



Contents

page 1: We stand firm in our resolve for freedom

page 3: Properly license your program for free software's sake

page 5: JShelter helps protect your online privacy: Here's how to contribute

page 7: Is it possible to buy a house in freedom?

page 10: Free software: An investment in human potential

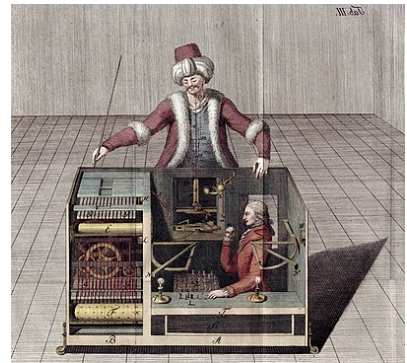
We stand firm in our resolve for freedom

By *Zoë Kooyman*
Executive Director

Led by farmers, advocating for the right to repair their (John Deere) tractors, the Right to Repair movement has been gaining momentum for more than a decade now, and is being advocated for all kinds of devices. The reparability of mobile phones and computers is fiercely defended by organizations like iFixit, and, this October, the movement took a major step forward thanks to the passage of a new law in California. The Right to Repair movement is one of the most successful movements of digital

autonomy we have seen in many years here at the Free Software Foundation (FSF).

We're excited to see where things head from here, since the Right to Repair and the free software movement are closely linked. In today's life, appliances and other personal technology are accelerating their pace in a transition from being purely mechanical toward becoming a technology reliant on software. This means that any meaningful notion of repair has to include user rights over that software. Without the freedom to run, modify, distribute, and share the software ourselves (i.e. software freedom), there will be no widespread right to repair. The top-down effect



On page 10, we examine the worrisome trend towards investing in machine potential in lieu of human potential.

of a bill becoming law, as Right to Repair has accomplished, can be the beginning of an unstoppable change.

But the real change has already happened, because change is like an iceberg or a snowball. The iceberg's large underwater surface symbolizes all the work we've done thus far, which has contributed to this relatively small moment of visible success. And, like a snowball, a movement gaining traction picks up activists along the way. A critical mass of ordinary people, by the millions, is formed and becomes an unstoppable force driving change.

We see Right to Repair's successes as more ice for the iceberg of software freedom—accumulating at the surface as another step toward freedom for all computer users. Since the GNU announcement, which initiated the free software movement forty years ago, we have seen many steps toward freedom. Today, 96% of all code bases incorporate free software—an immense achievement. But the FSF will only be satisfied when *every user* has the freedom to run, copy, distribute, study, change and improve the software. We want *all* computer users to be able to do everything they need to do on *any* computer using *only* free software. User freedom will increase the control users have over their technology, and therefore their lives. It will take away unjust power from technology corporations, and it

will redirect it to the individual. And in an increasingly digital world, free software becomes increasingly essential for freedom in general.

We have taken a lot of criticism for being “absolutists,” “inflexible,” “uncompromising,” “self-righteous,” and some even less elegant words for advocating exactly the above. Why? Because FSF brings every conversation back to *freedom*. Who would have thought that the organization that started this movement all those years ago is still advocating for the exact same thing and continues to stand by the values put forth all those years ago? The GNU General Public License (GPL) has enabled decades of constructive engagement, because it requires companies to give back improvements they distribute, under the same terms to everyone, and its terms don't change with new company leadership or after an acquisition. Demanding freedom is what led us to this 96%.

When the FSF says “freedom,” we mean it. Free software has many qualities that we are very proud of, including practical ones, yes. But we do not advocate for those practical benefits that free software merely *facilitates*, such as sustainable code, large communities, avoidance of vendor lock-in, or tailor-made solutions. They are major benefits, indeed, but *user freedom* is what we care about. It is also the standard

against which we measure the actions of others to judge whether they are true supporters of our movement or simply looking to gain from its benefits and successes.

Standing our ground is vitally important. Freedom is not a short-lived victory or something to turn into material gain. It's what protects users today and in the future. As long as we stand our ground, we continue to define the space between where we are now and where we want to be. The far end of the spectrum has an important role to play in every social movement—it's what moves the needle. If the FSF budes, even a little, the value of user freedom is diluted. We would send the message that it's okay to give up some freedoms as long as there's some gain on the other side. We reject that reasoning. We stand firm.

Properly license your program for free software's sake

*By Craig Topham
Copyright & Licensing
Associate*

For years, the free software movement has been defending its work toward a more free and just society. The antagonists of this movement, on the other hand, would rather operate in a world in which free software does not exist, but they can't. They can't because, along with free programs and other free licenses,

Right to Repair stands for increased user rights for everyday technology owners, allowing for more control over one's tools and creating less waste. In short, these are benefits that align well with many social movements of today. Because, these days most technology inherently integrates software, we celebrate the success of the Californian right-to-repair law. Meanwhile, we continue to push for user freedom everywhere. We know this doesn't make us the most popular, or the easiest to work with. As challenging as it may seem, we play our part as the lighthouse, the vanguard, and perhaps even the bullheaded. We practice what we preach, creating space for people to navigate towards software freedom, collecting critical mass, just below the surface like the mighty iceberg. 🧊

the GNU Project and the GNU General Public License (GPL) have earned us this victory for software freedom. What the antagonists of freedom can do, however, is keep software freedom under constant attack—which, to their shame, they have chosen to do. We urge you to respond to their attacks by taking the time to properly apply a free license to a program, which is an important way to make software freedom stronger. Freedom itself is at stake.

The good news is that properly licensing a program is simple—so

simple that I'm going to be able to provide a brief overview in this very short article. If you would like more details, you can visit our recommendations on "How to use GNU licenses for your own software" via u.fsf.org/418. If you have any questions, you can send them to licensing@fsf.org. Licensing a program is not difficult, but because there is no "official" way to do it, we unfortunately don't see enough consistency across the tens of thousands of free programs available. We developed and maintain a set of GNU licenses that serve the purpose of protecting the freedoms, each with a specific scenario in mind. You can learn more about each of them at gnu.org/licenses/. When you choose a GPL for your work you are sending a message to the world that you want your program to be free—and to stay free—for all its future users. This is how the free software movement gains strength. To be sure, our guides and recommendations are not legal advice, and you should consult a lawyer if you want to make sure how to take advantage of copyright and licensing in your particular situation.

First and foremost, you must be certain that you are the copyright holder and that nobody can make a claim on your work. Both employers and universities may be in a position to make such claims based on an

employment agreement or university policy. Like any legal document, it is wise to read those and, if you have doubts as to the status of your work, to consult legal counsel before signing. Optimally, you should ask your university or employer to sign a copyright disclaimer.

Per our recommendations, every source code file should contain the following in its header:

1. The name of the program and a short description of what it does.
2. The copyright notice with the year the file was created or modified and the name of the author, e.g. "Copyright 2023. Craig Topham."
3. The copying permission statement, which is a clear message that the software is freely available under the GPL, AGPL, or LGPL. This would include not only which version it is under, but whether it is only this specific version of the license or any later version the FSF might publish, e.g. "GPLv3 or later."
4. A disclaimer that there is no warranty or fitness for a particular purpose. (Alternatively, you may offer a warranty at your discretion. Doing so may be used as a strategic option for businesses who develop and sell free software.)
5. Finally, a note specifying that the recipient should have received a copy of the GPL so that they know their rights.

Including all five of these

important pieces of information in your source file headers makes it clear that the program is free software. With every source code file, it helps for people to see the words “free software,” so they understand those words have meaning and that those words matter. This is why we reject the term “open source.” We want to talk about freedom, and if you do too, then we urge you to put that message first. The reason to provide the notice on every file is so that this vital message and the necessary licensing information is not lost, even when the file is separated from the rest of the source.

Optionally, if you really want to spread the message of freedom and the GPL, you can also include these notices to be visible when the program starts up. This would reach a larger audience of users who might not be looking at source code files.

JShelter helps protect your online privacy: Here's how to contribute

*By Michael McMahon
GNU/Linux Systems
Administrator*

Protecting your online privacy has never been more important, as nowadays everything we do online leaves a digital footprint. As our world is ever-changing, one never knows what activity permitted today will be banned in the future. It is best practice,

The next important step is to place a copy of the GPL in the source code's main directory. Not only is it useful for recipients, it is a requirement of the GPL. Naming the file COPYING makes it clear to anyone that, if they plan to make copies of the program, they should read this file. If you are using the GNU LGPL, we recommend you name the file COPYING.LESSER.

Finally, besides having the file headers, it is helpful to recipients to offer a central location where the copyright and licensing information can be found. This is best done in a README or ABOUT file located in the source code's main directory.

In summary, if you care about software freedom, a properly applied GPL to a program furthers that cause. Anything less denies users the opportunity to learn more about free software. 🐶

in my opinion, to do as much of your online research and other browsing anonymously to prevent problems in the future. If you do not actively protect your privacy, your behavior will be tracked, for example, through increasingly invasive advertising practices. Once collected, your data can be sold to law enforcement to be used against you as evidence, to marketers to sell you things you do not want, and to malicious actors to trick you based on your desires. Every step that you take to enhance your

privacy online can go a long way to making your life more secure.

One step you can take is using JShelter together with thousands of people who do so already. What is JShelter? JShelter is a web browser extension used to thwart websites that try to identify users through their web activity, across websites whether or not they log in. Users who may assume their activity is anonymous on one site can be identified through a different site where they log in, for example, by comparing similarities in their browser environment. JShelter prevents web pages from turning the browser into a proxy to the local network. JShelter also blocks Web Workers, which is code running in the background of a browser that can be abused to install long-lived man-in-the-middle proxies and other attacks.

You might want to share location with certain sites during navigation. However, for other sites, you may want to reveal your location with less precision (like ordering a meal or looking for nearby places of interest). JShelter protects the user in other ways through features like JavaScript Shield, Network Boundary Shield, and Fingerprint Detector (known as “FPD”). The extension is free software and available on all major web browsers. You can find installation instructions at jshelter.org for your web browser. Discover how JShelter can enhance your privacy.

JShelter has gone through several changes since we last promoted it six months ago. Since that time, we have made improvements on usability, performance, and internationalization. We also recently added a way to identify scripts that call APIs that JShelter protects (for example, to create custom block lists), fixed several bugs, and improved the Frequently Asked Questions (FAQ) page. Everyone stands to benefit from JShelter’s privacy-protecting features, and journalists and activists would likely find the most benefit. As the number of people using privacy tools increases, the collective privacy from using those tools tend to increase as well.

You can make a difference and help make JShelter better by identifying, reporting, and fixing bugs. Even if it seems minor, every issue and workaround that you suggest might help someone else. Your reports can help influence the roadmap of development for the extension. Grow your technical skills by looking for existing issues, making your first contribution, and building your résumé. See our contribution page for ways to contribute: u.fsf.org/40z

Are you fluent in multiple languages? Help translate JShelter into your native language to help your region. Translating JShelter into more languages allows us to help protect the privacy of a wider audience from

around the globe. With each additional translation, we can break down barriers so that people from more countries are able to take back their privacy. This is a crucial effort because the risks related to losing privacy online vary greatly by country, and sometimes have dire consequences. If you can read and write in multiple languages and would like to contribute to this project, please visit the following web page with information on how to get started: jshelter.org/i18n/


Is it possible to buy a house in freedom?

*By Christopher Howard
Simulator Technician and Free Software Advocate*

Note: This article was submitted to us via email for publication in the Bulletin. A full-length version of the original is published on Christopher's personal site at: u.fsf.org/415

I am a free software advocate who tries to compromise as little as possible with my principles. For me, this means not only avoiding installation of nonfree software applications on my devices, but also blocking JavaScript in my web browser in order to avoid nonfree web apps. Recently, I have been trying to purchase a house, which has put me in a position to test how difficult it is to purchase a home nowadays without

Join our passionate group by making contributions. You can make a difference building something great. JSshelter needs your help. We are working to create a better world. Try it, and contribute, today!

Full details: u.fsf.org/40z 



JSshelter development is made possible by FSF, NLnet Foundation, and BRNO University of Technology.

using nonfree software. For those reading this who need more background on what nonfree JavaScript is and its dangers, please read: u.fsf.org/416

A clarification before going deeper: We are only dealing with the question of whether or not you can personally get through the process without using any JavaScript or web apps directly. Your real estate agent (we'll just say "agent" to save time), your lender, and your seller will most likely not be as vigilant in avoiding proprietary web apps.

The short version is that, in principle, it's possible to get through the whole process without JavaScript or proprietary web apps. However, in practice, it will be very difficult to do so without making a few compromises, and, in any case, the process will be painful and may involve some sacrifices.

Viewing listings

The first hurdle I had to deal with was to find and evaluate house listings. Friends and Internet searches all pointed me to listings on Zillow. On Zillow, it is possible to view the house address, image thumbnails, and a few minor details without enabling JavaScript. But if you want to view the photos at full size, or look at other details, you would need to enable (nonfree) JavaScript.

When I got hooked up with an agent, he was able to set up a system where listings would be automatically sent to me in an HTML email. His system allowed me to view most of the listing information without JavaScript enabled, except unfortunately most of the property photos.

File transfer and digital signatures

I'm not wealthy, so I needed a lender. I couldn't get help from my bank (long story), so I had to go with another in-state lender. Immediately, the lender wanted me to send a boatload of documents such as paystubs, bank balances, etc. The lender assumed that I would be sending the documents to them over their "secure portal" web app. Additionally, in order just to *find out* which documents I was supposed to send, I was supposed to download a file from the portal.

Several rounds of emails followed in which I tried to work out some other solution. Finally, the lender

called me, rather suspicious, wanting to know why I wasn't sending the documents, and what it was I was trying to hide. I stumbled through a brief explanation of software freedom principles, and eventually he grasped the idea well enough that he was willing to work further with me.

Of course, I still needed to send the documents, and I don't happen to have my own trusted, free-software file transfer system set-up. (Well... I took it as a given that I wouldn't be able to get them to agree to use SFTP.) So, the lender emailed the document list to me as an attachment, and then I mailed the documents via USPS. This introduced some extra delays and expenses, but it worked.

Then there were challenges to signing the documents. The lender, agent, and sellers were expecting every form and contract to be signed via DocuSign, which is a proprietary web app for signing documents. So, I had to explain to three or four different people why I didn't want to use that program. Fortunately, my agent has an office in town, so we worked out a system where all documents would be sent to my agent's assistant, who would print them off, have me sign them in the office, and then scan and send copies to whomever needed them. This kept everyone happy, and it had the added bonus of allowing me to easily ask questions about agreements before signing them.

The main difficulty with this system is time pressure. With a lot of the forms, it doesn't matter if it takes you a day or three to get over to the office. But for things like offers and counter-offers, in our very tight market, it was necessary to get contract forms sent through within a few hours, or the whole deal might fall through. And the agent's office wasn't generally open on the weekend. So, I made a compromise once or twice in using DocuSign for a few time-sensitive documents. But I didn't feel too bad about it, since at least everybody knew that is not what I normally do and the reasons why I felt it important to avoid.

First-time home-buyer course

To get me a rate-discount as a first-time home buyer, the lender is required by some law or regulation to put me through a Freddie Mac home-buying educational course. This is one of those proprietary web apps where you have to spend hours clicking through slides and doing multiple-choice tests.

The only ways that I see to take the test in relative freedom are: 1.) to take the test at a library, which would avoid running the app on one's own computer and 2.) asking for an exception so that the test could be mailed to you, taken, and returned in the mail, but I'm unsure if such an exception would be granted.

Conclusions

So, what did I learn from my experience? It *is* possible to go through the home-buying process without using proprietary web apps, at least if the first-time home buyer discount doesn't matter to you. But expect a roller-coaster ride as you navigate trying to get everyone else on board with your plan. Also, practically speaking, it is very difficult to evaluate listings without being able to look at the photos, which likely will require nonfree JavaScript.

It helps if you have a lender and an agent who both have an office nearby. In my case, nobody was opposed to printing out documents and having me sign them with "wet ink" provided that we could meet deadlines.

In the process, I also learned that brief explanations of software freedom principles usually don't communicate very well. To get the idea across, you need to explain the four freedoms, and then also explain how that ties into JavaScript and web browsing, since most people don't really know what a web app is or how JavaScript plays into that. 🍷



Free software: An investment in human potential

By Devin Ulibarri, Outreach & Communications Coordinator

Lately, there has been a lot of news that highlights the achievements of *machines*. We are constantly bombarded with updates about what new trick a large language model can do, the latest machine learning innovations that are being worked on, and the substantial investments being made into business startups that specialize in “A.I.” There is much speculation over the potential of these machines—what *they* can do and learn. While achievements generally should be celebrated, there is something unsettling about the fervor we are experiencing today. Let’s take a moment to examine what seems to be a dangerous trend toward investment in machine learning over human potential and explore the implications of this trend for the free software movement.

First of all, the free software movement has *always* been about human potential. Free software activists fight for freedoms on the software running their devices because it ultimately *protects and expands their own potential*—both as individuals as well as community members. When we lack computational freedom ourselves, we lack the important access we need to

unlock our potential. In many ways, it’s because we believe in our own (individual and shared) potential that we fight for our freedoms.

Proprietary software, on the other hand, *locks* people out of their potential by denying them their freedoms. Source code is hoarded away, along with the potential to study or modify it, for no one but the developer to see. Developers are tasked to design methods of restricting the ways in which software is used, or what we call “Digital Restrictions Management” (DRM). DRM denies people the potential to do many things, including strengthening their community through sharing. And, of course, all of these things require resources, whether those be financial or not. Creating a proprietary license typically requires financial resources spent on hiring a lawyer to draft an End User License Agreement (EULA). Implementing DRM requires hiring a team of developers to engineer it. People and businesses who develop proprietary software are ultimately making investments into methods and procedures—whether legal, digital, or otherwise—that have a consequential negative impact on the realization of potential of real-life human beings.

Free software requires resources, too, but it’s not spent on drafting a customized EULA or on designs that will prevent people (including the

licensee) from using the software as they wish. In fact, it could be argued that an investment in free software is one of the most efficient, least wasteful investments that could be made, because few, if any, resources are ever spent on such things. The majority of a project team's resources are spent developing and improving the software at hand, solving problems, and helping others do the same. And while an entire budget cannot go toward development, the ratios for resource allocation (i.e. investments) for free software projects tend to be more human-centric, by far.

And regardless of the particulars of a given project, an investment in free software offers benefits that proprietary software *never* can. Whether it's your time or money that you spend assisting free software development, those who use the software are empowered with freedoms that they can take with them. Future generations will be able to better fulfill their potential by freely studying and improving upon the software. Additionally, free software and DRM-free media make historical preservation and anthropological research possible in a way that restrictions cannot, allowing for future generations to better understand and reflect upon how we got to where we are today.

But what can we take away from all of this? Well, first we can insist on

technology that fundamentally respects human potential and whose development is not *at the expense* of progress in other human endeavors. Free software was designed to respect the dignity of the people using it. Only free software places “user” as equal to “developer” in a moralizing gesture of trust, democratization, and empowerment. As for endeavors whose overall consequences are less well understood, such as machine learning and large language models, we should insist on the same principles of community and the dignity of granting the same freedoms that we enjoy ourselves.

While we continue to make strides in technological innovation now and in the future, let's not forget about the potential that *people* inherently possess. As progress is made that improve how *machines* work, let's make even more of an effort to recognize all the *human* effort that has gone into such advancements. And when resources are being allocated to software projects, please insist that the software be free software, because an investment in free software is an investment in our future and our ever-growing potential as a species. 🤖

Join the FSF!

Visit: my.fsf.org/join



Donate to the FSF with Bitcoin:
1MiL7aKG3YAY8rKqW
HJaoE8w7ZWfFSLmjU

Copyright ©2023
Free Software Foundation, Inc.
The articles in this *Bulletin* are
individually licensed under the
Creative Commons Attribution-
ShareAlike 4.0 International license.
[https://
creativecommons.org/
licenses/by-sa/4.0/](https://creativecommons.org/licenses/by-sa/4.0/)

Published twice yearly by the Free
Software Foundation, 51 Franklin
Street, 5th Floor, Boston, MA 02110,
(617) 542-5942 info@fsf.org

This *Bulletin* was produced using all
free software, including GNU Emacs,
Inkscape, Scribus, and GIMP.

IMAGE CREDITS

Page 1: *The Turk*. Illustration by Joseph
Friedrich Freiherr von Racknitz (1744-1818).
Created 1789. Public Domain.

Page 7: Logo design by Manufactura
Independente. © 2023. Free Software
Foundation, Inc., licensed under a Creative
Commons Attribution ShareAlike 4.0 (CC BY-
SA 4.0) International license.

Page 9: *Clay house souvenir*. © 2023. Image
by Boaventuravinicius. CC BY-SA 4.0.

How to contribute

Associate membership:

Become an associate member of
the FSF. Members will receive a
bootable 16GB USB card, email
forwarding, and an account on
the FSF's Jabber/XMPP and Jitsi
servers. Plus: access to our
members forum at
forum.members.fsf.org! To
sign up or get more information,
visit member.fsf.org or write
to membership@fsf.org

Online: Make a donation at
donate.fsf.org, or contact
donate@fsf.org for more
information on supporting the
FSF.

Jobs: List your job offers on our
jobs page: fsf.org/jobs

Free Software Directory:
Browse and download from
thousands of different free
software projects:
directory.fsf.org

Volunteer: To learn more, visit
fsf.org/volunteer

LibrePlanet: Find local groups in
your area or start your own at
libreplanet.org! And join us
in person or online for the yearly
LibrePlanet conference next
spring.

Free Software Supporter: Receive
our monthly email newsletter:
fsf.org/fss