Some key terms:

Open source License- A software license, approved by the Open Source Initiative (OSI) as compliant with the Open Source Definition, granting permissions for anyone to inspect, use, modify, and distribute the software's source code for any purpose

Open source software - Software whose source code is under an open source license, by which the copyright holder grants to anyone the rights to inspect, modify, and distribute the source. Synonymous with open code.

Code Repository - A central storage location for the source code. Code repositories may contain source code in one or more programming languages. Repositories may provide tools for merging inputs from developers, automated testing to verify the proper functioning of source code, version control to track changes over time, and project management features. These sites may not promise long-term retention.

"Many journals and funding agencies require that you share your code at the time of publication. However, the prospect of opening code up to criticism, not receiving attribution, or missing a result that external researchers discover can deter scientists from making their code open-access. What if people find an error or criticize your coding style? What if they take your code and publish a new result without including you?"

Lets answer these in the next section of notes:

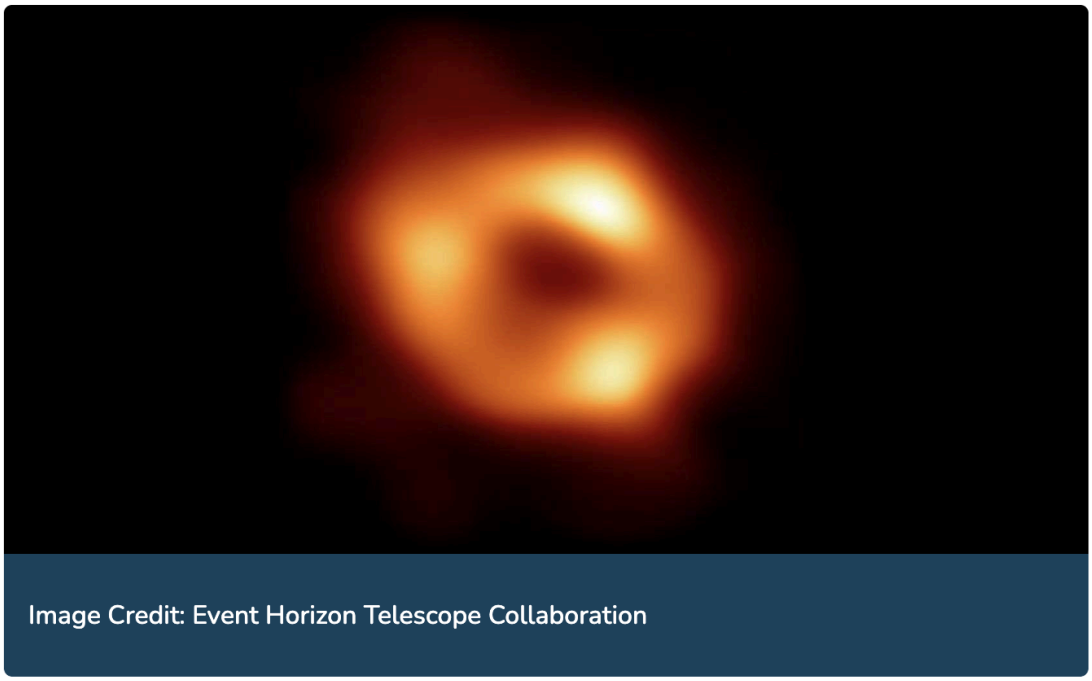


Image Credit: Event Horizon Telescope Collaboration

This famous first black hole image was created from the code contributions of 21,485 people!!

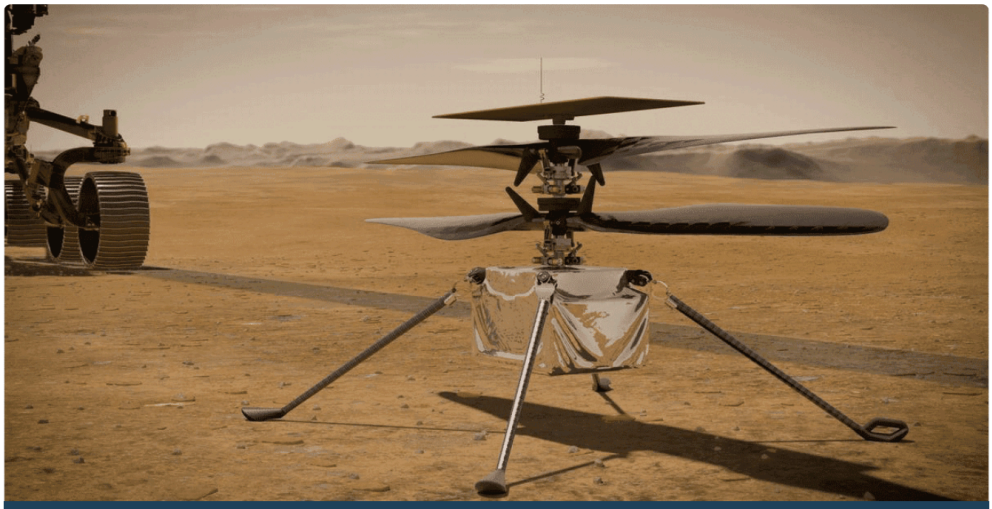


Image Credit: NASA/JPL-Caltech

This is the Ingenuity helicopter, or as the engineers call her, Ginny. She got to Mars by hitching a ride on the Perseverance rover, landing in the Jezero Crater in 2021.

This is a video of Ginny's first flight. She took off, got about ten feet off the ground, did a spin, and landed. This groundbreaking flight proved that powered flight on Mars is achievable, opening up the door for an entirely new era of exploration.

But Ginny's achievements also reflect another new era, one of truly open and collaborative science.

Behind that four-pound helicopter were more than 12,000 people who contributed code, documentation, design, and more, thanks to the open-source software which was used to power her. Everyone who contributed to the open-source software libraries that Ginny used received a badge on their GitHub page, showing that they helped fly the first helicopter on Mars.

In addition, Ginny's final software developed at the Jet Propulsion Lab, called F Prime, was itself open-source and has been used since in flight research, drones, and CubeSats. In fact, F Prime had been copied to other people's repositories more than 1,200 times.

Wow! https://github.com/nasa/fprime software of Ingenuity

All science builds on what has already been accomplished. Code is no different. *It's ok to use existing frameworks and build upon them, there is no need to do it from scratch every time you want to start a task.* ***This is still part of first principles thinking****, we*

aren't making the problem larger than it has to be by abstracting the fundamentals. Experiment, then optimize.

1. Have you used open code principles in your work?

I think I have. The Use Make Share framework is something I'm following with LAIKA, and I will include my orcid as I progress. The code is available on github, and I'll ensure those who work on it or create derivatives, do so using their orcid's in their software.

2. What are some of the successes and challenges you have encountered?

In line comments are how I best understand code, and often I find that open source software I use is scant on comments. I do my best to add these in when I am making new products with open software, for myself and for others to better understand the first time looking at it. Open software has encouraged others to join some of the projects I'm working on, especially AETHER, who has gained attention and new engineers because of its openly shared base.

3. What resources did you find useful for advancing open code in your work?

Github is a great place to share, and so is a personal webpage. I use go daddy and link as much as I can on there. It serves as my engineering diary for project operations.

Using open Code

How to find open code to help your projects:

A successful search for open code demands a clearly-defined purpose. Developers must first determine the tasks they expect their code to carry out. The requirements associated with these tasks can determine the best-suited programming language.

Next, if you want to find open code with similar requirements to your own, familiarize yourself with the terminology employed by others. The keywords affiliated with your programming purpose or requirements can serve as a starting point when searching for relevant code.

Biggest questions to ask: Can you **trust this code you discovered** on the web? **How much time** will it take to learn it? Could the **code contain malware**? Could you get in **legal trouble** for using it?

Consider these 4

- Functionality: Will it be useful for your scientific problem?
- Interoperability: How difficult will it be to use?
- Security: Is it safe? Would using the software create a security risk?
- Licenses/restrictions: Are you allowed to use it? Is it legal to use the software in your project?

Cite using the DOI or directly cite the GitHub repository

Making Open Code

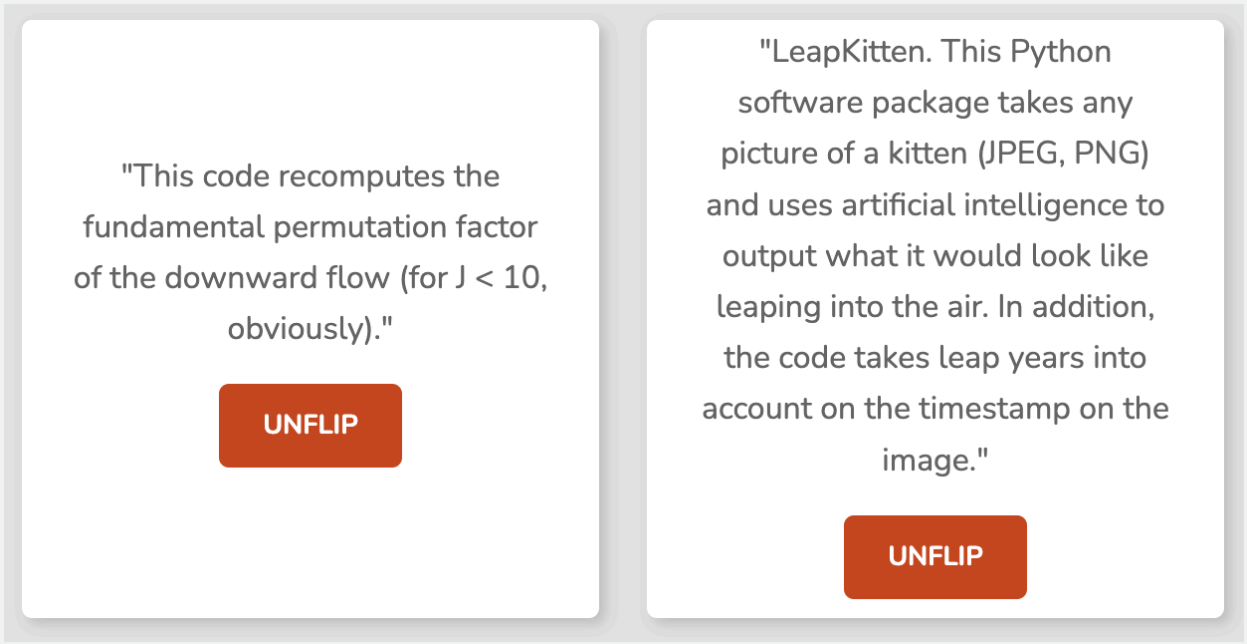
A good place to start: asking these questions before you begin can help when developing code to solve a problem.

- What problem am I trying to solve, and are others in my scientific community facing it as well (look for how they attempted it, break down the problem to fundamentals ?
- Are there existing solutions?
- If code does exist how alterable is it to match the needs of the project? (You could potentially contribute to existing code rather than writing something new.)

Even if a solution does exist, sometimes it's best to develop new code if the license isn't open enough to work with, it's in a language you don't use, or you have a different approach to solving the problem.

Starting a new project

1. Define the scope, functions and users
2. Consider resources required to run such a project
3. Consider who will manage and contribute



A BAD readme file vs a GOOD pictured above

Good documentation and ReadME files may also contain the following:

- A list of any code dependencies the software has, e.g. "Numpy, kitten-rng, and human-readable must be installed to run this software."
- How to install and a brief description of how to run the software.
- Detailed description of the software, especially if there is no external documentation.
- Examples of how to use the software.
- Acknowledgement of team members or sources of support.
-

While not every project has a CONTRIBUTING file, the existence of one is a clear indicator that contributions are welcomed.

- **Could be useful if you have a large open project or to include licensing. (in a citation file fist make a zenodo DOI if the repo is in a finished state to site it)**

Your **software should be documented within the source code.** Each function should have comments at the start that briefly state, in plain language, what the function is for. This is not only for other developers, but for your future self if you've forgotten those details.

Can also consider writing a how to use file

- **Create a virtual environment specific to your project to isolate its dependencies (and their versions) from those used for other projects.** WOW good tip.

Apache 2.0 license - permission open source license- https://opensource.org/license/apache-2-0

APPENDIX: How to apply the Apache License to your work

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets “[]” replaced with your own identifying information. (Don’t include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same “printed page” as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

Def. Protective Licenses allow users to reuse, but also require users to share their changes with the community using the same license.

"A NASA Open Science and Data Management Plan (OSDMP) describes how the scientific information that will be produced from NASA-funded scientific activities will be managed and made openly available. The OSDMP should include sections on data management, software management, and publication sharing. (NASA Science Mission Directorate, 2025)"

-NASA Science Mission Directorate, 2025

Example sections to include in an OSDMP:

- Data management plan (DMP)
- Software management plan (SMP)
- Publication sharing
- Other open science activities
- Roles and responsibilities

3. Repositories and Timeline for Sharing Software

This work will support the development of two peer-reviewed journal articles. All source code developed in Python to support each article will be archived on Zenodo no later than the article's publication date. The software will be made available under a permissive Apache License 2.0. Zenodo will assign a DOI to the archived software when it is archived.