Module-3

Understanding Open Source Licensing Models

- Understanding Intellectual Property Rights and Software Licenses
- Licensing models in OSS: Copyright, Copyleft, Permissive, Creative Commons
- Choosing an Open Source License

Intellectual property rights:

- By default: The one who writes the code owns it. *

EMPLOYEE/Soft ware Engineer

Company to its employees

- If you are a developer on regular payroll[employment contract]?
- If there is a written contract [freelance/on contract]?
- What if the development is outsourced to another company?

Intellectual property rights: relevant to software industry

- Patents
- Copyrights
- Trade secrets
- Trademarks

Patents

- Helps to protect functional features
  - May be hardware configuration, algorithm/steps.
- Patent violation/infringement usually leads to high financial penalty.

Copyrights

- Used to protect authorship: source code, diagrams, etc

Trade Secrets

- Used to protect business models/secrets[pricing, etc].

Trade Marks

- Used to protect brand recognition through logos, names etc.

Stake holders of source code

- Employees involved in the project.
- Consultants/Contractors
- Vendor Company developing the software

Types of source code in a software project

- Unique code
- Open source code
- Existing/Third party code
- Note: All 3 of the above come with different types of licenses.

Unique Code

- Specifically developed for a particular project and for a particular client.
  - Limited use to the vendor.
  - Ownership may get shared.

Open Source Code

- Neither the vendor nor the client company owns the IP.
- Must adhere to the license terms like GPL etc.
  - Detailed study of the license should be done before using.
  - Violations might lead to significant risks.
- Both client and the vendor should be aware while using the open source technologies.
  - Make sure all compliances are met.

Existing/Third party code

- Reusing the code written for other projects
  - Depends on the type of license of the code being used.
  - Some kind of royalty or license fee may be there[if its third party].
  - Apart from source code other artifacts like image and video can also be part of this.

So: Software license is:

- A legal instrument
  - Describing the manner in which a software can be used/redistributed in all forms[source/object code].

- If you have purchased a software license:[end user]
  - The licensee can use it as per the terms and conditions mentioned in the license document.
    - Example: renting of the house

Licensing models in open source software

- Copyright [ownership to creators [includes different forms of work like source code, music, artistic, etc]
  - Relicensing and Commercial use come under copyright license.
  - Copyright owner will dictate terms as per the license.

Copyleft [more free yet protective]

- Derivative works should be attributed to the creator.
- Open-sourced and copyleft.
- Study, modify.
- Called protective licenses
- Freedom to carry out the activities[4 freedoms].
- GNU-GPL is the first copyleft license. [strong copyleft]
  - Primarily used to protect open source softwares.
- Variations: Weak copyleft and Strong copyleft
  - Based on the provisions allowed[on derived works].
  - Mozilla public license[weak copyleft]

S.No.	Copyright	Copyleft
1.	Copyright is the right that enable you to prevent unauthorized copying or selling of your work.	Whereas Copyleft is a method using which you can modify the software or documentation and distribute it back to the open-source community.
2.	In Copyright the work is original and not the copy of other.	On the other hand Copyleft comes with an idea of collaboration.
3.	Copyrights protects you original ideas from others access.	While Copyleft allows you to make changes to other ideas and give them back.

4.	Copyright is all about granting individual permission.	Copyleft is all about user freedom.
5.	You can apply Copyright protection both to work that you have published into the public domain and work that you have not published.	While Copyleft allows users to distribute derivative works under a license that offers the same rights as the original work.

Permissive [more freedom]

- Minimal Restrictions
- Eg: BSD like/style licenses.
- Rules for usage – whatever user wants, but with few restrictions- derivative works must be attributed to the creator
- Source code need not be open or made available in the public domain
- Re-licensing allowed – derivative works can be released under any other license or used as proprietary products; Allows commercial usage.
- Examples:
  - GNU All-permissive License
  - MIT License (highly popular)
  - BSD licenses
  - Apple Public Source License and
  - Apache license

Major differences between **Copyleft**, **weak copyleft**, and **permissive** category licenses.

Aspect	Strong Copyleft License	Weak Copyleft License	Permissive License
Modification	Requires derivative works to be released under the same license.	Requires modifications to be open-sourced, but allows linking with proprietary code.	Permits modifications without requiring them to be open-sourced.
Distribution	Requires source code distribution for both original and derivative works.	Requires source code distribution for the modified portions, not necessarily the entire work.	Allows distribution in binary form without the obligation to provide source code.
Derivative Licensing	Forces derivative works to use the same copyleft license.	Allows derivative works to use a compatible license or the same copyleft license.	Allows derivative works to be licensed under any terms, including proprietary licenses.
Integration with Proprietary Software	Generally incompatible with proprietary software.	Allows for limited integration with proprietary software through dynamic linking.	Compatible with proprietary software; allows for closed-source extensions.
License Compatibility	Typically not compatible with other strong copyleft licenses.	Often compatible with other weak copyleft and permissive licenses.	Generally compatible with other permissive licenses.
Virality	Imposes a "viral" effect, requiring downstream works to be open-source.	Has a "weaker" viral effect, affecting only modifications, not the entire work.	Does not impose viral conditions; permits a wide range of licensing options.
Examples	GNU General Public License (GPL), Affero GPL (AGPL)	GNU Lesser General Public License (LGPL), Mozilla Public License (MPL)	MIT License, Apache License, BSD License

Important note

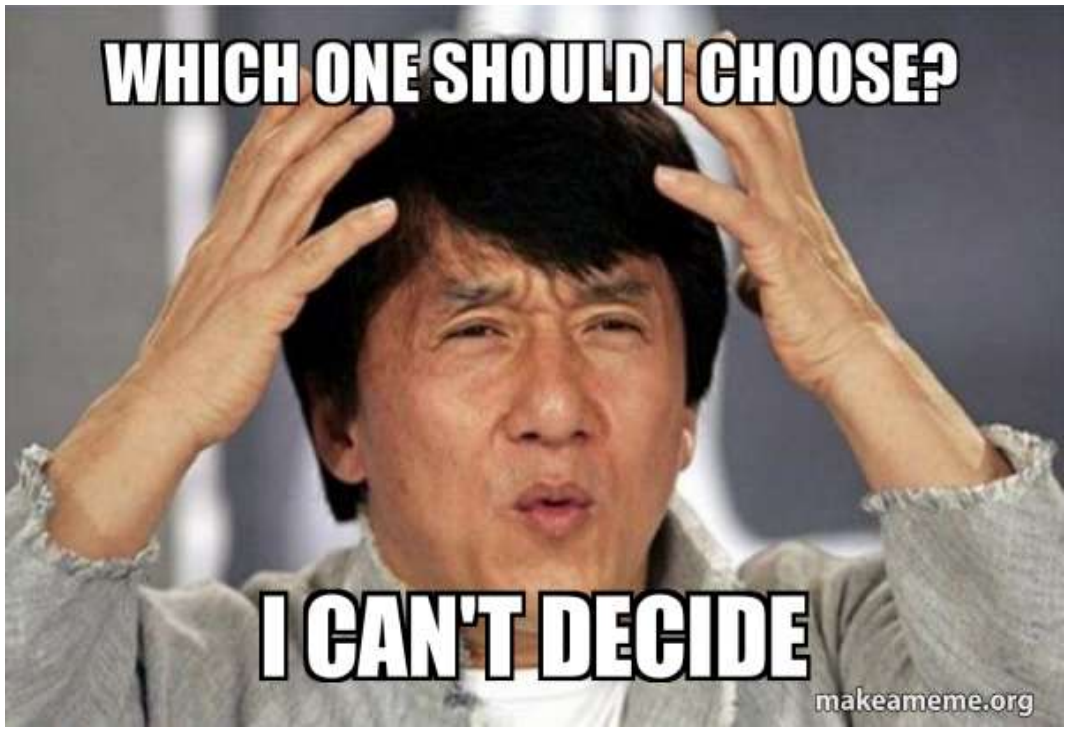
- It's important to note that these are general characteristics, and specific licenses within each category may have variations and nuances. The choice of a license should depend on your project's goals, the level of openness you desire, and compatibility with other software you may want to integrate with or use.

Creative commons License

- A Creative Commons (CC) license is a public copyright license that enables free distribution of a copyrighted work
- Can be applied to all works
  - including books, plays, movies, photographs, music, articles, blogs, and websites.
- NOT recommended for software –
  - Because they do not contain specific terms about the distribution of source code
- Linking with existing OSS is difficult:
  - Most of the CC licenses are not compatible with the major software licenses, so it would be difficult to integrate CC-licensed work with other free software

Choosing an open source license

- If you want to work with a community
- If you want to keep the license simple and permissive
- If you wish to share the improvements made
- If you wish to work without a license[no license]



Working with a community

- Working on an existing project[modify/adapt]
  - Continue using the original license.
    - Because of an requirement[terms].
      - Look into the license/readme/copying file.
      - If no data related to the license: reach out to the owner/creator.
  - Different ways of maintainers manage the ownership of OSS.
    - CLA[Contributor license Agreement].
    - Owner retains the copyright and ownership
    - Assigns the ownership to the project maintainer.
    - Not to define any ownerships: used in small projects.
- Note: there are no standard terms for CLA.

More on CLA

- Small projects usually do not come with a CLA.
- Small to very large CLAs exist.
- Large open source projects may require CLA from its contributor.
  - Example: OSS hosted by Apache, Google, Eclipse
- Simple to long/detailed CLA
  - Individual CLA, Corporate CLA.

More on CLA

- Contributor has to make certain clarifications.
  - Contributor is the author of the contribution
  - Contribution is an original work
  - Contribution is not subject to third party licenses/claims
  - Contributor has legal right to grant the copyright license
  - Contributor does not have an employer that can claim rights to the contribution.

Pros and cons of CLA

- Protection against copyright infringement
- Discourage contributions

If you want to keep it simple and permissive

- Go for MIT License
  - Important: its not protective
    - Your work can be derived and sold commercially
- Permissions under MIT License:

1. Use, modify, and distribute the software for any purpose.
2. Sublicense the software to others.
3. Incorporate it into your projects, open-source or proprietary.
4. Combine it with code under other licenses.
5. Sell software that includes MIT-licensed code.
6. No warranty, use at your own risk.
7. Include the original copyright notice and license text when distributing.