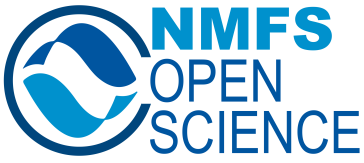
GitHub and Git

Guidance and Best Practices for NMFS Users
version 3.3

NMFS Open Science GitHub working group



1 Introduction

This "best practices" guide was developed by and is maintained by the NMFS Open Science GitHub working group (Section 6). This document provides practical help and guidelines for the wide variety of ways that Git and GitHub are used within NOAA Fisheries. This guide was heavily influenced by the work from the Fisheries Integrated Toolbox team who developed standardized GitHub "best practices" documents for NMFS tool developers, by the GitHub SOPs developed by SEFSC, NEFSC and NWFSC, and other guidance documents. See references (Section 7).

> **!** Important
>
> NOAA's official 2017 CIO memo authorizing GitHub use at NOAA is here. Note, this was written before GitHub Enterprise existed so refers only to GitHub Public. A NOAA GitHub policy document from 2017/8 may be found here, however this document appears to be for content on NOAAGov (top of page 2) and it is unclear which parts are general and which are specific to contributing to NOAAGov. Some individual centers and offices have their own GitHub Guidelines and SOPs. In addition, your division or program may already have a GitHub organization and have a SOP for contributing to that. This document is not intended to replace these SOPs but to supplement them and help NMFS staff follow them by giving pragmatic guidelines.

1.1 Background

In 2017, GitHub use was authorized for scientific products: CIO memo authorizing GitHub use at NOAA. Subsequently, in 2017, NOAA released official GitHub guidelines here. Some individual centers and offices prepared local GitHub guidelines and SOPs to help their staff interested in using GitHub. See references (Section 7). Since 2017, a lot has changed. GitHub use has expanded greatly across NMFS and all the federal agencies (link), and GitHub has become integrated into modern scientific workflow for a wide variety of reasons. The 2017 memo tells us what can be shared on GitHub (scientific products) but there is a need for guidelines that cover and recognize the wide variety of scientific work (and learning) that is done on GitHub and much has changed in the tools provided by GitHub since 2017. Local GitHub SOPs have been developed by individual centers (see references Section 7) to help their staff. These are very helpful, but not all centers have GitHub SOPs and even with these, staff are often confused by how to follow the guidelines in practice. For example, if I fork a repo in statistics workshop I am in, am I supposed to have a gold standard backup for that? What does it mean to

keep NOAA work separate? I was told that private repos are not allowed, but someone at another center said *public* repos are not allowed? I work with confidential fisheries data; what are my options? I am developing an R or Python package with functions for making common plots but this is completely generic; what are my options? I am collaborating with a regional office on an assessment that will be released as an official report from my center; what are my options?

> **Note**
>
> When the 2017 memo was written, GitHub was not Fedramp authorized. GitHub Enterprise became Fedramp authorized in 2018 and GitHub use expanded rapidly in other agencies (e.g. GSA and VA) using Enterprise. A number of the NOAA Fisheries science centers have a GitHub Enterprise account. The 2017 memo does not distinguish between GitHub (free) and GitHub Enterprise (paid). You can think of GitHub Enterprise as a secure private GitHub environment that you log into but functions otherwise mostly like the free GitHub once you are inside.

1.2 Scientific products

These guidelines are intended for scientific products that are low FISMA. Scientific products are not generally considered official communication or business, and a disclaimer is added to GitHub repositories to state this. However, those who are handling higher FISMA data, relying on GitHub as the main data repository for official data, or working on products that will be official communications will need to seek further guidance for using GitHub.

2 Guidelines for Use of GitHub at NOAA Fisheries

The information here is intended to provide employees and affiliates of NOAA Fisheries (NMFS) with practical guidance and "best practices" for how to use GitHub. "NOAA allows use of GitHub to share code and content in the spirit of collaboration and open government." (memo). NOAA has a strong history of scientific collaboration, coordination, and close engagement with other government partners, non-government organizations, academic institutions, international colleagues, and other members of the scientific research community.

2.1 Glossary

NOAA Branded Product - NOAA Branded Organizations are created by a science center, regional office, division, or program for its official products. These GitHub

organizations are distinguished by clear branding that indicates that the repositories are official agency products.

GitHub Organization (org) - Organizations are shared accounts where members of the org can collaborate across many repositories at once. Owners and administrators can manage member access to the organization's data and projects with sophisticated security and administrative features. Org in this context is not related to organizations on an org chart. It is more like a managed folder of folders.

Repository - A repository is hosted online and contains all of a project's files and each file's revision history.

2.2 What Content Can Be Shared on GitHub?

Generally, content on GitHub is limited to NOAA's scientific products as defined in the NOAA Scientific Integrity Policy. This policy defines such products as "The results of scientific activities including the analysis, synthesis, compilation, or translation of scientific information and data into electronic and hardcopy formats for the use of NOAA, the Department of Commerce, or the Nation. These products include, but are not limited to, experimental and operational models, forecasts, graphics, and verbal and written communications of all kinds relating to scientific activities, including NOAA social media accounts."

The open source nature of GitHub allows content to be available for other developers to build upon or contribute to via fork, clone, or pull request. Embracing this open source workflow facilitates open review by allowing others to comment and offer solutions for open issues, improving bug reports by allowing users to see source code, and providing the full history of the project changes (i.e., version control, usually Git). Note, "open source" is not equivalent to making content publicly accessible. The level of visibility of a repository to the general public is a separate decision and is project dependent.

3 Account Guidelines

3.1 GitHub Personal Account Settings

To collaborate with colleagues and contribute to open science and open government over GitHub, you will need a GitHub account. This will allow you to create GitHub repositories, participate in GitHub organizations, use version control with GitHub, fork or clone repositories, contribute to other GitHub repositories, among other features.

- There should be clear separation between any use of GitHub associated with your NOAA activities and non-NOAA activities. You are not required to append "-NOAA" to your NOAA GitHub username, but it may be helpful for distinguishing between accounts if you use GitHub for NOAA work and non-NOAA work[1].
- When creating an account on GitHub.com use your @noaa.gov email as primary (under Settings/account) and as the notification email (under Settings/notifications). The latter ensures that notifications on work related repositories are sent to your NOAA email, not to a personal or university email address.
- Fill out your profile with your name and NOAA affiliation.
- Provide a real photograph of yourself as your GitHub profile.
- Enable Two-Factor Authentication.

> **Note**
>
> Your NOAA supervisor should be aware of your use of GitHub and have a clear understanding of what content is being shared on GitHub.

3.2 GitHub Repository Guidelines

GitHub provides a platform to host official work products, however GitHub repositories are used for a variety of purposes and not all repositories are "products". Repositories are also used for project management, development, training, and testing out ideas.

All repositories, regardless of purpose, must follow these general guidelines:

- PII and BII should not be shared (on purpose or inadvertently) on GitHub regardless of whether the repository is in a private or public repository. Best practices and safeguards must be followed to prevent this.
- No sensitive information should be shared in repositories. Sensitive information includes, but is not limited to, usernames, passwords, login information, port numbers, IP addresses, server names, Application Programming Interface (API) keys, Personally Identifiable Information (PII), Business Identifiable Information (BII), or confidential data.
- GitHub is not a back-up service nor is it a data repository. Other tools are designed for this purpose. At the minimum, this means maintaining a clone on at least one government furnished laptop or server. *Managers of NOAA branded GitHub organizations should work with their IT department to ensure automated regular backups of the organization repositories.* In other guidelines, this is often referred to as a "gold standard copy".

[1] Work when not employed by NOAA or work that is not part of your job and you are not paid for this work by NOAA.

- Only scientific content (Section 2.2) that can be reasonably classified as FISMA Low (Section 4.1) should be shared on GitHub.
- Repositories that have code that interacts with APIs using IP addresses, usernames, passwords, secrets, or credentials must take steps to prevent committing of "secrets" to GitHub. (See Section 4.2).

3.3 Disclaimers and Licenses

If your repository represents something you produced in the course of your work at NOAA and could be considered a "work product", then your repository should include a README (Section 5.1), the government product DISCLAIMER (Section 5.2), and LICENSE file (Section 5.3). If the work is something that can be cited, then also include citation information and a DOI[2].

3.4 Repositories Under Individual Accounts

GitHub repositories can be created under your individual GitHub account or under a GitHub organization that you are a member of. If you have NOAA work products[3] hosted under your individual GitHub account, you need to do the following to ensure your NOAA work is transferred to a new NOAA owner when you leave NOAA.

- The repository ownership must be transferred to another NOAA individual GitHub owner at NOAA or to a NOAA GitHub organization when an individual leaves NOAA.
- Under Settings, specify a successor who can take over your account if you are unable to transfer your repositories.

What about a repository associated with a journal article if writing this was part of your NOAA job? Transferring that kind of repository would probably not make sense. The transferring policy is for things that someone who is on-boarding for your job will need. Think about the kinds of work you would transfer off your computer during off-boarding.

> GitHub Organizations
>
> You can also create your work repositories in a GitHub organization rather than your individual GitHub account. GitHub organizations can be set-up so that its

[2]DOIs are easy to generate with the Zenodo plugin for your GitHub repositories. Read how here.

[3]What is a NOAA work product? First, this is part of your job and you are being paid by NOAA to do this work. Second, it is a product. A repo that you throw onto GitHub as part of something you are testing out or during a workshop you are taking is not a 'product'. Would you put a DOI on this thing? Yes? That's probably a product.

members can create and manage their repositories freely as they would in their individual account. Creating work repositories in a GitHub organization greatly streamlines on-boarding and off-boarding. It also makes it easier for all team members to use similar templates for their repositories.

3.5 GitHub Organization Settings

GitHub organizations is simply an account where multiple GitHub users can be named as members who are then given permission to create repositories. The organization can have as many admins as you need, but 2-3 is common. Anyone can create a GitHub organization and it is a convenient way for teams or groups to organize and collaborate on a thematic set of repositories. In addition, a GitHub organization allows members to manage project boards, tasks, and discussions, and allows the admins to customize permission settings for the organization members.

The following are basic best practices for GitHub organizations.

- Fill out the organization profile with the NOAA affiliation.
- Provide contact information for the organization owner(s) or maintainer(s). Ensure that the organization has multiple owners who are able to manage the site. Ensure that when an organization owner leaves NOAA, another owner is able to manage the organization and all repositories.
- Create a `README.md` file for the organization in the `.github/Profile` folder. This README will then appear on the front page of the organization and can be used to describe the organization's purpose, affiliations, and repositories.

Naming Conventions

If your GitHub organization is specific to one center, be aware that some centers use a naming convention, e.g., `nwfsc-mathbio`.

3.5.1 NOAA Branded Organizations

There is nothing inherently official or formal about a GitHub organization; it is simply an account with multiple GitHub users (members). However a GitHub organization can be formal and highly regulated. A GitHub organization that delivers official products for the public would fall under this category and these organizations would use clear NOAA logos, text and other branding.

GitHub organizations that deliver official NOAA products will need to have formal guidelines for participation in the organization. Examples of the guidelines include only allowing NOAA members to participate, no direct push access by non-NOAA accounts,

guidelines for code review and tests, push access to repositories or visibility limited to specific members, naming conventions for repositories, and guidelines for repository structure.

3.6 Collaboration with non-NOAA GitHub users

From the 2017 guidelines: "NOAA has a strong history of scientific collaboration, coordination, and close engagement with other government partners, non-government organizations, academic institutions, international colleagues, and other members of the scientific research community." GitHub facilitates collaboration with non-NOAA collaborators and this is encouraged, however care needs to be taken for NOAA branded GitHub organizations and repositories. See Section 3.5.1 on common restrictions placed on NOAA branded GitHub organizations and repositories.

4 Security

4.1 FISMA Low

The scientific product on GitHub must be reasonably classifiable as FISMA Low, as outlined by the Federal Information Security Management Act of 2002. FISMA Low classification includes only information for which the unauthorized disclosure, unauthorized modification, unauthorized destruction, or disruption of access can be expected to have a limited adverse effect on organizational operations, organizational assets, or individuals. If the effect of such events would be serious, severe, or catastrophic, the information cannot be released under this authority.

4.2 Sensitive information cannot be shared on GitHub

No usernames, passwords, login information, port numbers, IP addresses, server names, Application Programming Interface (API) keys, Personally Identifiable Information (PII), Business Identifiable Information (BII), or confidential data may be stored in any file hosted on GitHub. Read Section 3.3 on how to properly store and use credentials. If you have GitHub Actions or Pages that use credentials, then Encrypted Secrets inside of GitHub is also acceptable for API (Application Programming Interface) keys and similarly credentialed interfaces.

4.3 Preventing inadvertent committing of secrets or credentials to GitHub

If your repository code uses confidential data or connects to APIs using IP addresses, usernames, passwords, secrets, or credentials, then you must take steps to ensure that you do not inadvertently commit these to GitHub. Ensuring that confidential data or secrets do not get pushed to GitHub requires diligence and deliberate choices in your workflow.

> **Note**
>
> Your approach must be tailored to the nature of your work and the content of the repository. For example, a repository that is a package for fitting species distribution models to generic data or a demo repository for teaching purposes is very different from a repository that produces an official report using confidential fisheries data.

4.3.1 Prevent Damage

Establish a workflow that keeps confidential information separate from non-confidential information

- Separate your files into "public" and "secret" folders. Confidential data or secrets cannot be dispersed throughout your code and files. Clearly distinguishing between "public" and "secret" in your code and files will require a deliberate and carefully designed file organization.
- Once you have "public" and "secret" folders delineated for your repository, use the `.gitignore` file to prevent these from being pushed to GitHub. You may need to add these files to the `.gitignore` file before you push to your GitHub repository.
- Alternatively, for code that integrates data or databases, develop a public version of the package. This public version of the package would not include any of the data or credentials and can then be used locally and called with arguments to create the output.

Passwords and API links

- Never leave usernames or passwords into your code or scripts. While it may be tempting to save time, this habit is a common cause of inadvertent sharing of credentials and is a security risk.
- The {dotenv} and {keyring} packages can be used to securely store API links and credentials for connecting to databases, APIs, and other services. These packages are easy to use and implement into your workflow. Other APIs (like Google Drive) use a secrets folder which has encrypted tokens for accessing their APIs.

- The basic idea is to keep the "keys" on your local computer so you'll need to add the folders or files with the secrets to your `.gitignore` file.
- GitHub Actions or Pages that use API (Application Programming Interface) keys and similarly credentialed interfaces can use Encrypted Secrets inside of GitHub.

Restrict push permissions and require pull-requests and code review

You can incorporate code and content review but only allow pull requests and no direct push access. This allows time for a manual review of the material being pushed to GitHub.

Customized pre-commit hooks to prevent committing secrets

Repositories that are in danger of committing access keys to cloud computing resources or similar credentials with high consequences will need to implement mechanisms to prevent committing secrets to GitHub in the first place. gitleaks should be used. It uses "git hooks", which are a special file in the `.git/hooks` folder within a repository. See the {gitleaks} documentation. Another option is the git-secrets add-on which you install locally.

Note, secrets scanners like gitlinks need to be customized and tested. Also, They are designed to catch secrets that look like AWS keys or API keys and not confidential fisheries data or a simple text password hard-coded into a script.

4.3.2 Detect Damage

If a secret has mistakenly made it onto GitHub, you will need to remove it as soon as possible and change keys/passwords that were shared. Enabling repository scanning via a secrets scanner such as gitleaks GitHub Action can provide an alert if someone inadvertently commit secrets to your repository. The scan will be automatically run using a "GitHub Action" every time a "push" or "pull request" is completed. Note, this is not a free service. If you have an Enterprise account, secrets scanning is included in the license and you should reach out to the Enterprise account administrator to get that set up. Always test your scanner set-up by committing some dummy secrets that should be caught.

5 GitHub Repository Components

5.1 README.md

This file should provide a description of the repository. The contents of the README file will vary greatly depending on the application. Here are some tips:

- **An official work product released on a NOAA-branded GitHub organization**. This is the most formal instance and the organization leaders should develop template READMEs to provide a consistent format across the products.
- **A software package (e.g., R package)**. Standard components are: Badges indicating build status and version, how to install, how to use or link to documentation, where to report issues, citation.
- **A report or paper.** Author, description and citation.
- Add the DISCLAIMER (below) for any repository that is a 'work product'.

5.2 DISCLAIMER.md

Repositories and web content shared on GitHub should make it clear to the audience that no information should be considered or interpreted as official communication of NOAA. The simplest method for doing this is to include the following disclaimer within a `DISCLAIMER.md` file at the root of each repository. Additionally, it's also good practice to include the disclaimer text as a footnote within the repository's `README.md` and any web content available from that repository. Be careful not to use NOAA logos and and NOAA Fisheries branding in a way that could imply an official communication. NOAA logos should be used to indicate your affiliation and acknowledge support and funding. The repositories within the NOAA Fisheries Integrated Toolbox GitHub organization are good examples of how to prepare README files with the Disclaimer and NOAA Fisheries logos.

The following `DISCLAIMER.md` file is put in the root of the repository.

DISCLAIMER.md

This repository is a scientific product and is not official communication of the National Oceanic and Atmospheric Administration, or the United States Department of Commerce. All NOAA GitHub project code is provided on an 'as is' basis and the user assumes responsibility for its use. Any claims against the Department of Commerce or Department of Commerce bureaus stemming from the use of this GitHub project will be governed by all applicable Federal law. Any reference to specific commercial products, processes, or services by service mark, trademark, manufacturer, or otherwise, does not constitute or imply their endorsement, recommendation or favoring by the Department of Commerce. The Department of Commerce seal and logo, or the seal and logo of a DOC bureau, shall not be used in any manner to imply endorsement of any commercial product or activity by DOC or the United States Government.

5.3 LICENSE Files

The work of U.S. Government employees is not subject to copyright in the U.S. Thus, you should include an appropriate LICENSE file with your repositories. There are two types of licenses that are commonly used: Creative Commons Zero v1.0 Universal (CC0 1.0 Universal) and (GNU General Public License v3.0) GPL-3. The CC0 license is commonly used for US Government products but the NOAA Fisheries Integrated Toolbox prefers the GPL-3 license since it is more tailored for code products. Other open source licenses (e.g., MIT) may also be appropriate.

To add a LICENSE file to your repository, navigate to the repository on GitHub.com, select "add file,", select "create new file", and type LICENSE. You will then see the button in the upper right hand of the page to choose a license file. Additionally, the following statement applies with regards to licensing for any code published in any NOAA repository. This should be included within a `LICENSE.md` file at the root of each repository. Repositories that are code packages should also add this to the repository `README.md`.

LICENSE.md

Software code created by U.S. Government employees is not subject to copyright in the United States (17 U.S.C. §105). The United States/Department of Commerce reserves all rights to seek and obtain copyright protection in countries other than the United States for Software authored in its entirety by the Department of Commerce. To this end, the Department of Commerce hereby grants to Recipient a royalty-free, nonexclusive license to use, copy, and create derivative works of the Software outside of the United States.

6 GitHub working group

The NMFS Open Science GitHub working group is a group of NMFS scientists who are active developers on GitHub and with wide experience is the many ways that GitHub is used within NOAA Fisheries. The group is involved in helping NMFS scientists with GitHub use by developing this best-practice document, helping with trainings and content, and coordinating with NMFS IT.

Lead editors:

- Eli Holmes, Northwest Fisheries Science Center
- Josh London, Alaska Fisheries Science Center
- Emily Markowitz, Alaska Fisheries Science Center
- Kathryn Doering, Office of Science and Technology

The editors assembled the material into a cohesive format, but significant sections were developed by other individuals in other contexts. See also the references (Section 7).

- Much of the section on setting up your GitHub individual account was adapted from the SEFSC October 2021 GitHub SOP. This SOP effort was led by James Primrose, SEFSC. Note the SEFSC SOP was developed from the NEFSC SOP led by David Chevrier, NEFSC.
- Most of the information on standard elements of a NMFS repository (disclaimers, readme, logos, etc) was adapted or taken directly from the Fisheries Integrated Toolbox Resource pages. Corinne Bassin, Christine Stawitz, Kathryn Doering, and Bai Li were the developers of the FIT material.
- The section on security used information from a presentation on "Keeping secrets secret" in the context of accessing fisheries databases via APIs by Adyan Rios, Southwest Fisheries Science Center.
- Some of the material was first assembled in a less formal format in the NMFS Open Science Resource Book, a project of the 2021 Openscapes cohorts at NWFSC and AKFSC.
- Thanks to the many NMFS R User Group members who commented on the initial drafts over 2022.

7 References

- 2017 CIO Memo authorizing GitHub use at NOAA Note this document was written before Microsoft bought GitHub and before GitHub Enterprise existed.
- 2017 NOAA GitHub Guidelines for NOAAGov GitHub Org This document appears to be intended to apply only to content on https://github.com/NOAAGov. See top of page 2.
- NOAA GitHub Checklist Note this checklist appears to be intended to apply to contributions to https://github.com/NOAAGov. References the GitHub Guidelines document above which is written for the NOAAGov GitHub organization.
- NOAA Scientific Integrity Policy (NAO 202-735D.2) website. final signed
- SEFSC GitHub Standard Operating Procedure (SOP), October 2021. James Primrose, SEFSC.
- NEFSC GitHub Standard Operating Procedure (SOP), October 2020. David Chevier, NEFSC.
- NWFSC GitHub Guidelines, 2018. David Berklund, NWFSC.
- Fisheries Integrated Toolbox GitHub Guide
- Using GitHub at NOAA Practice and Policy PMEL presentation by Eugene Burger, PMEL.
- DOC Open Source Policy
- Federal Open Access Memo 2022
- GitHub Enterprise Fedramp authorized