FLOSS Concept Booklet

Reading not your thing? Much of the content below is explained in audio and video here [1].

Intro /Concept

What does FLOSS stand for?

FLOSS stands for "Free/Libre Open-Source Software".

Legally, Free Software and Open-Source take quite different attitudes to sharing source code and what obligations those who share legally require. The different attitudes are a product of political ideology and cannot be easily reconciled - though they can be neutrally explained (only) by referring to the legal concepts of 'share-alike' and a 'consortium'. The term FLOSS emerged to simply avoid having to discuss or define these ideas to ordinary users who want to know about the implications for end users. Which do not normally include the legal negotiations between contributors.

What is free software?

Free software is software that anyone is free to use, copy, improve, examine or distribute, either free of cost or for a price. More precisely, it refers to four fundamental freedoms, which users of the software should have:

- Users should be able to run the software for any purpose. (freedom 0 — many things in computers start at 0)
- Users should be able to closely examine and study the software and should be able to freely modify and improve it to suit their needs better. (freedom 1)
- Users should be able to give copies of the software to other people for whom the software will be useful, either gratis or for a fee. (freedom 2)
- Users should be able to improve the software and freely distribute their improvements to the broader public so that they, as a whole, benefit. (freedom 3)

There is nothing new or special about this. This is how software used to be developed in the early 'Big Iron' days of companies which made money by selling hardware. But then, software companies came in, they started changing the rules of the game. They saw software only as a means of making money rather than as a means of making life easier, and with the advent of the 'Shrink-Wrap-License' even the need to provide a properly tested program was obviated.

Legally, free software is defined by a license to use and distribute the software that guarantees all four of these freedoms. Any user who grants all of these freedoms to other users retains their access to the software. In strict legal terms, the software is share-alike but only to those users who uphold these obligations. All others are disqualified and may lose their right to use, extend, distribute or derive new works from it.

It isn't open content, which has its own rules

Some related share-alike movements do not accept the so-called freedom 0 - it remains highly controversial. Accordingly those movements are neither free software nor open-source. They are part of the broader open content movement however. A typical license that restricts use by certain people or groups but permits it under share-alike terms to others is CC-by-nc-sa (the Creative Commons attribution-non-commercial-share-alike license) that is extremely popular among content and code developers that do not wish to grant commercial entities the power to build on their work without paying.

As a user, why would I want to examine and modify my software?

What is important is not that you modify or view the sources, but that you cannot be prevented from doing so or having it done (for you) and are not dependent on a particular person or entity to do it. Not to mention the fact that companies do go out of business - taking their proprietary products down with them - sometimes leaving the users who relied on them without any means of supporting some critical product.

As technology evolves, hardware, software and users' requirements change. And software, being a tool to make life easier, too has to be subject to easy and quick modification. So, even if you personally cannot change or modify the software, you want to be sure that you are not subject to monopoly power, or even simply the whims and fancies, of whoever created the software.

That apart, it is necessary to be able to examine the software, to see if it has malicious features. For example, to check whether the program is spying on you. One version of Windows was designed to report to Microsoft all the software on your hard disk. But Microsoft is not alone: the KaZaa music sharing software is designed so that KaZaa's business partner can rent out the use of your computer to their clients. You (or anyone else) need to be able to examine and modify your software to be able to protect yourself against such mistreatment. Even if you don't know what bad things to look for, someone who does will soon find this "bad thing" in the program and spread the word about it.

There are other reasons such as being able to fix bugs, and modify programs to your needs. These will be explained a little later.

Doesn't "free" mean that I do not have to pay for the software?

No. The word "free" has two meanings in the English language.

1. The "free" in "free beer", which refers to zero cost.
2. The "free" in "free speech" and "free market", which refers to freedom.

The free in free software refers to the freedoms that we've talked about above that people have. There's nothing in the definition of free software that says that you cannot sell it to someone for a price. Indeed, there are companies whose entire business model is centred around collecting, compiling and selling free software. However, since someone to whom free software is licensed is free to sell or give it away in turn, you can almost always easily find it openly (and legally) downloadable on the Internet or other places [2].

When you hear of "free software", think of liberty, freedom, and "free enterprise".

Well, what's not "free" about other kinds of software?

Most non-free software in the world today is not sold, it is licensed. From complex operating systems to tiny games or screen savers, the end users of the software have a license to use it under conditions laid out in an End User License Agreement. This agreement lists out the conditions under which the user can use the software – often restrictions are imposed on the use to which the software can be put. In almost all cases, users are explicitly prohibited from "taking the software apart" to study how it works, cannot modify or improve it, are only allowed to make a single copy of the software (for backup purposes) and are strictly prohibited from giving copies to other people.

What do you mean by "copyleft"?What's wrong with copyright?How is this different?

Legally, a copyleft is a share-alike clause in a license that requires certain conditions (or "freedoms") be upheld by the user, distributor or anyone basing a derivative work on the original. One of those conditions is to treat all other users, distributors and derived work authors equally under the conditions of the license. One of those equal conditions is the withdrawing of all rights under the license from anyone who does not do so.

Free software historically implemented copyleft by requiring all modified and extended versions of the program to be free software as well, using the copyright in the program (and associated rights to restrict use of it or derivations of it) as leverage. This did not deal with every problem. In particular, it did not prevent authors of derived works from filing a software patent on their improvements, nor anyone from building a web service on the original software and extending it by 'mashup' methods. *These deficiencies are supposed to be addressed in the GPL version 3.0.*

The original GPL, despite deficiencies, was an attempt to build a very minimal consortium among all users who would agree to its four freedoms. To understand this we must review the alternatives.

The simplest way to make a program free is to put it in the public domain, uncopyrighted. This allows people to share the program and their improvements, if they are so minded. But it also allows uncooperative people to convert the program into proprietary software. They can make changes, many or few, and distribute the result as a proprietary product. People who receive the program in that modified form do not have the freedom that the original author gave them; the middleman has stripped it away. Also the developers of free software will be forced to compete with improved versions of their own software (which open-source does permit).

Copyleft says that anyone who redistributes the software, with or without changes, must pass along the freedom to further copy and change it. If they do not, they lose their rights. As Stallman puts it, "Copyleft guarantees that every user has freedom, and ensures that somebody does not remove the freedom from free software."

To copyleft a program, first state that it is copyrighted; then add distribution terms in the form of a license document – they comprise a legal instrument that gives everyone the rights to use, modify, and redistribute the program's code or any program derived from it but only if the distribution terms are unchanged. Thus, the code and the freedoms become legally inseparable.

According to the Free Software Foundation, "Proprietary software developers use copyright to take away the users' freedom; we use copyright to guarantee their freedom. That's why

we reverse the name, changing 'copyright' into 'copyleft'." Historically however the term was simply a joke or pun added to a letter to Richard Stallman by Don Hopkins who had put "copyleft - all rights reversed" as one of many annotations.

The most popular copyleft license is the GNU General Public License [3].

What licenses exist to protect free software?

There are many licenses that make a piece of software free. But only some of them preserve user freedom aka copyleft. The non-copyleft license include X11, BSD, Artistic... The strongest copyleft and most widely used free software license is the GNU General Public License [4], or GNU GPL for short.

For a more complete list of licenses check out the GNU website [5].

Okay, so I can see that free software is legal but surely if I duplicate something that means that someone is losing out somewhere along the way?

We in the free software community feel that the harm caused by obstructing software use cannot be justified by the profit obtained through selling software. We use other means to earn money.

Also, chances are if someone didn't get the software for no price, they wouldn't have gotten it at all. Take all the people that get proprietary software illegally from peer-to-peer programs for example.

Contrary to your assumption that allowing distribution and modification causes loss, Richard Stallman lists three levels of material harm caused by restrictions on distribution and modification:

1. Fewer people use the program.
2. None of the users can adapt or fix the program.
3. Other developers cannot learn from the program, or base new work on it.

For a detailed analysis, check out the essays "Why Software Should Not Have Owners" [6] and "Why Software Should Be Free" [7] by Stallman.

This freedom with software programs is interesting.Can this be extended to other forms of information like books?

Yes. Most user manuals for free software programs, for example are released under either free or copyleft licenses. Already, much literature is available under permissive terms, such as the GNU Free Documentation License (GFDL) [8] and Creative Commons [9] Commercial-Attribution(-ShareAlike).

But unlike software, books and articles contains speech and personal opinion. And, personal credit for specific work is important. So, the benefit of unlimited modification is not always desirable in literature, science or other content. The open content [10] movement accordingly focuses more on share-alike clauses.

These clauses are not restricted to those free software or open-source require. They include for instance the Creative Commons NON-Commercial licenses which prevent use by commercial parties unless they negotiate a parallel commercial license (which may also be a share-alike license or may be more like a proprietary license). Other share-alike clauses have been proposed to require specific processes of dispute resolution, scientific method or

journalism be followed by creators of derived works.

Like the original free software foundation, each such clause would create a global consortium relying mostly on the license to enforce its rules and using the leverage of rights to use or improve the content as the leverage.

FSF itself has recognized the difference between open content and free software. The GNU Free Documentation License [11] used by both Wikipedia and Wikibooks itself contains clauses that deny the right to use the content to anyone who does not credit authors or other contributors, but these are similar to the free software conditions. GFDL does clearly and explicitly permit commercial use and it supports annotations and "Invariant Sections" that can reliably qualify the origin or reliability of the material or summarize objections to it - though these are not used by Wikipedia or Wikibooks at present, they serve necessary purposes for those services that do not allow every user to edit.

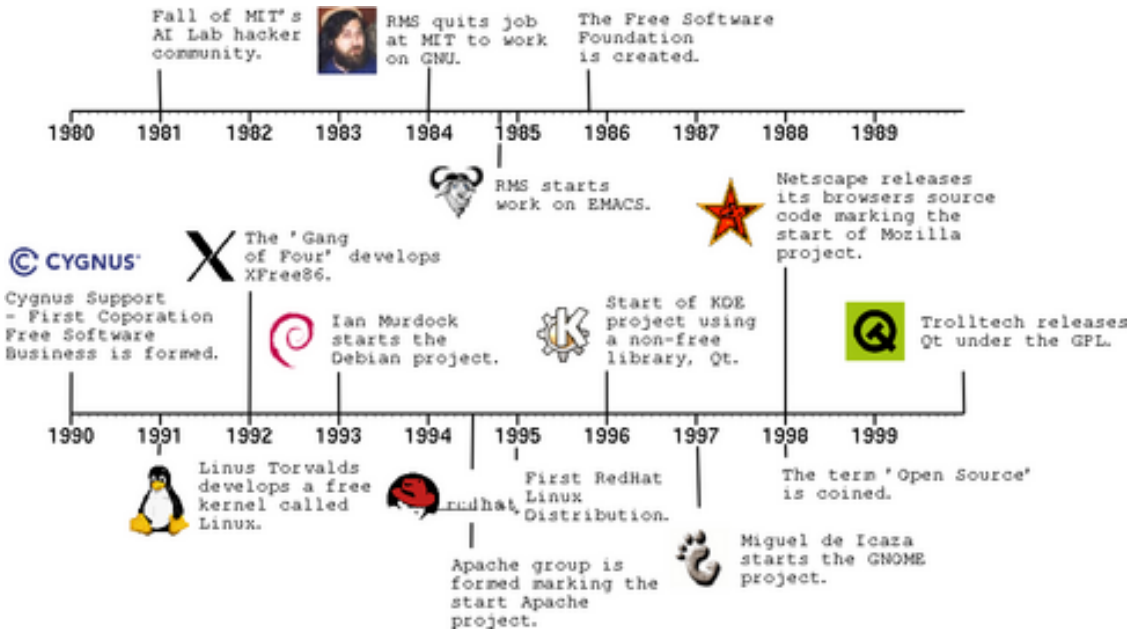
History of Free Software

When did this whole free software thing start?

Software sharing is as old as computers. But the Free Software Movement traces its history to a software sharing community at the MIT Artificial Intelligence Lab. Richard Stallman became a member of this software sharing community when he joined the lab in 1971.

In the early 1980s due to a series of events, the hacker community collapsed. Richard Stallman was one of the few hackers left out in the lab and he was faced with a stark moral choice. He could accept the world has changed and start using proprietary software. Or he could create a free operating system which could recreate a community of cooperating hackers. In short, he could either change himself or change the world.

Stallman decided to change the world. And thus the GNU (pronounced Guh-new, it rhymes with canoe) project, a free replacement for the Unix operating system was born. In January 1984, Stallman quit his job at MIT and began writing GNU software. One of the first pieces of GNU software that Stallman wrote was the Emacs text editor. Slowly more hackers joined Stallman and started putting together a complete body of free software. This ranged from tiny programs like ls and cp to the huge packages like the GNU compiler collection and the Bash shell.



Larger version

Okay, but now what is Linux?When did that come about?

Until 1991, most of the programs required to create a complete operating system had been created by the GNU project, except for one important piece - the kernel. In a nutshell, the kernel is software that provides secure access to a machine's hardware and decides what software get to use what hardware when. By then, a Finnish student named Linus Torvalds had written a free Unix compatible kernel called Linux (pronounced Lee-nux or Lin-ux). The kernel combined with the rest of GNU packages formed a complete usable operating system. Today this combination is called GNU/Linux, although it is often inaccurately called just Linux [12].

Open Source

What is this open source thing?

Open source is similar to free software. The biggest difference is that open source sees FLOSS as just a way of making better software, and doesn't value the freedom of the people.

How did open source start?

In 1998, some free software developers figured that by keeping quiet about ethics and freedom, and talking only about the practical benefits of free software, they might be able to spread free software better, especially to businesses. The term "open source" is offered as a way of doing this.

Why are open-source and free/libre software often grouped together?

Most (*not* all) open-source software can also qualify as free software, and vice versa.

What's better?Free Software or open-source software?

This is up to you. If you believe freedom is most important, free software. If you believe superior software is most important, open source.

Production Methodology

How is Free Software actually made?

Same way other software is made. People sit on the computer keyboard and type in commands and codes and compile them.

To properly understand how free software is made, we have to have a good idea how all software – free and proprietary is made.

Follow this link to find more about *Who is doing it?* http://widi.berlios.de/paper/study.html

Okay, how is software made then?

A software program is simply a set of instructions to a computer to do something. Since the computer is a machine without any capacity to think for itself, it can only understand instructions written in a particular language – this format is called "object code". Unfortunately, to human beings, object code looks like gibberish. While they are creating software, humans use a particular format that they can easily understand called "source code". Source code uses letters, numbers and punctuation and, like any human language, can be understood by humans who learn to do so. So now we have source code, a format that programs are written in or created by humans but which looks like gibberish to a computer and object code – a set of instructions that a computer can understand but which look like gibberish to a human. A special program called a compiler transforms the source code into the object code. To reiterate, a human being writes down what it wants the computer to do in a format called source code. It's then translated into the only language that computers understand – object code. A special program called a compiler does this translation.

So now that I understand how software is made, how is free software made?

Remember, most humans cannot easily understand object code. Therefore, if a human wants to closely study, modify or improve a piece of software, she has to have access to the source code of the software. Since being free to examine, modify or improve software is central to the concept of Free Software, it follows that humans have to have access to the source code of a piece of software for it to be considered free.

Unlike proprietary software where only the original software creators (or those they explicitly provide access) have access to the source code, anyone who is interested can get access to the source code of free software. Therefore, if a user of free software wants to modify or improve it, she is free to do so. In many cases, the people who make the

improvements make the improved software available to the broader public via the Internet. By definition and in practice, people who, in most cases, are connected to each other just through the Internet create free software collaboratively. One critical aspect of the creation of free software that is often overlooked is the feedback in the form of complaints and suggestions from normal users. This feedback is actively sought and many tools exist that make it easy for lay users to integrate these complaints, bug reports and suggestions into the production methodology.

So how is this different from the production of other kinds of software?

Typically, entities that produce non-free software usually have very tight restrictions on who has access to the source code of their programs and distribute their software only in object code format. The reason for this is that while it is very easy to compile source code into object code, it is very difficult to get the original source code from the object code. An analogy would be curds. While it's easy to make curds from milk, it's pretty much impossible to get milk from curds.

What kinds of people make free software?

Many people who write free software are volunteers, they probably have an unrelated day time job. These people spend their free time developing free software.

Commercial organizations that benefit from free software distribution or that provide free software support also develop free software by investing portions of their profit. An example of such an organization is RedHat.

There are many non-profit organizations that raise funds to develop free software, through donations from free software users. The Free Software Foundation is one such organization. Other examples are SPI, Gnome Foundation, Mozilla Foundation, and the like.

Some free software packages are developed by universities. The Festival text to speech engine, Octave - the Matlab clone are examples of software developed by universities.

Many commercial organizations also contribute to the development of free software, because these organizations benefit from the existing free software code base. For example IBM maintains the port of Linux to PowerPC CPUs, because it needs an OS for its CPU.

But I still don't understand why anyone would want to give away their work for free?What's in it for them!?

For love or for money.

Yes, people do make money by releasing software created by them as free software. Corporate bodies like MySQL and RedHat to name a few, make money because they release software created by them as free software (and offer support contracts for free software). And they do find that they are able to make more money than they ever would have made if they kept the software as non-free.

It's misleading to use the term "give away" to mean "distribute a program as free software". It implies the issue is price, not freedom. One way to avoid the confusion is to say "release as free software".

There are many reasons why people write free software. Some people might just want their work to spread out to the world. Many people would like to live in freedom. They contribute to free software so that they can continue to live in freedom. Some people write free software just for the fun of it. They love programming and hence use their programming skills to do something useful.

You might want to read Eric Raymond's paper titled Homesteading the Noosphere [13]. Eben Moglen in his paper titled Anarchism Triumphant: Free Software and the Death of Copyright [14] explains "Moglen's Metaphorical Corollary to Faraday's Law". This law also explains why people develop free software.

Arguments for using Free Software

I'm still not convinced.Surely a big computer company knows best when it comes to designing software?Why would I want to use software designed by an amateur?

Free software is NOT created by amateurs. The free software development process is open and transparent. If you want to include your code into a free software project, it will be scrutinised by several people. Amateur and or badly written code will be rejected outright.

Though your assumption that "big companies" are better at designing software is questionable, "big companies" are free to develop free software too. And they do develop free software. For example RedHat, Sun, Novell, and IBM, all develop free software.

If there are a group of people who would like to have a particular software written, they can bring together funds, hire a programming company, get the software written by professionals and then release it as free software. There is no reason why free software has to be written by amateurs.

Even if the free software designed by the amateur is inferior than the non-free software designed by a professional, you might want to use the free software because it gives you freedom, which is more important. Also, a better programmer that likes the idea could contribute to the project.

It is certainly not true that larger companies know better about designing software. They depend on a small group of programmers and are not generally in close contact with the users as in the case of free software. Thus, free software developers are often more aware of the needs of the users and their complaints about the existing versions. Some large companies allow technically incompetent marketing experts to make software design decisions that prioritize marketing requirements over user's needs and software soundness.

In any case, examinations of many free software applications show that today they are as good as or better than equivalent non-free applications, or are reaching there fast. And a large fraction of free software developers are not amateurs. And how do you know if the big company which sells non-free software did not hire an amateur programmer to write that code?

What we use depends on what we want. For computer users, software that can do the things they want done is a necessity. If such software does not exist, then they cannot do the particular kind of work. How well the work can be done, and how quickly, depends on the quality of the software and hardware available. It is, therefore, desirable to have software that enables the users to do the work with least effort and to get the best possible

output. These qualities of the software generally improve with time, the term 'standing on the shoulders of giants' applies to free software, so improvement begets improvement. Under non-free software it's always a matter of 'reinventing the wheel', or worse, to circumvent proprietary restrictions for another company or person to improve the code.

But more important than all these is the quality of freedom that the software has. If the software is restricted, and the company that makes it withholds all information about how the software is created and in what format the files are created, then the users become dependent on the company, and subject to exploitation. Because these are much more important in the larger context, it is important to use free software rather than non-free software.

But what about bugs?Surely free software is more likely to be virus prone?

The question involves two different types of computer related problems - bugs and viruses.

Bugs are unintentional errors in programs. With free software when you find a bug in the program, you have the freedom to exercise freedom two, the freedom to help yourself and correct the program. If you are not a programmer you are free to report the bug to the maintainer of the software or hire any programmer and correct the program. You are not under the mercy of any single organization. By submitting the bug fixes to the maintainer of the package the software package becomes better and better.

Virus is a malicious program that infects other programs by embedding a copy of itself in them. For more on these security risks, see the section below.

Since the source code is available, won't it be easy for someone to find out a security loop hole and exploit it?

Yes, but before the source gets to the hands of people who exploit loopholes, it passes through hands of people who develop free software. And they usually fix such loopholes.

And with free software, exploitation of a loophole is very, very, very quickly reported and fixed, often within hours.

Look at it this way, since the source is available, it would be easy for someone to find out a security loop hole, and make a patch for it, even before its exploited!

Also, the GNU operating system was designed with security in mind from the start.

So you're saying that free software actually evolves at a faster pace than non-free software?

Free software tends to evolve extremely rapidly. This is primarily because of the way it involves its users, who contribute bug reports and even code patches to the tool's development. The path and the pace of development is extremely open and only features and issues that are needed ever gets done. Release fast and release often is the main 'mantra' of this kind of development.

Note however that this benefits only tools which have reached the stage of being in popular use. Once this happens it is in the interest of the users too to promote and actually contribute to the development. In a way this ensures only the deserving and useful tools gets supported in this manner.

Contrary to popular assumptions this model is laissez faire in the truest and most efficient manner as possible. Demand and supply works best with more than finite number of suppliers and consumers. And this is what happens here.

The pace of evolution is actually controlled by the popularity and the usefulness of the software. The more popular a piece of software is, the faster it evolves. In fact if something hinders this pace in any manner, like a dis-interested maintainer or an abandoned company, often other people step up as developers of the project. Or sometimes, forks of the project spring up (not that they don't always other times) to continue with the development, under another banner since the original source is free to be used or maintained by anyone.

Compared to the isolated model of development employed by the proprietary software, this is a tremendous plus. There exists no questions about the continuity of the tool either, which is the ultimate barrier in the path of software evolution.

This concept is explained really well in Eric Raymond's paper The Cathedral and the Bazaar [15].

Free software is only something used by computer enthusiasts, right?

Wrong.

Free software is used by people who value their freedom more than anything else.

Many millions of people the world over use free software without actually realizing it the moment they access popular sites on the Internet - see the section below.

Have any established organizations actually used free software to their advantage?

- This page is hosted on a server running free software (GNU/Linux, Apache/1.3.29 (Unix), PHP/4.3.4).
- More than 98% of all the Domain Name Servers, which identify the machine on which a page (like http://wikibooks.org/) is situated, use free software called BIND.
- More than 80% of all web servers use free software, called Apache, to serve their sites. But sadly, since Apache isn't copylefted, many of these servers run proprietary versions of Apache.
- More than 60% of all network servers run the Linux kernel, another free software.
- The TCP/IP implementation on most computers, (including those running non-free operating systems) is free software.
- "Digital Domain used 105 DEC Alphas running RedHat [GNU/]Linux to simulate and render water for James Cameron's *Titanic*" -- http:// www. computer. org/ computer/ homepage/0202/ec/

Personal Relationship to Free Software

What kinds of problems might I expect to encounter using free software?

Exactly same kind of problems you face with non free software.

Some problems with free software today is,

- Incoherence. Take manuals for example. Some free software come with info manuals, some with man pages, some with html documents, some others with plain text files, and yet some others with only source code comments!
- Inability to use certain patented algorithms. But that is not a problem with free software as such.
- Non-availability of drivers for certain devices, especially "WinModems."
- Free software cannot be legally used (at least in the US) for certain activities that involve copy-protection techniques. This includes playing encrypted DVDs.

Okay, but why would I want to modify my software anyway?

There are plenty of reasons. Say for example a software does not support your local language. You would like this software to be available in your local language so that you can use it. If the software is proprietary you will have to go and beg the "owner" of the software. If he finds making the change wouldn't be profitable, he will not make the change. With free software, you can make the change yourself or you can got to a programming company and ask them to make the changes for you (Which is how Richard Stallman supported himself for some time). With free software you are not helpless.

Also consider this ever plausible situation. Some time during your education, you would have written a project, or thesis or dissertation. In all probabilities, you may have used a computer belonging to a friend or the university to create this work. There is no assurance that you would get the same software or package which you used to create this file a few years from now, when in a nostalgic mood, you decide to go through your past work. This is a time when you will feel like modifying whatever is available to read a new file format.

If you are any kind of organisation which needs (of as it often happens, obliged by law) to retain certain info over a long time, the right to modify your software through a third party vendor is 'the' most important right.

Look, I'm no computer whiz! Isn't it easier for me to just use packaged software?Who do I turn to when something goes wrong?

Using free software is all more important if you are not a computer expert. That way, you do not have to depend on the company from whom you purchased non-free software.

First, read the documentation that came with the software. If that doesn't help, try searching the web with a search engine like mozDex [16]. *Last* try one of the things below.

If you are looking for "free of cost" help, there will always be a free software user group or Linux User Group in your locality. Find one. Or ask any of the innumerable mailing lists or forums which provide support. You will be surprised at the response and support you receive. One forum is Nuxified.org [17]. There are more popular forums, however they don't set a good example by having nonfree software (vBulletin) power their sites.

If you are willing to spend money, you can always hire a company or a consulting programmer to help you.

You can also buy support from expert companies. Since the software itself is open, the support vendor cannot lock you down like it happens with proprietary softwares. You are always free to go to another support vendor.

Also if your requirement is pretty huge, it will be highly cost effective to have an in house software development and support team who will take the free software and customize it for your needs and constantly maintain it.

But how do I know I can trust someone not linked to a big company that has a reputation to uphold?

Several big companies do provide support for free software. Several big companies charge several times more than several individual or small companies for their services. Remember, in the free software scenario, you are paying only for the services, not the product or package. So, there will be immense savings, and companies, or consultants who work on free software will always have lesser turnover, even if they experts and market leaders in their field.

Plus a little-known company/person can gain a bad reputation.

Which GNU/Linux distribution (or distro for short) should I use?

There are many publicly available GNU/Linux distros. Distrowatch [18] maintains a list of distros with summaries of their features. Fedora Core [19] (formerly RedHat), OpenSuse [20] and Ubuntu [21] are popular ones for new users (listed in alphabetical order). There are also many more distros such as Damn Small Linux, Gentoo, Morphix, Slackware, etc.

Ubuntu [22] is a distribution based on Debian GNU/Linux but aimed at being easy to use. They have non-free software available, but it is not enable by default. It has a Live CD available, which allows you to test the distro without installing it on your computer.

OpenSuse [23] is a powerful yet easy-to-use distro. Their Evaluation edition has non-free software, but the Open Source Software edition does not.

Fedora Core [24] is also new user friendly.

Try Debian if you are good at computers and have a little working knowledge of the GNU/Linux system. The primary strength of Debian is the sheer number of packages. It also has support for many different architectures. Debian also has distributions which don't use Linux as the kernel such as Debian GNU/Hurd and Debian GNU/kFreeBSD. Use these only if you are really interested in developing free software. Be warned though, Debian has nonfree software in its package repositories that are suffixed with "-nonfree.

You should stay out of Mandriva (formerly Mandrake), MEPIS, KNOPPIX, Linspire... These distributions come with non-free software packages by default.

If you are looking for a 100% free distribution, you should try gNewSense [25].

Depending on your need there are even specialised GNU/Linux distributions. There is one which plays only Video CDs. There are several which are used only as firewalls. Some distros run from one or two floppies. Some distros are intended to be used as rescue disks. There are also distros even for musicians and Geographical Information Systems (GIS). Most major distros come with several Compact Disks, but you can have a running and usable system with usually the first one or two CDs.

Where can I find free software?

Some good sources are:

- GNU Savannah [26]

NOTE: The next two also list non-free software, although most of it is free.

- SourceForge [27]
- FreshMeat [28]

I have some software for Windows that I can't find a replacement for on GNU/Linux! Where can I find one?

Look here [29].

Okay, so how would I begin to install free software on my machine?

Once again, the same way you begin to install non-free software on your machine. You can buy computers with free software pre-installed. Or you can get a friend to do it for you. Or find a professional for doing it.

If you decide to do it yourself, try lots of different software until you find what you like.

Most distributions have a built-in way of easily installing software and finding all its dependencies, called a package manager. Some package managers are:

- Smart [30] supports many types of repositories, including yum, apt, apt4rpm and urpmi.
- Yum [31] for Fedora Core
- Apt [32] for Debian
- Apt4RPM [33] for RPM distributions
- Portage [34] for Gentoo
- Ports [35] for FreeBSD

How can I make software I've written free software?

See the last section of the GNU GPL [36]. Be sure to write free documentation [37] (or get someone else to do it)!

How can I make money off of free software?

The easiest is selling free software. Be sure to explain to them that it's a matter of freedom, not price. Distributing this book with the software might be a good idea. Remember that if the software's under the GGPL and you distribute binaries, you must include the source with the binaries and/or offer the source separate.

You could also write free documentation [38] and sell that. The same things in the above paragraph apply here.

Another way is, if you can write software, put yourself for hire to add/change something in free software. If you are the maintainer of the software, you could have people pay you to make a(n) addition/change higher on the priority list for the project. This is how Richard Stallman used to support himself, and he's still here.

External links

[1] http://www.gnu.org/philosophy/audio/audio.html
[2] http://www.freedomtoaster.co.za/
[3] http://www.gnu.org/copyleft/gpl.html
[4] http://www.gnu.org/copyleft/gpl.html
[5] http://www.gnu.org/licenses/license-list.html
[6] http://www.gnu.org/philosophy/why-free.html
[7] http://www.gnu.org/philosophy/shouldbefree.html
[8] http://www.gnu.org/copyleft/fdl.html
[9] http://creativecommons.org
[10] http://en.wikipedia.org/wiki/open_content
[11] http://en.wikipedia.org/wiki/GFDL
[12] http://www.gnu.org/gnu/why-gnu-linux.html
[13] http://www.catb.org/~esr/writings/homesteading/homesteading/
[14] http://emoglen.law.columbia.edu/my_pubs/anarchism.html
[15] http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/
[16] http://www.mozdex.com/
[17] http://www.nuxified.org/forums/
[18] http://distrowatch.com/
[19] http://fedora.redhat.com/
[20] http://en.opensuse.org
[21] http://ubuntulinux.org
[22] http://ubuntulinux.org
[23] http://en.opensuse.org
[24] http://fedora.redhat.com/
[25] http://www.gnewsense.org
[26] http://savannah.gnu.org/
[27] http://sourceforge.net/
[28] http://freshmeat.net/
[29] http://www.nawaz.org/wiki/index.php?title=Table_of_Equivalents
[30] http://labix.org/smart
[31] http://fedora.redhat.com/docs/yum/
[32] http://www.debian.org/doc/manuals/apt-howto/index.en.html
[33] http://apt4rpm.sourceforge.net/
[34] http://www.gentoo.org/doc/en/handbook/handbook-x86.xml?part=2&chap=1
[35] http://www.freebsd.org/ports/
[36] http://www.gnu.org/copyleft/gpl.html#SEC4
[37] http://www.gnu.org/philosophy/free-doc.html
[38] http://www.gnu.org/philosophy/free-doc.html

Source: http://en.wikibooks.org/w/index.php?oldid=1360101

Contributors: Darrelljon, Paul tomlinson, Robert Horning, Spiritia, Whiteknight, 15 anonymous edits

License

Version 1.2, November 2002

0.PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.
This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.
We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1.APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.
A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.
A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.
The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.
The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.
A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".
Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.
The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.
A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.
The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2.VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.
You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3.COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.
If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.
If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.
It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4.MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5.COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6.COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7.AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8.TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9.TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10.FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.