The History of the GNU General Public License

A test document for site search indexing — tracing the origins and evolution of the world's most influential free software license

The Free Software Foundation and the Origins of Copyleft

The GNU General Public License, universally known as the GPL, is the foundational legal instrument of the free software movement. Its origins lie in the experiences and convictions of Richard M. Stallman, a programmer at the MIT Artificial Intelligence Laboratory in the late 1970s and early 1980s. Stallman's famous account of his frustration with a Xerox printer whose source code he could not modify to fix a recurring paper jam — and whose manufacturer refused to share the code — crystallized for him a broader problem: the increasing enclosure of software as proprietary property was destroying the collaborative culture that had made computing thrive.

In 1983, Stallman announced the GNU Project, an effort to create a complete free Unix-compatible operating system. In January 1984 he resigned from MIT to work on GNU full time, and in 1985 he published the GNU Manifesto in Dr. Dobb's Journal, articulating the philosophical and practical case for free software. That same year he founded the Free Software Foundation (FSF) to support the project legally and financially. Critically, Stallman defined 'free' not in terms of price but in terms of freedom: the freedom to run, study, modify, and redistribute software.

Copyleft: Turning Copyright on Its Head

The central legal innovation of the GPL is the concept of copyleft — a deliberate play on the word copyright. Stallman and his collaborators realized that simply releasing software into the public domain was insufficient: anyone could take the code, make modifications, and redistribute the result as proprietary software, effectively re-enclosing the commons. Copyleft solved this by using copyright law itself as the enforcement mechanism. Under a copyleft license, users are granted extensive freedoms, but any distribution of modified versions must carry the same freedoms. The license propagates forward through the software supply chain.

This mechanism is sometimes described as 'viral' by critics and 'reciprocal' by proponents. The terminology reflects the same underlying reality: GPL-licensed code, when incorporated into a larger work that is then distributed, requires the entire combined work to be distributed under GPL terms. This property has made the GPL both enormously influential and, in some commercial contexts, deeply controversial.

GPL Version 1 (1989)

The first version of the GNU General Public License was released in January 1989. GPLv1 established the core framework that all subsequent versions would build upon: the four freedoms (to run, study, modify, and share), the requirement that source code accompany binary distributions, and the copyleft obligation that derivative works be licensed under the same terms. The license was drafted by Stallman with legal input, and it was notably written in relatively plain language for a legal document — a deliberate choice to make the terms comprehensible to the programmers who would be using and relying on it.

GPLv1 also contained a provision addressing a loophole: a distributor could not impose additional restrictions on recipients beyond those in the license itself. This prevented the strategy of distributing GPL software while adding contractual terms that clawed back the freedoms the license granted.

GPL Version 2 (1991)

GPLv2, released in June 1991, is arguably the most historically significant version of the license, as it governed the Linux kernel (adopted by Linus Torvalds in 1992) and remained Linux's license for over a decade. GPLv2 addressed a problem that had emerged with the growing diversity of computing platforms: a distributor could comply with the letter of GPLv1 by providing source code while making the binary impossible to run on the target hardware through technical means. GPLv2's 'Liberty or Death' clause — formally Section 7 — addressed this by stating that if a distributor could not comply with all of the license's terms simultaneously, they could not distribute the software at all.

The Liberty or Death clause was a direct response to software patent threats. If a patent holder required royalties for software that also happened to be GPL-licensed, the distributor would be caught between incompatible obligations — and would therefore be prohibited from distributing the software. This provision made the GPL a significant force against software patent encumbrance of free software projects.

GPLv2 also introduced the concept of 'mere aggregation,' clarifying that including GPL software on the same distribution medium as proprietary software — such as on a CD-ROM containing both — did not by itself trigger copyleft obligations for the proprietary components, so long as the programs were genuinely independent and did not form a combined work.

The Lesser GPL and the LGPL Strategy

Recognizing that strict copyleft was not always strategically appropriate, the FSF released the GNU Library General Public License in 1991 alongside GPLv2, later renamed the GNU Lesser General Public License (LGPL). The LGPL was designed for software libraries — particularly the GNU C Library (glibc) — where requiring all programs that linked against the library to be GPL-licensed would have been counterproductive. The LGPL permits proprietary programs to link against LGPL libraries without the copyleft obligation propagating to the proprietary code, while still requiring that modifications to the library itself remain free software.

The FSF has historically been ambivalent about the LGPL, viewing it as a necessary concession in some contexts but not a general model. Stallman wrote in 1999 that using the ordinary GPL rather than the LGPL is often the better choice for libraries when there is no significant free alternative, since the stronger copyleft gives the free software community more leverage.

The Tivoization Problem and the Road to GPLv3

By the early 2000s, the GPL had achieved enormous success — it governed the Linux kernel, the GNU tools, and thousands of other projects. But new challenges had emerged that the drafters of GPLv2 had not fully anticipated. The most significant was the practice that came to be called 'Tivoization,' named after TiVo's digital video recorders. TiVo shipped devices running Linux under GPLv2 compliance: they provided full source code as required. But the devices used cryptographic signing to verify that only TiVo-approved binaries would run on the hardware. Users could modify the source, compile it — and then find that their modified version would not run on the device they owned.

From Stallman's perspective, this violated the spirit if not the letter of the GPL. The freedom to modify software is meaningless if you cannot run your modifications on the hardware you own. From the perspective of companies like TiVo and, more significantly, Linus Torvalds and many Linux kernel developers, the hardware lockdown was a legitimate business and security practice that the GPL had not prohibited.

This disagreement became one of the central fault lines in the free software community during the GPLv3 drafting process, which began formally in 2006 with an unprecedented global public consultation organized by the FSF.

GPL Version 3 (2007)

GPLv3 was released on June 29, 2007, after an eighteen-month drafting process involving four public drafts, hundreds of comment submissions from individuals and organizations, and intense debate. The final text addressed several major concerns beyond Tivoization. It contained explicit provisions addressing Digital Rights Management (DRM), stating that GPL software should not be used as part of effective technological measures that restrict users. It included an explicit patent retaliation clause: any contributor to GPL software implicitly grants a patent license to all recipients, and any party that initiates patent litigation against GPL software loses their right to distribute it.

GPLv3 also included a compatibility provision for the Apache License 2.0, resolving a long-standing incompatibility between the two licenses. And it added a 'Corresponding Source' definition that required distributors to provide not just source code but the scripts and tools needed to compile and install modified versions — directly targeting the Tivoization loophole for user products.

The Linux kernel community, led by Torvalds, declined to upgrade to GPLv3. Torvalds had publicly criticized the anti-Tivoization provisions, arguing that hardware manufacturers have legitimate reasons

to control what runs on their devices. As of the mid-2020s, the Linux kernel remains under GPLv2 with the 'or any later version' clause removed — a deliberate choice to lock the kernel to GPLv2 perpetually. This split between the FSF and the Linux community remains one of the defining tensions of the free software ecosystem.

The Affero GPL and Network Services

A further gap in the GPL framework emerged with the rise of web applications and Software as a Service (SaaS). Under GPLv2 and GPLv3, the copyleft obligation is triggered by distribution of software. But running a modified GPL program on a server and offering its functionality to users over a network does not constitute distribution — users never receive the software, only its outputs. A company could therefore take GPL software, make extensive modifications, run it as a web service, and never be required to share those modifications.

The GNU Affero General Public License (AGPL), first published in 2002 and significantly revised as AGPLv3 in 2007 to be compatible with GPLv3, closes this loophole. The AGPL adds a 'network use' provision: if a modified version runs on a server accessible to users over a network, those users must be able to obtain the source code of the modified version. The AGPL has been adopted by projects including MongoDB (in its earlier history), Nextcloud, and many others that operate primarily as networked services.

Legacy and Contemporary Relevance

The GPL family of licenses — GPL, LGPL, and AGPL — remains the most widely used category of free software license. According to various surveys of open source software repositories, GPL variants consistently govern a plurality of open source projects, though permissive licenses like the MIT License and Apache License 2.0 have gained significant ground since the mid-2000s. The shift toward permissive licensing reflects in part the success of the free software ecosystem: with so much high-quality free software available, the defensive logic of copyleft became less urgent for many developers, while the frictionless adoption of permissive software by commercial actors became more attractive.

For developers and organizations working with open source software — including those building on the Drupal content management system and its ecosystem, which is itself licensed under GPLv2 or later — understanding GPL compliance remains a practical necessity. Key compliance obligations include preserving copyright notices, providing or offering source code, and ensuring that GPL-licensed components are not combined with incompatible licenses in ways that create distribution obligations the distributor cannot fulfill.

The GPL's legacy extends beyond its direct legal effect. It established that software licensing could be used as an instrument of social policy — not merely to protect intellectual property rights, but to actively shape the structure of the software industry and the freedoms available to software users. Whether one

views this as an inspired subversion of copyright or an ideological imposition on practical software development, the GPL's influence on the history of computing is beyond dispute.

— Test document generated for RSVP-System site search index validation —