WI-Project: Open source project

Topics

Prof. Dr. Gerit Wagner
Faculty of Information Systems and Applied Computer Sciences
Otto-Friedrich-Universität Bamberg

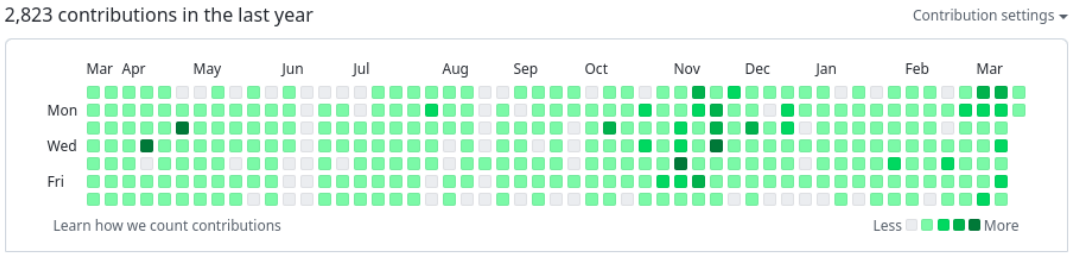
Your instructor

- Prof. Dr. Gerit Wagner
- At Bamberg University since October 2022
- I have worked with Git, Python, and R since 2014
- I enjoy coding, solving programming puzzles, and building tools
- My latest and most significant project: **CoLRev**

Gerit Wagner
geritwagner

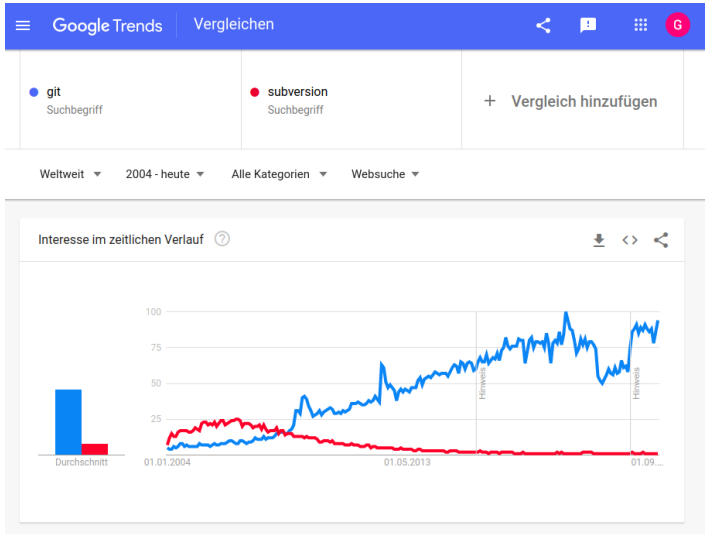
Edit profile

21 followers · **11** following



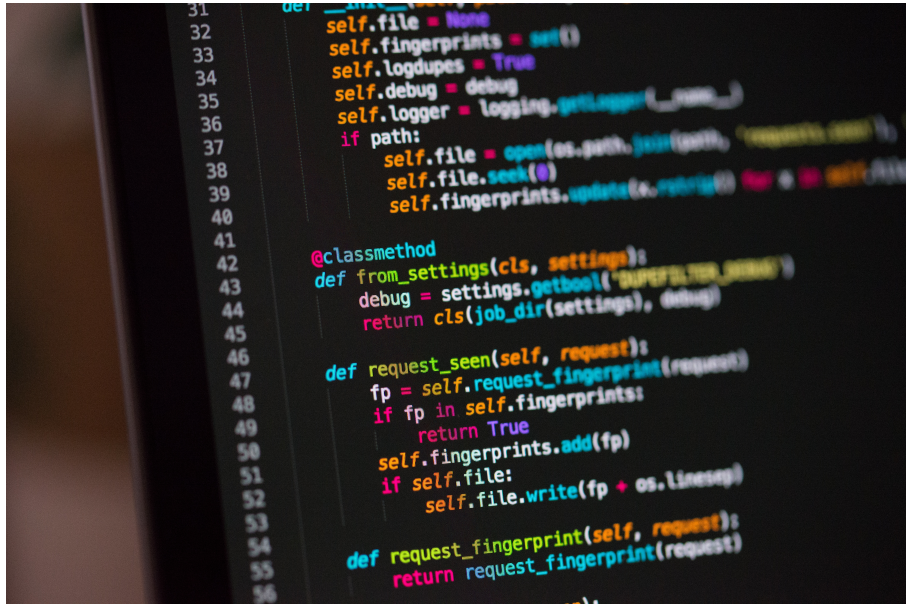
What you will learn (I): Git/GitHub

- Use Git and GitHub for versioning and collaboration
- Git quickly established itself as the most popular version control system
- As a prominent example, Microsoft has acquired GitHub and moved the source code of Windows to Git (1)
- Many companies have public GitHub programs and share parts of their work as open source (e.g., Google or Meta)



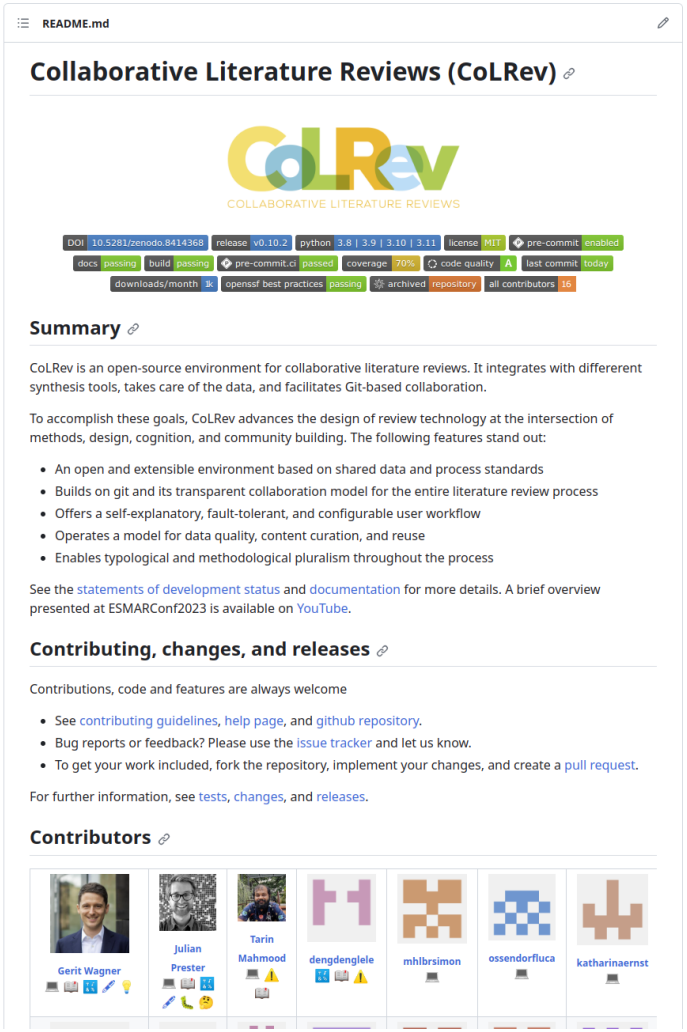
What you will learn (II): Python

- Write Python code and contribute to a Python package
- Python is the leading programming language in several rankings, including the TIOBE Index and the PYPL (PopularitY of Programming Language) ranking
- Job requirements for **Data Scientists**, **Full-stack Software Engineers**, **DevOps Engineers**, or **Data Engineers** commonly include Python (1)



What you will learn (III): Open-sourcing

- Open source software plays a major role in the industry (see HBS working paper) and firms are starting to use open source principles to organize their work (see innersource)
- You will
  - Work with the open source workflows of GitHub
  - Make a first contribution to a public open source project
  - Adopt the role of a maintainer and review the code of your peers
  - Have an opportunity to create your *developer portfolio*


README.md

Collaborative Literature Reviews (CoLRev)

COLLABORATIVE LITERATURE REVIEWS

DOI 10.5281/zenodo.8414368 release v0.10.2 python 3.8 | 3.9 | 3.10 | 3.11 license MIT pre-commit enabled
docs passing build passing pre-commit.ci passed coverage 70% code quality A last commit today
downloads/month 3k openssf best practices passing archived repository all contributors 16

Summary

CoLRev is an open-source environment for collaborative literature reviews. It integrates with differerent synthesis tools, takes care of the data, and facilitates Git-based collaboration.

To accomplish these goals, CoLRev advances the design of review technology at the intersection of methods, design, cognition, and community building. The following features stand out:

- An open and extensible environment based on shared data and process standards
- Builds on git and its transparent collaboration model for the entire literature review process
- Offers a self-explanatory, fault-tolerant, and configurable user workflow
- Operates a model for data quality, content curation, and reuse
- Enables typological and methodological pluralism throughout the process

See the statements of development status and documentation for more details. A brief overview presented at ESMARConf2023 is available on YouTube.

Contributing, changes, and releases

Contributions, code and features are always welcome

- See contributing guidelines, help page, and github repository.
- Bug reports or feedback? Please use the issue tracker and let us know.
- To get your work included, fork the repository, implement your changes, and create a pull request.

For further information, see tests, changes, and releases.

Contributors

Gerit Wagner | Julian Prester | Tarin Mahmood | dengdenglele | mhlbrsimon | ossendorfluca | katharinaernst

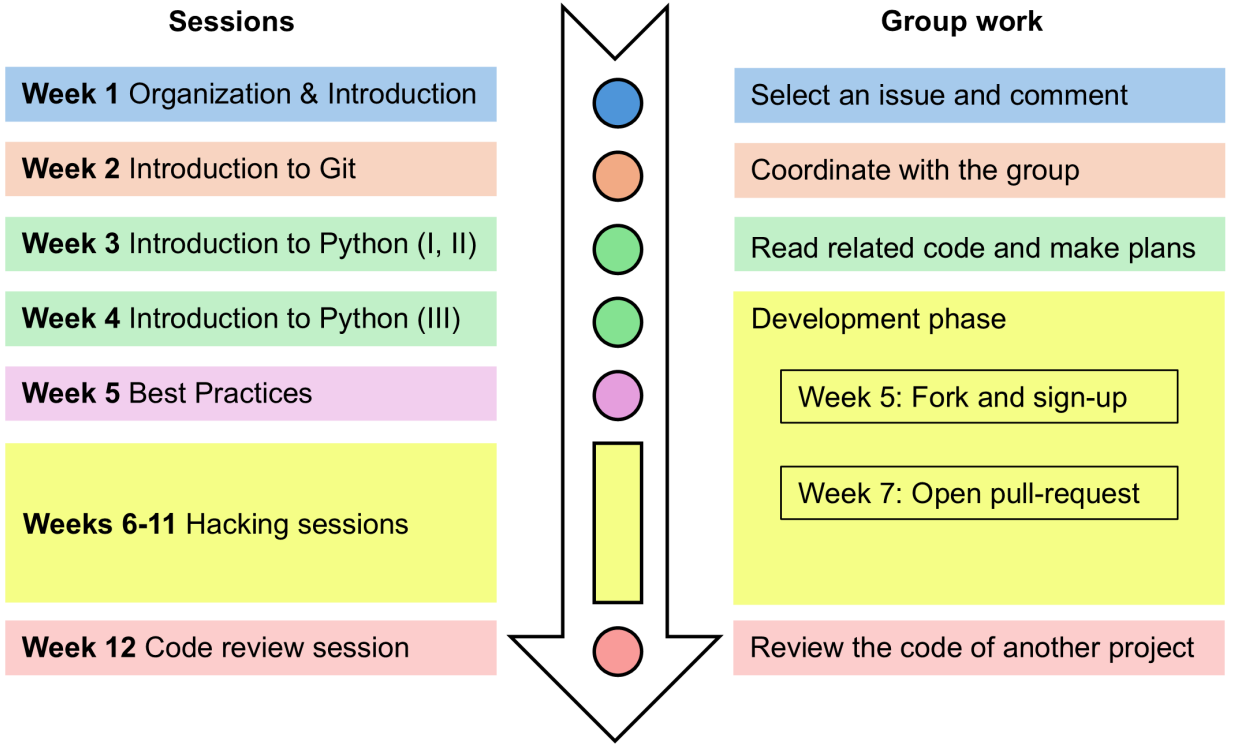
The project: CoLRev

Collaborative Literature Reviews (CoLRev) is an open source environment for collaborative literature reviews. It integrates with different synthesis tools, takes care of the data, and facilitates Git-based collaboration.

The following features stand out:

- Supports all literature review steps: problem formulation, search, deduplication, (pre)screening, PDF retrieval, preparation, and synthesis
- An open platform based on shared data and process standards
- It builds on Git and its transparent collaboration model for the entire literature review process
- Focus: command-line interface

Agenda



Dates and rooms are available online.

How groups will form

You **assign yourself to a group** when you contribute code, following these steps:

1. **Signal your intent** to contribute by joining the issue discussion on GitHub (in the issue discussion, before the **Git session**, for two topics at most).
2. **Organize your work**, meet potential group members, and select a project leader (in the following sessions). Keep in mind that **no more than 5 people** will be accepted per group.
3. **Officially sign up for a group** by contributing a non-trivial code part (*). Team members' contributions must be made in separate commits. Select a project lead who sends your GitHub ID together with a link to your code contribution, your student ID, and email address to gerit.wagner@uni-bamberg.de.

This process ensures that **all group members contribute fairly** and typically **a single grade is given for the entire group**. However, if issues arise, it is essential to raise concerns early. We reserve the right to adjust the grading policy as needed, including assigning different grades within the group and awarding a bonus to those who make exceptional contributions.

> - You can only contribute to a group if you have signaled your intent in the issue discussion beforehand.

Evaluation criteria

Process: Open source practices

- Follow established Git conventions for commit messages, branching, and pull requests.
- Actively engage in the community by reviewing code and providing and incorporating constructive feedback.

Code: Functionality

- Implement all required features with input validation.
- Provide unit tests that cover all critical functionality.

Code: Quality

- Comply with Python coding standards enforced by the repository’s pre-commit hooks.
- Include documentation, including docstrings and a README with setup and usage instructions.
- Ensure clear code structure and maintain high readability.

Evaluation scope

Evaluation is contingent on completing the deliverables listed here.

Evaluation components

- **Code submitted in the pull request (70%)**
  Assessment of functionality, overall code quality, and implementation of feedback.
- **Code review session (30%)**
  Focus on collaboration, version control practices, and understanding of code.

Not considered in the evaluation

- Training sessions (Topics, Git, Python)
- Hacking sessions

How you can make the project a success

- Embrace the challenge and adopt a problem-solving mindset
- Take full responsibility for setting up your programming environment
- Be prepared. Know your code, be able to explain it, and ask prepared questions (Google it, consider different options)
- Do not use generative AI such as ChatGPT (risk of copyright infringement)
- Adopt an open source approach (work publicly, communicate in English, create a profile)
- Reach out and schedule individual hacking sessions via Calendly to discuss challenges and get feedback

Finding things in an open source project

Go to the CoLRev project repository.

Form groups of three and try to find the following information in 10 minutes:

1. How many commits, contributors, and downloads does the project have? What is the test coverage?
2. How does CoLRev compare to related tools? Which tool would you choose and why?
3. Where is the documentation, and how can I install CoLRev?
4. What is the license, and where can we find information on how to contribute to the project?
5. What information should be provided for bug reports? Where can I open a feature request?
6. How many issues are open vs. closed? How many pull requests are open vs. closed?
7. How long do workflows with tests or code formatting run? When did the last one fail?
8. What is "unpaywall" used for in the project?
9. Where can we find the features planned for the next milestone?

Until the next session

Familiarize yourself with the CoLRev documentation

- An overview of the process and the workflow are available in the documentation.
- Watch the demo to understand the use of CoLRev.

Find a topic

- Go to the issue tracker and read the open issues tagged as `#good first issue` and `#search_source` (available topics).
- Comment on the issues to find others who are interested in the same topic.

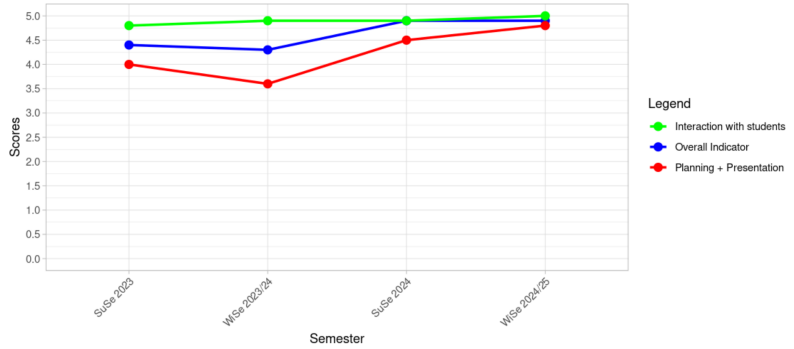
Complete the setup

- Register for the course on the *Virtual Campus* platform: here (password: OSP25#stud)
- Create a GitHub account (using your student email address) to use Codespaces in the next sessions
- Read how to use Codespaces

> Challenge (optional): You can **set up your programming environment** (see instructions in the CoLRev documentation). This requires you to install and configure Git, Docker, pre-commit hooks, and venv. It will give you more control over and a deeper understanding of your programming environment.

🛠️ We work hard to improve the course for you

- We run evaluations at the end of each semester and make the results publicly available (view results).
- We document how we implement your feedback and continuously improve the course (see ongoing updates).
- We invite you to contribute directly to our teaching materials by submitting an issue ♻️ or suggesting edits 🛠️.
- We actively engage in the pedagogical discourse in Information Systems by sharing teaching tips and inviting feedback from peers:



 Wagner, G., and Thurner, L. 2025. "Rethinking how we teach Git: Pedagogical recommendations and practical strategies for the Information Systems curriculum". Journal of Information Systems Education, 36(1). link Wagner, G., Thurner, L., Tang, C., Ott, S. "Teaching Python package development: A structured course with learning resources and an instructor's guide". Submitted to The Journal of Open Source Education. link Thurner, L. and Wagner, G. "CONTRIBUTE: A pedagogical framework for open