  attribute2: valueB
"""

output = yaml.safe_load(output_yaml)

with open('output.yaml', 'w') as file:
    yaml.dump(output, file)
```

Extremely simple to use in python:

- `pip install pyyaml`
- `import yaml`

Automating metadata in python

```python
import yaml

output_yaml = """
# Sequences/lists
sequence:
  - value1
  - value2

# Mapping
key: value

# Nested mapping
toplevel_key:
  attribute1: valueA
  attribute2: valueB
"""

output = yaml.safe_load(output_yaml)

with open(’output.yaml’, ’w’) as file:
    yaml.dump(output, file)
```

Extremely simple to use in python:

- `pip install pyyaml`
- `import yaml`
- Start writing!

Automating metadata in C++

```cpp
#include "yaml-cpp/yaml.h"
int main(){
  YAML::Node output_node;  YAML::Emitter output_emit;

  // Sequences/lists
  output_node["sequence"].push_back("value1");
  output_node["sequence"].push_back("value2");

  // Mapping
  output_node["key"] = "value";

  // Nested mapping
  output_emit << YAML::BeginMap; //start map 1
  output_emit << YAML:Key << "toplevel_key";
  output_emit << YAML:Value << YAML::BeginMap; //start map 2
    output_emit << YAML:Key << "attribute1"
                << YAML::Value << "valueA";
    output_emit << YAML:Key << "attribute2"
                << YAML::Value << "valueB";
  output_emit << YAML::EndMap << YAML::EndMap; //end maps 1&2
}
```

Less fluid in C++:

Automating metadata in C++

```cpp
#include "yaml-cpp/yaml.h"
int main(){
  YAML::Node output_node;  YAML::Emitter output_emit;

  // Sequences/lists
  output_node["sequence"].push_back("value1");
  output_node["sequence"].push_back("value2");

  // Mapping
  output_node["key"] = "value";

  // Nested mapping
  output_emit << YAML::BeginMap; //start map 1
  output_emit << YAML:Key << "toplevel_key";
  output_emit << YAML:Value << YAML::BeginMap; //start map 2
    output_emit << YAML:Key << "attribute1"
                << YAML::Value << "valueA";
    output_emit << YAML:Key << "attribute2"
                << YAML::Value << "valueB";
  output_emit << YAML::EndMap << YAML::EndMap; //end maps 1&2
}
```

Less fluid in C++:

- `https://github.com/jbeder/yaml-cpp`

Automating metadata in C++

```
#include "yaml-cpp/yaml.h"
int main(){
  YAML::Node output_node;  YAML::Emitter output_emit;

  // Sequences/lists
  output_node["sequence"].push_back("value1");
  output_node["sequence"].push_back("value2");

  // Mapping
  output_node["key"] = "value";

  // Nested mapping
  output_emit << YAML::BeginMap; //start map 1
  output_emit << YAML:Key << "toplevel_key";
  output_emit << YAML:Value << YAML::BeginMap; //start map 2
    output_emit << YAML:Key << "attribute1"
                << YAML::Value << "valueA";
    output_emit << YAML:Key << "attribute2"
                << YAML::Value << "valueB";
  output_emit << YAML::EndMap << YAML::EndMap; //end maps 1&2
}
```

Less fluid in C++:

- `https://github.com/jbeder/yaml-cpp`
- Build with CMake, linking detailed in their docs

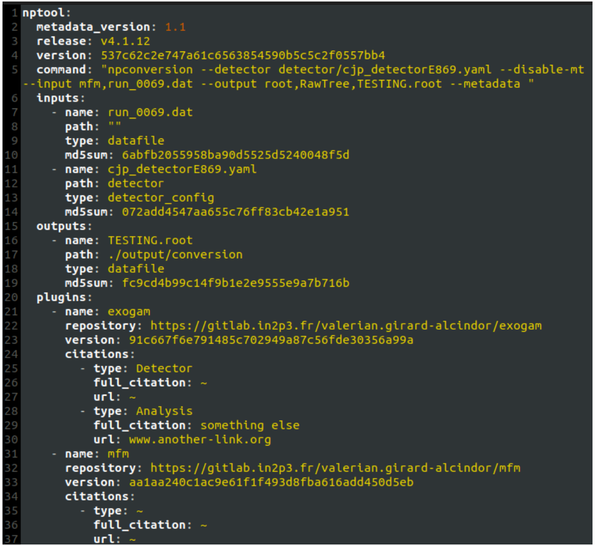
Automating metadata in C++

```cpp
#include "yaml-cpp/yaml.h"
int main(){
  YAML::Node output_node;  YAML::Emitter output_emit;

  // Sequences/lists
  output_node["sequence"].push_back("value1");
  output_node["sequence"].push_back("value2");

  // Mapping
  output_node["key"] = "value";

  // Nested mapping
  output_emit << YAML::BeginMap; //start map 1
  output_emit << YAML:Key << "toplevel_key";
  output_emit << YAML:Value << YAML::BeginMap; //start map 2
    output_emit << YAML:Key << "attribute1"
                << YAML::Value << "valueA";
    output_emit << YAML:Key << "attribute2"
                << YAML::Value << "valueB";
  output_emit << YAML::EndMap << YAML::EndMap; //end maps 1&2
}
```

Less fluid in C++:

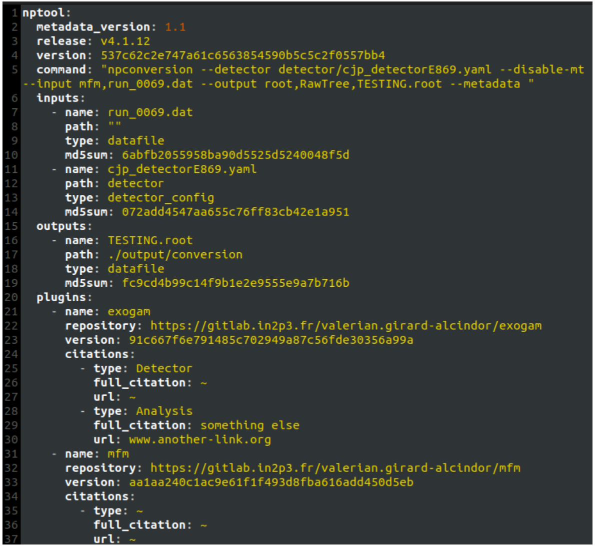
- `https://github.com/jbeder/yaml-cpp`
- Build with CMake, linking detailed in their docs
- YAML::Node or YAML::Emitter objects

Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

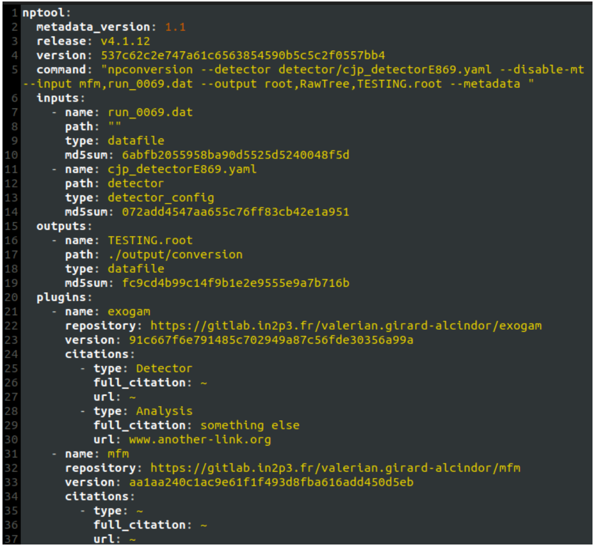
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata

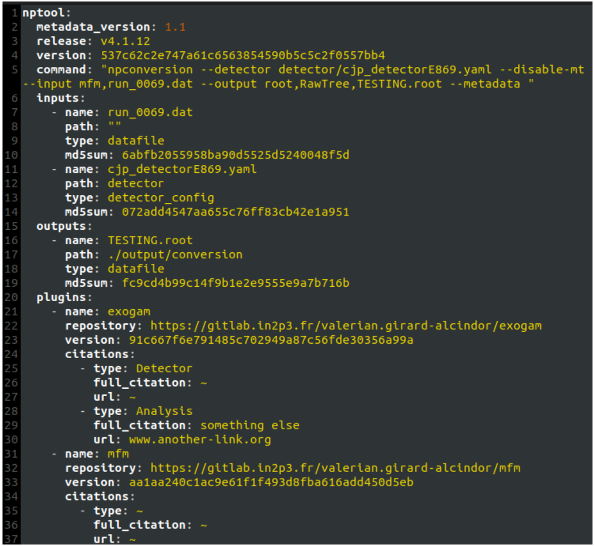
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable

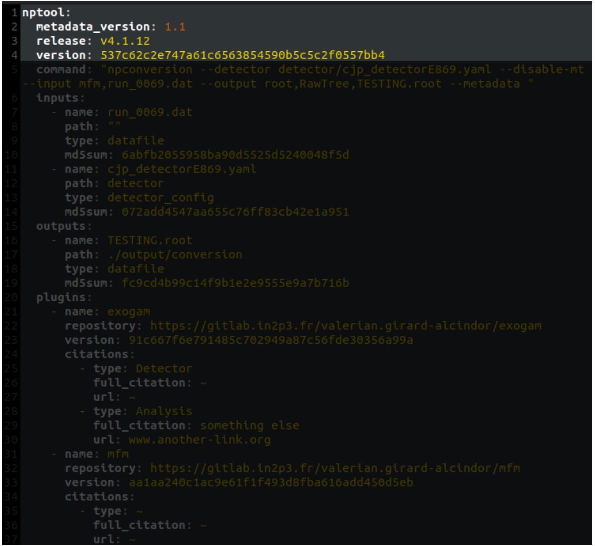
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt --input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable
- Stored with the output it describes
  - `/path/file.root`
  - `/path/file_metadata.yaml`

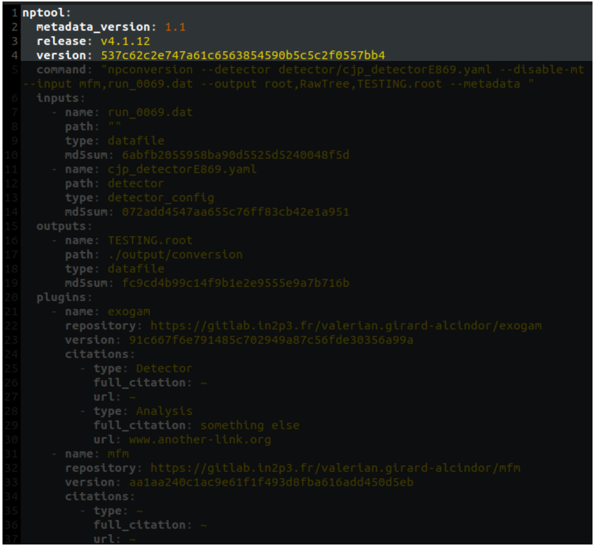
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable
- Stored with the output it describes
  - `/path/file.root`
  - `/path/file_metadata.yaml`
- Major-Minor-Patch version

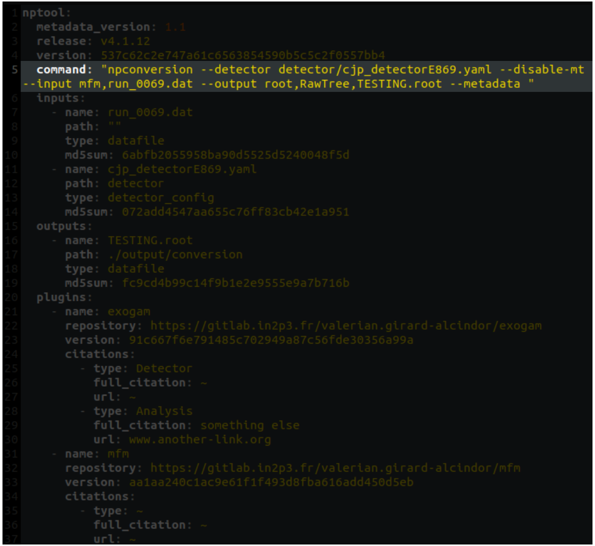
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable
- Stored with the output it describes
  - `/path/file.root`
  - `/path/file_metadata.yaml`
- Major-Minor-Patch version
- Exact git commit hash from `.git/HEAD`
  - Either hash, or path to branch hash

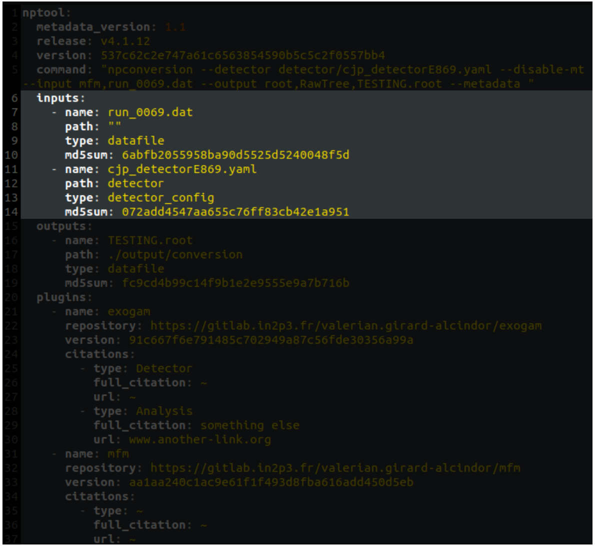
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt --input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable
- Stored with the output it describes
  - `/path/file.root`
  - `/path/file_metadata.yaml`
- Major-Minor-Patch version
- Exact git commit hash from `.git/HEAD`
  - Either hash, or path to branch hash
- Exact command including runtime options

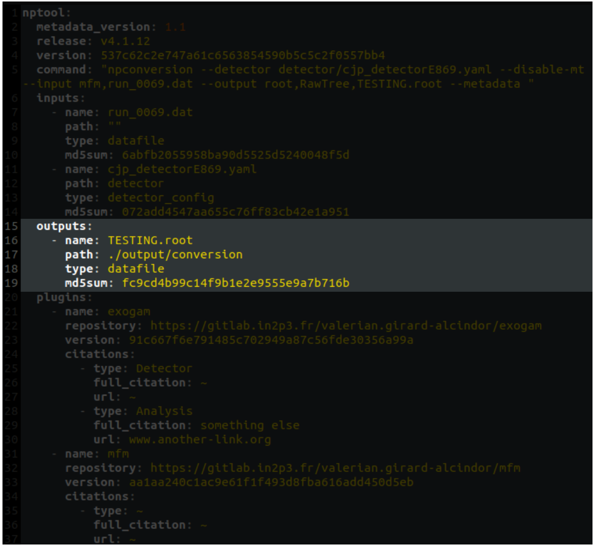
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable
- Stored with the output it describes
  - `/path/file.root`
  - `/path/file_metadata.yaml`
- Major-Minor-Patch version
- Exact git commit hash from `.git/HEAD`
  - Either hash, or path to branch hash
- Exact command including runtime options
- Details of input and output files (+MD5)
  - Unique ID! Hash of the bits of the file

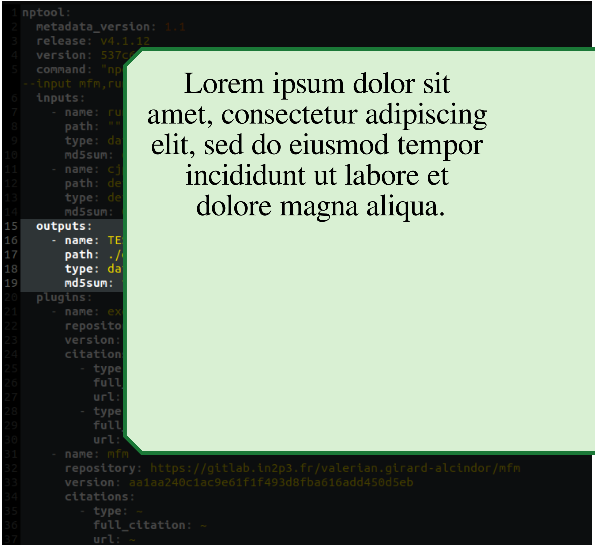
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable
- Stored with the output it describes
  - `/path/file.root`
  - `/path/file_metadata.yaml`
- Major-Minor-Patch version
- Exact git commit hash from `.git/HEAD`
  - Either hash, or path to branch hash
- Exact command including runtime options
- Details of input and output files (+MD5)
  - Unique ID! Hash of the bits of the file

Example: Automatic metadata in nptool v4 (NPMetadataManager)



```
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c
  command: "np
--input mfm,ru
  inputs:
    - name: ru
      path: ""
      type: da
      md5sum:
    - name: cj
      path: de
      type: de
      md5sum:
  outputs:
    - name: TE
      path: ./
      type: da
      md5sum:
  plugins:
    - name: ex
      reposito
      version:
      citation
        - type
          full
          url:
        - type
          full
          url:
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

- Human- and machine-readable metadata

ecutable

/HEAD

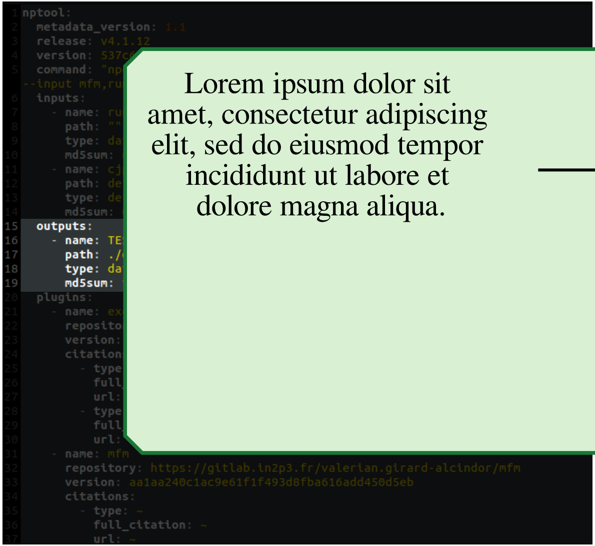
ash

options

MD5)

- Unique ID! Hash of the bits of the file

Example: Automatic metadata in nptool v4 (NPMetadataManager)



```
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c
  command: "np
--input mfm,ru
  inputs:
    - name: ru
      path: ""
      type: da
      md5sum:
    - name: cj
      path: de
      type: de
      md5sum:
  outputs:
    - name: TE
      path: ./
      type: da
      md5sum:
  plugins:
    - name: ex
      reposito
      version:
      citation
        - type
          full
          url:
        - type
          full
          url:
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

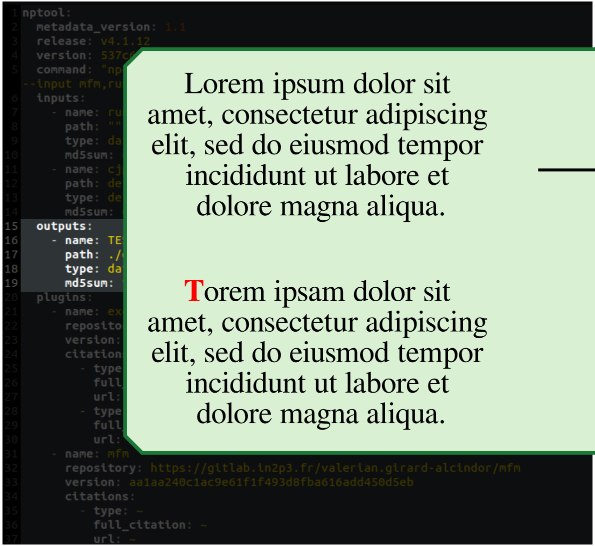
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

$$\xrightarrow{\text{md5sum}}$$

59322f2b
15f812e8
34d38da6
00a84517

- Human- and machine-readable metadata
- ecutable
- /`HEAD`
- ash
- options
- MD5)
- Unique ID! Hash of the bits of the file

Example: Automatic metadata in nptool v4 (NPMetadataManager)



```
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c
  command: "np
--input mfm,ru
  inputs:
    - name: ru
      path: ""
      type: da
      md5sum:
    - name: cj
      path: de
      type: de
      md5sum:
  outputs:
    - name: TE
      path: ./
      type: da
      md5sum:
  plugins:
    - name: ex
      reposito
      version:
      citation
        - type
          full
          url:
        - type
          full
          url:
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata

ecutable

`/HEAD`

ash

options

MD5)

- Unique ID! Hash of the bits of the file

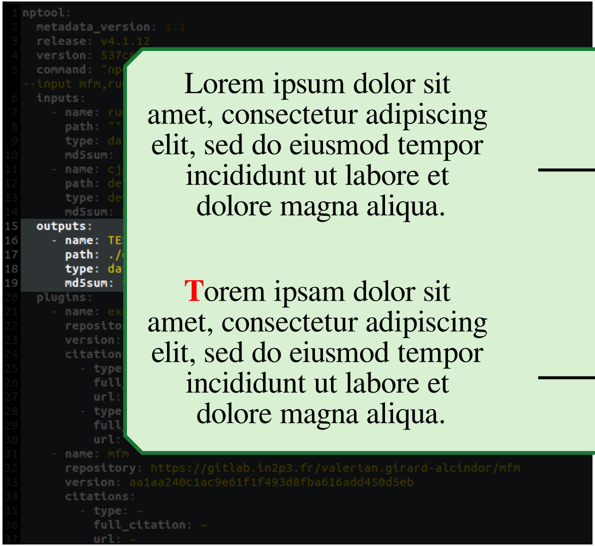
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

$\xrightarrow{\text{md5sum}}$

59322f2b
15f812e8
34d38da6
00a84517

Torem ipsam dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Example: Automatic metadata in nptool v4 (NPMetadataManager)



```
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c
  command: "np
--input mfm,ru
  inputs:
    - name: ru
      path: ""
      type: da
      md5sum:
    - name: cj
      path: de
      type: de
      md5sum:
  outputs:
    - name: TE
      path: ./
      type: da
      md5sum:
  plugins:
    - name: ex
      reposito
      version:
      citation
        - type
          full
          url:
        - type
          full
          url:
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

Lorem ipsum dolor sit
amet, consectetur adipiscing
elit, sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua.

$\xrightarrow{\text{md5sum}}$

59322f2b
15f812e8
34d38da6
00a84517

Torem ipsam dolor sit
amet, consectetur adipiscing
elit, sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua.

$\xrightarrow{\text{md5sum}}$

ad81f5c0
de2c820b
c757440b
8911956a

- Human- and machine-readable metadata

ecutable

`/HEAD`

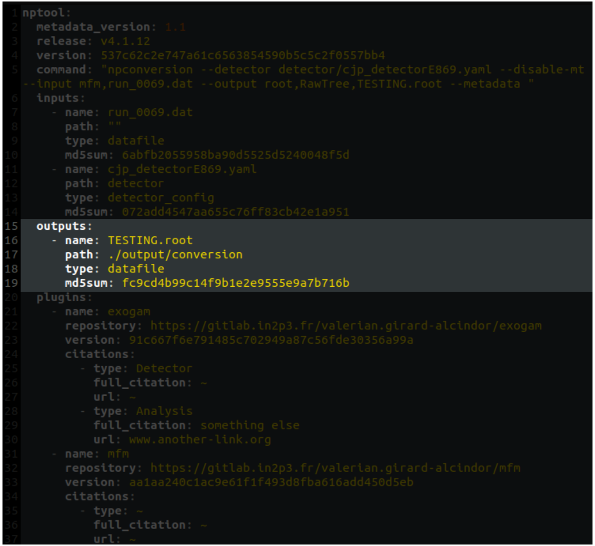
ash

options

MD5)

- Unique ID! Hash of the bits of the file

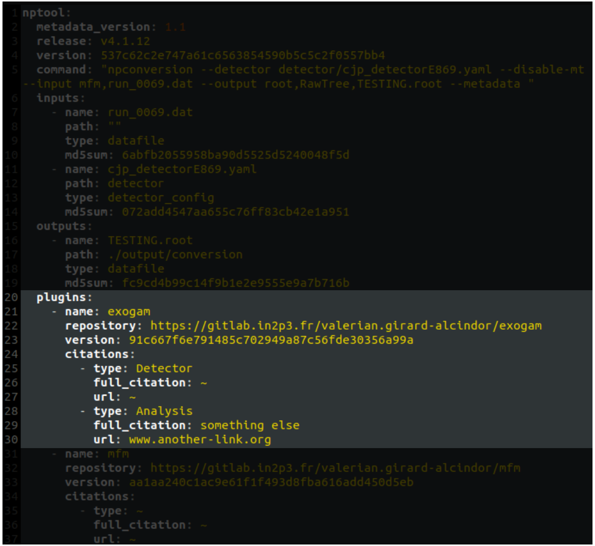
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable
- Stored with the output it describes
  - `/path/file.root`
  - `/path/file_metadata.yaml`
- Major-Minor-Patch version
- Exact git commit hash from `.git/HEAD`
  - Either hash, or path to branch hash
- Exact command including runtime options
- Details of input and output files (+MD5)
  - Unique ID! Hash of the bits of the file

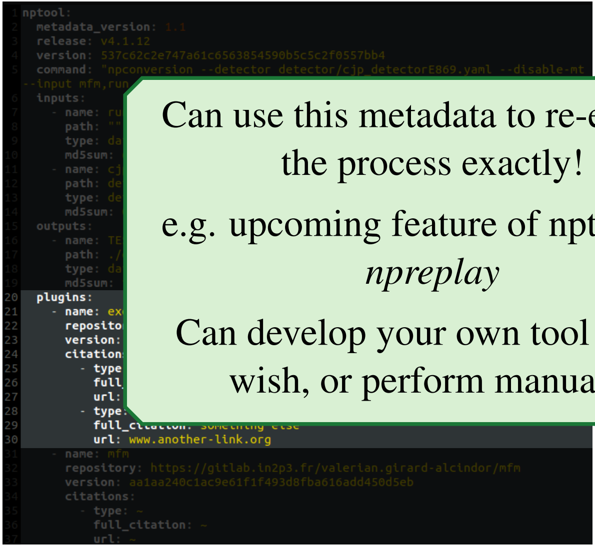
Example: Automatic metadata in nptool v4 (NPMetadataManager)



```yaml
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt --input mfm,run_0069.dat --output root,RawTree,TESTING.root --metadata "
  inputs:
    - name: run_0069.dat
      path: ""
      type: datafile
      md5sum: 6abfb2055958ba90d5525d5240048f5d
    - name: cjp_detectorE869.yaml
      path: detector
      type: detector_config
      md5sum: 072add4547aa655c76ff83cb42e1a951
  outputs:
    - name: TESTING.root
      path: ./output/conversion
      type: datafile
      md5sum: fc9cd4b99c14f9b1e2e9555e9a7b716b
  plugins:
    - name: exogam
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/exogam
      version: 91c667f6e791485c702949a87c56fde30356a99a
      citations:
        - type: Detector
          full_citation: ~
          url: ~
        - type: Analysis
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable
- Stored with the output it describes
  - `/path/file.root`
  - `/path/file_metadata.yaml`
- Major-Minor-Patch version
- Exact git commit hash from `.git/HEAD`
  - Either hash, or path to branch hash
- Exact command including runtime options
- Details of input and output files (+MD5)
  - Unique ID! Hash of the bits of the file
- Repos and versions of other software

Example: Automatic metadata in nptool v4 (NPMetadataManager)



```
nptool:
  metadata_version: 1.1
  release: v4.1.12
  version: 537c62c2e747a61c6563854590b5c5c2f0557bb4
  command: "npconversion --detector detector/cjp_detectorE869.yaml --disable-mt
--input mfm,run
  inputs:
    - name: ru
      path: ""
      type: da
      md5sum:
    - name: cj
      path: de
      type: de
      md5sum:
  outputs:
    - name: TE
      path: ./
      type: da
      md5sum:
  plugins:
    - name: ex
      reposito
      version:
      citation
        - type
          full
          url:
        - type:
          full_citation: something else
          url: www.another-link.org
    - name: mfm
      repository: https://gitlab.in2p3.fr/valerian.girard-alcindor/mfm
      version: aa1aa240c1ac9e61f1f493d8fba616add450d5eb
      citations:
        - type: ~
          full_citation: ~
          url: ~
```

- Human- and machine-readable metadata
- Generated automatically during executable

HEAD

ash

options

MD5)

  - Unique ID! Hash of the bits of the file
- Repos and versions of other software

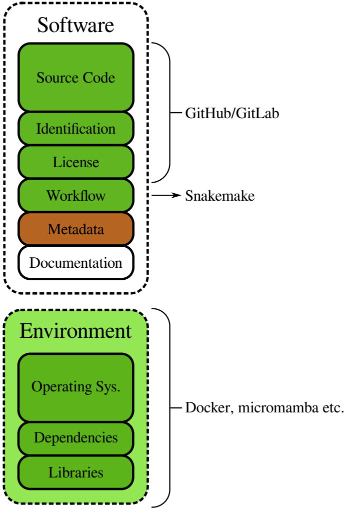
Can use this metadata to re-execute the process exactly!

e.g. upcoming feature of nptool v4, *npreplay*

Can develop your own tool if you wish, or perform manually

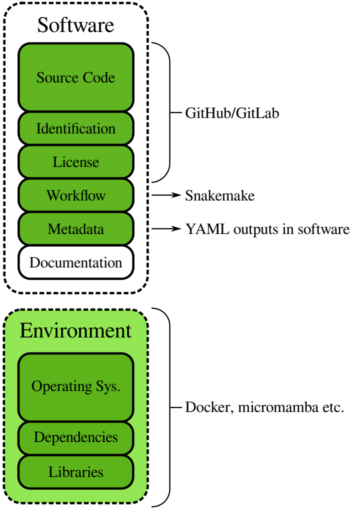
nptool

Getting closer...



Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

Getting closer...


Software

Source Code

Identification

License

Workflow

Metadata

Documentation

GitHub/GitLab

Snakemake

YAML outputs in software

Environment

Operating Sys.

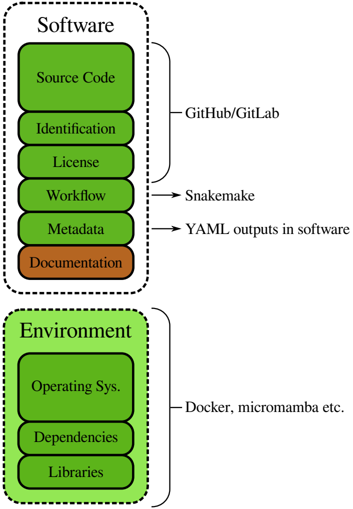
Dependencies

Libraries

Docker, micromamba etc.

Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

Getting closer...



Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

Documentation for reproducibility

Documentation for reproducibility

- Code without documentation is a nightmare
  - Purpose or limitations unclear
  - Unknown inputs/outputs
  - Chase parameters from function to function

Documentation for reproducibility

- Code without documentation is a nightmare
  - Purpose or limitations unclear
  - Unknown inputs/outputs
  - Chase parameters from function to function
- Documentation answers:
  - What does this code do?
  - How do I run it?
  - How do I modify or extend it?

Documentation for reproducibility

- Code without documentation is a nightmare
  - Purpose or limitations unclear
  - Unknown inputs/outputs
  - Chase parameters from function to function
- Documentation answers:
  - What does this code do?
  - How do I run it?
  - How do I modify or extend it?
- Helpful for collaborators:
  - Explain methods, choices, future plans
  - Understand intention, so easier to contribute

Documentation for reproducibility

- Code without documentation is a nightmare
  - Purpose or limitations unclear
  - Unknown inputs/outputs
  - Chase parameters from function to function
- Documentation answers:
  - What does this code do?
  - How do I run it?
  - How do I modify or extend it?
- Helpful for collaborators:
  - Explain methods, choices, future plans
  - Understand intention, so easier to contribute
- Helpful for you:
  - Forces you to think again about what & why
  - Understand in a few month's time

Types of documentation

Narrative documentation

API/Code documentation

Types of documentation

Narrative documentation

- More conceptual

API/Code documentation

- More technical

Types of documentation

Narrative documentation

- More conceptual
- Parts *can* be read in order (in theory)

API/Code documentation

- More technical
- To be skimmed as needed

Types of documentation

Narrative documentation

- More conceptual
- Parts *can* be read in order (in theory)
- Describes how to use the code
  - Installation instructions
  - README files
  - Tutorials & examples

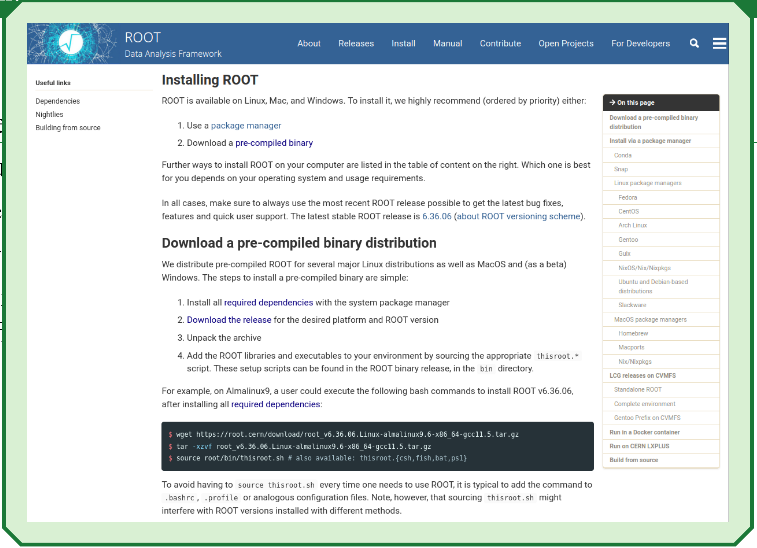
API/Code documentation

- More technical
- To be skimmed as needed

Types of documentation

Narrative docume

- More conceptu
- Parts *can* be re
- Describes how
  - Installation
  - README f
  - Tutorials &



Useful links

- Dependencies
- Nightlies
- Building from source

Installing ROOT

ROOT is available on Linux, Mac, and Windows. To install it, we highly recommend (ordered by priority) either:

1. Use a package manager
2. Download a pre-compiled binary

Further ways to install ROOT on your computer are listed in the table of content on the right. Which one is best for you depends on your operating system and usage requirements.

In all cases, make sure to always use the most recent ROOT release possible to get the latest bug fixes, features and quick user support. The latest stable ROOT release is 6.36.06 (about ROOT versioning scheme).

Download a pre-compiled binary distribution

We distribute pre-compiled ROOT for several major Linux distributions as well as MacOS and (as a beta) Windows. The steps to install a pre-compiled binary are simple:

1. Install all required dependencies with the system package manager
2. Download the release for the desired platform and ROOT version
3. Unpack the archive
4. Add the ROOT libraries and executables to your environment by sourcing the appropriate `thisroot.*` script. These setup scripts can be found in the ROOT binary release, in the `bin` directory.

For example, on Almalinux9, a user could execute the following bash commands to install ROOT v6.36.06, after installing all required dependencies:

```
$ wget https://root.cern/download/root_v6.36.06.Linux-almalinux9.6-x86_64-gcc11.5.tar.gz
$ tar -xzvf root_v6.36.06.Linux-almalinux9.6-x86_64-gcc11.5.tar.gz
$ source root/bin/thisroot.sh # also available: thisroot.{csh,fish,bat,ps1}
```

To avoid having to `source thisroot.sh` every time one needs to use ROOT, it is typical to add the command to `.bashrc`, `.profile` or analogous configuration files. Note, however, that sourcing `thisroot.sh` might interfere with ROOT versions installed with different methods.

→ On this page

- Download a pre-compiled binary distribution
- Install via a package manager
  - Conda
  - Snap
  - Linux package managers
    - Fedora
    - CentOS
    - Arch Linux
    - Gentoo
    - Guix
    - NixOS/Nix/Nixpkgs
    - Ubuntu and Debian-based distributions
    - Slackware
  - MacOS package managers
    - Homebrew
    - Macports
    - Nix/Nixpkgs
- LCG releases on CVMFS
  - Standalone ROOT
  - Complete environment
  - Gentoo Prefix on CVMFS
- Run in a Docker container
- Run on CERN LXPLUS
- Build from source

Types of documentation

Narrative documentation

- More conceptual
- Parts *can* be read in order (in theory)
- Describes how to use the code
    - Installation instructions
    - README files
    - Tutorials & examples

API/Code documentation

- More technical
- To be skimmed as needed

Types of documentation

Narrative documentation

- More conceptual
- Parts *can* be read in order (in theory)
- Describes how to use the code
  - Installation instructions
  - README files
  - Tutorials & examples

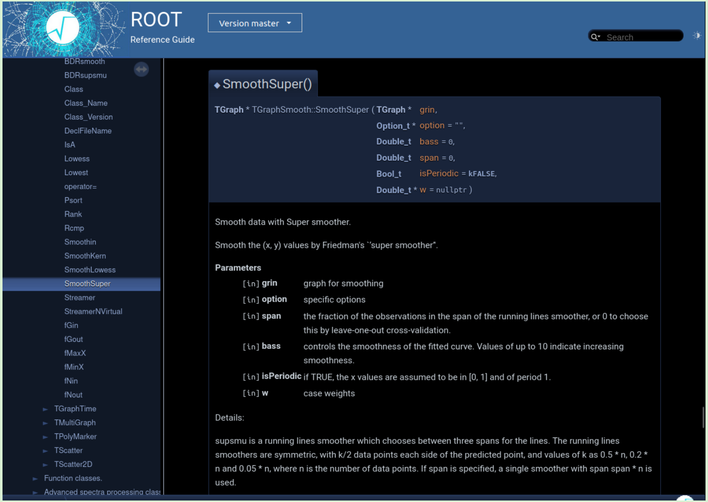
API/Code documentation

- More technical
- To be skimmed as needed
- Describes what the code is doing
  - Functions, classes
  - Inputs, outputs
  - What it does, how it does it

Types of documentation

Narrative docume

- More conceptu
- Parts *can* be re
- Describes how … oing
  - Installation i
  - README f
  - Tutorials & … it



◆ SmoothSuper()

```
TGraph * TGraphSmooth::SmoothSuper ( TGraph *   grin,
                                     Option_t * option = "",
                                     Double_t   bass = 0,
                                     Double_t   span = 0,
                                     Bool_t     isPeriodic = kFALSE,
                                     Double_t * w = nullptr )
```

Smooth data with Super smoother.

Smooth the (x, y) values by Friedman's ``super smoother".

Parameters

[in] **grin**	graph for smoothing
[in] **option**	specific options
[in] **span**	the fraction of the observations in the span of the running lines smoother, or 0 to choose this by leave-one-out cross-validation.
[in] **bass**	controls the smoothness of the fitted curve. Values of up to 10 indicate increasing smoothness.
[in] **isPeriodic**	if TRUE, the x values are assumed to be in [0, 1] and of period 1.
[in] **w**	case weights

Details:

supsmu is a running lines smoother which chooses between three spans for the lines. The running lines smoothers are symmetric, with k/2 data points each side of the predicted point, and values of k as 0.5 * n, 0.2 * n and 0.05 * n, where n is the number of data points. If span is specified, a single smoother with span span * n is used.

Types of documentation

Narrative documentation

- More conceptual
- Parts *can* be read in order (in theory)
- Describes how to use the code
    - Installation instructions
    - README files
    - Tutorials & examples
- In a specific folder of the repo, e.g. `docs/`

API/Code documentation

- More technical
- To be skimmed as needed
- Describes what the code is doing
    - Functions, classes
    - Inputs, outputs
    - What it does, how it does it

Types of documentation

Narrative documentation

- More conceptual
- Parts *can* be read in order (in theory)
- Describes how to use the code
  - Installation instructions
  - README files
  - Tutorials & examples
- In a specific folder of the repo, e.g. `docs/`

API/Code documentation

- More technical
- To be skimmed as needed
- Describes what the code is doing
  - Functions, classes
  - Inputs, outputs
  - What it does, how it does it
- Written next to the code it describes

Types of documentation

Narrative documentation

- More conceptual
- Parts *can* be read in order (in theory)
- Describes how to use the code
    - Installation instructions
    - README files
    - Tutorials & examples
- In a specific folder of the repo, e.g. `docs/`
- ... and/or a website

API/Code documentation

- More technical
- To be skimmed as needed
- Describes what the code is doing
    - Functions, classes
    - Inputs, outputs
    - What it does, how it does it
- Written next to the code it describes
- ... and/or a website

Guidance

General rules from the *WriteTheDocs* team [7]

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID Accept (some) Repetition In Documentation (≠ DRY, Don't Repeat Yourself)

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation (≠ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation ($\neq$ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation (≠ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation ($\neq$ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.
Current	Consider incorrect documentation to be worse than missing documentation.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation (≠ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.
Current	Consider incorrect documentation to be worse than missing documentation.
Nearby	Store sources as close as possible to the code which they document.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation (≠ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.
Current	Consider incorrect documentation to be worse than missing documentation.
Nearby	Store sources as close as possible to the code which they document.
Unique	Eliminate content overlap between separate sources – easier to maintain!

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation ($\neq$ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.
Current	Consider incorrect documentation to be worse than missing documentation.
Nearby	Store sources as close as possible to the code which they document.
Unique	Eliminate content overlap between separate sources – easier to maintain!
Discoverable	Funnel users intuitively towards publications through all likely pathways.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation (≠ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.
Current	Consider incorrect documentation to be worse than missing documentation.
Nearby	Store sources as close as possible to the code which they document.
Unique	Eliminate content overlap between separate sources – easier to maintain!
Discoverable	Funnel users intuitively towards publications through all likely pathways.
Addressable	Provide addresses to readers that link directly to content at a granular level.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation (≠ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.
Current	Consider incorrect documentation to be worse than missing documentation.
Nearby	Store sources as close as possible to the code which they document.
Unique	Eliminate content overlap between separate sources – easier to maintain!
Discoverable	Funnel users intuitively towards publications through all likely pathways.
Addressable	Provide addresses to readers that link directly to content at a granular level.
Cumulative	Content should be ordered to cover prerequisite concepts first.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation (≠ DRY, Don’t Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.
Current	Consider incorrect documentation to be worse than missing documentation.
Nearby	Store sources as close as possible to the code which they document.
Unique	Eliminate content overlap between separate sources – easier to maintain!
Discoverable	Funnel users intuitively towards publications through all likely pathways.
Addressable	Provide addresses to readers that link directly to content at a granular level.
Cumulative	Content should be ordered to cover prerequisite concepts first.
Complete	Within each publication, cover concepts in full, or not at all.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Guidance

General rules from the *WriteTheDocs* team [7]

ARID	Accept (some) Repetition In Documentation (≠ DRY, Don't Repeat Yourself)
Skimmable	Structure content to help readers identify and skip over concepts
Exemplary	Include (some) examples and tutorials in content.
Consistent	Use consistent language and formatting in content.
Current	Consider incorrect documentation to be worse than missing documentation.
Nearby	Store sources as close as possible to the code which they document.
Unique	Eliminate content overlap between separate sources – easier to maintain!
Discoverable	Funnel users intuitively towards publications through all likely pathways.
Addressable	Provide addresses to readers that link directly to content at a granular level.
Cumulative	Content should be ordered to cover prerequisite concepts first.
Complete	Within each publication, cover concepts in full, or not at all.
Beautiful	Visual style should be intentional and aesthetically pleasing.

---

[7] `www.writethedocs.org/guide/writing/docs-principles/` (Accessed on 29/01/2026)

Code documentation

Critical, load-bearing functions you wrote 14 months ago, expecting to never look at them again:

```python
def cut(e):
    return e[e > 3]
```

```cpp
double calc(double e, double t)
{
    return e * std::sin(t) / (t * t);
}
```

Code documentation

Critical, load-bearing functions you wrote 14 months ago, expecting to never look at them again:

```python
def cut(e):
    return e[e > 3]
```

```cpp
double calc(double e, double t)
{
    return e * std::sin(t) / (t * t);
}
```

Now, your collaborator wants to use and expand your code...

Code documentation

Critical, load-bearing functions you wrote 14 months ago, expecting to never look at them again:

```python
def cut(e):
    return e[e > 3]
```

```cpp
double calc(double e, double t)
{
    return e * std::sin(t) / (t * t);
}
```

Now, your collaborator wants to use and expand your code...

Assuming `e` is energy, what are the units? keV? MeV?
Why is the threshold of `cut` set to 3?
What is `calc` calculating? Why?

Code documentation

```python
def energy_cut(events, threshold_MeV):
    """
    Apply a lower energy threshold cut.

    Parameters
    ----------
    events : array-like
        Event data structure containing an ’energy’ field in MeV.
    threshold_MeV : float
        Lower energy threshold in MeV.

    Returns
    -------
    selected_events : same type as events
        Events with energy >= threshold_MeV.

    Notes
    -----
    This cut does not correct for detector resolution or efficiency.
    """
    return events[events["energy"] >= threshold_MeV]
```

- Use docstrings!
- In/outputs
- Notes on usage
- Readable by documentation generators!
- e.g. **Sphinx**

Sphinx for python

`www.sphinx-doc.org`

- Many options available for python, but this is a strong tool, commonly used!

Sphinx for python

`www.sphinx-doc.org`

- Many options available for python, but this is a strong tool, commonly used!
- Parses your code, builds static sites (e.g. HTML) to be deployed as you wish

Sphinx for python

`www.sphinx-doc.org`

- Many options available for python, but this is a strong tool, commonly used!
- Parses your code, builds static sites (e.g. HTML) to be deployed as you wish
- Install with `pip install sphinx`, or through `conda-forge`, or many other methods...

Sphinx for python

`www.sphinx-doc.org`

- Many options available for python, but this is a strong tool, commonly used!
- Parses your code, builds static sites (e.g. HTML) to be deployed as you wish
- Install with `pip install sphinx`, or through `conda-forge`, or many other methods...
- Easy for narrative documentation, *and* code documentation

Sphinx for python

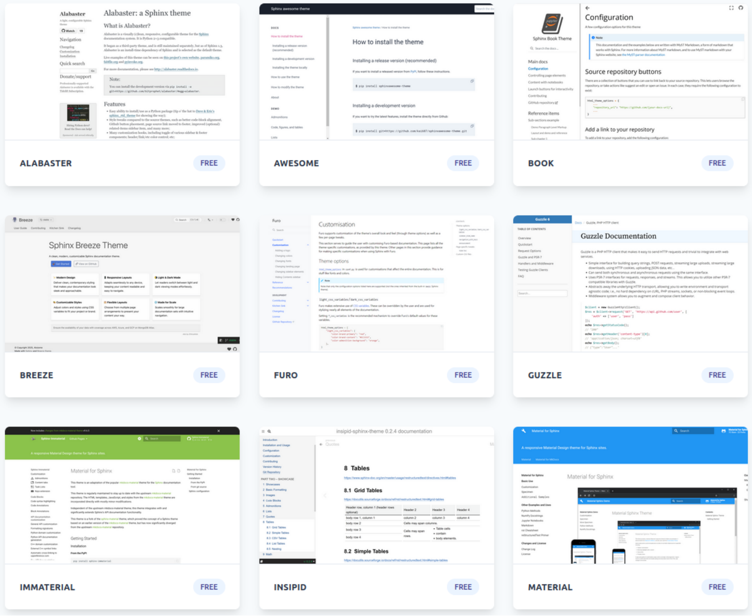
`www.sphinx-doc.org`

- Many options available for python, but this is a strong tool, commonly used!
- Parses your code, builds static sites (e.g. HTML) to be deployed as you wish
- Install with `pip install sphinx`, or through `conda-forge`, or many other methods...
- Easy for narrative documentation, *and* code documentation

Pros

- Narrative & code docs at once
- Very readable, loads of themes available (installed with `pip`)

Sphinx for pythor



- Many options :
- Parses your co
- Install with `pi` nethods...
- Easy for narrat

Pros

- Narrative & cc
- Very readable, (installed with

Sphinx for python

`www.sphinx-doc.org`

- Many options available for python, but this is a strong tool, commonly used!
- Parses your code, builds static sites (e.g. HTML) to be deployed as you wish
- Install with `pip install sphinx`, or through `conda-forge`, or many other methods...
- Easy for narrative documentation, *and* code documentation

Pros

- Narrative & code docs at once
- Very readable, loads of themes available (installed with `pip`)

Sphinx for python

`www.sphinx-doc.org`

- Many options available for python, but this is a strong tool, commonly used!
- Parses your code, builds static sites (e.g. HTML) to be deployed as you wish
- Install with `pip install sphinx`, or through `conda-forge`, or many other methods...
- Easy for narrative documentation, *and* code documentation

Pros

- Narrative & code docs at once
- Very readable, loads of themes available (installed with `pip`)

Cons

- Claims to do C++, but not really!
  Cannot parse, have to maintain by hand!

Sphinx for python

`www.sphinx-doc.org`

- Many options available for python, but this is a strong tool, commonly used!
- Parses your code, builds static sites (e.g. HTML) to be deployed as you wish
- Install with `pip install sphinx`, or through `conda-forge`, or many other methods...
- Easy for narrative documentation, *and* code documentation

Pros

- Narrative & code docs at once
- Very readable, loads of themes available (installed with `pip`)

Cons

- Claims to do C++, but not really!
  Cannot parse, have to maintain by hand!

For C++ users, life is never as simple...

Code documentation

```
/**
 * @brief Calculate angular-dependent scattering amplitude.
 *
 * Computes a simplified scattering amplitude proportional to:
 *     A(E, theta) = E * sin(theta) / theta^2
 *
 * @param energy_MeV Incident particle energy in MeV
 * @param theta_rad Scattering angle in radians
 * @return Scattering amplitude (arbitrary units)
 *
 * @note Diverges for theta -> 0; caller must avoid this region.
 */
double scattering_amplitude(double energy_MeV, double theta_rad)
{
    return energy_MeV * std::sin(theta_rad) /
           (theta_rad * theta_rad);
}
```

- Doxygen-style comments
- Other comment styles, /// or //!
- Specify with `@brief`, `@exception`, `@related`... (`@` or `\`)
- Readable by Doxygen

Doxygen for C++

doxygen

`https://doxygen.nl/`

Doxygen for C++

`https://doxygen.nl/`

- Like Sphinx, builds static websites which are deployable however you want

Doxygen for C++

doxygen

`https://doxygen.nl/`

- Like Sphinx, builds static websites which are deployable however you want
- Download as `.tar` file, CMake installation (but, installation is *very* well-documented!)

Doxygen for C++

doxygen

`https://doxygen.nl/`

- Like Sphinx, builds static websites which are deployable however you want
- Download as `.tar` file, CMake installation (but, installation is *very* well-documented!)
- Only real option for C++ code docs. Other projects have risen and fallen (cldoc, standardese)

Doxygen for C++

doxygen

`https://doxygen.nl/`

- Like Sphinx, builds static websites which are deployable however you want
- Download as `.tar` file, CMake installation (but, installation is *very* well-documented!)
- Only real option for C++ code docs. Other projects have risen and fallen (cldoc, standardese)
- Code-focussed, but can handle narrative documentation too

Doxygen for C++

doxygen

`https://doxygen.nl/`

- Like Sphinx, builds static websites which are deployable however you want
- Download as `.tar` file, CMake installation (but, installation is *very* well-documented!)
- Only real option for C++ code docs. Other projects have risen and fallen (cldoc, standardese)
- Code-focussed, but can handle narrative documentation too

Pros

- Very stable and reliable for many years
- Full C++ parsing (and Fortran, Python...)
- Can generate linking diagrams

Doxygen for C++

doxygen

`https://doxygen.nl/`

- Like Sphinx, builds static websites which are deployable however you want
- Download as `.tar` file, CMake installation (but, installation is *very* well-documented!)
- Only real option for C++ code docs. Other projects have risen and fallen (cldoc, standardese)
- Code-focussed, but can handle narrative documentation too

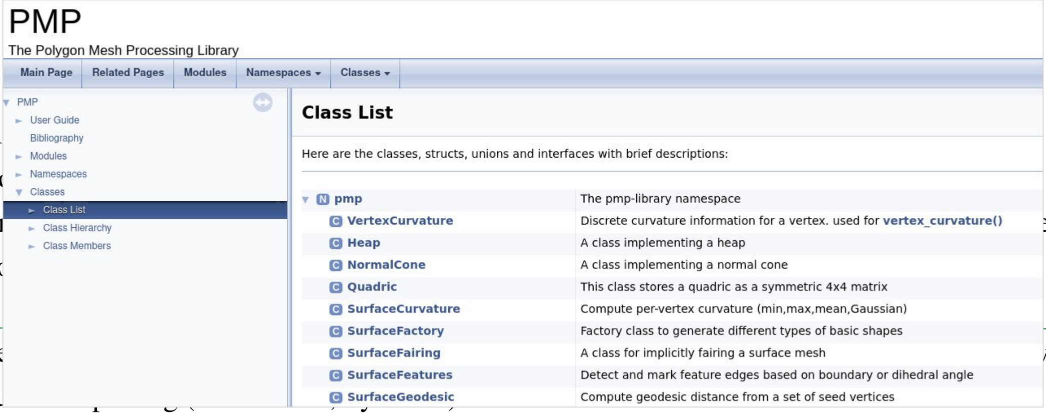
Pros

- Very stable and reliable for many years
- Full C++ parsing (and Fortran, Python...)
- Can generate linking diagrams

Cons

- Very ugly, requires a little bit of work give it a ‘modern’ feel

Doxygen for C++



PMP

The Polygon Mesh Processing Library

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

pmp	The pmp-library namespace
VertexCurvature	Discrete curvature information for a vertex. used for **vertex_curvature()**
Heap	A class implementing a heap
NormalCone	A class implementing a normal cone
Quadric	This class stores a quadric as a symmetric 4x4 matrix
SurfaceCurvature	Compute per-vertex curvature (min,max,mean,Gaussian)
SurfaceFactory	Factory class to generate different types of basic shapes
SurfaceFairing	A class for implicitly fairing a surface mesh
SurfaceFeatures	Detect and mark feature edges based on boundary or dihedral angle
SurfaceGeodesic	Compute geodesic distance from a set of seed vertices

- Li
- Do
- Or e)
- Co

Pros

- Ve ve it
- Fu
- Can generate linking diagrams

Doxygen for C++

doxygen

`https://doxygen.nl/`

- Like Sphinx, builds static websites which are deployable however you want
- Download as `.tar` file, CMake installation (but, installation is *very* well-documented!)
- Only real option for C++ code docs. Other projects have risen and fallen (cldoc, standardese)
- Code-focussed, but can handle narrative documentation too

Pros

- Very stable and reliable for many years
- Full C++ parsing (and Fortran, Python...)
- Can generate linking diagrams

Cons

- Very ugly, requires a little bit of work give it a ‘modern’ feel

Doxygen for C++

doxygen

`https://doxygen.nl/`

- Like Sphinx, builds static websites which are deployable however you want
- Download as `.tar` file, CMake installation (but, installation is *very* well-documented!)
- Only real option for C++ code docs. Other projects have risen and fallen (cldoc, standardese)
- Code-focussed, but can handle narrative documentation too

Pros

- Very stable and reliable for many years
- Full C++ parsing (and Fortran, Python...)
- Can generate linking diagrams

Cons

- Very ugly, requires a little bit of work give it a ‘modern’ feel

Now you have your HTML, how do you get it online?

Hosting documentation with GitLab Pages

- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`

Hosting documentation with GitLab Pages

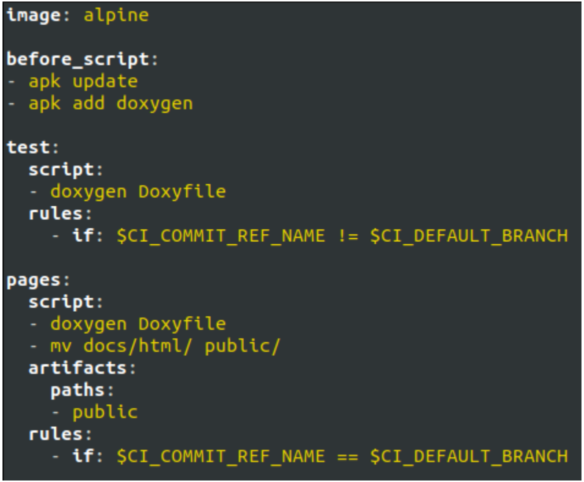
- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)

Hosting documentation with GitLab Pages

- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline

Hosting documentation with GitLab Pages

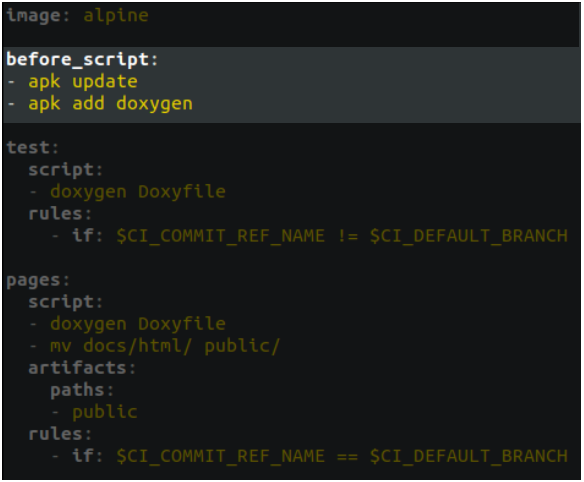
- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline



```
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

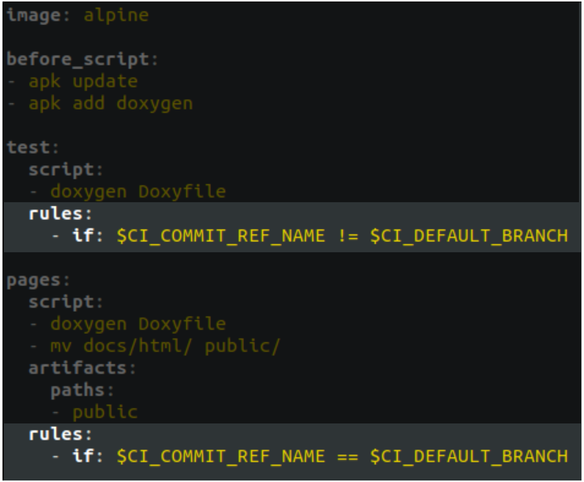
- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline



```
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
      - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

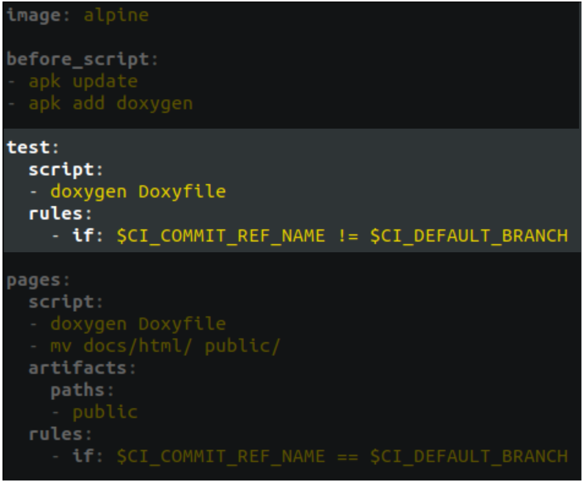
- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline



```yaml
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

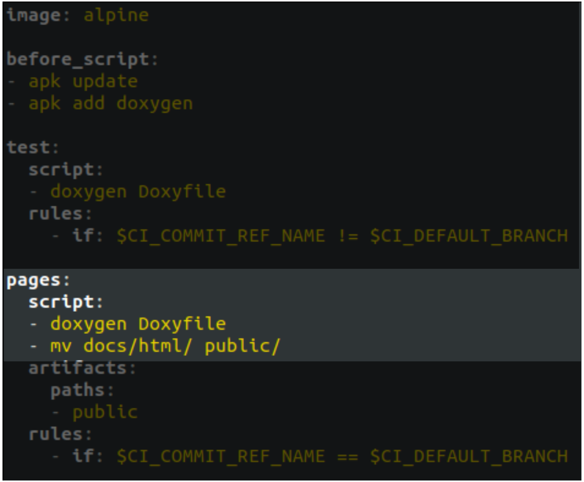
- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline



```
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

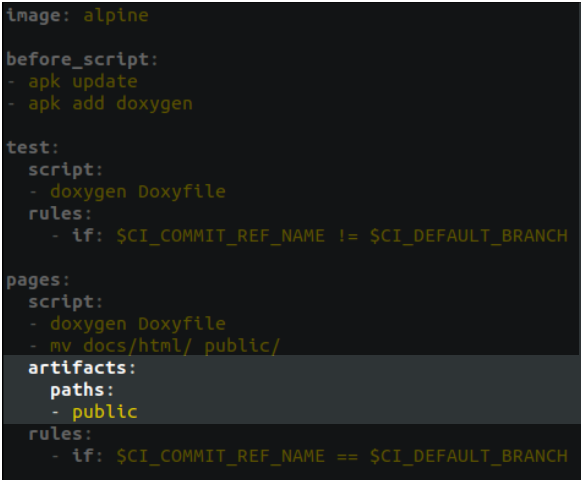
- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline



```yaml
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

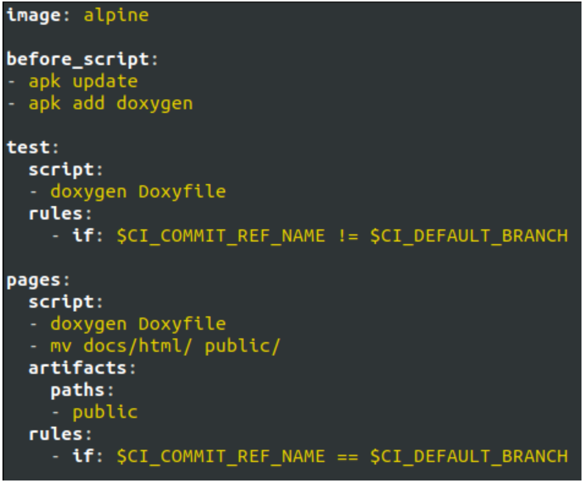
- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline



```yaml
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline
- **See also:** ReadTheDocs, free hosting for open-source projects with git integration



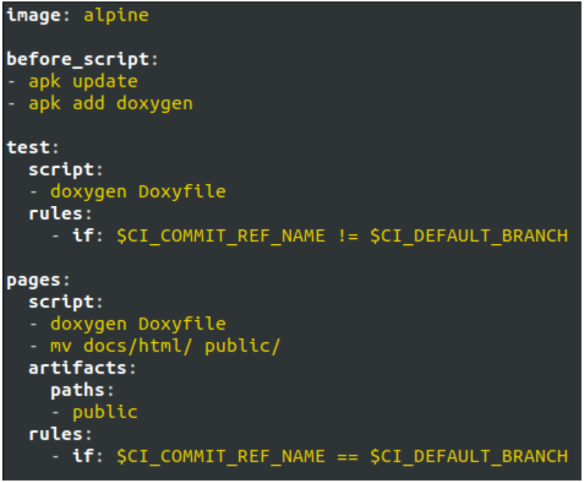
```
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline
- **See also:** ReadTheDocs, free hosting for open-source projects with git integration

---

- **Warning:** Development on MkDocs (static generator) has now stalled!



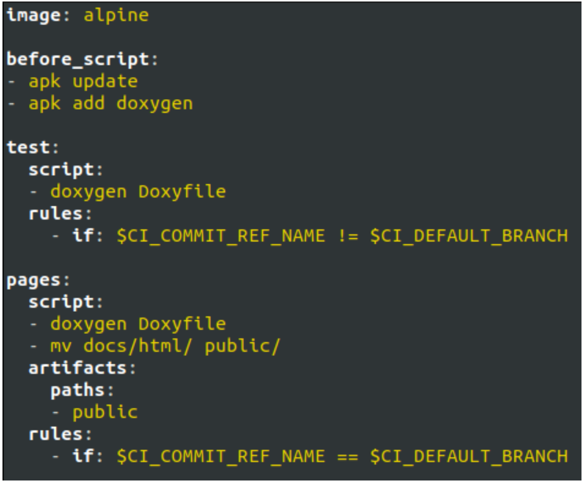
```yaml
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline
- **See also:** ReadTheDocs, free hosting for open-source projects with git integration

---

- **Warning:** Development on MkDocs (static generator) has now stalled!
- **Other generators:** Zensical, Docusaurus...



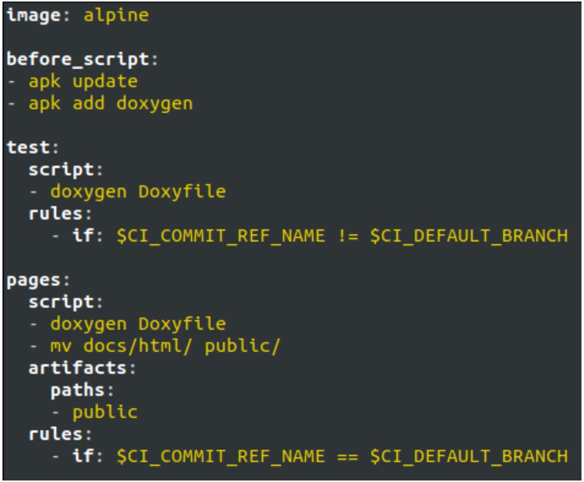
```
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

Hosting documentation with GitLab Pages

- Free static site hosting
  - Self-hosted: standard `.gitlab.io`
  - Lab-hosted: custom, e.g. `.in2p3.fr`
- Tied to the repo (therefore versioned)
- Can build and deploy in CI/CD pipeline
- **See also:** ReadTheDocs, free hosting for open-source projects with git integration

---

- **Warning:** Development on MkDocs (static generator) has now stalled!
- **Other generators:** Zensical, Docusaurus...



```
image: alpine

before_script:
- apk update
- apk add doxygen

test:
  script:
  - doxygen Doxyfile
  rules:
    - if: $CI_COMMIT_REF_NAME != $CI_DEFAULT_BRANCH

pages:
  script:
  - doxygen Doxyfile
  - mv docs/html/ public/
  artifacts:
    paths:
    - public
  rules:
    - if: $CI_COMMIT_REF_NAME == $CI_DEFAULT_BRANCH
```

I don't have time and/or don't care, what is the **easiest option**?

Narrative-only documentation – GitLab Wiki

GitLab

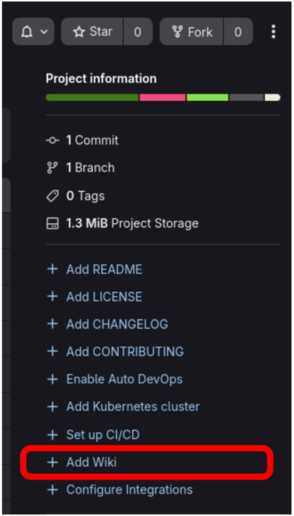
`https://docs.gitlab.com/user/project/wiki/`

Narrative-only documentation – GitLab Wiki

GitLab

`https://docs.gitlab.com/user/project/wiki`

- No download, no command line! Just a few clicks

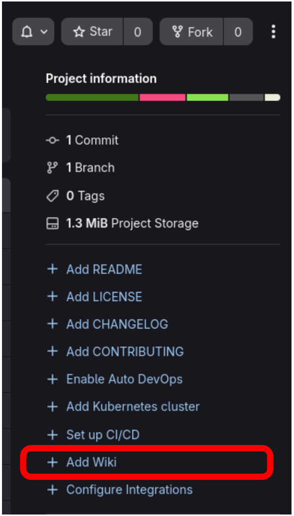


Narrative-only documentation – GitLab Wiki

GitLab

`https://docs.gitlab.com/user/project/wiki`

- No download, no command line! Just a few clicks
- Narrative documentation, in a separate (but linked) repository



Narrative-only documentation – GitLab Wiki

`https://docs.gitlab.com/user/project/wiki/`

- No download, no command line! Just a few clicks
- Narrative documentation, in a separate (but linked) repository
- No automatic API/code documentation, but better than one huge README file

Narrative-only documentation – GitLab Wiki

GitLab

`https://docs.gitlab.com/user/project/wiki/`

- No download, no command line! Just a few clicks
- Narrative documentation, in a separate (but linked) repository
- No automatic API/code documentation, but better than one huge README file
- Can make page templates to encourage consistent styling between pages

Narrative-only documentation – GitLab Wiki

GitLab

`https://docs.gitlab.com/user/project/wiki/`

- No download, no command line! Just a few clicks
- Narrative documentation, in a separate (but linked) repository
- No automatic API/code documentation, but better than one huge README file
- Can make page templates to encourage consistent styling between pages

Pros

- Simplest option for narrative documentation
- Can leave comments on wiki pages
- Can ‘subscribe’ for notification of changes

Narrative-only documentation – GitLab Wiki

GitLab

`https://docs.gitlab.com/user/project/wiki/`

- No download, no command line! Just a few clicks
- Narrative documentation, in a separate (but linked) repository
- No automatic API/code documentation, but better than one huge README file
- Can make page templates to encourage consistent styling between pages

Pros

- Simplest option for narrative documentation
- Can leave comments on wiki pages
- Can ‘subscribe’ for notification of changes

Cons

- No automatic API documentation
- Not obvious on the site! Best to point out the existence of a wiki in the README

Narrative-only documentation – GitLab Wiki

GitLab

Whatever you do, decide early in the project and document it as you go!

Easier to document when you are writing it, rather than spending an hour documenting 1000’s of lines of code

I live in a glass house...

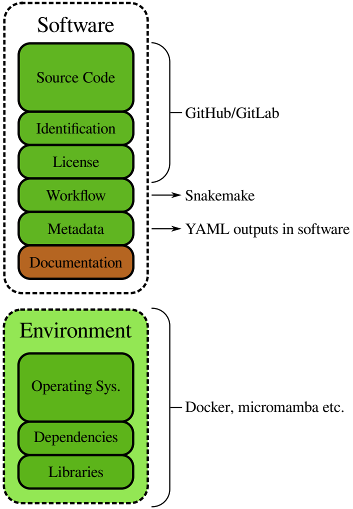
- No dow
- Narrati
- No auto
- Can ma

Pros

- Simple
- Can lea
- Can ‘subscribe’ for notification of changes

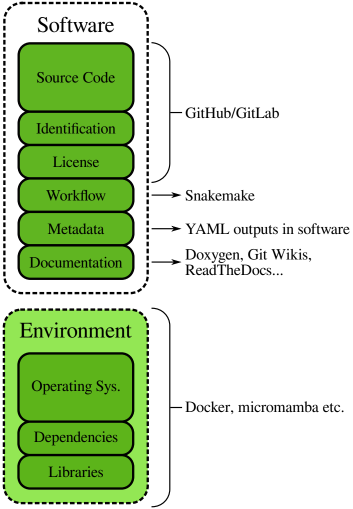
int out the existence of a wiki in the README

...and we're there!



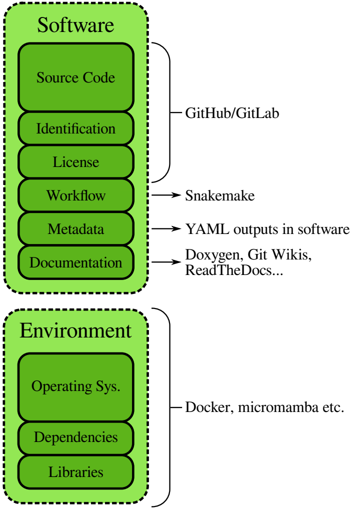
Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

...and we're there!



Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

...and we're there!



Adapted from Fig. 1 of D. S. Katz *et al.*, Patterns **2** 3 100222, `https://doi.org/10.1016/j.patter.2021.100222`

Summary

Software Metadata

- Data about data, gives necessary context

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
  - Comprehensive (there is no ‘overkill’!)
  - Versions, commands, in/outputs
  - For humans & machines, e.g. YAML
  - As close to outputs as possible

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
  - Comprehensive (there is no ‘overkill’!)
  - Versions, commands, in/outputs
  - For humans & machines, e.g. YAML
  - As close to outputs as possible
- How to do produce metadata:
  - Automate it! Hand-made *will* decay
  - `pyyaml` for python, `yaml-cpp` for C++

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
    - Comprehensive (there is no ‘overkill’!)
    - Versions, commands, in/outputs
    - For humans & machines, e.g. YAML
    - As close to outputs as possible
- How to do produce metadata:
    - Automate it! Hand-made *will* decay
    - `pyyaml` for python, `yaml-cpp` for C++
- Critical to achieve FAIR4RS-ness!

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
  - Comprehensive (there is no ‘overkill’!)
  - Versions, commands, in/outputs
  - For humans & machines, e.g. YAML
  - As close to outputs as possible
- How to do produce metadata:
  - Automate it! Hand-made *will* decay
  - `pyyaml` for python, `yaml-cpp` for C++
- Critical to achieve FAIR4RS-ness!

Software Documentation

- Guide for anyone not in your head *now*

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
    - Comprehensive (there is no ‘overkill’!)
    - Versions, commands, in/outputs
    - For humans & machines, e.g. YAML
    - As close to outputs as possible
- How to do produce metadata:
    - Automate it! Hand-made *will* decay
    - `pyyaml` for python, `yaml-cpp` for C++
- Critical to achieve FAIR4RS-ness!

Software Documentation

- Guide for anyone not in your head *now*
- Narrative documentation:
    - What the software does, how to use it
    - Installation, tutorial, examples

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
  - Comprehensive (there is no ‘overkill’!)
  - Versions, commands, in/outputs
  - For humans & machines, e.g. YAML
  - As close to outputs as possible
- How to do produce metadata:
  - Automate it! Hand-made *will* decay
  - `pyyaml` for python, `yaml-cpp` for C++
- Critical to achieve FAIR4RS-ness!

Software Documentation

- Guide for anyone not in your head *now*
- Narrative documentation:
  - What the software does, how to use it
  - Installation, tutorial, examples
- API/Code documentation:
  - Granular docs of specific components
  - Classes & functions, what, why & how

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
    - Comprehensive (there is no ‘overkill’!)
    - Versions, commands, in/outputs
    - For humans & machines, e.g. YAML
    - As close to outputs as possible
- How to do produce metadata:
    - Automate it! Hand-made *will* decay
    - `pyyaml` for python, `yaml-cpp` for C++
- Critical to achieve FAIR4RS-ness!

Software Documentation

- Guide for anyone not in your head *now*
- Narrative documentation:
    - What the software does, how to use it
    - Installation, tutorial, examples
- API/Code documentation:
    - Granular docs of specific components
    - Classes & functions, what, why & how
- Write as you go! Retro-fitting is tedious

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
  - Comprehensive (there is no ‘overkill’!)
  - Versions, commands, in/outputs
  - For humans & machines, e.g. YAML
  - As close to outputs as possible
- How to do produce metadata:
  - Automate it! Hand-made *will* decay
  - `pyyaml` for python, `yaml-cpp` for C++
- Critical to achieve FAIR4RS-ness!

Software Documentation

- Guide for anyone not in your head *now*
- Narrative documentation:
  - What the software does, how to use it
  - Installation, tutorial, examples
- API/Code documentation:
  - Granular docs of specific components
  - Classes & functions, what, why & how
- Write as you go! Retro-fitting is tedious
- Keep near what it describes

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
  - Comprehensive (there is no ‘overkill’!)
  - Versions, commands, in/outputs
  - For humans & machines, e.g. YAML
  - As close to outputs as possible
- How to do produce metadata:
  - Automate it! Hand-made *will* decay
  - `pyyaml` for python, `yaml-cpp` for C++
- Critical to achieve FAIR4RS-ness!

Software Documentation

- Guide for anyone not in your head *now*
- Narrative documentation:
  - What the software does, how to use it
  - Installation, tutorial, examples
- API/Code documentation:
  - Granular docs of specific components
  - Classes & functions, what, why & how
- Write as you go! Retro-fitting is tedious
- Keep near what it describes
- ...and (maybe) online, easier navigation

Summary

Software Metadata

- Data about data, gives necessary context
- What metadata should be:
  - Comprehensive (there is no 'overkill'!)
  - Versions, commands, in/outputs
  - For humans & machines, e.g. YAML
  - As close to outputs as possible
- How to do produce metadata:
  - Automate it! Hand-made *will* decay
  - `pyyaml` for python, `yaml-cpp` for C++
- Critical to achieve FAIR4RS-ness!

Software Documentation

- Guide for anyone not in your head *now*
- Narrative documentation:
  - What the software does, how to use it
  - Installation, tutorial, examples
- API/Code documentation:
  - Granular docs of specific components
  - Classes & functions, what, why & how
- Write as you go! Retro-fitting is tedious
- Keep near what it describes
- ...and (maybe) online, easier navigation

Requires a bit of setup, but improves your software and saves time down the road!