OPEN SOURCE LICENSES

Presented by Bob Primak

To the Lexington Computers and technology Users Group (LCTG)

Wednesday, November 19, 2025

Much of this material was generated using GeminiAI within Google Search.

OPEN SOURCE LICENSES

What Is a Software License

A software license is a legal contract that grants a user permission to use a software product under specific terms and conditions, which can include restrictions on how it can be copied, modified, and shared. This agreement is between the copyright holder (the software author or publisher) and the user (the licensee), and it outlines the user's rights and the author's expectations. It is not a sale of ownership but rather a rental of rights to use intellectual property.

OPEN SOURCE LICENSES

Key components and types of software licenses

Grant of use: The license defines the scope of use, such as the number of devices or users, and what actions are permitted, like copying or modification.

Copyright and intellectual property: The license establishes how the software's intellectual property can be used. It also protects the author by defining unacceptable uses.

OPEN SOURCE LICENSES

Types of licenses:

Proprietary: Often includes restrictions on copying, distribution, and modification.

Open Source: Allows users to view, modify, and distribute the source code, with different variations like:

Copyleft licenses: Require that modifications also be made available under the same license terms.

OPEN SOURCE LICENSES

Types of licenses:

Permissive licenses: Have very few conditions, often only requiring that copyright information be retained.

Public Domain: The developer forgoes all copyright and gives the software away with essentially no restrictions.

OPEN SOURCE LICENSES

Open Source License Types: Permissive and Copyleft

The main types of open-source licenses are

permissive and **copyleft**.

Permissive licenses have minimal restrictions, allowing code to be used in most projects, including proprietary ones, with simple attribution requirements.

Copyleft licenses are more restrictive, requiring that any derivative works be distributed under the same, or a compatible, copyleft license. A third category is weak or limited copyleft licenses, which provide a middle ground by requiring only certain parts of the code to remain open-source.

OPEN SOURCE LICENSES

Copyleft is a legal term, and a specific type of license that uses copyright law to allow the free distribution and modification of a work, with the condition that any derivative works must also be distributed under the same copyleft license. It is not the opposite of copyright, but rather a method of licensing that works within the existing copyright framework to guarantee ongoing freedom for the work and its future versions.

Based on copyright: Copyleft licenses are only possible because the author first owns the copyright to the original work, giving them the right to decide how it can be distributed.

OPEN SOURCE LICENSES

A licensing method: It is a way to grant certain rights to users while imposing conditions, primarily that any modified versions must be shared under the same terms.

Open Source Licenses by Category

Page created on September 19, 2006 | Last modified on November 2, 2022

https://opensource.org/licenses-old/category

This list contains around 118 different open source licenses.

I only have limited time, so I'll cover 3 of the most popular open source licenses. Some are copyleft, some more permissive. If time allows, I can circle back to cover two more open source license types.

OPEN SOURCE LICENSES

Five of the most common open-source license types are

(1) the MIT License, which is permissive.

(2) the Apache License 2.0, which I may not get to.

(3) the GNU General Public License (GPL), which is restrictive.

(4) the Mozilla Public License (MPL), which is in-between, and

(5) the BSD Licenses, which I may not get to.

OPEN SOURCE LICENSES

(I'll skip the GNU Lesser General Public License (LGPL) due to time constraints.)

These licenses vary in how they handle software modification and distribution, with some being more "permissive" (like MIT, Apache and BSD) and others being "copyleft" (like GPL and MPL).

Creative Commons Licensing is also used, but its structure is complex, and I don't have enough time to go into its nuances.

OPEN SOURCE LICENSES

License Category	Examples	Key Characteristics
Permissive Licenses	MIT Apache 2.0 BSD	- Allow broad use, modification, and redistribution. - Can be used in proprietary (closed-source) software. - Typically only require attribution and copyright notices to be preserved.
Copyleft Licenses	GNU GPL Mozilla Public License (MPL)	- **Strong Copyleft (GPL):** Requires derivative works to be licensed under the same terms. - **Weak Copyleft (MPL):** Applies the copyleft provisions to the licensed code itself, but allows for linking with non-copyleft code.

OPEN SOURCE LICENSES

(1) MIT License – which is permissive.

The MIT license is a permissive open-source license that allows users to use, copy, modify, and distribute the software freely for any purpose, including commercial use. The primary requirement is that the original copyright and license notice must be included in all copies or substantial portions of the software. The license includes a disclaimer of warranty and does not hold the author liable, but it is often interpreted as allowing patent usage despite lacking an explicit patent grant.

OPEN SOURCE LICENSES

Key aspects of the MIT license

Permissive nature: It places minimal restrictions on how the software can be used, making it a popular choice for open-source projects that want to encourage widespread adoption and collaboration.

Broad permissions: You can use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the software.

Commercial use: The license allows for commercial use, and you can incorporate MIT-licensed code into proprietary, closed-source products.

Attribution requirement: You must include the original copyright notice and a copy of the license text in your redistributed software.

OPEN SOURCE LICENSES

Key aspects of the MIT license

No warranty: The license disclaims all warranties, meaning the software is provided "as-is" with no guarantee of functionality.

No liability: The original author is not held liable for any claims arising from the software.

Patent rights: The license doesn't include an explicit patent grant, which can create some legal ambiguity compared to licenses like Apache 2.0, though courts often interpret it as allowing patent usage.

OPEN SOURCE LICENSES

Controversies Surrounding the MIT License

The main controversy surrounding the MIT License is its lack of explicit patent protection, which leaves users vulnerable to patent lawsuits from the original copyright holder. Other controversies include how its permissiveness can be seen as problematic when large companies use open-source code for proprietary products without contributing back, and confusion over licensing when a project is forked. Some projects have addressed this by adding separate patent licenses or switching to more comprehensive licenses.

OPEN SOURCE LICENSES

Patent issues

No explicit patent grant: The MIT license doesn't mention patents, and while some interpret an implicit patent license, it is not a guarantee. This leaves users open to "submarine patent" threats, where the license holder could later sue over patent infringement.

A need for explicit grants: Some projects, like Facebook's React, have previously addressed this by adding a separate patent license to the MIT license. More modern licenses, like Apache 2.0, have explicit patent provisions to avoid this ambiguity.

OPEN SOURCE LICENSES

"Abuse" by large corporations

One-way use: While the license is designed to be permissive, some argue that it can be "abused" when large companies use open-source code in proprietary products without contributing back to the community.

Examples: Some forks of projects like Chromium are turned into proprietary products (like Chrome and Edge) while others, like Microsoft's VS Code, restrict access to official extensions for forks, making them less useful.

OPEN SOURCE LICENSES

Forcing forks and other issues

Forking issues: The permissiveness of the MIT license has been criticized in projects like Bitcoin, where its wide adoption has led to a proliferation of forks, causing confusion and debate among developers about project ownership and governance.

Trivial misuse: In some instances, developers have been upset when their work was used without attribution, but the license only requires the original license and copyright notice to be included, not attribution for all modifications.

OPEN SOURCE LICENSES

Potential solutions and alternatives

Explicit patent licenses: For projects that are concerned about patent issues, adding a separate patent grant to the MIT license is a potential solution.

Switching to a more comprehensive license: Projects can opt for a more comprehensive license like the Apache License 2.0, which addresses patents, copyright, and trademark separately.

GPL-style licenses: Some developers switch to copyleft licenses like the GNU General Public License (GPL) to ensure that modifications are also shared under the same license terms, which can help prevent proprietary forks from being created.

OPEN SOURCE LICENSES

(3) GNU General Public License – which is Strong Copyleft.

The GNU GPL 3.0 is a strong copyleft license that allows users to copy, modify, and distribute software, but requires that any modifications or derivative works must also be distributed under the same GPL 3.0 license. This ensures the software remains free and open, but also means that combining it with non-GPL code can require the entire work to be released under GPL 3.0. Key features include protection against "tivoization" (devices that prevent users from running modified software), patent restrictions, and the requirement to provide source code for modified versions.

OPEN SOURCE LICENSES

Key features

Protection against "tivoization": GPL 3.0 prevents devices from being designed to deny users the ability to install modified versions of the software.

Patent protection: It provides explicit protection against patent restrictions that could limit the use of the software.

Compatibility: It was updated to improve compatibility with other free software licenses.

OPEN SOURCE LICENSES

Implications for businesses

While you can sell software that uses GPL 3.0 code, you must provide the source code to your customers.

Companies that want to keep their own code proprietary generally avoid using GPL 3.0 code because of its "infectious" nature, which requires derivative works to also be open-sourced.

OPEN SOURCE LICENSES

GPL Versions

The two most widely used versions are GPL v2 (1991) and GPL v3 (2007). GPL v3 offers updated provisions, including clearer procedures for addressing non-compliance and protections against "tivoization" (preventing users from running modified software on a device).

This last item refers to **a controversy involving the TiVo video recording devices.** (Full disclosure: I own a TiVo device.)

OPEN SOURCE LICENSES

The GPL TiVo controversy, known as "tivoization," stemmed from TiVo's use of the GPLv2-licensed Linux operating system in its DVRs. While TiVo complied with GPLv2 by releasing its modified source code, it circumvented the spirit of the license by using hardware-level digital restrictions (DRM) to prevent users from running modified or third-party software on the devices. This sparked a major debate within the free and open-source software community, leading the Free Software Foundation (FSF) to create the new, more restrictive GPLv3 license to specifically prevent this practice.

OPEN SOURCE LICENSES

The issue:

TiVo's action: TiVo used GPL-licensed Linux but implemented a digital signature check to ensure that only its own authorized software could run on its hardware.

The problem: The Free Software Foundation (FSF) argued that this practice, known as "tivoization," violated the spirit of the GPL by denying users the right to install modified versions of the software on the hardware they owned.

The debate: Although TiVo released its source code as required, the hardware locked users out of running modified versions, which the FSF saw as circumventing the user's freedom to modify and run the software.

OPEN SOURCE LICENSES

The response:

GPLv3: The controversy was a primary reason for the development of the GNU General Public License version 3 (GPLv3).

Anti-tivoization clause: GPLv3 included a specific anti-tivoization clause to prevent this type of hardware restriction, which aims to ensure users can run modified versions of the software they have the source code for.

Mixed reactions: The addition of this clause was controversial. While many free software supporters welcomed it, some, like Linus Torvalds and the Linux community, disagreed with its restrictive nature.

OPEN SOURCE LICENSES

Truth Social and Mastodon

The AGPL 3.0 license is a strong copyleft license that extends the requirements of the GPL to software used over a network. Its main feature is that if you modify the software and use it to provide a service over a network, you must make the complete source code of the modified version available to users. This ensures that the freedom of the software remains available to anyone who interacts with it, even if they only access it remotely.

OPEN SOURCE LICENSES

Key features

Strong copyleft:

Like the GPL, it requires that any derivative works are distributed under the same AGPL 3.0 license, but it adds a crucial condition for network use.

Network copyleft:

The license ensures that if you modify the software and offer a service over a network, you must provide the source code of your modified version to anyone who interacts with it.

OPEN SOURCE LICENSES

Availability of source code:

To comply, you must provide the full source code of the modified program and any other software that forms part of the service.

Preservation of rights:

It requires that all copyright and license notices are preserved and includes an express grant of patent rights from contributors.

OPEN SOURCE LICENSES

Dual licensing:

Some software projects, like MinIO, are available under both the AGPL 3.0 and a separate commercial license. This allows developers to use the software commercially if they purchase the commercial license instead of adhering to the open-source requirements.

OPEN SOURCE LICENSES

Who it's for

Developers:

The AGPL 3.0 license is a good choice for developers who want to ensure that their open-source projects and any modifications remain free and open, even for network-based services.

Users:

It protects user freedom by ensuring that everyone can access the source code for any modified version they use over a network.

Businesses:

Businesses may need to use a commercial license if they want to avoid the AGPL 3.0 requirements of making their modified code public, such as when building a proprietary service on top of AGPL software.

OPEN SOURCE LICENSES

The Truth Social Controversy

The Truth Social open-source licensing controversy arose when the platform used code from the open-source project Mastodon without complying with the open-source license's terms. **Mastodon is released under the AGPLv3 license**, which requires that any network-connected modifications must have their source code made publicly available to all users. Critics like the Software Freedom Conservancy accused Truth Social of violating this license by not providing its source code and misrepresenting the platform as proprietary.

OPEN SOURCE LICENSES

Key details of the controversy

Initial use of code: A beta version of Truth Social was found to be using Mastodon's open-source code, including elements like the software's frontend and underlying HTML.

License violation: By not making its source code publicly available, Truth Social was violating the AGPLv3 license, which is a "copyleft" license requiring all users who operate the software over a network to make their modifications available.

Truth Social's response: The platform initially claimed its source code was "proprietary" and removed references to Mastodon. It eventually published its source code in a ZIP file on its website after public pressure and formal requests from the Software Freedom Conservancy and Mastodon.

OPEN SOURCE LICENSES

Legal action: Mastodon sent a formal notice to Truth Social, giving the company 30 days to comply or face potential legal action, including the possibility of having its right to use the code revoked.

Current status: While Truth Social eventually complied by publishing its source code, the initial controversy highlighted the tension between open-source principles and commercial interests. The source code is now available in the website's legal section and on GitHub.

OPEN SOURCE LICENSES

(4) Mozilla Public License – Weak Copyleft

The Mozilla Public License (MPL) is a weak copyleft open-source license that allows users to freely use, modify, and distribute software. It imposes a file-level copyleft, meaning modifications to MPL-licensed files must be released under the MPL, but it permits mixing MPL code with non-MPL code (including proprietary code) within the same project, as long as the MPL-licensed components remain accessible under the MPL. This "file-level" approach provides a balance between stricter licenses like the GNU GPL and more permissive ones like the MIT or BSD licenses.

OPEN SOURCE LICENSES

Key features of the MPL

File-level copyleft: The copyleft requirement applies only to the specific files that are modified, not the entire project.

Flexibility for commercial use: It allows developers to incorporate MPL-licensed code into proprietary projects, provided the MPL-licensed parts remain under the MPL and their source code is accessible.

Balance: The MPL is positioned as a middle ground between strong copyleft licenses and permissive licenses.

OPEN SOURCE LICENSES

Contribution requirements: Users must include a file documenting their changes and retain original license and copyright notices.

Attribution: Modified code must indicate it is derived from the original and include the original developer's name.

No warranty: Like most open-source licenses, the MPL disclaims warranties and limits liability.

OPEN SOURCE LICENSES

Controversies surrounding the Mozilla Public License

The main controversy surrounding the Mozilla Public License involves user confusion and backlash over its **updated terms of service for the Firefox browser**, particularly concerning data use and licensing.

Users expressed concern over vague language that seemed to grant Mozilla a broad license to their data, which they feared could lead to misuse.

Mozilla clarified the license was necessary for basic browser functions and did not grant them ownership or rights to misuse personal data.

Other MPL controversies

License abuse: A separate controversy involved a situation where a project developer used the Mozilla Public License to file complaints and enforce copyright, leading to the shutdown of another open-source project.

Debate on ethical use: This incident sparked a debate within the open-source community about whether it is ethical to use the license in this manner to shut down a project, even if the claims were legally valid.

"Exploitation" risk: Another issue noted by some developers is that the file-level copyleft nature of MPL 1.1 might allow proprietary modules to be bundled without fully exposing underlying changes, a gap that could lead to exploitation by large companies.

OPEN SOURCE LICENSES

There was a notable controversy in 2021 where the **Pale Moon project** team was accused of misusing the terms of the Mozilla Public License (MPL) to shut down a fork of their project named Mypal. The Mypal project aimed to continue supporting older Windows operating systems (XP and Vista), which Pale Moon had ceased supporting.

The dispute centered on the interpretation of the MPL's requirements for providing source code:

OPEN SOURCE LICENSES

Origins of the conflict

Initial fork: MyPal was initially a fork of Pale Moon, designed to continue support for older systems like Windows XP after Pale Moon moved to support more modern systems.

Licensing issues: A dispute arose between the two projects regarding the licensing of the code.

Code rebase: The conflict led MyPal to rebase its codebase on Firefox Quantum, causing a split from the Pale Moon project.

OPEN SOURCE LICENSES

Subsequent developments

MyPal's path: After the split, MyPal's code was based on Firefox's ESR68-78 codebase, rather than Pale Moon's own development.

Pale Moon's path: Pale Moon continued to develop independently, focusing on its own features and requiring more modern systems, while MyPal continued to support legacy systems like Windows XP.

Branding and trademarks: Pale Moon's lead developer has asserted ownership of the project's name, logo, and trademarks.

OPEN SOURCE LICENSES

Key takeaways

The controversy was primarily a licensing and code-licensing dispute, not a difference in technical direction.

The two browsers ultimately diverged, with MyPal focusing on supporting older operating systems and Pale Moon focusing on modern systems.

OPEN SOURCE LICENSES

Conclusions:

Open source licenses are designed to allow the free exchange of software, code and developer resources, without the opaqueness and restrictions of closed-source, proprietary licensing types.

Copyleft is an important term for understanding how open source code and software projects differ from proprietary, copyrighted code and software projects.

OPEN SOURCE LICENSES

Issues have arisen when differing licensing types have resulted in differing interpretations of how much sharing and forking is permitted, and how to provide attribution where it’s required. Some abuses have included taking open source code into projects which are then made proprietary. Some issues also involve developers withdrawing their code from open source projects despite the requirement to keep the code available. (The Linux kernel developers had such a controversy.)

OPEN SOURCE LICENSES

The idea behind open source development is not to let everyone have cost-free access to code or products which take developer time and effort to produce. Instead, the main idea is to allow code to be seen and evaluated by everyone, and used in a variety of projects. There are ways other than charging money for code to create a revenue stream, and leaving a fork of a project open can provide a good test bed for a paid, proprietary product or service. Even if the code and the software are free, support options can still be paid, providing a further revenue stream.

From an end user perspective, open source software provides a low cost or no cost way to try out software, and to have a basic library of software for everyday tasks.

OPEN SOURCE LICENSES

Personally, while I am a heavy user of open source software, I do pay for utilities like backup programs, the Windows operating system, and other specialized software for which I find open source alternatives inadequate.

Now is the time for discussion, comments, opinions and questions.

I invite those with better legal and technical knowledge than myself to fill in the details of how open source licensing works.

OPEN SOURCE LICENSES

-- Bob Primak --

-- for the Lexington Computers and Technology Users Group (LCTG) --

-- Wed., November 19, 2025 --

-- Revised Sunday, November 16, 2025. 2:30 PM EST --