# 2025-11-19-ds-rss-Collaborative Document Day 2

Welcome to the Collaborative Document of the Research Software Support Training.

This Document is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

---

**Content website** (https://esciencecenter-digital-skills.github.io/research-software-support/)

This is the Document for today: https://edu.nl/yyqbr

Collaborative Document day 1: https://edu.nl/qvm9f

Collaborative Document day 2: https://edu.nl/yyqbr

## Code of Conduct

Participants are expected to follow these guidelines:

- Use welcoming and inclusive language.
- Be respectful of different viewpoints and experiences.
- Gracefully accept constructive criticism.
- Focus on what is best for the community.
- Show courtesy and respect towards other community members.

If you feel that the code of conduct is breached, please talk to one of the instructors (if the complaint is for one of the participants) or send an email to training@esciencecenter.nl (if the complaint is for one of the instructors).

## ⚖ License



## Getting help

To ask a question, just raise your hand.

If you need help from a helper, place a pink post-it note on your laptop lid. A helper will come to assist you as soon as possible.

## Workshop website

link (https://esciencecenter-digital-skills.github.io/2025-11-05-ds-rss-masterclass/)

**Content website** (https://esciencecenter-digital-skills.github.io/research-software-support/)

## Instructors

Fenne Riemslagh, Ole Mussmann, Ewan Cahen, Lourens Veen

## □□□□□□□□□□♂□ Roll Call

Name/ pronouns (optional) / job, role / social media (twitter, github, ...) / background or interests (optional) / city

- 

## □ Agenda

| Time | Topic |
|---|---|
| 09:30 | Welkom, recap and homework |
| 10:00 | Version Control and Software Testing |
| 11:00 | Break |
| 11:15 | Software Documentation |
| 12:15 | Lunch Break |
| 13:15 | Distributing Software |
| 14:15 | Break |
| 14:30 | Software Licenses |
| 15:15 | Break |
| 15:30 | Software Publication and Citation |
| 16:15 | Wrap-up |
| 16:30 | End & Drinks |

## Location logistics

- Coffee and toilets are in the hallway, just outside of the classroom.
- If you leave the building,
  be sure to be accompanied by someone from the escience center to let you back in through the groundfloor door
- For access to this floor you might need to ring the doorbell so someone can let you in
- In case of an emergency, you can exit our floor using the main staircase.
  Or follow green light signs at the ceiling to the emergency staircase.
- **Wifi**: Eduroam should work. Otherwise use the 'matrixbuilding' network, password should be printed out and available somewhere in the room.

## Certificate of attendance

If you attend the full workshop you can request a certificate of attendance by emailing to training@esciencecenter.nl.
Please request your certificate within 8 months after the workshop, as we will delete all personal identifyable information after this period.

# Icebreaker

What software did you use this morning when coming here? How do you usually stay updated on the latest trends and technologies?

# Exercises

## Software Version Control

### Which of the following do you think should be part of a code repository

Link (https://esciencecenter-digital-skills.github.io/research-software-support/modules/version-control/version_control_exercise)

Don't include the full dataset, include the rest. Split user and developer documentation. Maybe add a test case so user can see if installation was successful.

## Software Testing

### Identify the test types in this story

Link (https://esciencecenter-digital-skills.github.io/research-software-support/modules/testing/exercise)

- Put out fire: smoke test
- Each component separately: unit test
- Check if parts fit: integration test
- Check if pen writes: system test
- New features: regression test

### Self test

Link (https://esciencecenter-digital-skills.github.io/research-software-support/modules/testing/self-test)

## Software Documentation

### When to use which documentation?

Link (https://esciencecenter-digital-skills.github.io/research-software-support/modules/documentation/exercise-levels)

Answers are more of a guideline, depends on context.

- Scenario A: In-code documentation, naming, maybe a README
- Scenario B: Overview of components, API reference
- Scenario C: User documentation, tutorial

**Tip:** when you find yourself explaining something to someone else, also write it down as variable.

On the topic of documenting what goes into and comes out of functions (type safety), Lourens wrote a blog post on that (https://medium.com/escience-center/dynamic-types-static-types-oh-my-py-25c9743b72c4).

### Software Distribution

#### How to distribute this software?

Link (https://esciencecenter-digital-skills.github.io/research-software-support/modules/distributing/exercise-distributing)

Group 1 Case 2 - The Miracle Model:

- Is the code in a repo? Is there a DOI? Is there a README file? Is there a citation and licence doc?
- This is Scenario 1: make a script/notebook avaiable for download
- make environment files
- Users uses a package manager and run the script

Group 2 Case 1:

- What are the conditions under which you can use the program?
- How does the program work? Is it a script? A piece of software that needs web hosting?

### Software Publication

Link to the RSD exercise (https://esciencecenter-digital-skills.github.io/research-software-support/modules/publication/register_software_rsd_exercise)

Link to RSD playground (https://research-software.dev/)

### Software citation

Link to citation exercise (https://esciencecenter-digital-skills.github.io/research-software-support/modules/citation/create_citation_file)

CFFinit tool (https://citation-file-format.github.io/cff-initializer-javascript/#/)

## Collaborative Notes

### SMP (homework) feedback

- Some questions are too in-depth, like the quality check.
- Sometimes overwhelming.
- "I didn't think about it" is also a valid answer.
- Should it be as easy as possible or should people be more in depth?
- Who gets to decide the licence aor who is the owner? (On purpose not too many guidelines given, as this differs per institute.)
- How can software not be part of a project? (We assume there is one but we want you to spell it out.) So the question should be "What research project is the software part of?".
- How to go about new versions and the DOI?

- Code quality secure software: scientists don't really care about secure software? How to define secure software? (Is indeed not applicable for most research software, but could be when working with sensitive data or when you have a portal where users work in. It is important that you thought about it and "not applicable" is also a valid answer.)
- The example text in the text boxes disappear as soon as you type something and they are not copy-able.
- Would an AI chatbot help?
- Feedback button on the SMP tool would help, for now use training@esciencecenter.nl

SMP is intended to be a living document that can change over the lifetime of the software. It can help to sit down with researchers an explain/talk with them.

Kanban boards allow you to work together on tasks as they organise workflows and tasks. GitHub has a built-in Kanban board.

## Software Version Control

A commit is not a save state (you probably save more often than you commit) but a coherent unit of work with a dedicated commit message explaining what you did. Commits should form a logical history of what happened.

Git has hooks that allow you to run sripts on certain actions, e.g. run a script on each commit. GitHub and many other platforms also allow you to run certain actions/workflows on different hooks.

With SemVer, you don't have to restrict to numbers between `1` and `9`, so `6.22.100` and `0.40.0` are also valid.

When looking for dependencies, you can also look at the date of the last commit, which could indicate whether or not the software is still maintained.

It is a good idea to always have the main branching in a working state. You could hide away development in a dedicated branch.

Ideally, data is not put into (the same) version control as the software. You might version control it in a tool dedicated to data.

Sensitive data should never be put on the public internet, even if in a private repositoty, as usually foreign goverments can get access to this data.

**Tip:** prepend your branch name with the issue number you are working on, e.g. `15-add-documentation` if the issue has number 15.

## Software Testing

Testing also takes away fear of refactoring.

**Tip:** [Pure functions (https://en.wikipedia.org/wiki/Pure_function)] are easy to test.

## Software Documentation

## Software Distribution

Libraries are used by programmers (like NumPy).
Programs are used by end-users (with a point and click interface, like Firefox).

It depends on the package if the dependencies are declared with an exact version or with a range of verions (or no version at all, in which case typically the newest version is used).

Depending on the package manager, you can or cannot have multiple versions of the same dependency installed.

### Software licenses

Copyright laws can differ per country.
Software automatically is protected by copyright.
To distribute/use it, you need a license.

### Software publication

Registries enhance the findability of your software!

## Feedback

### Feedback on the SMP-tool

- Hoe ver/diep moet een onderzoeker gaan om het SMP in te vullen? Bij bv een vraag over aan welke software standaarden hij voldoet, kan hij dan gewoon zeggen ‘ik ben niet bekend met de standaarden’? Weinig onderzoekers zullen voor het SMP een studie willen gaan doen van die standaarden. (Mijn persoonlijke advies zou zijn het heel laagdrempelig te maken en te suggereren dat als ze het niet weten, dat ze gewoon iets kunnen invullen als 'dit is de eerste keer in mijn leven dat ik deze term hoor'. Uiteraard zou je ook willen motiveren dat mensen verder zoeken en meer leren voor een volgende keer, dus het is een design keuze tussen mensen die het voor de eerste keer doen niet afschrikken en mensen die al meer weten motiveren om dieper te gaan. Ik zou nu dus zelf kiezen voor niet afschrikken)
- Wie is de eigenaar van de software, dus wie mag de licentie bepalen?
- Bij kopje RESEARCH PROJECT CONTEXT. De vraag is nu: IS er een Research Project. Maar zou moeten zijn; WAT is het research project
- Hoe ga je om met nieuwe versies en (kleine updates) en een DOI? Via Zenodo lijkt iedere nieuwe versie een nieuwe DOI te moeten hebben? Waarom zou dat zijn? (NB: ik denk dat het het idee van Zenodo alleen de eindversie zet. Of iig stable versions die direct door andere gebruikers gebruikt zouden worden)

### Tips 

- 
- 
- 

### Tops

- 
- 
- 

## Resources

- Conventional commits (https://www.conventionalcommits.org/en/v1.0.0/)
- GitHub Actions (https://github.com/features/actions)
- GitHub workflows (https://docs.github.com/en/actions/how-tos/write-workflows)
- Can I pre-reserved a DOI before a GitHub release? (https://support.zenodo.org/help/en-gb/24-github-integration/73-can-i-pre-reserved-a-doi-before-a-github-release)
- Zenodo and version and concept DOIs (https://zenodo.org/help/versioning)
- Kanban board (https://en.wikipedia.org/wiki/Kanban_board)
- Test-driven development (https://en.wikipedia.org/wiki/Test-driven_development)
- Lesson material on testing & other good practices (https://carpentries-incubator.github.io/good-practices-lesson/)
- Intermediate software development lesson material (https://esciencecenter-digital-skills.github.io/python-intermediate-development/)
- Python template (https://github.com/NLeSC/python-template)
- R usethis package (https://usethis.r-lib.org/) templates for, among others, CI/CD setup on Github
- Functional Programming Paradigm (https://github.com/readme/guides/functional-programming-basics)
- TADA! The bare minimum for reproducible code (https://doi.org/10.32942/X2D93K)

### Lunch Discussion about Notebooks

Problem with Jupyter Notebooks: Presentation: I don't like Notebooks (https://docs.google.com/presentation/d/1n2RIMdmv1p25Xy5thJUhkKGvjtV-dkAIsUXP-AL4ffI/preview#slide=id.g362da58057_0_1)

Marimo Notebooks solve some of those problems: https://marimo.io/

## Post-workshop survey

https://www.surveymonkey.com/r/M8RPPX2