Introduction to the Licenses

GNU philosophy

Written by Brett Smith and Richard Stallman.

This document is part of GNU philosophy, the GNU Project's exhaustive collection of articles and essays about free software and related matters.

Introduction to the Licenses

This part contains the text of the latest versions of the primary GNU licenses: the GNU General Public License (GNU GPL), the GNU Lesser General Public License (LGPL), and the GNU Free Documentation License (FDL). Though they are legal documents, they belong in this book of essays because they are concrete expressions of the ideals of free software.

Software development for the GNU operating system began in 1984. Once Richard Stallman had parts of the GNU system that were worth releasing, he needed a license to release them under. Some free software licenses already existed; these gave users permission to modify and redistribute the software, but they also allowed using the code in proprietary versions and proprietary programs. Using those licenses, GNU would have failed to achieve its goal of delivering freedom to all users, because middlemen would have converted the GNU code into proprietary software.

So Stallman devised a license to assure every user the freedom to modify and redistribute the software. It granted these permissions under one key condition: whoever distributed the software must pass along the authorization to modify and redistribute that same software, along with the source code making it practical to do so. Stallman coined the term "copyleft" (see *What Is Copyleft?*) to describe this key twist of using the legal power of copyright to ensure freedom for all users.

GNU copyleft licenses were first developed for software, and later for related areas such as software documentation. In them, the principles of the free software movement, explained throughout the essays in this book, take practical form. Each of their successive revisions has had to wrestle with free software's legal and practical obstacles and offers numerous illustrations of how free software ideals are codified into legal terms.

The Origins of the GPL

The first version of the GNU General Public License was published in 1989—but Stallman had been releasing software under copyleft licenses as part of the GNU Project since as early as 1985. Prior to 1989, each published GNU program had been covered by a license specifically tailored for it. Instead of a single GNU General Public License, there was a GNU CC General Public License, a GDB General Public License, and so on. These licenses were identical except for minor differences: for instance, terms about displaying license notices to users were different for different programs and, unless it covered a program that was just one source file, each license contained the name of the program it applied to.

By 1989, Stallman had had enough experience with different GNU packages under slightly different licenses to conclude that it was crucial to unify them into one license that would cover all these packages. He worked with Jerry Cohen, an attorney at Perkins Smith & Cohen LLP, to collect concepts from all the different licenses written up to that point, and bring them together into one license. It was thus that on 1 February 1989 the GNU General Public License was born.

The first version of the license sought to ensure two results: first, that all derived works of the software would be released under the same license and, second, that everyone who received the software would have a chance to get the source code. These requirements im-

plement a strong copyleft by blocking the three main ways of making programs proprietary: with copyright, with end user license agreements, and by not distributing source code.

In comparison to the program-specific licenses that had preceded it, GPL version 1 featured few substantial changes—the GPL was evolutionary, not revolutionary—but it made a big practical difference. Previously, developers who had wanted to copyleft a program had needed to tailor one of the existing licenses to that program. Many had not bothered. With the release of the GPL, those developers had a license they could use out of the box to provide all of their users with freedom to share and change the software. It was a powerful tool.

Version 2

After the 1981 US Supreme Court decision in *Diamond v. Diehr*, the US Patent and Trademark Office began issuing patents for software. Software patents threaten free software and proprietary software alike (see part IV in this book), and Stallman realized that they could subvert the copyleft in the GNU GPL.

By selectively issuing patent licenses, patent holders can arbitrarily control how the software under them is distributed or modified. A patent holder can give one party permission to resell the program, another permission to develop and use a modified version at her company, and a third permission to do all the activities that the GPL itself allows. They can demand whatever they wish in exchange for these permissions. They have this power over any software that implements the patented idea, whether or not they have modified or distributed it themselves. This power threatens free software because third parties with patents can impose restrictions on free software users and developers.

If patent holders don't distribute or modify software, then a software license based on copyright like the GPL can't control their activities: they haven't done anything that requires permission under the license. But the software license can stop each of the program's distributors from entering limiting agreements with the patent holder. Enter GPL version 2: a new section in the license (sec. 7) explicitly says that if parties are subject to other legal agreements—such as a patent license—that contradict the GPL's terms, then the licensee must refrain from distributing the software at all. As a result, any party that wants to distribute or modify the software, and also obtain a patent license, must ensure that the terms of that license are consistent with all of the GPL's conditions: recipients of the software must receive it under the same terms, with no additional restrictions, and have the means to get the source code.

This new section protected the integrity of the distribution system for GPL-covered software. A fundamental principle of the license is that every licensee, from the most humble individual to the largest corporation, has the exact same rights to share and change the software. Patent holders who do not distribute the software themselves and selectively issues patent licenses could potentially interfere with this goal, splitting licensees into different groups however they see fit. Section 7 of GPL version 2 prevents this abuse.

The LGPL

The GPL worked well for the programming tools, utilities, and games that were released by the GNU Project in the early years; however, Stallman recognized that releasing the recently developed GNU C Library the same way could backfire. Aside from some extensions, the

GNU C Library was to be a compatible replacement for the UNIX C Library, so any C program would be able link with either one. If proprietary C programs were not allowed to use the GNU C Library, they would simply use the UNIX library. Being strict in this case would gain nothing.

Stallman decided to compromise with a modified copyleft: one that would protect the freedom of the library itself, but not that of the programs that use it. This idea was implemented in a license originally called the GNU Library General Public License, first published as version 2.0, in June 1991. The original LGPL stated Conditions like the GPL's—with an important exception: if someone else's program used the library only by referring to it as a library, that program's source could be distributed under license terms of the author's choosing. However, the executable made by combining the program and the library had to come with a copy of the LGPL and source code for the library, and provide some mechanism for users who have modified the library to update the executable to use their modified library.

How does a developer use the work as a library in order to take advantage of the special set of conditions provided by LGPLv2? Think of a computer program as a series of instructions for doing a particular job: compiling or linking the program with a library provides the programmer with a means to say, "When the program gets to this point, get further instructions from the library, and come back here when those are done." Libraries are commonly used in software development because they make the effort less repetitive and less error prone: programmers don't have to reinvent the wheel—and perhaps introduce bugs in the process—every time they want to accomplish a particular task. Because libraries are so widely created and used, developers have the means to readily take advantage of the LGPL's additional permissions.

Version 2.0 of the license worked as intended: in some situations, proprietary software developers chose to use an LGPL-covered library over a proprietary alternative, and users received the freedom to share and change that library. This did not produce an "ideal" outcome—where the user had complete control over the entire program—but in these cases the GPL would not have achieved that ideal outcome either. The LGPL assured the users some freedom where they would have otherwise had none.

The name "Library GPL" led some free software developers to assume all libraries ought on principle to be licensed this way, but that was not the intent—when a free library has no proprietary competitor, releasing it under the GNU GPL can benefit free software. To avoid this unintended message, Stallman renamed this license to the Lesser General Public License, and incremented the version number to 2.1 to reflect the relatively minor changes in the text: the license sported a new preamble, a few wording clarifications, and allowed programs to make their calls to the library through special system facilities for shared libraries where those are available. The Lesser General Public License version 2.1 was released in February 1999.

The FDL

At the turn of the century, free software was growing much faster than it had been previously; the documentation, however, was not keeping pace. Stallman was concerned about this failure and wrote about it in *Why Free Software Needs Free Documentation*.

While there are some similarities between software and documentation—they are both works that are meant for practical use—there are important differences in the ways they can be used. The GPL and the LGPL were not suitable for manuals.

For some time, GNU packages had been using an untitled, simple, ad hoc copyleft license for each manual. Since each manual's license was different, text could not be copied from one manual to another. So Stallman wrote the GNU Free Documentation License, a copyleft license designed primarily for software documentation and other practical written works.

The FDL was first published in March 2000. The principles of the copyleft remain the same: everyone who receives a copy of the work should be able to modify and redistribute it. Where the FDL differs from the software licenses is in the details of its implementation: conditions about how to attribute the work and provide "source code"—an editable version of the document—are different.

Version 3

During the 1990s, as free software became more popular, the GPL emerged as the clear copyleft license of choice for the community, and was adopted by the majority of free software projects; at the same time, however, proprietary developers had come up with methods of effectively denying users the freedoms that the GPL was meant to protect, without actually violating the GPL. In addition, there were other practices that the GPL did not handle conveniently. To deal with these issues called for an updated version of the license.

Around 2002, Stallman and others at the Free Software Foundation began discussing how to update the GPL, and the LGPL along with it. The FSF established a public review process, run with help from attorneys at the Software Freedom Law Center, to catch possible problems before actually releasing the new licenses. Committees of advisors from the community studied issues raised by public comments and reported the various positions and arguments to Stallman, who decided what policy to adopt; then he wrote license text with advice and suggestions from the attorneys. The importance of the changes made are explained in *Why Upgrade to GPLv3*.

Version 3 used new terminology to promote uniform interpretations in different jurisdictions, and modified some requirements to fit new practices in the free software community. Beyond that, it introduced several new conditions to strengthen the copyleft and thereby the free software community as a whole. For instance, it

- blocked distributors from restricting users by building hardware that rejects the users' modified versions ("tivoization");
- allowed code to carry limited additional requirements, for compatibility with some other popular free software licenses;
- and strengthened patent requirements by providing clear terms to handle patent cross-licenses, which are common arrangements between large patent-holding companies.

Both GPLv3 and LGPLv3 included terms to address all of these issues, and were finally released on 29 June 2007. These licenses are the state of the art in copyleft, going farther than any other software license to protect users' freedom and bring about a world in harmony with the ideals expressed in this book.