Open Source Software Paradigms

Lecture - 04

Software Licensing Models

Lecturer: Biniam Behailu
Addis Ababa Science and Technology University

Learning Objectives

By the end of this lecture, you will be able to:

- Define what a software license is and explain its core purpose.
- Differentiate between proprietary and open-source software licensing models.
- Identify and describe key proprietary license models: Perpetual, Subscription (SaaS), and Concurrent.
- Compare and contrast permissive (e.g., MIT, Apache) and copyleft (e.g., GPL) open-source licenses.
- Understand the real-world implications of choosing a specific software license.

Contents

What is Software Licensing?

A **software license** is a legal agreement between the developer/owner (the "licensor") and the user (the "licensee") that defines the rights and restrictions for using the software.

Why Licenses Matter?

Protects Intellectual Property (IP), clarifies liability, and sets the commercial terms.

***Image source**: PNGTree, "3d White People With Shopping Cart Merchandise," PNGTree. [Online]. Available: https://pngtree.com/freepng/3d-white-people-with-shopping-cart-merchandise_13318828.html. [Accessed: Oct. 1, 2025].*

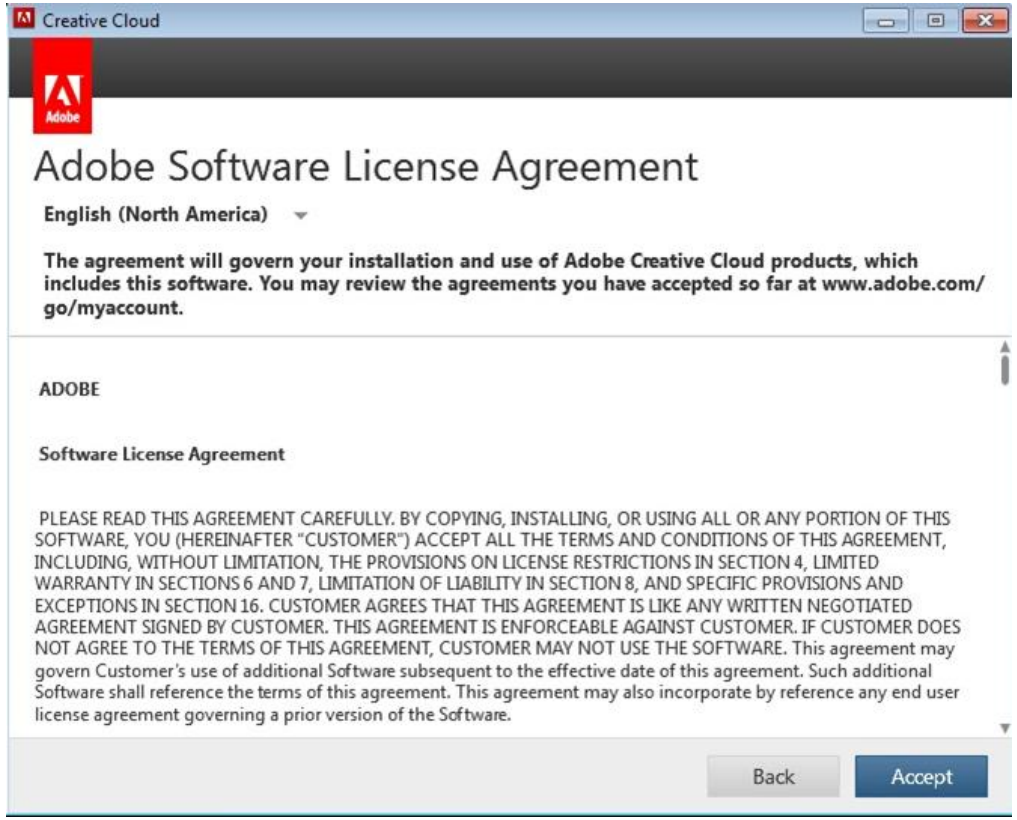
What is Software Licensing?

Core Rights Covered

- **Installation** and **use** (how many copies/users).
- **Modification** (can the code be changed?).
- **Distribution** (can the software be shared/sold?).
- **Liability** & **Warranty** (Is the software is provided "as is" with no warranty).

***Image source**: PNGTree, "3d White People With Shopping Cart Merchandise," PNGTree. [Online]. Available: https://pngtree.com/freepng/3d-white-people-with-shopping-cart-merchandise_13318828.html. [Accessed: Oct. 1, 2025].*

What is Software Licensing?



- A License agreement governs the use of licensed software.
- A license agreement on an installer is an End User License Agreement (EULA), a legally binding contract presented during software installation that grants users permission to use the software under specific terms set by the developer, without actually buying the software itself.

***Image source**: Adobe, "Unable to accept End User License Agreement (EULA)," Adobe Help Center. [Online]. Available: https://helpx.adobe.com/x-productkb/global/unable-to-accept-end-user-license.html. [Accessed: Oct. 1, 2025].*

Why do installers include a EULA?

Protect intellectual property

- Developers use EULAs to safeguard their ownership of the software.

Define usage rights

- They grant users permission to use the software while setting limits on activities like copying, distributing, or modifying it.

How does it work?

- **Presentation**: The installer displays the EULA text in a dialog box.
- **Agreement**: The user must read and agree to the terms, often by selecting an "I agree" button.
- **Continuation or termination**: If the user accepts, the installation proceeds; if they do not accept, the software is not installed, and the process ends.

Types of Software Licensing

The Two Major Camps of Software Licensing

Proprietary (or Closed-Source) Licensing

Open Source Licensing

Public Domain

***Image source**: Hypertec SP, "Software Licensing for Businesses," Hypertec Solutions Partner. [Online]. Available: https://hypertecsp.com/knowledge-base/how-software-licensing-effects-businesses/. [Accessed: Oct. 1, 2025].*

Proprietary (Closed-Source) Licensing

- Proprietary (or Closed-Source) Licensing is the most common model for commercial software, where the developer retains full control and ownership of the source code.
- Proprietary licensing refers to software that is owned by an individual or a company.
- The source code is not available for public access or modification.
- The is to protect intellectual property, maintain control over the software, and generate revenue through sales or licensing fees.
- This model is defined by a set of restrictive rights granted to the user, enforced primarily through the End User License Agreement (EULA).

Proprietary License Grant Models

Perpetual License

Subscription License (SaaS)

Concurrent / Floating License

Perpetual Licensing (Own It Forever)

- The traditional model where the customer pays a single, upfront fee for the right to use the software indefinitely.
  - ☞ **One-time purchase**: The license grant is permanent, providing stability for the user.
  - ☞ **Maintenance & Support fees**: Often requires optional separate Maintenance & Support contract fees to receive updates, patches, and technical support.

Note
This model is declining as vendors shift to subscription based services for recurring revenue.

Subscription & SaaS (The Access Model)

- Pay a recurring fee (monthly/annual) for access to the software for a defined period. Access is revoked if payments stop.
- It is a dominant delivery model for Software-as-a-Service (SaaS).

- The "**As-a-Service**" Stack:
  - ☞ **IaaS (Infrastructure):** Rent virtualized computing resources. (AWS EC2, Azure VMs).
  - ☞ **PaaS (Platform):** Rent a platform to build, deploy, and manage applications. (Heroku, Google App Engine).
  - ☞ **SaaS (Software):** Rent a fully functional, ready-to-use application. (Salesforce, Slack, Dropbox, Gmail).

Subscription & SaaS (The Access Model)

Core Characteristics of SaaS Licensing

- **Access over Ownership**: Customers subscribe to a service; they do not own the software.
- **Centralized Management**: The vendor manages all updates, security, and infrastructure.
- **Recurring Revenue**: The business model is based on subscription fees (Monthly or Annual Recurring Revenue - MRR/ARR).
- **Multi-Tenancy**: A single software instance serves multiple customers ("tenants") with data segregation.

Subscription & **SaaS** (The Access Model)

1. Subscription-Based Licensing

- Users pay a recurring fee (monthly or annually) to access the software. This model often includes updates and support.
- **Example**: Microsoft 365, Adobe Creative Cloud

2. Freemium Model

- Basic features are offered for free, while advanced features or additional services require payment. This model helps attract a large user base.
- **Example**: Dropbox, Slack

Subscription & **SaaS** (The Access Model)

3. Pay-As-You-Go Model

- Users are charged based on actual usage, such as the number of transactions, data storage, or active users. This model allows for flexibility and scalability.
- **Example**: Amazon Web Services (AWS), Twilio

4. Tiered Pricing Model

- Different pricing tiers are offered based on features, user limits, or performance levels. This model allows businesses to choose a plan that fits their needs and budget.
- **Example**: HubSpot, Mailchimp

Subscription & **SaaS** (The Access Model)

5. User-Based Licensing

- Licenses are purchased based on the number of users who will access the software. This model is common in enterprise applications.
- **Example**: Salesforce, Zendesk

6. Site Licensing

- A single license allows unlimited access to the software within a specific organization or location, often used by educational institutions or large enterprises.
- **Example**: Google Workspace for Education, Microsoft Teams

7. Enterprise Licensing

- Customized licensing agreements for large organizations, often negotiated based on specific needs, user counts, and features.
- **Example**: SAP, Oracle Cloud

Concurrent & Floating Licenses (The Optimization Model)

The Problem Statement

"We have **1,000** engineers, but statistical usage shows only **150** can use the software **simultaneously**. Buying **1,000** licenses is financially irresponsible."

***Image source**: Adobe Stock, "White 3D Man Images – Browse 531,806 Stock Photos, Vectors, and Video," Adobe Stock. [Online]. Available: https://stock.adobe.com/. [Accessed: 1 October 2025].*

Concurrent & Floating Licenses (The Optimization Model)

- Concurrent and floating licenses allow multiple users to share a limited number of licenses for software [1].
- This model optimizes software usage and costs, particularly in environments where not all users need access at the same time.

Key solution

- A **License Server** (or License Manager) hosts a pool of licenses that are shared ("float") among all potential users.

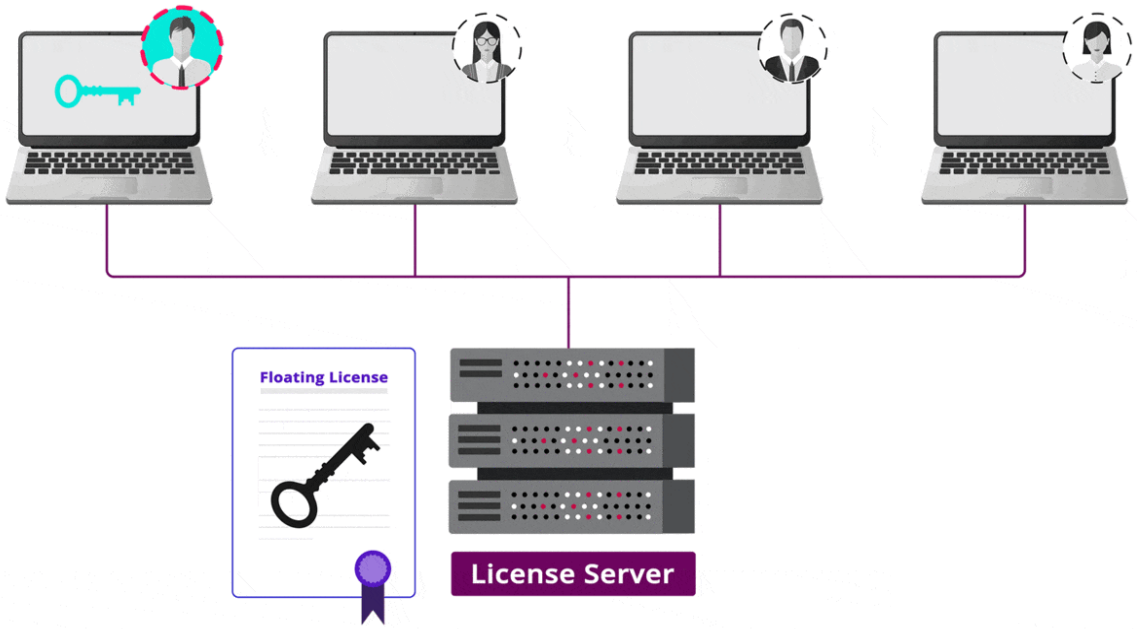


***Image source**: RealPars, "What Is a Floating License?," RealPars. [Online]. Available: https://realpars.com/floating-license/. [Accessed: Oct. 1, 2025].*

Concurrent & Floating Licenses (The Optimization Model)

1. The software client is installed on all user machines.
2. Upon launch, the client contacts the central license server.
3. The server checks the pool. If a license is available, it "checks out" the license to the user.
4. The available license count is decremented.
5. When the user closes the application, the license is "checked in" and the count is incremented.
6. If the pool is empty, the next user is put in a queue or denied access.

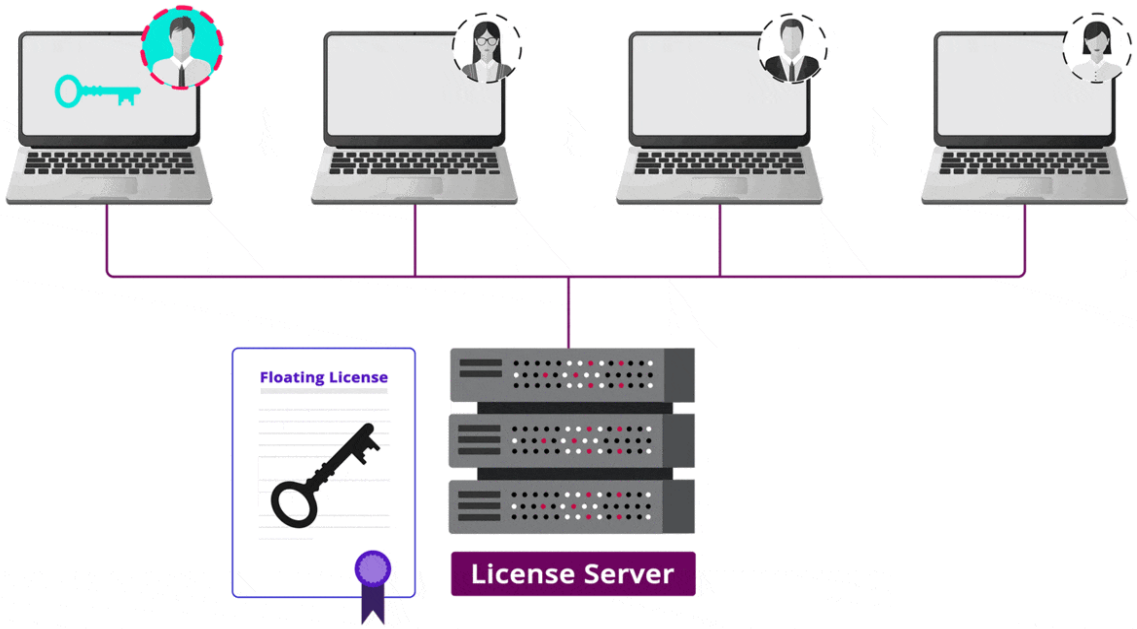


***Image source**: RealPars, "What Is a Floating License?," RealPars. [Online]. Available: https://realpars.com/floating-license/. [Accessed: Oct. 1, 2025].*

Proprietary Licensing Summary

Feature	Description
Source Code	The source code is **not made available** to the licensee. It is the proprietary trade secret of the vendor.
Modification	Users are strictly **prohibited** from modifying, adapting, translating, or creating derivative works from the software.
Redistribution	Users cannot copy, distribute, or sell the software (or any part of it) to others without explicit permission from the vendor.
Reverse Engineering	The EULA explicitly **forbids** attempting to deconstruct the software (e.g., through decompiling or disassembling) to reveal the source code.
Ownership	The **copyright and intellectual property (IP) remain entirely with the vendor** (licensor). The user (licensee) is only granted a non-exclusive, non-transferable right to use the software.

Open Source Licensing

- Open source licenses are legal instruments that grant users the right to access, use, modify, and share software source code under defined terms.
- These definitions are strictly governed by the **Open Source Initiative's** (OSI) Open Source Definition, ensuring a common understanding and adherence to principles of transparency and collaboration.
- To be OSI-approved, a license must comply with the criteria set in the Open Source Definition [2].

Open Source Licensing

- Key aspects to consider when choosing a license:

  - **Permissiveness**: How restrictive the license is.
  - **Copyleft**: Whether the license requires derivative works to also be open source.
  - **Patent Grants**: Whether the license includes patent grants.
  - **Commercial Use**: Whether the license allows commercial use.
  - **Compatibility** with other types of free licenses
  - **Enforcement**: Consider how you will enforce the license and the implications of any violations.

Two Main Types of Open Source Licenses

- Understanding the difference between Permissive and Copyleft is fundamental to navigating the OSS landscape.

Permissive Licenses

Minimal restrictions. Primarily focused on allowing proprietary use and easy integration.

Copyleft Licenses

Stronger restrictions. Ensures all derived works also remain open source under the same terms.

1. Permissive Licenses

- Permissive licenses are, also known as **Non-Copyleft/ Liberal** licenses.
- They are a category of Free and Open Source Software (FOSS) licenses that **grant maximum freedom** to users with **minimal restrictions** on how they can use, modify, and redistribute the software.
- Allow users to do almost anything with the software, including using it in proprietary products.

> **Philosophy**
>
> "Do whatever you want with this code, just give me credit and don't sue me."

- Commonly used Permissive Licenses are **MIT License**, **Apache License 2.0**, **BSD License**.

The MIT License

- The MIT License, also known as the **MIT Open Source License**, is one of the **most permissive** software license that grants users a wide range of freedoms to use, modify, and distribute software.
- Its permissive nature makes it compatible with many other licenses, facilitating integration with other projects.
- Highly compatible with almost every other license, including strong copyleft licenses (like GPL) and proprietary licenses.

Important Links

The MIT License – Open Source Initiative

Exploring the MIT Open Source License: A Comprehensive Guide | MIT Technology Licensing Office

***Image source**: Licen.cc, "Licencia MIT," Licen.cc, [Online]. Available: https://www.licen.cc/es/licencias/mit/. [Accessed: Oct. 1, 2025].*

Important Links

The MIT License – Open Source Initiative

Exploring the MIT Open Source License: A Comprehensive Guide | MIT Technology Licensing Office

Key Features of MIT License

- **Permissiveness**: It places **minimal** restrictions on how you can use the software. You can use it for any purpose, **including commercial use**.
- **Attribution**: The only requirement is to include the **original copyright notice** and the **license text** in any copies or modifications you make.
- **No Warranty**: The software is provided **"as is"** without any warranty or guarantee of its quality or performance.

Limitation

It does not contain an explicit patent grant, which means a contributor who holds a relevant patent is not explicitly prevented from suing users of the code.

***Image source**: Licen.cc, "Licencia MIT," Licen.cc, [Online]. Available: https://www.licen.cc/es/licencias/mit/. [Accessed: Oct. 1, 2025].*

Important Links

The MIT License – Open Source Initiative

Exploring the MIT Open Source License: A Comprehensive Guide | MIT Technology Licensing Office

Popular software projects that are licensed under the MIT License

- Ruby on Rails
- jQuery
- Node.js
- Vue.js
- Bootstrap
- Django
- Electron
- AngularJS
- Pillow
- .Net Core

***Image source**: Licen.cc, "Licencia MIT," Licen.cc, [Online]. Available: https://www.licen.cc/es/licencias/mit/. [Accessed: Oct. 1, 2025].*

Important Links

Apache License, Version 2.0

Apache License, Version 2.0 – Open Source Initiative

The Apache License

- It is a permissive open-source software license developed by the Apache Software Foundation (ASF).
- It's known for its balance between permissive use and copyright protection.
- The most commonly used version is Apache License 2.0, released in January 2004.
- The explicit patent protection makes it the preferred choice for large enterprise contributions where IP is a major concern.

***Image source**: The Apache Software Foundation, "Introducing The ASF's New Logo," The ASF Blog, Sep. 11, 2025. [Online]. Available: https://news.apache.org/foundation/entry/introducing-the-asfs-new-logo. [Accessed: Oct. 1, 2025].*

Important Links

Apache License, Version 2.0

Apache License, Version 2.0 – Open Source Initiative

Key Features of Apache License

- **Permissiveness**: Like the MIT License, the Apache License allows for a wide range of use, including commercial purposes.
- **Patent Grant**: A significant feature of the Apache License is the patent grant. If you sue someone for patent infringement related to the software, the license holder can revoke your rights to use the software. This clause is designed to discourage patent trolls and protect the open-source community.
- **Attribution**: You must include the original copyright notice and the license text in any copies or modifications you make.
- **No Warranty**: The software is provided "as is" without any warranty.

***Image source**: The Apache Software Foundation, "Introducing The ASF's New Logo," The ASF Blog, Sep. 11, 2025. [Online]. Available: https://news.apache.org/foundation/entry/introducing-the-asfs-new-logo. [Accessed: Oct. 1, 2025].*

Important Links

Apache License, Version 2.0

Apache License, Version 2.0 – Open Source Initiative

Popular software projects that are licensed under the Apache License

- Apache HTTP Server
- Apache Hadoop
- Apache Tomcat
- Apache OpenOffice
- Android
- Docker
- ElasticSearch
- Netflix OSS
- OpenStack
- Red Hat OpenShift
- Kubernetes

***Image source**: The Apache Software Foundation, "Introducing The ASF's New Logo," The ASF Blog, Sep. 11, 2025. [Online]. Available: https://news.apache.org/foundation/entry/introducing-the-asfs-new-logo. [Accessed: Oct. 1, 2025].*

Important Links

The 2-Clause BSD License – Open Source Initiative

The 3-Clause BSD License – Open Source Initiative

The BSD License

- The BSD License (**Berkeley Software Distribution License**) is a permissive free software license that originated from the Berkeley Software Distribution (BSD), a Unix-like operating system by University of California, Berkeley.
- BSD License is a family of permissive free software licenses that impose **minimal** restrictions on the use and distribution of covered software.

***Image source**: The FreeBSD Project, "BSD wordmark," Wikimedia Commons, Nov. 19, 2009. [Online]. Available: https://commons.wikimedia.org/wiki/File:BSD_wordmark.svg. [Accessed: Oct. 1, 2025].*

Important Links

The 2-Clause BSD License – Open Source Initiative

The 3-Clause BSD License – Open Source Initiative

Key Features of BSD License

- **Permissiveness**: BSD licenses are **highly permissive**, allowing users to freely use, modify, and distribute the software, even for commercial purposes.
- **Attribution**: The primary requirement is to retain the **original copyright notice** and **license text** in any distribution of the software.
- **No Warranty**: The software is provided **"as is"** without any warranty or guarantee.
- **Modification** and **Distribution**: Users can modify the software and distribute it in both **original** and **modified** forms.

Limitation

It does not contain explicit patent protections.

***Image source**: The FreeBSD Project, "BSD wordmark," Wikimedia Commons, Nov. 19, 2009. [Online]. Available: https://commons.wikimedia.org/wiki/File:BSD_wordmark.svg. [Accessed: Oct. 1, 2025].*

The 2-Clause BSD License

- The 2-Clause BSD License is also known as "**Simplified BSD License**" and the "**FreeBSD License**".
- The BSD 2-Clause license is a permissive open-source license that allows for a high degree of freedom for reuse, including for proprietary software.
- Retain copyright notice in source distributions
- Reproduce copyright notice and disclaimer in binary distributions

Important Links

The 2-Clause BSD License – Open Source Initiative

The 3-Clause BSD License – Open Source Initiative

***Image source**: The FreeBSD Project, "BSD wordmark," Wikimedia Commons, Nov. 19, 2009. [Online]. Available: https://commons.wikimedia.org/wiki/File:BSD_wordmark.svg. [Accessed: Oct. 1, 2025].*

Important Links

The 2-Clause BSD License – Open Source Initiative

The 3-Clause BSD License – Open Source Initiative

The 3-Clause BSD License

- The 3-Clause BSD License, also known as the **New BSD License** or **Modified BSD License**, is a widely adopted permissive open-source license.
- It is essentially identical to the 2-Clause BSD License, but with the **addition** of a **crucial third clause** to **protect** the **names** of the **original contributors**.
- **Prohibits** the **use** of the **author's name** to **endorse** or **promote products** derived from the software.
- More **restrictive** than the **2-clause** version but still highly permissive.

***Image source**: The FreeBSD Project, "BSD wordmark," Wikimedia Commons, Nov. 19, 2009. [Online]. Available: https://commons.wikimedia.org/wiki/File:BSD_wordmark.svg. [Accessed: Oct. 1, 2025].*

Important Links

The 2-Clause BSD License – Open Source Initiative

The 3-Clause BSD License – Open Source Initiative

4-Clause BSD License

- The 4-Clause BSD License, often called the **Original BSD License** or **Old BSD License**.
- In addition to the **copyright notice** and **disclaimer**, it requires **attribution** in **advertising** or **promotion** of the software.
- Due to the controversial nature of its fourth clause, the 4-Clause BSD License is not widely used for new projects today.

***Image source**: The FreeBSD Project, "BSD wordmark," Wikimedia Commons, Nov. 19, 2009. [Online]. Available: https://commons.wikimedia.org/wiki/File:BSD_wordmark.svg. [Accessed: Oct. 1, 2025].*

Important Links

The 2-Clause BSD License – Open Source Initiative

The 3-Clause BSD License – Open Source Initiative

Popular software projects that are licensed under the BSD License

- Golang
- Django
- Flask
- Nginx
- DrJava
- PostgreSQL
- Flutter
- Dart
- Chromium

***Image source**: The FreeBSD Project, "BSD wordmark," Wikimedia Commons, Nov. 19, 2009. [Online]. Available: https://commons.wikimedia.org/wiki/File:BSD_wordmark.svg. [Accessed: Oct. 1, 2025].*

Feature	MIT License	Apache License 2.0	BSD License
Freedom to Use	Yes	Yes	Yes
Modification Rights	Yes	Yes	Yes
Distribution Requirements	Must include copyright notice and license	Must include copyright notice and license; also includes a NOTICE file for attributions	Must include copyright notice and license
Patent Grant	No	Yes	No (3-Clause includes some patent protections)
Attribution Required	Yes	Yes	Yes
Compatibility	Highly compatible with other licenses	Compatible with many other licenses, but can have complexities with certain licenses	Generally highly compatible
No Warranty	Yes	Yes	Yes

Permissive Licenses Summary

2. Copyleft Licenses

- Copyleft is a type of open-source license that ensures software remains free and open for all subsequent users and developers.
- Require derivative works to be licensed under the same or compatible terms.
- This ensures that the freedoms granted by the original license are preserved in all subsequent versions of the software.

Philosophy

You can use, modify, and distribute the software, but if you distribute it, you must also share your modifications under the same license.

Core Idea of Copyleft

Action	Condition
Use the software privately	✅ Allowed
Modify it for personal or internal use	✅ Allowed
Redistribute original	✅ Must include source + license
Redistribute modified version	✅ Must also use the same copyleft license
Incorporate into proprietary software	❌ Not allowed (unless under special terms)

- **Examples**
  - ☞ GNU General Public License (GPL)
  - ☞ GNU Lesser General Public License (LGPL)
  - ☞ Affero General Public License (AGPL)

GNU General Public License (GPL)

- GPL **ensures** that any software **derived** from GPL-licensed software must also be **GPL-licensed**.
- One of the most widely used copyleft licenses.
- **Strong** copyleft

Versions

- **GPLv2**: The classic. Used by the Linux kernel. Simpler, but does not address patents or hardware restrictions ("Tivoization").
- **GPLv3**: Modernized. Addresses Tivoization (the practice of using GPL code in locked-down hardware), provides explicit patent grants from contributors, and is compatible with the Apache 2.0 license.
- It also has provisions against using the software for digital rights management (DRM) restrictions.

Affero General Public License (AGPL)

- Similar to the GPL, but it **extends the copyleft** requirement to software that interacts with the AGPL-licensed software through a **network**.
- **Strong** copyleft
- Suitable for: Web applications and server-side software.
- **Extends the GPLv3.** If you run a modified version of the program on a server and allow users to interact with it remotely (e.g., as a web application/SaaS), you are considered to be "distributing" the software and must make the complete corresponding source code of your modifications available to those users.

GNU Lesser General Public License (LGPL)

- A more **permissive** version of the GPL, the LGPL allows software to be linked with proprietary software as long as the core library remains open-source.
- Allows linking to proprietary software without requiring the entire work to be open source, as long as modifications to the LGPL-covered components themselves are made available under the LGPL.
- **Weak** copyleft

The Mozilla Public License (MPL)

- It is a free and open-source **weak copyleft license**. It's designed to balance the interests of both open-source and proprietary developers.
- **Weak Copyleft**: The MPL allows for the integration of MPL-licensed code into proprietary codebases, as long as the MPL-licensed components remain accessible under the terms of the MPL.
- **Permissive Use**: The MPL grants liberal copyright and patent licenses, allowing for free use, modification, distribution, and exploitation of the work.
- **No Trademark Rights**: The MPL does not grant the licensee any rights to the contributor's trademarks.
- **Patent Grant**: Contributors to the licensed code provide an explicit grant of patent rights to users.

Considerations to use Copyleft Licenses

- **Compatibility**: Some copyleft licenses can be **incompatible** with other licenses, which may limit how software can be combined.
- **Commercial Use**: While copyleft licenses allow for commercial use, businesses must be cautious about how they distribute modified versions to comply with the license requirements.
- **Large-Scale Projects**: Copyleft licenses are often used for large-scale projects where you want to ensure that the software remains open-source and community-driven.
- **Protecting User Freedom**: If you want to protect users' rights to access, modify, and distribute the software, a copyleft license is a good choice.

Popular Copyleft licensed software

- Audacity (GPL v2 with some components under GPL v3)
- LibreOffice (MPL v2.0 with some components under GPL v3)
- VLC Media Player (GPL v2 with some components under GPL v3)
- WordPress (GPL v2 with an option for GPL v3)

- Linux Kernel (GPL v2)
- GIMP (GPL v3)
- GNU Emacs (GPL v3)
- Blender (GPL v2 or later)
- Moodle (GPL v3)
- OpenJDK (GPLv2)

Feature	GPL	LGPL	AGPL
Strictness	Strictest	Less Strict	Stricter than GPL
Freedom to Use	Yes	Yes	Yes
Modification Rights	Yes	Yes	Yes
Distribution Requirements	Must distribute derivative works under GPL	Can distribute derivative works under LGPL	Must distribute derivative works under AGPL
Linking with Proprietary Software	No	Yes	No
Source Code Availability	Yes, for distributed versions	Yes, for modified versions if distributed	Yes, must provide when used over a network
Network-Based Services	Not Explicitly Addressed	Not Explicitly Addressed	Requires Open-Source License
Commercial Use	Allowed, but must comply with GPL	Allowed, but can link proprietary software	Allowed, must comply with AGPL

Copyleft Licenses Summary

3. Public Domain

Public Domain

- Public Domain is not a license; it is the **absence** of **copyright restrictions**.
- The software is **owned** by the **public**, not by an individual or organization.
- When a work is in the public domain, it is **free** for **anyone** to **use**, **modify**, **distribute**, and even **sell**, for **any purpose**, without any obligations whatsoever.
- **Example**: SQLite

Creative Commons

Primarily used for non-software works (like content, art, etc.) with various levels of permissions.

Summary

- Licensing Paradigms: Proprietary: Closed source, controlled, revenue-focused and Open Source: Collaborative, transparent, freedom-focused.
- Key Proprietary Models: Perpetual (Buy once), Subscription (Pay regularly), and Concurrent (Share licenses).
- Open Source Philosophies: Permissive (Freedom for developers - MIT/Apache) and Copyleft (Freedom for the code - GPL/AGPL)

Thank you!

"Open source software is a testament to the power of collaboration; it transforms ideas into innovations, empowering individuals and communities to build a better future together."

Lecturer: Biniam Behailu
Addis Ababa Science and Technology University

References

[1] Reprise Software, “Floating Licensing Explained,” Reprise Software. [Online]. Available: https://reprisesoftware.com/floating-licensing-explained/. [Accessed: Oct. 1, 2025].

[2] Open Source Initiative, "The Open Source Definition," Open Source Initiative, Jul. 7, 2006, last modified Feb. 16, 2024. [Online]. Available: https://opensource.org/osd. [Accessed: Oct. 1, 2025].