Not just for programmers: A friendly guide on the versatility/benefits of GitHub for accelerating collaborative research in Ecology and Evolution

This manuscript (permalink) was automatically generated from SORTEE-Github-Hackathon/manuscript@830a2c1 on January 28, 2022.

Authors

- **Dylan G. E. Gomes**
  0000-0002-2642-3728 · dylangomes
  Cooperative Institute for Marine Resources Studies, Hatfield Marine Science Center, Oregon State University, Newport, OR, United States

- **Cole B. Brookson**
  0000-0003-1237-4096 · colebrookson
  Department of Biological Sciences, University of Alberta, Edmonton, AB, Canada

- **Robert Crystal-Ornelas**
  0000-0002-6339-1139 · robcrystalornelas · rob_c_ornelas
  Earth and Environmental Sciences Area, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

- **Ali Guncan**
  0000-0003-1765-648X · Aguncan · aliguncan
  Department of Plant Protection, Faculty of Agriculture, Ordu University, 52200, Ordu, Turkey

- **Brandon P.M. Edwards**
  0000-0003-0865-3076 · BrandonEdwards
  Department of Biology, Carleton University, Ottawa, ON K1S 5B6, Canada

- **Kaitlyn M. Gaynor**
  0000-0002-5747-0543 · kaitlyngaynor · kaitlyngaynor
  Departments of Zoology and Botany, University of British Columbia, Vancouver, BC, Canada; National Center for Ecological Analysis and Synthesis, Santa Barbara, CA 93101, USA

- **Katherine Hébert**
  0000-0001-7866-6775 · katherinehebert · hebert_kat
  Département de biologie, Université de Sherbrooke, Québec, Canada

- **Emma J. Hudgins**
  0000-0002-8402-5111 · emmajhudgins
  Department of Biology, Carleton University, Ottawa, ON K1S 5B6, Canada

- **Saeed Shafiei Sabet**
  0000-0001-5919-2527 · shafieisabets
  Fisheries Department, Faculty of Natural Resources, University of Guilan, Sowmeh Sara, Iran

- **Eric R. Scott**
  0000-0002-7430-7879 · Aariq
  Department of Wildlife Ecology and Conservation, University of Florida, Gainesville, FL, USA

Abstract

Importance

Introduction

High-level/general background about GitHub

Contributors to this section: RCO, .

With over 73 million registered users, GitHub and its underlying version control system Git, are the *de facto* platform for collaboration on computer code [1]. GitHub has become an indispensable tool for software developers because, through version control, users can track changes to multiple files and folders over time [2]. Thus, users have an "audit trail" on the files they choose to store on their GitHub repositories which simultaneously less *ad hoc* than passing files back and forth yet able to scale up as projects take on more files or collaborators [3].

Researchers in ecology and evolutionary biology (EEB) are starting to collaborate on software as part of their research, and some are interacting with GitHub for the first time [4]. For first-time users, the GitHub learning curve can seem overwhelming because the creation of the platform and its features were initially centred in collaboration for software development [CITE]. However, by leveraging existing tools on GitHub and the wide range of collaborations they can enable, researchers in EEB can make the most out of their research and collaborative projects.

What's already been written about GitHub

Contributors to this section: RCO, PHPB.

Git is the version control system that enables all the collaborative tools available on GitHub. In Git, changes performed to files are registered as uniquely identified "commits", which are a snapshot of the line-by-line changes that have been voluntarily performed at that moment. Because the details of interacting with Git can get very technical very quick, we focus instead on the web platform GitHub. However, we suggest those interested in Git explore the many papers [5] [6] and books [7/] that can provide an introduction to git. Despite the prevelance of technical papers and books that focus on Git or GitHub for the software development community, there are much fewer resources for EEB researchers who want to begin collaborating through GitHub. We acknowledge that GitHub is not the only way for productive collaboration on cloud-based research documents and code, so we encourage researchers in EEB to take the elements of a GitHub that fit into their workflow. Because GitHub as web platform is so well documented and has a robust user community, scientists can take advantage of many collaborative aspects without knowing even a line of Git code.

What's already been done with GitHub in EcoEvo

Contributors to this section:

Very friendly description of what GitHub is and the main uses and advantages of using it in the natural sciences back in 2016 (Perkel 2016). What's missing about GitHub in EcoEvo and our objective: Introducing the GitHub ecosystem that's composed of many different elements!

What's missing about GitHub in EcoEvo and our objective

Contributors to this section:

Simple habits (of which github is one component) can do a lot to make research more reproducible and collaborative (Alston and Rick 2021). In EcoEvo Github use is predicated on an understanding in R. This close connection has some benefits, but other programming languages are frequently used by researchers (e.g. Python, Julia). Lots of ways to use GitHub that are independent from R. We have in this hackathon a definite focus on R tools for interacting with GitHub, but sometimes the issues we present at 'Github' issues might be more about the ways that we interact with Github (i.e. through R vs. bash shell)

I am just testing what happens if I add a sentence. (Saeed) :) It is also important to consider what are the aims of researchers and how they want contribute in...

Box 1: Definitions

- **repository**:

- **commit**: Commits are like snapshots in the development of a project. Commits can include changes in multiple files and must include a brief commit message describing the changes made. A typical workflow is to make some related changes in files, make a commit (e.g. "generate and include fig1 in results"), and after several commits to **push** those commits to the remote GitHub **repository**.

- clone: Cloning a **repository** is a way of making a local copy (i.e. on your computer) of a GitHub **repository**. If you have access to **push** to a **repository**, this can be a first step to contributing to a project.

- branch: Development branches can be created at any point in time and work on each branch can continue independently. This is useful for testing out new ideas (both code and text) which may or may not eventually get integrated into the main branch of the project. Branches can also be used to isolate contributions of multiple contributors. Each person working on their own branch eliminates problems that arise when conflicting edits are pushed to the same branch. Changes in a development branch can be merged into the main branch via **pull requests**. Branches can only be made by those who are given access to the project **repository**.

- fork: A fork is a copy of a **repository** hosted on GitHub. If a repository is public, then anyone can make a fork. Even if they do not have access to push to the original repository, they can make a fork and edit it independently. Forks are linked to the original GitHub repository and "upstream" changes (those in the original repository) can be merged to keep the fork up to date with the original project. Changes made in the fork can be integrated into the original project via **pull requests**.

- push/pull: When **commits** are made in a project locally, they must be synced with the remote GitHub repository by "**pushing**" them. Changes on a GitHub repository can then be "**pulled**" to keep your local version of the project up to date.

- pull request: A pull request is a request that the owner of a GitHub repository integrate changes you've made on either a **branch** in the repository or in your own **fork**. When you initiate a pull request, you must provide a description of what changes are made. Some automated tests may be run and review may be required before integrating your changes.

-

GitHub in EcoEvo examples (Part 1)

Storing and archiving version-controlled data

contributors to this section: Dylan Gomes

Many researchers often start their use of GitHub to backup their working data and code to a remote server (Just push and pull, see Box 1, from their own repo). This saves the user time from backing up data and code on their own portable devices, such as hard drives. This also offers some peace of mind, as this information is retrievable even if one's laptop ends up at the bottom of a lake. Thus, an additional benefit of this 'cloud' storage is that one's GitHub repository can be accessed by any machine with internet access, allowing the user to be more mobile if they wish to both work from home and the office from different computers. Each time a user pushes changes to their repository, GitHub tracks what these changes are and stores this history. This feature allows for version control, such that users can re-visit previous versions of data and code. This is particularly useful if a mistake has been made where a user has unknowingly overwritten or deleted information that would otherwise be irretrievable without GitHub having saved that information.

GitHub also integrates with Zenodo, a free, long-term data archiving service funded by CERN. After linking your GitHub account to Zenodo and turning on archiving, any time a release (Box 1) is made, a snapshot of the entire repository is archived in Zenodo with a versioned, citable DOI (see `### Making code citable` below for more information).

Virtual lab notebook

contributors to this section: commits as a way to record daily progress issues as a way to keep track of short-term objectives/goals, and progress towards them

Classroom teaching / educational materials

contributors to this section: Cole Brookson

GitHub provides a large variety of uses for hosting teaching/educational materials. In fact, through taking advantage of the suite of GitHub features, the entire process of running a course, workshop, or even just a lecture, can all be done openly on GitHub. As a matter of gross simplification, organizing a course (for example) could be broken down into: 1) developing the material (i.e. slides, examples, relevant readings, labs, etc.), 2) hosting the course on some online platform for students to access, 3) delivering the content, and 4) dealing with student submissions and the subsequent grade returns. While of course there are other purpose-built platforms for this type of activity, few of them provide the usability at the price point GitHub does. First, developing your course material, from slides to labs and everything in between, can be done on GitHub, out in the open, where others can see, review and offer feedback on your process! Making presentations can be done through most major high-level programming languages such as R, with `RMarkdown`, Python, with `python-ppt`, and Julia, with `Remark.jl`. Since all these programs work via code bases, they can be version-controlled through git and GitHub. Once you've made all the content for your course, hosting a course website can be done through GitHub pages, and there are lots of templates available to borrow from. This way, not only can the course content be available to your enrolled students, but also to anyone interested in the course material. Since the course material can be easily housed on a GitHub pages website, it is then simple enough to deliver the content via that website, and/or a GitHub organization with template repositories for assignments etc. Student submissions are perhaps the least seamless component, but for assignments submitted as code files (i.e. `.R` & `.Rmd` as two of the most common) and/or `.pdf` files, GitHub has a new and far-from-perfect but still useful tool GitHub classroom where instructors can host private assignments, and even build custom autograding tests, that will autograde assignments!

The previous section is meant to highlight the myriad tools GitHub can provide to centralize the delivery of educational materials. While most instructors will likely choose to pick from this selection and end up having a mix of tools to deliver their content to students, it is still valuable to utilize some of these, if only for the reason that it can encourage students to even *begin* learning about version control through interacting with git/GitHub, however minimally, through the course. There are (as always) no "points" awarded for using ALL GitHub materials ALL the time, but if a central tenant of a given course or educational unit is to introduce or give students experience to version control and the tools that working professionals in the biological sciences use, then adopting a few of these tools can be a great way to do so.

Matthew D. Beckman, Mine Çetinkaya-Rundel, Nicholas J. Horton, Colin W. Rundel, Adam J. Sullivan & Maria Tackett (2021) Implementing Version Control With Git and GitHub as a Learning Objective in Statistics and Data Science Courses, Journal of Statistics and Data Science Education, 29:sup1, S132-S144, DOI: 10.1080/10691898.2020.1848485

GitHub in EcoEvo examples (Part 2)

Project management

Contributors to this section: Kaitlyn Gaynor, Rob Crystal-Ornelas

GitHub can be a powerful tool for team-based project management, allowing collaborators to share feedback, brainstorm ideas, and troubleshoot problems. The "Issues" feature of GitHub allows for discrete tasks and sub-tasks to be identified, assigned to team members, and categorized with custom labels, and the new "Discussion" feature serves as a message board for conversation. Scripts, commit messages, and pull requests can be linked directly to issues and discussions, providing a clear record of project workflow. The use of GitHub for all project-related conversation and planning, rather than e-mail or messaging tools, makes it easier to keep track of progress throughout the lifespan of a project and less likely for issues to slip through the cracks. It is not essential for all team members to have proficiency in git or programming, as users can interact with Issues and Discussions via web browser or e-mail. By default, GitHub repositories are publicly visible, and so anyone with a GitHub account can not only view content, but also engage with repository administrators through Issues and Discussions.

Can talk about ESS-DIVE's project management using ZenHub/Jira to manage customer support requests, feature updates to our data sharing platform.

Building website

Contributors to this section: Emma Hudgins

Seems like the technical aspect of this is discussed in Dawson, Chris (2016). Building Tools with GitHub: Customize Your Workflow. O'Reilly Media GitHub pages allows any .html document to be rendered as a website with a URL. This could be, for example, a report written in markdown or R Markdown rendered into a .html file.

Jekyll and Hugo are both "static website generators", which you can use as template libraries for websites that can be hosted freely via GitHub pages. Both of these tools require some additional learning because they are deployed locally via the terminal, but they are a great resource for creating free, eye-catching websites. If you wish to use your own domain name, you can purchase a domain for your GitHub pages site. It is also possible to fork the website of someone else who has publicly hosted their website on GitHub in order to use it as a template.

Making code citable

Contributors to this section: Rob Crystal-Ornelas, Emma Hudgins
GitHub makes it easy to store and share a variety of data files in the cloud. But for a variety of reasons (e.g., privately owned company, ability to make repositories private, accounts can be deleted at will) GitHub is not considered a long-term data or code repository like zenodo and figshare [8] [6]. Also, unlike the long-term repositories, GitHub does not issue Digital Object Identifiers (DOIs) for content uploaded to their servers. DOIs are persistent and unique alpha-numeric IDs assigned to research products like papers, code, and data. DOIs allows tracking and citing research products. For this reason, scientists who share code and data through GitHub are strongly encouraged to also submit GitHub repository content to a long-term data archive [9]. Fortunately, both long-term repositories mentioned above (zenodo and figshare) have integrations with GitHub which facilitates archiving a snapshot of all repository content with the click of a button.

GitHub Linking with Zenodo, etc. to achieve a DOI helps work become findable, gives proper attribution, and that can ensure long-term stability (Hampton et al. 2015) Another key step researchers should consider taking when they receive a DOI for the content they archive on GitHub is choose a usage license [10]. This helps

The standard GitHub licensing options are best suited for software. If your code is intended only for your specific analysis, consider a Creative Commons License. The Choose a License website can offer further guidance. If you wish to allow anyone to re-use your code, consider a CC0 1.0 public domain dedication. If you wish to receive attribution for any reuse of your code, consider a CC BY 4.0 license, which requires attribution upon reuse.

Many researchers believe that their code is not useful because their analysis is context-specific and not designed for re-use like software. However, even if code is rough, it shows the exact steps taken to conduct an analysis, and therefore provides the most detailed look into how to reproduce a given analysis [11].

Collaborative (code) editing

Contributors to this section: Kaitlyn Gaynor, Rob Crystal-Ornelas

From its inception, one of the primary uses of GitHub has been for collaborative coding. We acknowledge that the average software developer and EcoEvo researcher using GitHub

Is it worth walking through how collaborative code editing works through GitHub, or just pointing to all the available resources for this? (e.g. https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/incorporating-changes-from-a-pull-request/merging-a-pull-request). Rob: I think pointing to available resources and citing them here is great [10].

GitHub can facilitate interactions between research advisors and advisees, providing a platform for students or other trainees to share in-progress code, and flag specific challenges or questions for their supervisors or mentors. Periodic code review can also help advisors to identify errors early in the process, and inform further training and mentorship to fill gaps in skills.

Writing manuscript

contributors to this section:
Caveat that GitHub has been called out for not being so user-friendly for manuscript development (Ram 2013). But getting better? Tools that link with GitHub have been developed with synchronous

writing in mind. HackMD provides a collaborative writing platform based on Markdown that integrates with GitHub.
We used this platform early on in the process of writing this manuscript to generate an outline.

Peer-Review

Peer review of research software by rOpenSci (https://ropensci.org/software-review/) and of research software and associated manuscripts by the Journal of Open Source Software (https://joss.readthedocs.io/en/latest/submitting.html) requires that submitted work is hosted on GitHub and their review processes make use of GitHub issues (Box 1). GitHub can also be used as a hub for reviewers and authors during the peer review process of an ordinary research manuscript. If the code associated with a manuscript is made available at the time of submission (e.g. via a link to a GitHub repository in a Data Availability Statement), peer-reviewers may be able to offer more helpful suggestions on written methods and may even make comments on the code itself, potentially catching bugs or errors before publication . GitHub issues (Box 1) can also be used to organize and discuss reviewer suggestions and to assign them to co-authors (See example here). When reviewer comments are posted as separate issues, authors can comment on the issues to discuss possible changes and assign themselves to indicate which comments they intend to handle. Co-authors can then integrate their edits and responses to reviewers using pull requests (Box 1).

GitHub in EcoEvo examples (Part 3)

Open science discussion

Contributors to this section: https://github.community/ GitHub new discussion tool

Project continuity

Contributors to this section: 'thus preserving the long-term integrity of the project even as collaborations form and shift.' (Hampton et al. 2015) better to have old versions on GitHub than on somebody's personal hard drive!

Asycnhronous working

Contributors to this section:

GitHub organizations

Contributors to this section: Katherine Hébert, Cole Brookson

Whether experiments are done in a wetlab, data are gathered in a field site, or analyses are run in a shared office, even conceptually distinct projects are often carried out in a common physical space. GitHub Organisations offer a shared virtual space that allows a team to work in different repositories, while remaining tied together under a larger figurehead, such as a laboratory, a department, an organisation, or a large project involving several teams. Organisations are well-suited to ensure larger projects with many steps or moving parts are constrained to one virtual space, where outputs and sub-projects can be easily accessed and located without relying on any one individual. Because the repositories are grouped in one virtual space, members can reference and contribute to each other's work without necessarily being part of the same repository, broadening the accessibility and longevity of code and writing contributions.

Contributors can be assembled into teams within an organisation, which allows administrators to assign roles and tasks to groups of people. Whereas access to repositories is usually assigned to individual contributors, Organizations facilitate the management of access permissions by allowing each team to be granted access to certain repositories, and not to others. This ensures that more sensitive repositories remain as restricted as needed, while repositories with greater general interest can be easily accessible to many members at once.

As an example, GitHub Organizations are particularly well-suited to house documents and projects within a laboratory, such as research compendia, codes of conduct, protocols, training documents, and other such documents that evolve collaboratively over time and are relevant to many colleagues. In this way, students or teams can have full ownership of repositories within an organization, while ensuring that these materials stay accessible to the laboratory after people have moved on (or upgraded their computers). This application extends to research centres, which may include several distinct projects that remain linked under a given institution, such as the German Centre for Integrative Biodiversity Research (iDiv). Of course, the utility of this tool goes beyond laboratories - they are useful to structure the organisation, presentation, and outcomes of working groups such as the hackathon which inspired this paper (SORTEE-Github-Hackathon) by keeping track of all materials as ideas develop and take shape in one virtual space. Organisations are also convenient for hosting a set of related learning materials such as a set of lectures or workshops, such as the Québec Centre for Biodiversity Science R Workshop Series (QCBSRworkshops) or the University of Edinburgh's Coding Club (Coding Club), which may be updated by an ever-evolving group of contributors over time.

Utilizing GitHub organizations as a research group or even for a handful of individuals working on a group of projects can be incredibly useful for all involved. GitHub organizations are relatively easy to set up, and especially easy to manage as membership to the organization changes through time. Not only is it a useful way to store repositories of lab-related research products, but it's also incredibly helpful for storing "living documents" that may be edited frequently, and may be linked to a lab website (that could also be generated via a repository that lives within the organization!). The use of the "Teams" feature can allow certain groups to have varying levels of access to repos in the organization with a select group having push access to some repos but not others. This can manifest in a group working on some common dataset(s) (e.g. some genetic data) to have push access to the handful of repositories used for processing sequence data, while another group of students/researchers may have push access to an entirely different set of repos. The organization structure also allows for easy tracking of issues, projects, and discussions related to the research group, and provides PIs/group leads an easy birds-eye view of the progress going on across multiple projects.

As well, organizations provide a convenient location for students to archive the code for their projects, for use/reference by future students in the research group, thus providing a type of knowledge communication that may not exist otherwise. Indeed, providing new students with access to the organization and ideally a template repository for lab projects can soften the burden on those new to the software, in that it provides them with examples to work off of, and an online location to ask for help from their labmates and/or advisors through tools like projects, discussions, and issues.

Misc other uses

Contributors to this section: RCO There are many more potential uses of GitHub for EEB researchers, and we briefly highlight several of them here. First, community-driven data standards include instructions and templates that can help researchers format their data and metadata more consistently [12]. Often, these documents and templates are hosted on static websites as PDFs. However, GitHub is now seen as a useful site for storing the data standard documents since they can be version controlled, and commented on by the user community e.g., ESS-DIVE's GitHub Community Space [9].

Second ...

Code review rOpenSci's code review process, and also caught mistakes in code of published papers that could have been caught in peer code review. Also maybe say something about ReproHack.

Discussion

General pargrah on what GitHub can enable in EcoEvo

General paragraph on GitHub on how, given all the potential uses of GitHub, it can enable more collaborative EcoEvo research Despite all the awesomeness of GitHub, there are still plenty of times when you might look to other plantforms for collaboration

Why aren't more people using GitHub?

Learning to use Github requires time, but the payoff is *[may be?]* worth it. Time vs. effort examples or analyses to demonstrate the payoff can help drive the point home to convince people to learn these tools

Limitations

Our own limitations since we are mostly writing from the EcoEvo perspective/ additional github limitation Reliance on R since we are generally in EcoEvo Discussion of free vs. paid plans. When projects get highly collaborative may have to add / pay for accounts. At this point, little difference between paid and free.

Using GitHub is a good start, but lots of practices to make repo more user friendly

end off with our 5/10 tips for how to gain knowledge/practice with GitHub here

Conclusion

Acknowledgements

This manuscript arose from a hackathon at the Society for Open, Reliable, and Transparent Ecology and Evolution (SORTEE) virtual meeting in 2020.

(add funding as needed!)

Code and data availability

The source code and data for this manuscript are available at https://github.com/SORTEE-Github-Hackathon/manuscript.

References

1. **Build software better, together**
   GitHub
   https://github.com

2. **Excuse Me, Do You Have a Moment to Talk About Version Control?**
   Jennifer Bryan
   The American Statistician (2018-04-24) https://doi.org/gdhzdp
   DOI: 10.1080/00031305.2017.1399928

3. **Git can facilitate greater reproducibility and increased transparency in science**
   Karthik Ram
   Source Code for Biology and Medicine (2013-02-28) https://doi.org/krv
   DOI: 10.1186/1751-0473-8-7 · PMID: 23448176 · PMCID: PMC3639880

4. **Our path to better science in less time using open data science tools**
   Julia SStewart Lowndes, Benjamin D Best, Courtney Scarborough, Jamie C Afflerbach, Melanie R Frazier, Casey C O'Hara, Ning Jiang, Benjamin S Halpern
   Nature Ecology & Evolution (2017-06) https://doi.org/gc4jb3
   DOI: 10.1038/s41559-017-0160 · PMID: 28812630

5. **A Quick Introduction to Version Control with Git and GitHub**
   John D Blischak, Emily R Davenport, Greg Wilson
   PLOS Computational Biology (2016-01-19) https://doi.org/gbqsnf
   DOI: 10.1371/journal.pcbi.1004668 · PMID: 26785377 · PMCID: PMC4718703

6. **Ten Simple Rules for Taking Advantage of Git and GitHub**
   Yasset Perez-Riverol, Laurent Gatto, Rui Wang, Timo Sachsenberg, Julian Uszkoreit, Felipe da Veiga Leprevost, Christian Fufezan, Tobias Ternent, Stephen J Eglen, Daniel S Katz, … Juan Antonio Vizcaíno
   PLOS Computational Biology (2016-07-14) https://doi.org/gbrb39
   DOI: 10.1371/journal.pcbi.1004947 · PMID: 27415786 · PMCID: PMC4945047

7. **Let's Git started | Happy Git and GitHub for the useR**
   Jenny Bryan Hester the STAT 545 TAs, Jim
   https://happygitwithr.com/

8. **Democratic databases: science on GitHub**
   Jeffrey Perkel
   Nature (2016-10-06) https://doi.org/gdz6dq
   DOI: 10.1038/538127a · PMID: 27708327

9. **A Guide to Using GitHub for Developing and Versioning Data Standards and Reporting Formats**
   Robert Crystal-Ornelas, Charuleka Varadharajan, Ben Bond-Lamberty, Kristin Boye, Madison Burrus, Shreyas Cholia, Michael Crow, Joan Damerow, Ranjeet Devarakonda, Kim S Ely, … Deborah A Agarwal
   Earth and Space Science (2021-08) https://doi.org/gmbs9c
   DOI: 10.1029/2021ea001797

10. **Barely sufficient practices in scientific computing**
    Graham Lee, Sebastian Bacon, Ian Bush, Laura Fortunato, David Gavaghan, Thibault Lestang, Caroline Morton, Martin Robinson, Philippe Rocca-Serra, Susanna-Assunta Sansone, Helena

Webb
Patterns (2021-02) https://doi.org/gjpcb6
DOI: 10.1016/j.patter.2021.100206 · PMID: 33659915 · PMCID: PMC7892476

11. **Elevating The Status of Code in Ecology**
KAS Mislan, Jeffrey M Heer, Ethan P White
Trends in Ecology & Evolution (2016-01) https://doi.org/gg43mk
DOI: 10.1016/j.tree.2015.11.006 · PMID: 26704455

12. **FAIRsharing as a community approach to standards, repositories and policies**
the FAIRsharing Community, Susanna-Assunta Sansone, Peter McQuilton, Philippe Rocca-Serra, Alejandra Gonzalez-Beltran, Massimiliano Izzo, Allyson L Lister, Milo Thurston
Nature Biotechnology (2019-04) https://doi.org/gfxsq8
DOI: 10.1038/s41587-019-0080-8 · PMID: 30940948 · PMCID: PMC6785156