HIFIS HELMHOLTZ FEDERATED IT SERVICES

RESEARCH FOR GRAND CHALLENGES

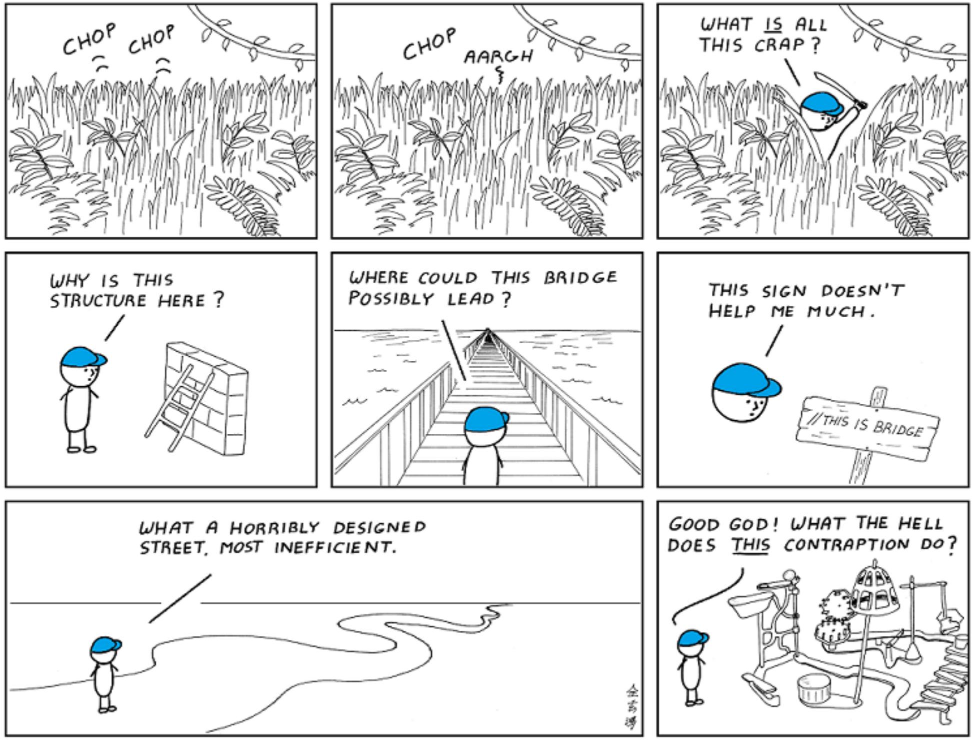
Solutions for the Ages - A Short Crash Course on Sustainable Software Development

Tobias Huste

HIFIS, Helmholtz-Zentrum Dresden – Rossendorf

2021-04-29, International Virtual COVID-data-Challenge

How It Should Not Look Like



CHOP CHOP
CHOP
AARGH
WHAT IS ALL THIS CRAP?
WHY IS THIS STRUCTURE HERE?
WHERE COULD THIS BRIDGE POSSIBLY LEAD?
THIS SIGN DOESN'T HELP ME MUCH.
//THIS IS BRIDGE
WHAT A HORRIBLY DESIGNED STREET. MOST INEFFICIENT.
GOOD GOD! WHAT THE HELL DOES THIS CONTRAPTION DO?

I hate reading other people's code.

Comic taken from https://abstrusegoose.com/432.

6 Steps to Make Your Code Ready For Publication

Put your code under version control

Make sure that your code is in a sharable state

Add essential documentation

Add a license

Mark the stable version of your code

Make your code citable

Talk built upon the HIFIS course *Let us make your script ready for publication*. (CC-BY-4.0)
https://gitlab.com/hifis/hifis-workshops/make-your-code-ready-for-publication/workshop-materials

Step 1

Put Your Code Under Version Control

Why?

- Sharing and collaborating.
- Being able to go back to a specific state at any time.

Where?

- Bare Minimum: Use a local Git repository (`git init`).
- Collaboration platforms like GitLab or GitHub.

What?

- Everything that is required to create a usable version of your code and to produce the intended results.
- Typically avoid adding generated artifacts (Keyword: `.gitignore`).

Step 2

Make Sure That Your Code Is In a Sharable State

Why?

- Otherwise, others are not able to use your code.
- You might accidentally share things you do not want to share.

General Requirements

- Code can be run outside your organization.
- Create a suitable directory structure and structure code in suitable building blocks.
- Apply to *community of practice* in your programming language, domain, etc.
- Clarify your dependencies.
- Do not share secrets!

Step 2

Make Sure That Your Code Is In a Sharable State

Understandable Code:

- Consistent code style.
- Use meaningful and consistent names.
- Do not *over-comment*; but comment *clever tricks* or the *big picture.*
- Experiment with light-weight code reviews.

- Standard style guide: PEP8
- Multiple formatters and linters exist
  - Black
  - Flake8
  - Pylint
  - Integrate into your CI pipeline
- Structure your Python project
  - "Hitchhiker's Guide" to "Structuring Your Project"
  - "Application Layout" reference
- Poetry – Useful for dependency mgmt.

Step 3

Add Essential Documentation

Why?

- Otherwise, potential **users** do not want to use or do not know how to use your software.
- Otherwise, potential **contributors** do not know how to provide their contributions in an efficient manner.

Typical Documentation Files

- **README**: The front page of your code. Should be created in any case!
- Other typical documentation files:
  - *CONTRIBUTING*,
  - *CODE_OF_CONDUCT*,
  - *LICENSE* file or *LICENSES* folder,
  - *CHANGELOG*,
  - *CITATION*.

Step 3

Add Essential Documentation

Documentation as Code
"Code and documentation, created and maintained equally."

- Use markup languages: *Markdown*, *Asciidoc*, *RestructuredText.*
- Typical minimal README structure.
- Typically required documentation for Open Source.
  - GitHub's community profiles
  - Open Source Guides
- For python™
  - Use Sphinx to generate professional documentation.
  - Use docstrings to document your Python objects, …

Step 4

Add a License

Why?

- Potential users cannot (re-)use your software from the legal point of view.

Copyright

- Software is protected by Copyright.
- Copyright holder has certain exclusive rights: Usage, creation of copies, distribution, creation of derivative works.
- Copyright gives other persons no rights, unless the copyright holder explicitly grants them.

Step 4

Add a License

Software licenses are a way for a copyright holder to grant rights to other persons or legal entities.

- A software license **grants** certain rights (e.g., use, copy, distribute) and **demands** certain obligations (e.g., disclosure of source code under a certain license, constraints concerning the distribution, attribution).
- **Every software that you use has to be covered by a license.**

1. Choose a license.
2. Ask your boss for permission to share your software.
3. Prepare your code**.**

Step 5

Mark the Stable Version of Your Code

Why?

- Otherwise users do not know which version is considered stable.
- Otherwise users do not exactly know which version has been used to produce a specific result.

Release Basics

- A *release* is a stable version of your software.
- The *release number* uniquely identifies the released software version.
- The *release tag* marks the release content in the source code repository.
- The *Changelog* documents all released versions.

Step 5

Mark the Stable Version of Your Code

Minimal Release Checklist

- Define which release number scheme you want to use.
    - Semantic Versioning
    - Calendar Versioning
- Define how release tags are named.

1. Prepare your code for release.
    - Test your code on the basis of the package you provide to your users.
    - Define the release number.
    - Document user-visible changes in your Changelog.
2. Create a release tag.
    - Use a tag to mark the version in the repository.

Step 6

Make Your Code Citable

Why?

- Software is a **research product**, just like a paper or a monograph.
- Creating and maintaining research software is **academic work** and should allow for academic credit and careers.
- Citing software is an important part of the **provenance of research** results and enables reproducibility.

How to cite software?

- Cite all software packages (also your own) in the reference list of academic work.
- Follow guidelines.

Step 6

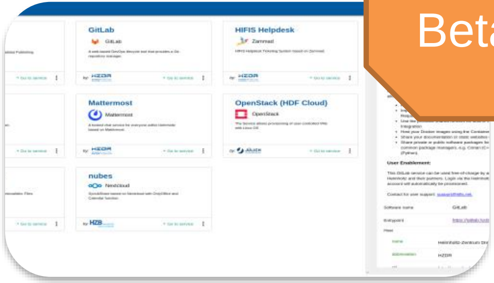
Make Your Code Citable

- Allow others to easily cite your software, by
  - Providing citation metadata,
  - Obtaining a persistent identifier (PID),
  - Providing a citation hint.

- Two practical solutions
  1. Deposit software in a digital repository. (See https://zenodo.org)
  2. Publish software on a public code hosting platform, add citation metadata and use the Software Heritage PID for reference.

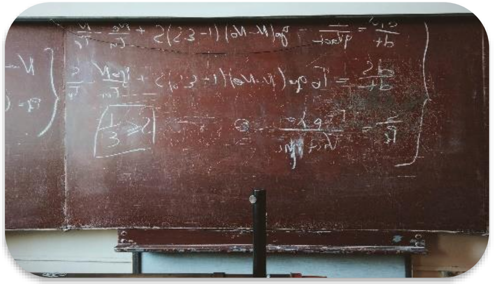
- In addition, consider writing a software paper. Consider the *Journal of Open Source Software.*

HIFIS For You



Helmholtz Cloud

https://cloud.helmholtz.de



Education & Training

https://software.hifis.net/events

Consulting

https://software.hifis.net/services/consulting

Helmholtz GitLab

https://gitlab.hzdr.de