Licensing: How to open-source your work

Skylar John Evans Jeffrey Fisher

Introduction

Presenters

What this presentation is

We intend that after this presentation you:

- Know what permissions popular licenses grant you
- Have a good idea of what licenses you may want to use for your projects
- (maybe) Have some understanding of the issues if you want to make money from open-source work

What this presentation is **not**

We will avoid promoting a particular choice. However, we may mention pros and cons of particular choices. Also, this presentation is focused on free and open-source licenses.

We are not lawyers. We focus on well-understood truths, or point out when something is not well-understood. We cite our sources. If you want to use a custom or less-understood license, you should probably consult a lawyer.

Terminology

Copyright, works, licenses, oh my!

- Work = Noun. As in "a work of art".
- Copyright notice = Statement of who owns the copyright for something. Example: "Copyright © 2012 Jane Doe"
  - Include year of first publication.
  - A year range may be included for works that are frequently updated: "Copyright © 2000-2004 Jane Doe"
- License = Agreement that grants permissions, possibly with some restrictions.

Free and Open Source Software (FOSS)

The free software movement and the open-source software movement are separate but closely related.

Most licenses that fit the free software definition also fit the open-source software definition, and vice versa. ("Categories of Free and Nonfree Software," n.d.)

Just stick it online, right?

No license = All rights reserved

> *When you make a creative work (which includes code), the work is under exclusive copyright by default. Unless you include a license that specifies otherwise, nobody else can copy, distribute, or modify your work without being at risk of take-downs, shake-downs, or litigation. Once the work has other contributors (each a copyright holder), "nobody" starts including you. ("Choose an Open Source License," n.d.)*

- If you are the only user, this is probably fine
- It's not required, but it's a good idea to put a copyright notice and statement of "all rights reserved" in an easy-to-find place like a file named README.
- If you want other people to reuse and share your work, this is probably not what you want
- Choose a license to specify your terms more clearly

Permissive licenses

I want it simple and permissive

Permissive licenses let people do almost anything they want with your project, like making and distributing closed source versions. ("Choose an Open Source License," n.d.)

- Includes public domain (there is no copyright owner)

Examples

- MIT: NodeJS
- BSD: scikit-learn
- Apache: Qiskit
- Unlicense: yt-dlp

Copyleft licenses

I care about sharing improvements

> *Copyleft licenses lets people do almost anything they want with your project, /except/ distributing closed source versions. ("Choose an Open Source License," n.d.)*

- Changes must be redistributed with a copyleft license and with source code
- Some copyleft licenses (GPL) may be incompatible with proprietary code
  - LGPL probably doesn't have these problems
- Easier to retain control and share improvements

Examples

- GPLv2: Linux kernel
- GPLv3: LLaMA
  - provides additional clarifications on license compatibility, digital rights management, and patents
- AGPL: RStudio
  - requires servers hosting the code to release their
- LGPL: TODO (many low-level libraries)

Licenses for non-software works

Creative Commons

- CC0 = public domain
- CC-BY = attribution (like MIT)
- CC-BY-SA = share alike (like GPL)

Hardware

3D Printing

Home 3D printing has made sharing 3D models common practice.

Similar to source code files, 3D model files can be considered creative works, and can be licensed as such.

Note: This does not mean that the design or the physical prints are protected. Patents would be required for that.

Working with others and in a community

Making a new project

- Your license conveys to the community how you intend your project to be used and shared
- Permissive licenses are easier to use and distribute, but harder to retain control and share improvements
- Copyleft licenses are harder to use and distribute, but easier to retain control and share improvements

Contributing to an existing project

- Review the license
- Read the contributor agreements, if there are any:
  - GNU Emacs: Contributing to Emacs itself or the official package repository (ELPA) requires you to assign your copyright to the FSF. Emacs is GPLv3 licensed so this is reasonable.
  - Developer Certificate of Origin (DCO): Contributing to the Linux kernel requires signing this. In brief, the certificate verifies that the developer wrote the open-source changes and allows the project to use them in a way consistent with the license. This is reasonable because the developer owns their changes to the project. https://developercertificate.org/
  - Contributor License Agreement (CLA): Refers to a restrictive class of agreements that might allow the owner of the project (usually a foundation or a company) to relicense your patches. Controversial because if the company makes the project proprietary, developers no longer own the open-source code that they contributed. The company does. This is contentious among some communities so contribute to these with caution.

Dependencies and using other people's code

- License compatability: you must comply with the licenses of code you depend on
- Rule of thumb:
  - copyleft code can only be reused in copyleft code
  - permissive code can be reused almost anywhere
  - proprietary code may have restrictions (frequently noncommercial)

You can check the licenses of your code's dependencies for various languages:

- Rust
- Python
- Javascript

Non-FOSS licenses

Proprietary licenses

An important requirement of FOSS is that it can be used for any purpose (commercial or otherwise).

- Just because you can look at the code doesn't mean it's FOSS
- Leaked/reverse-assembled proprietary code is still proprietary

Source-available licenses

- Business Source License (“permissive”)
  - Releases are source available with a promise to open-source it later
  - QT, MariaDB, Codon
- Server Side Public License (“copyleft”)
  - Copyleft extends much farther than the AGPL
  - Withdrawn from consideration to the Open-Source Initiative: restricts the right to make use of the program for any purpose

Noncommercial Creative Commons licenses

In recent years, the Creative Commons family of licenses have become popular.

3D printing repository websites such as Thingiverse and Printables have these licenses available for users to select from when uploading their designs.

While it is not well explained on these sites, the Non-Commercial versions of these licenses are not free or open source. ("Creative Commons 4.0 BY and BY-SA Licenses Approved Conformant with the Open Definition," n.d.)

Ethical-source license

Ethical source licenses have provisions that restrict uses for ethical purposes.

- JSON license: “The Software shall be used for Good, not Evil.”
  - What counts as an “ethical” usage?
  - Maybe unenforceable

Practical and economic considerations

Practical and economic considerations

- Whatever license you pick, consider the community
- If you need a legally-contentious or custom license, consult a lawyer
- Now you know the basics of software licensing

Bibliography

Bibliography

"Categories of Free and Nonfree Software." n.d. Free Software Foundation. https://www.gnu.org/philosophy/categories.html.

"Choose an Open Source License." n.d. GitHub, Inc. https://choosealicense.com/.

"Creative Commons 4.0 BY and BY-SA Licenses Approved Conformant with the Open Definition." n.d. Creative Commons. https://creativecommons.org/2013/12/27/creative-commons-4-0-by-and-by-sa-licenses-approved-conformant-with-the-open-definition/.