# Making Research Software Findable, Accessible, Interoperable, Reusable (FAIR) with Codefair

## Authors

Bhavesh Patel, <bpatel@calmi2.org>, FAIR Data Innovations Hub, California Medical Innovations Institute, San Diego, CA, United States, 0000-0002-0307-262X

Dorian Portillo, <dportillo@calmi2.org>, FAIR Data Innovations Hub, California Medical Innovations Institute, San Diego, CA, United States, 0000-0002-4306-4464

Sanjay Soundarajan, <ssoundarajan@calmi2.org>, FAIR Data Innovations Hub, California Medical Innovations Institute, San Diego, CA, United States, 0000-0003-2829-8032

## Abstract

Software developed as part of a scientific research process or specifically to support research is designated as research software.[1] It is developed for various purposes such as data visualization, data analysis, machine learning (ML), artificial intelligence (AI), and more. It is created and shared in different formats such as Python package, Jupyter notebook, or web app. Research software has progressively become a critical element of research projects. Recognizing its importance, the research software community and other stakeholders established the Findable, Accessible, Interoperable, and Reusable (FAIR) Principles for Research Software (or FAIR4RS Principles).[2] These principles provide a high-level framework for making research software optimally findable and reusable. We established the first minimal and actionable step-by-step guidelines that biomedical researchers can follow for making their research software FAIR, called the FAIR Biomedical Research Software (FAIR-BioRS) guidelines.[3] We are now also leading the Actionable FAIR4RS Task Force under the Research Software Alliance (ReSA) that is establishing domain-agnostic actionable guidelines for making any software FAIR.[4] Despite these efforts, the two-year FAIR4RS survey (2024) revealed that making software FAIR is not a common practice, with the lack of support cited as a barrier.[5]

To tackle these challenges, we are developing Codefair, a free and open-source (MIT license) platform designed as a personal assistant for making software FAIR.[6] Researchers simply have to install the Codefair GitHub app on their software's GitHub repo, and it then integrates seamlessly into their development workflow by leveraging GitHub tools such as Probot. Once installed, the Codefair GitHub app creates a GitHub issue that lists how the software complies with FAIR requirements. This issue is updated with each new commit to the main branch. From the issue message, users can go to the Codefair web app to easily address any missing requirements. Whether developing AI/ML models in Python, visualizations in Jupyter Notebooks, or analysis code in R, Codefair supports researchers throughout the software lifecycle. When we introduced Codefair at US-RSE 2024, it was in its early development days and only helped with including essential metadata elements such as license, CITATION.cff, and codemeta.json files, and with the archival of software on Zenodo. Codefair now supports automated validation of Common Workflow Language (CWL) files, maintenance of documentation, and provides a user-friendly web dashboard for easily addressing FAIR compliance across users' GitHub repositories. Additional features are being added to provide support for complying with coding best practices, archiving on Software Heritage, and addressing broader software management areas, including quality and security.

In our presentation, we will highlight the current features of Codefair, summarize progress since the presentation at US-RSE'24, discuss upcoming features, and explain how the community can not only benefit from it but also contribute to it. We believe Codefair is an essential tool for enabling software curation at scale and turning FAIR software into reality. It is now installed on over 400 repositories across 35 GitHub users and organizations. We believe this work is fully aligned with the US-RSE'25 topic of "Code, Practices, and People", and is particularly related to the "Code assist tools, templates, and frameworks" and "Best practices for research software engineering" topics. The conference participants will benefit greatly from this talk as they will learn about a tool that can enhance their software development practices.

## References

1. Gruenpeter, M. *et al.* Defining Research Software: a controversial discussion. Preprint at https://doi.org/10.5281/zenodo.5504016 (2021).
2. Barker, M. *et al.* Introducing the FAIR Principles for research software. *Sci Data* **9**, 622 (2022). https://doi.org/10.1038/s41597-022-01710-x
3. Patel, B., Soundarajan, S., Ménager, H. & Hu, Z. Making Biomedical Research Software FAIR: Actionable step-by-step guidelines with a user-support tool. *Sci. Data* **10**, 557 (2023). https://doi.org/10.1038/s41597-023-02463-x
4. The Research Software Alliance. Actionable FAIR4RS Task Force. https://www.researchsoft.org/tf-actionable-fair4rs/ (2025).
5. Honeyman, T. & Martinez-Ortiz, C. Update: FAIR4RS two year review. Preprint at https://doi.org/10.5281/zenodo.15381365 (2025).
6. *Codefair-App: Your Coding Assistant to Make Research Software Reusable without Breaking a Sweat!* (Github). https://github.com/fairdataihub/codefair-app