Cal State Fullerton

CPSC 254

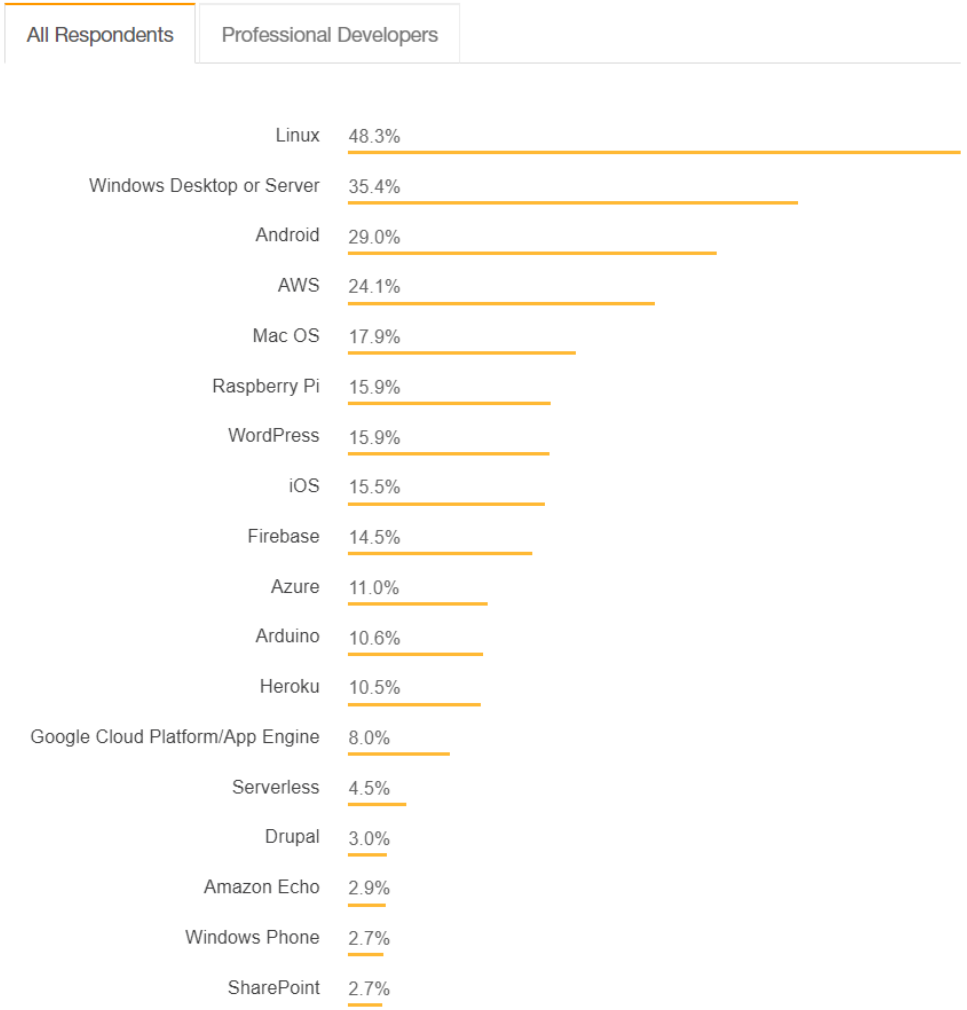
Software Development With Open Source Systems - Open Source Libraries

Instructor: Tejaas Mukunda Reddy

Linux Beats Windows to Become the most popular development platform

https://insights.stackoverflow.com/survey/2018#technology

https://opensource.org/faq



Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

Basis of Open Source

What is "Open Source" software?

- Generally, Open Source software is software that can be freely accessed, used, changed, and shared (in modified or unmodified form) by anyone. Open source software is made by many people, and distributed under licenses that comply with the Open Source Definition.
- The internationally recognized Open Source Definition provides ten criteria that must be met for any software license, and the software distributed under that license, to be labeled "Open Source software." Only software licensed under an OSI-approved Open Source license should be labeled "Open Source" software.

Can Open Source software be used for commercial purposes?

- Absolutely. All Open Source software can be used for commercial purpose; the Open Source Definition guarantees this. You can even sell Open Source software.
- However, note that commercial is not the same as proprietary. If you receive software under an Open Source license, you can always use that software for commercial purposes, but that doesn't always mean you can place further restrictions on people who receive the software from you. In particular, copyleft-style Open Source licenses require that, in at least some cases, when you distribute the software, you must do so under the same license you received it under.

Can I restrict how people use an Open Source licensed program?

- No. The freedom to use the program for any purpose is part of the Open Source Definition. Open source licenses do not discriminate against fields of endeavor.

Can I stop "evil people" from using my program?

- No. The Open Source Definition specifies that Open Source licenses may not discriminate against persons or groups. Giving everyone freedom means giving evil people freedom, too.

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

What is free software is it same as open source

- "Free software" and "open source software" are two terms for the same thing: software released under licenses that guarantee a certain specific set of freedoms.
- The term "free software" is older, and is reflected in the name of the Free Software Foundation (FSF), an organization founded in 1985 to protect and promote free software. The term "open source" was coined by Christine Peterson and adopted in 1998 by the founders of the Open Source Initiative. Like the FSF, the OSI's founders supported the development and distribution of free software, but they disagreed with the FSF about how to promote it, believing that software freedom was primarily a practical issue rather than an ideological one (see for example the entry "How is `open source' related to `free software'?" from the OSI's original 1998 FAQ page).
- Many who later adopted the term "open source" broadly shared the ideological perspective of the FSF but had some disagreements over strategy and rhetoric. Today some people use both terms, choosing according to context and audience.
- One of the tactical concerns often cited by adopters of the term "open source" was the ambiguity of the English word "free", which can refer either to freedom or to mere monetary price; this ambiguity was also given by the OSI founders as a reason to prefer the new term (see "What Does `free' Mean, Anyway?", and similar language on the marketing for hackers page, both from the original 1998 web site).
- In the 1990s, the term "open" applied to software source code was sometimes used to imply source code being merely inspectable or visible or available. Going back further, in the 1980s there were uses of "open" in the computing industry that primarily connoted something like "absence of hardware vendor lockin". OSI's term "open source", as defined in the Open Source Definition, makes clear that open source specifically entails not mere inspection access but also conveying to recipients the perpetual right to fork covered code and use it without additional fees.

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

What is free software is it same as open source

- The FSF uses a shorter, four-point definition of software freedom when evaluating licenses, while the OSI uses a longer, ten-point definition. The two definitions lead to the same result in practice, but use superficially different language to get there.
- This history has led to occasional confusion about the relationship between the two terms. Sometimes people mistakenly assume that users of the term "open source" do not intend to communicate a philosophical point of view via that term, even though many actually do use it that way. Another mistake, which has occasionally been seen since about 2008, is to assume that "free software" refers only to software licensed under copyleft licenses, since that is how the FSF typically releases software, while "open source" refers to software released under so-called permissive (i.e., non-copyleft) licenses. In fact, both terms refer to software released under both kinds of license.
- Neither term binds exclusively to one set of associations or another, however; it is always question of context and intended audience. When you sense a potential misunderstanding, you may wish to reassure your audience that the terms are essentially interchangeable, except when being used specifically to discuss the history or connotations of the terminological difference itself. Some people also prefer to use the term "free and open source software" (or FOSS, FLOSS [free, libre and open source software]) for this reason.
- See also our history page for more information about the history and usage of the term "open source".

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

You should read the license to understand its requirements for the source code distribution. What is “copyleft”? Is it the same as “open source”?

- "Copyleft" refers to licenses that allow derivative works but require them to use the same license as the original work. For example, if you write some software and release it under the GNU General Public License (a widely-used copyleft license), and then someone else modifies that software and distributes their modified version, the modified version must be licensed under the GNU GPL too — including any new code written specifically to go into the modified version. Both the original and the new work are Open Source; the copyleft license simply ensures that property is perpetuated to all downstream derivatives. (There is at least one copyleft license, the Affero GPL, that even requires you to offer the source code, under the AGPL, to anyone to whom you make the software's functionality available as a network service — however, most copyleft licenses activate their share-and-share-alike requirement on distribution of a copy of the software itself
- Most copyleft licenses are Open Source, but not all Open Source licenses are copyleft. When an Open Source license is not copyleft, that means software released under that license can be used as part of programs distributed under other licenses, including proprietary (non-open-source) licenses. For example, the BSD license is a non-copyleft Open Source license. Such licenses are usually called either "non-copyleft" or "permissive" open source licenses
- Copyleft provisions apply only to actual derivatives, that is, cases where an existing copylefted work was modified. Merely distributing a copyleft work alongside a non-copyleft work does not cause the latter to fall under the copyleft terms.
- For more information, look at the text of the specific copyleft license you're thinking of using, or see the Wikipedia entry on copyleft. C.f. "permissive" licensing.

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

What is a "permissive" Open Source license?

- A "permissive" license is simply a non-copyleft open source license — one that guarantees the freedoms to use, modify, and redistribute, but that permits proprietary derivative works. See the copyleft entry for more information.

Is <SOME PROGRAM> Open Source?

- Only if it uses one of the approved licenses, and releases appropriate software.

Can I call my program "Open Source" even if I don't use an approved license?

- Please don't do that. If you call it "Open Source" without using an approved license, you will confuse people. This is not merely a theoretical concern — we have seen this confusion happen in the past, and it's part of the reason we have a formal license approval process. See also our page on license proliferation for why this is a problem.

Is <SOME LICENSE> an Open Source license, even if it is not listed on your web site?

- In general, no. We run all licenses through an approval process to provide an accepted standard on which licenses are Open Source, and we list the approved ones. Be dubious of claimed Open Source-ness for licenses that haven't gone through the process. See also the license proliferation page for why this matters so much.

Is a license an Open Source license if it is just a few words different from a license listed on your web site?

- Many older licenses, such as the BSD license, have several variations that differ only by a few words or sentences. Unfortunately, it is not possible for OSI to approve every variation, so we cannot say if a specific variation is approved. Where possible, please avoid using these variants, and instead use the same version used on the OSI website.

What about software in the "public domain"? Is that Open Source?

- For most practical purposes, it is — sort of. This is a complicated question, so please read on.
- "Public domain" is a technical term in copyright law that refers to works not under copyright — either because they were never in copyright to begin with (for example, works authored by U.S. government employees, on government time and as part of their job, are automatically in the public domain), or because their copyright term has finally lapsed and they have "fallen into" the public domain.
- Not all jurisdictions have a public domain, and it doesn't always mean exactly the same thing in the jurisdictions that do have it. Furthermore, even where it is clear what it means, it's still not a license. To be subject to a license, a work must still be in copyright. That means there is no way for the "public domain", as a concept, to go through the OSI evaluation and approval process. We wouldn't be evaluating a license text. Instead, we would have to somehow evaluate the laws themselves, in different jurisdictions, and say which jurisdictions have a public domain that meets the Open Source Definition and does not create problems for software authors and users. This would be very difficult, because it would mean evaluating not just the statutes but various bodies of case law (for example, open source licenses usually have a strong disclaimer of liability for the copyright holder — but we don't know how or whether the author would be protected from liability for software released into the public domain in various jurisdictions). This approach would not be useful to the OSI's mission, because open source is an international phenomenon and we only want to approve licenses that meet the Open Source Definition everywhere.
- Thus we recommend that you always apply an approved Open Source license to software you are releasing, rather than try to waive copyright altogether. Using a clear, recognized Open Source license actually makes it easier for others to know that your software meets the Open Source Definition. It also enables the protection of attribution, and various other non-restrictive rights, that cannot be reliably enforced when there is no license.
- There are certain circumstances, such as with U.S. government works as described above, where it is not easy to apply a license, and the software must be released into the public domain. In these cases, while it would be inaccurate to display the OSI logo or say that the license is OSI-approved (since there is no license), nevertheless we think it is accurate to say that such software is effectively open source, or open source for most practical purposes, even though it is not officially released under an open source license. (This is assuming, of course, that in the laws of releasing jurisdiction the meaning of "public domain" is compatible with the Open Source Definition.) After all, the freedoms guaranteed by open source licenses are still present, and it is possible for the familiar dynamics of open source collaboration to arise around the software.
- For a detailed discussion of the complexities of the public domain and open source, search for the words "public domain" and "PD" in the subject headers of the January 2012, February 2012, and March 2012 archives of the OSI License Review mailing list. And if the thought of reading all those conversations is daunting, please take that as more evidence that it's just better to use an approved Open Source License if you can!

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

What about software in the “public domain”? Is that Open Source

What about the Creative Commons "CC0" ("CC Zero") public domain dedication? Is that Open Source?

- At this time, we do not recommend releasing software using the the CC0 public domain dedication.
- Creative Commons Zero is a legal device known as a "public domain dedication". It is essentially a statement of intent by the copyright holder to waive copyright ownership in the work — that is, the copyright holder wishes to place the work into the public domain.
- Because such a waiver is (perhaps surprisingly) not possible in all jurisdictions, CC0 also contains a "Public License Fallback" clause that goes into effect "should any part of the Waiver for any reason be judged legally invalid or ineffective under applicable law". The fallback is essentially a copyright license that is very similar to an Open Source license, in that it gives up most of the restrictive powers associated with copyright, and allows redistribution and modification of the work.
- In February 2012, Creative Commons submitted CC0 to the OSI for approval as an open source license, requesting that the OSI evaluate the public license fallback section, since the rest of the text is a waiver of rights rather than a license. An unexpectedly intense and detailed discussion followed — search for "CC0" and "Creative Commons Zero" in the subject headers of the February 2012 and March 2012 archives of the OSI License Review mailing list.
- CC0 was not explicitly rejected, but the License Review Committee was unable to reach consensus that it should be approved, and Creative Commons eventually withdrew the application. The most serious of the concerns raised had to do with the effects of clause 4(a), which reads: "No ... patent rights held by Affirmer are waived, abandoned, surrendered, licensed or otherwise affected by this document.". While many open source licenses simply do not mention patents, it is exceedingly rare for open source licenses to explicitly disclaim any conveyance of patent rights, and the Committee felt that approving such a license would set a dangerous precedent, and possibly even weaken patent infringement defenses available to users of software released under CC0.

Distributing and Using Open Source Software

What if I do not want to distribute my program in source code form? Or what if I don't want to distribute it in either source or binary form?

- If you don't distribute source code, then what you are distributing cannot meaningfully be called "Open Source". And if you don't distribute at all, then by definition you're not distributing source code, so you're not distributing anything Open Source.
- Think of it this way: Open Source licenses are always applied to the source code — so if you're not distributing the source, then you're not distributing the thing to which an Open Source license applies. You might or might not distribute binaries; that's a separate question. But while some Open Source licenses allow you to distribute binary code without distributing the corresponding source, it is only the source code that can be "open source". The binaries alone cannot be Open Source, because you're not making any source code available to be open. (If someone else distributes the source code under an Open Source license, then that's still Open Source, of course.)
- Note that copyleft Open Source licenses require redistributors to make source code available under certain circumstances; for example, see the GNU General Public License and GNU Affero General Public License.
- See also the "What is 'distribution'?" question.

Which Open Source license should I choose to release my software under?

- You can choose any license from the open source licenses listed starting here: opensource.org/licenses. Most people select one from the "popular" category, but you are free to choose any listed license.
- If this is your first time choosing an open source license, we recommend that you find someone who has experience with open source licensing and talk to them about your project — that will help you choose the most appropriate license. The person doesn't have to be a lawyer; it could be a developer who has experience releasing open source code. The section Choosing a License at the Civic Commons wiki may be useful, and you can learn more about open source licenses from, Section 3.2 of the eBook Introduction to Free Software by Hernandez, Jimenez, Barahona, Pascual, and Robles, and Karl Fogel's book, Producing Open Source Software, Chapter 2, "Choosing a License and Applying It."

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

Which Open Source license is best?

- Unlike bilateral copyright licenses, which are negotiated between two parties and embody a truce between them for business purposes, multilateral copyright licenses — of which open source licenses are a kind — are “constitutions of communities”, as Eben Moglen and others have observed. They express the consensus of how a community chooses to collaborate. They also embody its ethical assumptions, even if they are not explicitly enumerated.
- When that consensus includes giving permission to all to use, study improve and share the code without prejudice, the license is an open source license. The Open Source Definition provides an objective test of evaluating that such a license is indeed an open source license and delivers the software freedom we all expect.
- Since licenses are the consensus of communities, it is natural that different communities will have different licenses, that communities with different norms will find fault with the licenses used by others, and that all will regard their way as optimum. The arguments over this will be as deep as the gulf between the philosophical positions of the communities involved.
- Ultimately, there is no license that is right for every community. Use the one that best aligns with your community’s objectives and ethos.

How do I apply <SOME OPEN SOURCE LICENSE> to software I'm releasing?

- This question isn't actually specific to open source licenses — it's really just about how to apply some particular copyright license (whether open source or not) to your software. Please note that the OSI is not a legal services organization and does not provide legal advice. However, many licenses come with instructions on how to apply them: for example, see the section "APPENDIX: How to apply the Apache License to your work" in the Apache-2.0 license, or the section "How to Apply These Terms to Your New Programs" in the GPL-3.0 license. If the license you want to apply has such instructions, just follow them. If it does not, then look at the previous two examples (or at other licenses that contain similar instructions) and follow a similar recipe, adjusting for the license you're actually using of course. The Software Freedom Law Center also maintains a guide on managing copyright information within open source projects. Finally, this guide may also help, though please note again that neither it nor this FAQ item constitutes legal advice.
- Note that releasing software under an open source license does not involve contacting the OSI, signing up to some process, or handing a copy of your software to the OSI or any other organization for evaluation. You just publish the software with an OSI-approved open source license attached, in the manner described above — that's all you need to do.

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

What are "contributor agreements"? Are they like open source licenses?

- Many open source projects will only accept patches (code contributions or documentation contributions) from people who have submitted a legal document known as a contributor agreement. Contributor agreements are not open source licenses — rather, they are a way for the contributor to tell the project that it has the right to distribute the new contributions under the project's existing open source license. (Some contributor agreements also allow for the project to distribute the contributions under other open source licenses too, which enables projects to change their license in the future, and some agreements even allow the project to distribute the contributions under any license the project wants.)
- There are two kinds of contributor agreements. In a Contributor License Agreement (CLA), the original contributor retains copyright ownership of their contributions, but grants the project a broad set of rights such that the project can incorporate and distribute the contributions as it needs to. In a Copyright Assignment Agreement (CAA), the contributor actually transfers copyright ownership of the contributions to the project, who can then license it however they want since they own it (but a CAA typically grants very broad non-exclusive rights back to the contributor so that they too can use, distribute, sublicense etc their contribution freely).
- With both CLAs and CAAs, it is of course necessary that "the project" be some kind of legal entity able to enter into agreements. Sometimes the project is incorporated itself, usually as a non-profit entity; sometimes it is represented by an umbrella non-profit organization (such as the Apache Software Foundation or the Software Freedom Conservancy); sometimes a for-profit corporation considers itself the main sponsor of the project and requests contributor agreements in order to manage the development community and maintain a public distribution of the software in question.
- For more about contributor agreements in general, and some examples, see civiccommons.org/Contributor_Agreements. See also the Project Harmony, "...a community-centered group focused on contributor agreements for free and open source software (FOSS)."

Can I strip out the copyrights on Open Source code and put in my own?

- Definitely not! This isn't even about Open Source, really: in general, you should not remove a valid copyright notice, no matter what license it specifies. Copyright notices are legal notices; they are also a source of information about the provenance of source code, and if that information is stripped out, recipients of downstream copies have no easy way to rediscover it.

Can I write proprietary code that links to a shared library that's open source?

- Sometimes you can; it depends on the Open Source license. Authors often want you to be able to do this, so most shared libraries are licensed under a permissive license or one that allows linking under certain circumstances (e.g., the LGPL). A very small number of libraries use the GPL, which only allows linking with proprietary works if the licensor grants an explicit exception. Thus, you are wise to check the licenses that your program links to. The community expects that all code linked to GPL code will be licensed under the GPL, even if the link is made at runtime using a shared library.

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

I want to publish some code as Open Source code — can I get a license from you?

- As long as you own that source code, all that you need to do is choose one of the approved Open Source licenses, include a copy of the license text, typically in a filenamed "COPYRIGHT", including a statement saying that you are licensing the code under that copyright, and give it to somebody else! Of course, you probably want to give it to a lot of people in order to gain the maximum benefit from giving away your code. A number of websites will help you do that: berlios.de, sourceforge.net, code.google.com, and others.

Is <SOME PROGRAM> Open Source simply because it's written in <SOME OPEN SOURCE LICENSED LANGUAGE>?

- No. While languages like PHP, Perl or Python have implementations that are licensed under Open Source licenses, that doesn't turn all code written in these languages or run under such implementations into Open Source. The code written in such languages or run under such implementations would need to be licensed under an approved Open Source license in order to be Open Source.

What is "distribution"? What does it mean to "distribute" a program? Is letting people use it on my server the same as distribution?

- Colloquially, to "distribute" a program means to give someone else a copy of its code — either its source code, or its binary (executable) code, or both. Merely allowing people to invoke a program on your server, for example via networked API calls, does not constitute distribution of the program as generally understood.
- To avoid confusion, some licenses use the terms "propagate" and "convey" instead of "distribute". For example, in the GNU General Public License, version 3, to "propagate" means "...to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well." Even that leaves some question as to what "making available to the public" means. The definition for "convey" narrows it down, however: "...any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying."
- In a legal context, you may wish to use similar precision. However, in informal communications, "distribute" is usually understood to mean "deliver copies in source code and/or binary form".
- Not all programs have separate source and binary forms. For programs written in so-called "scripting" languages there is generally only a source code form (though sometimes compressed, for example via the minification often performed on Javascript code prior to distribution). But other programming languages are typically compiled to an architecture-specific executable form and can optionally be distributed as executables without source code. The distinction between source code and executable form is important for understanding the terms and conditions of some open source licenses, so if you don't have the necessary technical background, you should consult someone who does..

Prepared By: Tejaas Mukunda Reddy, Shivansh Vijay Nathan

Reference Prof. David Heckathorn

Someone is violating a copyleft license, for example by refusing to give me source code when they are required to. What can I do?

- The Open Source Initiative is not a legal services organization and generally cannot help you when someone is violating a copyleft license. However, as of late 2011, one of the organizations below may be able to help (note that most of the enforcement they do is about the GNU GPL and AGPL licenses, though in theory they can help enforce other copyleft licenses too):
    - The Software Freedom Law Center.
    - The GPL Violations Project.
    - If the violation includes software that is one of Software Freedom Conservancy's Member Projects, you can contact the Conservancy at <compliance@sfconservancy.org> and they will help you help that project enforce the GPL.
- If you know of other organizations that provide similar legal assistance, please let us know at osi {_AT_} opensource.org.
- **Commerce and Open Source**
- **How do I make money if anybody can sell my code?**
- You can sell services based on the code (i.e., sell your time), sell warranties and other assurances, sell customization and maintenance work, license the trademark, etc. The only kind of profit strategy that is incompatible with Open Source is monopoly-based sales, also known as "royalties". See this article for how to think about business strategies that make money from Open Source. Also, this 2015 survey of open source leaders (including many OSI Directors) provides several business models for Free and Open Source software.
- **Can I sell Open Source programs? Even if I haven't written it?**
- Yes, you can. But depending on the license, you probably can't stop your customers from selling it in the same manner as you. See the commercial use for more details.

Does Open Source mean anybody else can use my name and logo?

- No, at least not any more than they could otherwise. Open Source is about software source code, not about identity. That is, letting people use your code under an Open Source license is not the same as letting them use your trademarks or other identifying attributes, except insofar as they would be permitted to anyway (for example, in nominative use doctrine). There are many companies and other organizations that release open source code while exercising tight control over their trademarks.
- Trademarks and other marks of attribution are primarily about preventing public confusion over identity and provenance, and therefore trademark regulation is useful in Open Source software in the same way it is useful generally.

Where can I find open source libraries?

- Here is a comprehensive list of C++ libraries; http://en.cppreference.com/w/cpp/links/libs
- List of most popular Java libraries on GitHub

The Main Insights From the Top 100 Libraries List

The Unexpected

- Hadoop Blows Spark Out of the Water – Hadoop comes in at #42 with no mention of Apache Spark in the top 100 list whatsoever. Apache Zookeeper made it to #75, helping maintain Hadoop clusters and keeping the elephants at bay.
- SQL > MongoDB > PostgreSQL – The Java SQL connector came in at #27, MongoDB showed up in #87, and PostgreSQL barely made the list at #97.
- ElasticSearch has the Most Justified Buzz Around a Java Library – ElasticSearch, the search server based on Apache Lucene (which made #90 in the list), the E in the ELK stack, and a personal favorite of ours, is the library with the most justified buzz we have on the list.
- And… The Usual Suspects
- JUnit is the Undisputed King of Java Libraries – With 3,345 entries, 64% of Github's top Java projects imports are set on JUnit. Followed by spring-test on the Spring front and testng, these are the top 3 Java testing libraries that we saw in the top 20 list.
- SLF4J is the Most Popular Logging Library – Whether you're using Log4j, Logback or any other logging engine, with 1,184 entries over 22% of Github's top Java are using slf4j has their logging facade.
- 14 Out of the Top 100 Libraries are Coming From the Spring Framework – The most popular framework among the top 100 libraries (even more than apache-commons which has 12 libraries in the top 100), with spring-context as its most popular library.
- Google Guava Rocks the Charts as the #4 Most Popular Java Library – With 815 entries which make 15.6% of Github's top Java projects. We actually love using Guava here at OverOps as well and recently published a post about some of its useful yet lesser known features.

- **apache-commons is Really Common Coming in at #5** – With its top representative holding 659 import statements (12.63%) in Github's top Java projects and 12 of its libraries in the top 100, apache-commons continues to justify its name.
- **Mockito is the Most Popular Java Mocking Framework** – 559 entries (10.72%) show that mocking makes it big in Java, ranking as the 7th most popular library.
- **Developers Love Using joda-time** – This comes as no surprise but it's interesting to see the joda-time library by Stephen Coulbourne reach the 18th place.

5 More Entries Worth Mentioning

- **#65 – Bukkit** – The only gaming library in the top 100 list, you guessed it right, Minecraft servers.
- **#66 – Jetty** – Because Netty didn't make it to the list.
- **#81 – PowerMock** – A fresh entry to the top 100 list, states that "it can be used to solve testing problems that are normally considered difficult or even impossible to test".
- **#90 – Google Protobuf** – A language-neutral, platform-neutral, extensible way of serializing structured data for use in communications protocols, data storage, and more.
- **#100 – AssertJ** – Rising in popularity over the last year and also included in the new version of Dropwizard, one of the popular new testing libraries, accepting migrations from FEST Assert.
- .