FREE DVD

Double-Sided DVD
INSIDE!

MANAGING PDFs in LibreOffice

SNAPSHOTS
PROTECT YOUR DATA

LINUX MAGAZINE PRO

LINUX PRO MAGAZINE

MAY 2019

SNAPSHOTS

A better way to protect your data?

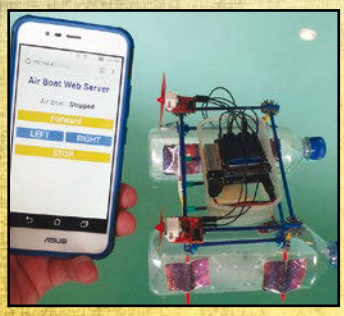
Plasma 5.14

Exploring KDE's new age desktop

Optimize Gimp Images with Python Plugins

tmate

Emergency repairs from the command line



New in Bash

Cool new commands for the terminal window

WiFi Airboat

Plastic bottles, duct tape, and a little MicroPython

OpenWRT

2 Internet connections from one home router

LINUXVOICE

- CherryTree and Piggydb outline editors
- Filesystem Tools
- maddog: The rise of open hardware

FOSSPicks

- Neuronify
- Gnome Internet Radio Locator
- Chrysalis

Tutorials

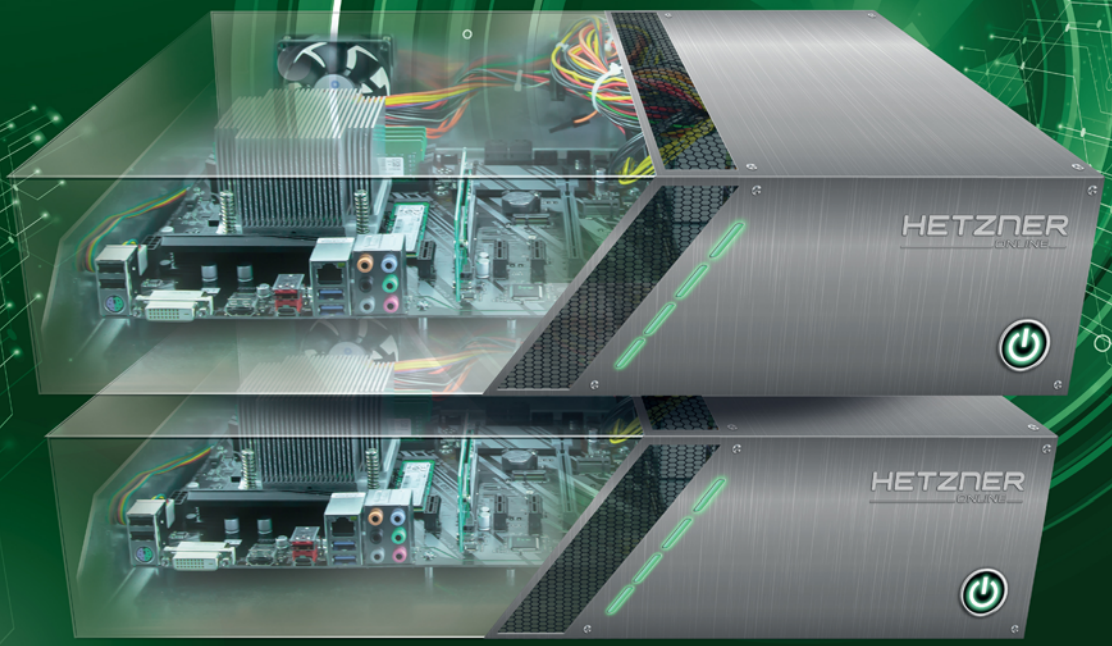
- Shell Tests
- OpenSCAD

Issue 222
May 2019
US$ 15.99
CAN$ 17.99

HETZNER ONLINE

OUR NEWEST ALL STAR. DEDICATED ROOT SERVER EX52

Pick your preference: Enterprise HDDs or NVMe SSDs



Dedicated Root Server EX52

- ✔ Intel® Core™ i7-8700 Hexa-Core Coffee Lake incl. Hyper-Threading Technology
- ✔ 64 GB DDR4 RAM
- ✔ 2 x 8 TB SATA Enterprise HDD 7200 rpm
- ✔ 100 GB Backup Space
- ✔ Traffic unlimited
- ✔ Location Germany or Finland
- ✔ No minimum contract
- ✔ Setup Fee $66.80

monthly from $ **61.10**

Dedicated Root Server EX52-NVMe

- ✔ Intel® Core™ i7-8700 Hexa-Core Coffee Lake incl. Hyper-Threading Technology
- ✔ 64 GB DDR4 RAM
- ✔ 2 x 1 TB NVMe SSD
- ✔ 100 GB Backup Space
- ✔ Traffic unlimited
- ✔ Location Germany or Finland
- ✔ No minimum contract
- ✔ Setup Fee $66.80

monthly from $ **61.10**

All prices exclude VAT and are subject to the terms and conditions of Hetzner Online GmbH. Prices are subject to change. All rights reserved by the respective manufacturers.

www.hetzner.com

BIG DEAL

Dear Reader,

When I started this job 15 years ago, Debian was a really big deal. The whole Linux community paid attention to Debian elections, and this magazine even had a regular monthly column devoted to Debian affairs. It would not be an exaggeration to reveal that I even knew a guy who had the Debian spiral tattooed on his neck.

Over the years, though, the Debian project has faded from the foreground, with Arch Linux filling the über-geek niche and enterprise variants from Red Hat and others capturing the greater share of media attention.

But if you're really tuned in to Linux, you'll know that Debian is still really important. Dozens of the best known Linux variants grab the Debian codebase and adapt it to their needs, including cult favorites like Knoppix and CrunchBang, security and privacy distros like Kali Linux and Tails, and game distros like SteamOS. The Raspbian Linux running on your Raspberry Pi is based on Debian. All flavors of Ubuntu are based on Debian, and by extension, all distros based on Ubuntu, such as Linux Mint, Zorin OS, and elementary OS are based on Debian code.

The Debian project was back in the headlines this month, when Debian developer and maintainer Michael Stapelberg revealed in a blog post that he is "winding down" his contributions to the project. With over a thousand Debian contributors, the loss of one package maintainer (even a prominent one) is not necessarily big news. The real news is Stapelberg's detailed critique of Debian development methods, which scored a Slashdot link and has already generated lots of comments from columnists like me.

Although it is neither possible nor necessary for me to recount his entire discussion (which you will find at his blog [1]), I will touch on a few of the highlights. The general theme is that Debian development methods are antiquated, and the decentralized nature of the Debian development process creates headaches for developers. Stapelberg's detailed commentary is organized by topic:

- Change process: He writes, "While it is great to have a lint tool (for quick, local/offline feedback), it is even better to not require a lint tool at all. The team conducting the change (e.g., the C++ team introduces a new hardening flag for all packages) should be able to do their work transparent to me."
- Fragmented workflow and infrastructure: "Debian generally seems to prefer decentralized approaches over centralized ones…non-standard hosting options are used rarely enough to not justify their cost, but frequently enough to be a huge pain when trying to automate changes to packages."
- Package Uploads: Package uploads occur through a system of batch jobs. Stapelberg claims that, depending on timing, you might wait for over seven hours before your package is actually installable.
- Bug tracker: The Debbugs bug tracker was created in 1994 and lacks some important features available with contemporary tools.
- Email archive: Stapelberg laments that Debian still doesn't have a conveniently browsable threaded archive of mailing list discussions.
- Machine readability: According to Stapelberg, "While it is obviously possible to deal with Debian packages programmatically, the experience is far from pleasant."

The problems of aging code are well known to all software developers. Left to its own minimum energy state, a codebase is an artifact of the time in which it was envisioned. The tools and methods used to build the code become an integral part of the product. In some respects, the decentralized nature of a project like Debian forces a kind of continuous evolution, which tends to attenuate some of the aging effects. But, as Stapelberg points out, the decentralization causes other issues, especially for a very large project. The current trend in software development is automation, which puts the emphasis on uniformity and centralization – traits that are deemphasized in a vast, decentralized project.

The problems outlined in Stapelberg's blog post seem to call for decisive action from a committee or individual empowered to impose uniformity and standardization in a culture where these traits have no history. The question is whether a community project like Debian could muster the unity and strong leadership needed for big and decisive changes. Of course, it is fair to say that Stapelberg's critique is just the opinion of one contributor; many Debian developers might not agree with him. Perhaps more importantly, some developers who agree with him about the headaches of Debian development still might not want to change Debian culture, which has always been based on principle rather than on considerations of efficiency and business.

Whether or not the Debian developers decide to address the issues raised in Michael Stapelberg's critique, I hope his blog post starts a fair, open, and honest discussion about what they really want and what kind of development environment would serve them best. Linux distributions come and go, but Debian really is a big deal – just as it was 15 years ago.

Joe Casad,
Editor in Chief

Info

[1] "Winding down my Debian Involvement," by Michael Stapelberg: *https://michael.stapelberg.ch/posts/2019-03-10-debian-winding-down/*

LINUX MAGAZINE

WHAT'S INSIDE

Modern filesystems make it easy to take a snapshot – a precise copy of your data at a moment in time. This month we review some leading Linux snapshot tools. Also in this issue:

- **Redundant Internet Links** – With a little help from OpenWrt, you can support two Internet uplinks from your little home router (page 32).
- **tmate** – access a remote shell session (page 48).

Stop by MakerSpace to build your own WiFi-controlled airboat. Check out LinuxVoice for a helpful tutorial on Bash test conditions and a study of two useful outline tools: CherryTree and Piggydb.

SERVICE

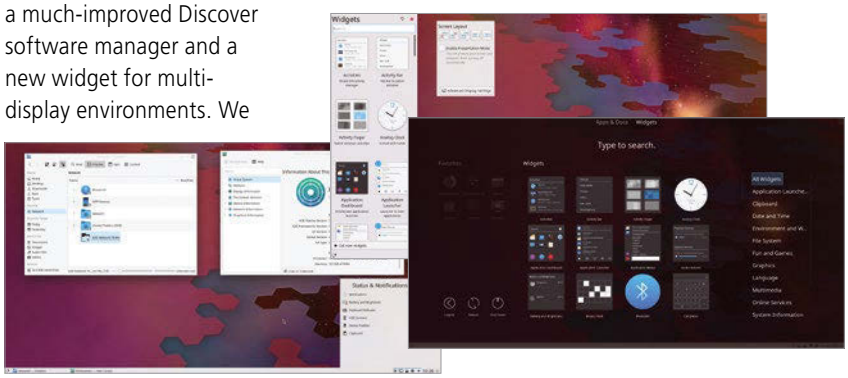
NEWS

COVER STORIES

Experts agree about the need to keep a copy of your data, but restoring from a set of incremental backups takes time and sometimes doesn't work as expected. The other option is to capture your data in a snapshot. We review some leading Linux snapshot tools.

REVIEWS

Plasma 5.14 brings some highly visible improvements for KDE users, including a much-improved Discover software manager and a new widget for multi-display environments. We tour the latest version of KDE's iconic desktop.



IN-DEPTH

MAKERSPACE

SEE PAGE 6 FOR DETAILS

LINUXVOICE

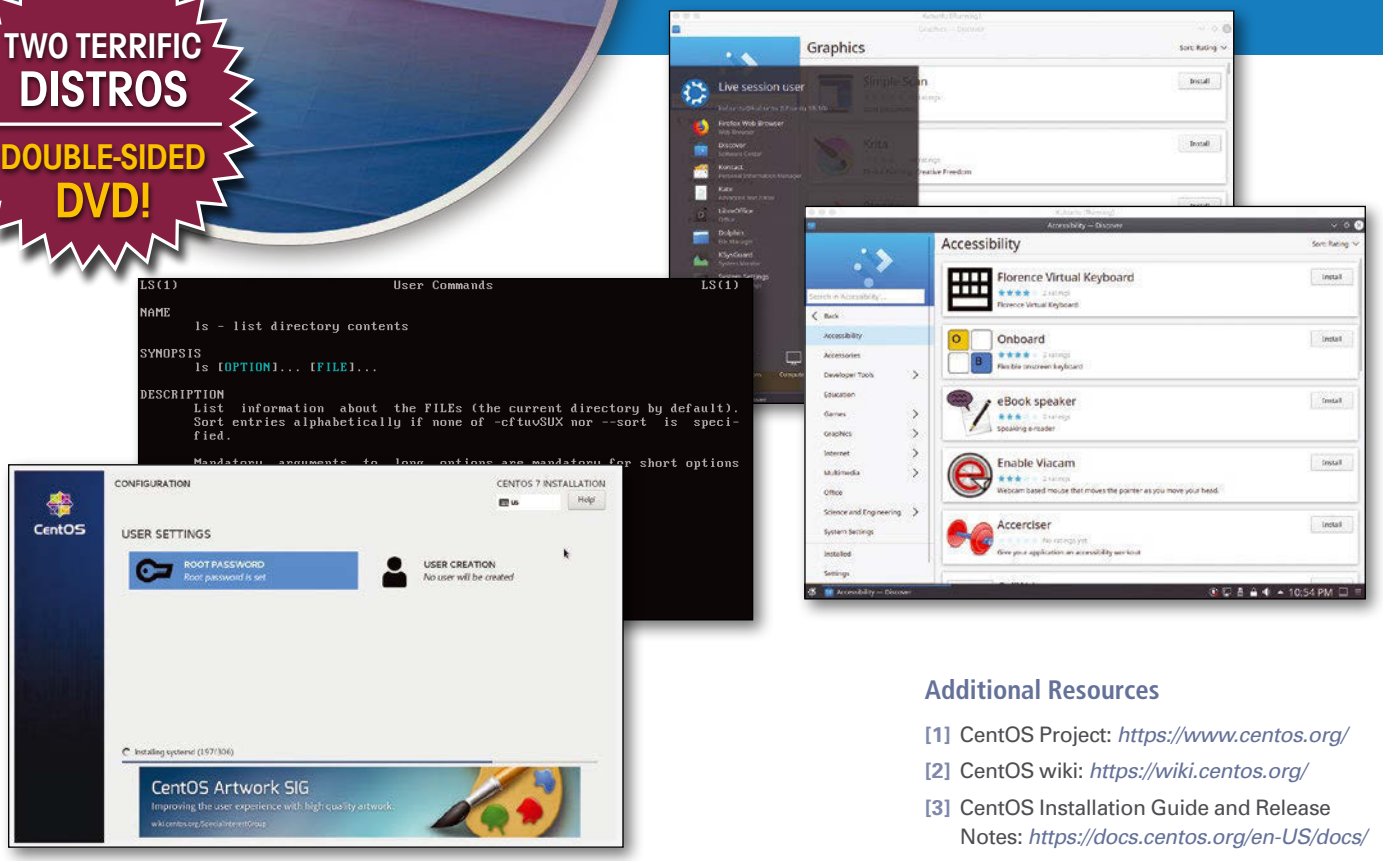
On the DVD

CentOS 7.6 (1810)

CentOS is a community-sponsored distro closely tied with Red Hat Enterprise Linux (RHEL). CentOS Linux is a popular and inexpensive option for web servers and other systems that benefit from enterprise-grade engineering without the need for costly subscription agreements. The latest release comes with source code derived from RHEL 7.5.

Kubuntu 18.10 "Cosmic Cuttlefish"

Kubuntu is an Ubuntu flavor based on the innovative KDE desktop environment. According to the Kubuntu developers, version 18.10 includes "...some exciting improvements, with newer versions of Qt, updates to major packages like Krita, KDEConnect, KStars, Peruse, Latte Dock, Firefox, and LibreOffice, and stability improvements to KDE Plasma. In addition, Snap integration in Plasma Discover software center is now enabled by default, and Flatpak integration is also available."



Defective discs will be replaced. Please send an email to subs@linux-magazine.com.

Although this Linux Magazine disc has been tested and is to the best of our knowledge free of malicious software and defects, Linux Magazine cannot be held responsible and is not liable for any disruption, loss, or damage to data and computer systems related to the use of this disc.

Additional Resources

[1] CentOS Project: *https://www.centos.org/*
[2] CentOS wiki: *https://wiki.centos.org/*
[3] CentOS Installation Guide and Release Notes: *https://docs.centos.org/en-US/docs/*
[4] Kubuntu: *https://kubuntu.org/*
[5] Kubuntu Manual: *https://kubuntu.org/support/*
[6] Ubuntu help: *https://help.ubuntu.com/*

ISC HIGH PERFORMANCE 2019

FUELING INNOVATION

FRANKFURT, GERMANY

- 5 DAYS
- 450 SPEAKERS
- 160 EXHIBITORS
- 3500 ATTENDEES

JUNE 16 – 20
2019
FRANKFURT, GERMANY

SAVE THE DATE!

ISC High Performance
The HPC Event.

Platinum Sponsors:

Atos · Hewlett Packard Enterprise · HUAWEI · intel

Next-Generation High Performance Components | Exascale Systems | Extreme-Scale Applications | HPC and Advanced Environmental Engineering Projects | Parallel Ray Tracing -- Visualization at its Best | Blockchain Technology and Cryptocurrency | Parallel Processing in Life Science | Quantum Computers / Computing | What´s New with Cloud Computing for HPC | Parallel Programming Models for Extreme-Scale Computing | Workflow Management | Machine Learning and Big Data Analytics | Deep Learning and HPC – State of the Art

isc-hpc.com

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

VMware Rolls Out Essential PKS

VMware is launching a new Kubernetes solution (*https://blogs.vmware.com/cloud-native/2019/02/26/launching-vmware-essential-pks/*) called VMware Essential PKS. Essential PKS includes an upstream Kubernetes version backed with commercial support by VMware. The new solution is designed to address the needs of users who want vendor support without the heavy customization and modification often associated with vendor-distributed Kubernetes alternatives. Essential PKS also comes with reference architectures to inform design decisions.

Why would someone want Essential PKS (with an upstream version of Kubernetes) rather than VMware's own fully managed Enterprise PKS? According to Scott Buchanan, senior director, product marketing at VMware, many potential customers have already invested in Kubernetes and would prefer to keep the open source, upstream version, but without the overhead of in-house maintenance.

Linux 5.0 Is Here

Linus Torvalds, the creator of the Linux kernel, has announced the release of Linux 5.0. Despite any excitement around the major release number, the fact is these numbers really don't mean much. Torvalds has often said that he chooses a new number when the version number becomes too long. He simply doesn't want a situation where "the numbers are big enough that you can't really distinguish them."

Announcing 5.0, Torvalds wrote, "I'd like to point out (yet again) that we don't do feature-based releases, and that '5.0' doesn't mean anything more than that the 4.x numbers started getting big enough that I ran out of fingers and toes."

That said, there are many new features in this release, including support for GPUs. Linux 5.0 comes with improvement for AMD FreeSync, NVidia RTX Turing, and Raspberry Pi Touch Display support. It also comes with Google's Adiantum storage encryption system.

© guesswho, 123RF.com

As we reported earlier, 50 percent of the Linux 5.0 codebase consists of driver updates, 20 percent is architecture updates, 10 percent is tooling, and the remaining 20 percent is the rest of the stuff, including documentation, networking, filesystems, header file updates, and core kernel code.

Linux 5.0 also alleviates the performance hit that was caused by previous mitigations of Spectre and Meltdown bugs.

Kali Linux 2019.1 Available

Kali Linux is a distribution that shows up on *Mr. Robot* quite a bit. It's one of the best penetration testing and hacking tools out there. Ahead of the next season of *Mr. Robot*, the Kali Linux project has announced the first release of 2019, to package more tools for both Elliot and his sister Darlene.

Kali Linux 2019.1, updates the Linux kernel to version 4.19.13, fixes bugs, and includes many updated packages.

According to the project, "the big marquee update of this release is the update of Metasploit to version 5.0, which is their first major release since version 4.0 came out in 2011."

Metasploit 5.0 includes database and automation APIs, new evasion capabilities, and usability improvements throughout.

Kali Linux 2019.1 also includes updated packages for theHarvester and DBeaver. Kali now also supports ARM, which means you can use a single-board computing device as a swiss army knife that you can slip into your pocket.

"The 2019.1 Kali release for ARM includes the return of Banana Pi and Banana Pro, both of which are on the 4.19 kernel. Veyron has been moved to a 4.19 kernel and the Raspberry Pi images have been simplified so it is easier to figure out which one to use.

If you want to try Kali Linux, you can download the latest images from the Kali download page (*https://www.kali.org/downloads/*).

Linux Foundation Releases a New Draft of OpenChain Specification

The OpenChain Project is releasing a draft of its OpenChain Specification 2.0 (*https://www.openchainproject.org/news/2019/02/15/comment-on-the-next-generation-of-the-openchain-specification*).

© Irina Tischenko, 123RF.com

OpenChain is a critical open source project that offers a standard for open source compliance in the supply chain. Open source is powering the modern world; every company is consuming open source in one way or the other. It's becoming critical that they comply with the license used. "OpenChain provides a specification as well as overarching processes, policies, and training that companies need to be successful in managing open source license compliance so that it becomes more efficient, understandable, and predictable for participants of the software supply chain," said the OpenChain blog post.

The Linux Foundation has also announced that Microsoft is joining the OpenChain Project as a platinum member (*https://www.openchainproject.org/news/2019/02/06/microsoft-joins-openchain-platform*). Under the leadership of Satya Nadella, Microsoft has become more active in its support of open source initiatives. As a lot of open source code flows through Microsoft's own products and services, it's critical for the company to ensure that it is totally in compliance with open source.

"By joining the OpenChain Project, we look forward to working alongside the community to define compliance standards that help build confidence in the open source ecosystem and supply chain," said David Rudin, assistant general counsel, Microsoft.

Other platinum members of the OpenChain project include Adobe, ARM Holdings, Cisco, Comcast, Facebook, GitHub, Google, Harman International, Hitachi, Qualcomm, Siemens, Sony, Toshiba, Toyota, Uber, and Western Digital.

Hackers Start Exploiting Drupal Bug

Hackers have started exploiting a security flaw in Drupal that was patched last week. Imperva reported that they started seeing attacks on February 23, after the two vulnerabilities were patched and proof-of-concept (PoC) exploit code was made

MORE ONLINE

Linux Magazine
www.linux-magazine.com

Linux Administration Focus
http://www.linux-magazine.com/tags/view/administration

Wireshark • Ken Hess
Wireshark fills the gap between security and system administration for those who need to know more about what's flowing through the wires or over the airwaves in the corporate network.

Keep All Your Linux Servers in Check
Mayank Sharma
Use the browser-based Cockpit tool to monitor and administer multiple Linux servers without leaving your desk.

ADMIN HPC
http://www.admin-magazine.com/HPC/

Porting Code to OpenACC • Jeff Layton
OpenACC directives can improve performance if you know how to find where parallel code will make the greatest difference.

ADMIN Online
http://www.admin-magazine.com/

Migrate Your Workloads to the Cloud
Martin Loschwitz
Move a workload to the cloud without trouble, and leverage cloud benefits for a conventional setup.

Improved Defense Through Pen Testing
James Stanger
Discover indicators of compromise with open source pen testing tools.

Link Aggregation with Kernel Bonding and the Team Daemon • Martin Loschwitz
The Linux world has different implementations for integrating multiple network cards: kernel old-timer ifenslave and the younger libteam.

available publicly. Attackers tried to install CoinIMP, a JavaScript cryptocurrency miner on unpatched sites.

Drupal wrote in an advisory that CVE-2019-6340 and SA-CORE-2019-003 can lead to arbitrary PHP code execution in some cases, as some field types do not properly sanitize data from non-form sources.

The advisory said that a site can be affected if it meets one of these conditions: the site has the Drupal 8 core RESTful Web Services (REST) module enabled and allows `GET`, `PATCH`, or `POST` requests; or the site has another web services module enabled, like JSON:API in Drupal 8 or Services or RESTful Web Services in Drupal 7.

Drupal doesn't have any automated update mechanism (*https://www.drupal.org/project/ideas/issues/2940731*), and updating Drupal is more involved than updating WordPress, which means many sites may still be unpatched.

The vulnerabilities affect only Drupal 8 sites, unless you have Services or RESTful Web Services enabled in Drupal 7.

According to ZDNet (*https://www.zdnet.com/article/it-took-hackers-only-three-days-to-start-exploiting-latest-drupal-bug/*), there are only 63,000 Drupal 8 sites, which means there might not be enough incentive for hackers to spend their time searching out Drupal 8 sites to attack. Still, Drupal 8 admins are advised to install the patch as soon as possible.

LibreOffice Vulnerable to Remote Code Execution Flaw

Security researcher Alex Inführ has discovered a vulnerability in OpenOffice and LibreOffice that allows remote code execution (*https://insert-script.blogspot.com/2019/02/libreoffice-cve-2018-16858-remote-code.html*).

In a blog post, Inführ wrote that he found a way to achieve remote code execution as soon as a user opens a malicious ODT file and moves their mouse over the document, without triggering a warning dialog.

He demonstrated PoC, in which he created a hyperlink and changed its color from the default blue to white, so it would not raise suspicion. The link covered the whole page, increasing the chance of the user hovering the mouse over it. Remember, no clicking was needed; just hovering the mouse over the hyperlink was required to execute the payload.

The culprit here is the Python interpreter (`pydoc.py`) that comes with LibreOffice. It accepts commands and executes them via the command line.

LibreOffice has already released a patch; OpenOffice has not yet.

Early Warning for Seismic Events?

Scientists at the Caltech Seismology Laboratory are using deep learning technology to develop an early warning system for earthquakes. The research is based on the idea that electronic communication happens faster than seismic waves can pass through the ground. If you detect an earthquake at the point where it occurs, you have a few seconds to notify surrounding areas that the quake is coming. Those few seconds aren't enough for a major evacuation or human intervention, but they could be enough to stop trains or shut down power to power lines, which could help to reduce the damage caused by the quake.

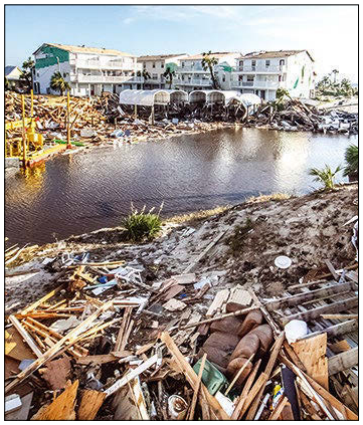


Photo by Alex Hall on Unsplash

According to the announcement (*https://blogs.nvidia.com/blog/2019/02/28/deep-learning-seismology-earthquake-early-warning/*), the deep learning models "...use convolutional neural networks to look at a single sensor at a time to identify seismic waves, narrowing down the sensor's data stream to a handful of discrete times with seismic activity. A second model, a recurrent neural network, recognizes wave patterns from several sensors over the course of a seismic event. The system unscrambles events that include multiple earthquakes in quick succession and can reduce false triggers by a factor of 100 — greatly improving the reliability for early warning systems."

Shop the Shop shop.linuxnewmedia.com

DISCOVER LibreOffice

Explore the FREE office suite used by busy professionals around the world!

Create your own:

- Word processing docs
- Spreadsheets
- Presentations
- Databases

Whether you work on a Windows PC, a Mac, or a Linux system, you have all you need to get started with LibreOffice today. This single-volume special edition will serve as your guide!

Order online:
shop.linuxnewmedia.com/specials

For Windows, macOS, and Linux users!

Zack's Kernel News

Chronicler Zack Brown reports on the latest news, views, dilemmas, and developments within the Linux kernel community. *By Zack Brown*

printk() Wrangling

Everyone hates the `printk()` function, because it's supposed to do something very simple – print output to the console – but in order to do that it has to overcome tremendous obstacles, like producing its output while the system itself is in the very midst of total destruction. Unlike other functions, `printk()` can't rely on any other part of the system actually working. At least, that's the ideal.

John Ogness recently tried to improve matters. Specifically, among other things, he wanted to free `printk()` from the constraint of requiring a global lock. He wanted to make it pre-emptible, so that even the deep, dark parts of the kernel that absolutely needed to be able to interrupt everything could still call `printk()` in times of crisis.

It's a worthy goal. If those deep, dark parts of the kernel can't use `printk()`, then they can't report what went wrong if they were involved in a system crash. Making `printk()` pre-emptible would allow every nook and cranny of the kernel to give relevant information that could help to debug the crash.

But it's not so simple. As Sergey Senozhatsky pointed out, if it's possible for `printk()` to be interrupted, then it's also possible for the very cause of the crash to interrupt `printk()` before `printk()` can finish making its final report on that cause. In this case, instead of preserving that final piece of debugging information, it would simply be lost.

John explained that, in fact, his patches implemented two separate cases for `printk()`. In most cases, `printk()` would be pre-emptible and would be used for plain old system messages of no particular importance. But if the system started to crash, `printk()` would revert to its old method of locking out all interruptions in order to use the final microseconds of life to give its best guess as to what happened.

John's code included other improvements. Traditionally, `printk()` timestamps were taken from the end of the `printk()` operation, even though `printk()` might have been waiting for a long time before obtaining its global lock and creating its output. John's code changed this so that the timestamp would be calculated by calling the routine that actually invoked `printk()` – a more accurate representation of when the reported event actually occurred.

The discussion was fairly short, but the patch may have trouble making it into the kernel. John's code does more than simply speed up `printk()` and make it more reliable; it also changes some of the philosophy behind `printk()`, from a general logging tool to a tool that requires more thought and consideration from the user. The sticking point is the new concept that some messages have a higher priority than others and that `printk()` will actually behave differently depending on the type of message it tries to output.

Persistent Memory as Generalized Resource

Dave Hansen recently lamented the fact that persistent RAM, which retains its state across reboots, could not easily be used as regular RAM when needed. Instead, applications had to be coded up special to recognize and use persistent memory. Dave felt that if the user didn't care about persistence, then there should be no obstacle to simply using persistent

Author
The Linux Kernel Mailing List comprises the core of Linux development activities. Traffic volumes are immense, often reaching 10,000 messages in a week, and keeping up to date with the entire scope of development is a virtually impossible task for one person. One of the few brave souls to take on this task is **Zack Brown**.

RAM as regular RAM, with no special coding needed.

There's some slight controversy about this. Intel in particular, as Jeff Moyer pointed out, has already implemented memory modes for its Optane RAM chips to let the system choose to treat the chips either as regular memory or as persistent memory. Presto, no problem.

But this wasn't good enough for Dave, nor for Keith Busch, who pointed out that an architecture-specific solution would only apply to a specific chip or family of chips rather than providing general support for all persistent RAM chips that might come along.

Dave's idea was to write a single driver that would detect all persistent RAM chips and make them available to the kernel as regular memory. And – in response to Keith pointing out that persistent RAM tended to be slower than regular RAM – Dave's code would prioritize faster RAM and only use the slower persistent chips when no faster option was available.

Meanwhile, Dave had nothing bad to say about Intel's memory modes – he saw memory modes as complementary to his own work. However, he saw his own code as providing a generalized solution for chips that didn't implement their own memory mode support. And beyond that, he did point out that memory modes weren't as flexible as his proposed driver.

For example, his driver could allocate just a portion of persistent RAM to be used as regular RAM, leaving the rest available as persistent RAM to any application that was written to make use of it. Also, Dave said, his driver would remain able to identify persistent RAM that was made available as regular RAM, so that applications could decide whether to use it or plain RAM chips, which tended to be faster. Memory modes allowed no such distinction, forcing applications to use RAM without knowing whether it was faster or slower.

No decision was made on Dave's patches per se during the discussion, but it's clear that something like these patches will make it into the kernel at some point. Ultimately, the entire Linux kernel – and all operating system kernels throughout the universe – are resource management tools. As new resources become available, and as existing resources splinter off into diverse categories of related resources, Linux will inevitably come to recognize them and provide whatever sort of access is most useful to user code.

Making Kernel Headers Available on Running Systems

Developers have plenty of opportunity to compile user code that interacts with the operating system kernel, but this requires having the running kernel's header files available for the compiler to read. So, anyone wanting to write code to interoperate with the kernel must install the Linux kernel source for that kernel version. That can be a bit of a time suck, especially if you're writing software to be compiled on a wide array of machines that are each running their own kernel version.

Joel Fernandes thought that it would be nice if the kernel simply stored its header files at compile time and automatically made them available via a `/proc` subdirectory. Then it wouldn't be necessary to determine which kernel was running and subsequently arrange installing those sources in order to develop your project.

The idea did not appeal to Christoph Hellwig, who thought it as a complete waste of memory. Storing the kernel headers forever in RAM, he said, would only deplete system resources. He felt a better idea would be to simply have the build system produce a compressed archive and unpack it at the appropriate time during bootup.

Greg Kroah-Hartman, on the other hand, said there was nothing wasteful about Joel writing a module to export the kernel headers, given that users were not obliged to load that module. Also, he pointed out that there was already precedent for doing something along the same lines, given that `/proc/config.gz` was stored in RAM and contained the kernel's build configuration options.

People like Daniel Colascione were more than happy to sacrifice RAM on the altar of easily available kernel headers. Daniel was thrilled at how much easier Joel's module would make his various kernel experiments.

However, H. Peter Anvin sided with Christoph, saying that even though Greg was right that the user could

choose not to load the module, there was still no justification for building those header files into the kernel in the way Joel proposed. Peter felt that simply constructing a compressed filesystem and mounting it as needed was a much simpler solution that accomplished all the same goals.

There was no immediate resolution, though the disagreement was between several relatively heavyweight kernel hackers. I'd expect Linus Torvalds to let them duke it out amongst themselves for awhile before issuing a ruling.

Ultimately though, the disagreement seemed to be over the best implementation and not over whether to have the feature at all. So, especially given how excited the feature made someone like Daniel, it seems as though something along the lines of Joel's idea will get into the kernel at some point.

Kernel Licensing Hell

From time to time, the Linux kernel has to deal with copyright and licensing issues. The kernel has always been available under the terms of the GNU General Public License (GPL) v2. An interesting detail is that it would be virtually impossible to change that license, because of the tens of thousands of people who have contributed code to Linux over the decades. Changing the license would mean first getting permission from each and every one of those people to release their contribution under the terms of the new license. As a result, this seems to be unlikely.

At the same time, it's possible for contributors to release their own contributions under different licenses if they choose. If Barack Obama submits a patch, for example, he retains the copyright himself. So he could license that patch for use in the Linux kernel under the terms of the GPL v2, but he could also license that same code for use in a different project under any other terms he wanted, including terms that were incompatible with the GPL. The reasoning behind this is essentially the same as the reason why any change to Linux's licensing would require getting his permission to release his kernel contributions under the new license. He holds the copyright for his contribution, so he would be free to agree to the new license, just as he is free to release his code separately under any other license.

At the same time, it's also possible for Linux to load proprietary drivers that are released only as binary code. There may be some question as to whether those binary drivers are thereby obligated to be released under the GPL – but that's a question for lawyers, and I'm not one. However, traditionally, a running kernel is able to identify whether it has been "tainted" by a binary-only driver by asking the driver to identify its license. If the driver claims to be licensed under the GPL, then presto! But if the driver is licensed under some other license, it would not be able to claim GPL licensing without potentially creating some legal problems for itself. This is useful for the kernel developers, because they're unlikely to waste time chasing down bug reports from users with tainted kernels. What's the point of hunting for a bug that is in all likelihood buried in a binary-only blob of code doing goodness knows what? It's also the case that some parts of the kernel API only make themselves available to drivers that are released under the GPL.

There's another interesting case that came up recently on the Linux Kernel Mailing List: Drivers released into the Linux kernel under multiple licenses, such as the GPL and BSD licenses.

Thomas Gleixner recently pointed out that as far back as 2003 a patch went into the kernel that created some serious ambiguity in how modules should identify their license.

Before the patch, Thomas said, a module could simply use the string "GPL" to identify that it was released under the terms of GPL v2. The goal of the patch itself was to implement the feature I mentioned above – allowing portions of the kernel API to refuse to link to proprietary modules.

Thomas pointed out that as an undocumented side effect, the 2003 patch also changed the meaning of the string used to identify a module's license. Specifically, "GPL" no longer meant only GPL v2, but included any later version of the GPL, at the user's discretion.

The new string to specify only GPL v2 was "GPL v2." Additionally, three other options were also added, for dual

licensing and licensing with other additional rights.

One of the main problems with the 2003 change was that, as Thomas said:

"It broke all modules which were using the 'GPL' string in the `MODULE_LICENSE()` *already and were licensed under GPL v2 only.*

"A quick license scan over the tree at that time shows that at least 480 out of 1,484 modules have been affected by this change back then. The number is probably way higher as this was just a quick check for clearly identifiable license information.

"There was exactly one *instance of a 'GPL v2' module license string in the kernel back then –* `drivers/net/tulip/xircom_tulip_cb.c`*, which otherwise had no license information at all. There is no indication that the change above is any way related to this driver. The change happened with the 2.4.11 release, which was on Oct. 9, 2001 – so quite some time before the above commit. Unfortunately there is no trace on the intertubes to any discussion of this."*

Thomas also added, "The dual licensed strings became ill-defined as well because following the 'GPL' vs. 'GPL v2' distinction, all dual-licensed (or additional rights) `MODULE_LICENSE` strings would either require those dual-licensed modules to be licensed under GPL v2 or later or just be unspecified for the dual-licensing case. Neither choice is coherent with the GPL distinction."

To add to the horror of it, Thomas went on to say, "As of 5.0-rc2, 2,873 out of 9,200 instances of `MODULE_LICENSE()` strings are conflicting with the actual license in the source code (either SPDX or license boilerplate/reference). A comparison between the scan of the history tree and a scan of [the] current Linux tree shows to the extent that the `git` rename detection over [the] Linux tree grafted with the history tree is halfway complete that almost none of the files which got broken in 2003 have been cleaned up vs. the `MODULE_LICENSE` string. So subtracting those 480 known instances from the conflicting 2,800 of today, more than 25 percent of the module authors got it wrong, and it's a high probability that a large portion of the rest just got it right by chance."

To address this problem, Thomas posted a patch intended to:

"Remove the well-meant, but ill-defined, distinction between 'GPL' and 'GPL v2' and document that:

- *'GPL' and 'GPL v2' both express that the module is licensed under GPL v2 (without a distinction of 'only' and 'or later') and is therefore kernel license compliant.*
- *None of the* `MODULE_LICENSE` *strings can be used for expressing or determining the exact license.*
- *Their sole purpose is to decide whether the module is free software or not.*
- *Add a* `MODULE_LICENSE` *subsection to the license rule documentation as well."*

Joe Perches supported Thomas's patch, as did Greg Kroah-Hartman and Jessica Yu. Jessica also offered some textual fixes.

Rusty Russell also supported Thomas's patch, saying, "Thanks [for] taking on such a thankless task Thomas." And remarked, "Frankly, this should be autogenerated rather than 'fixed' if we want this done properly."

In fact, Rusty was the originator of the 2003 patch. By way of explanation, he said that he, "might have been overzealous in assuming a versionless GPL string meant 'or later' (I'm happy for that for my own code, FWIW). My memory is fuzzy, but I don't think anyone cared at the time."

Alan Cox also came in with his recollection of which string meant what in 2003. He said, "Versionless always meant 'or later' outside of the tags. It's the default version of the licence. (Whether 'v2' only has any meaning beyond intent is another debate that I guess some year a lawyer will have to figure out)."

In terms of additional licensing strings, Alan remarked, "They should just have used GPL and additional rights. Either you have GPL rights (and it's ok to use in the kernel) or you don't (and it's proprietary, and the rest is down to derivative works). We don't actually care whether its dual-licensed BSD, or whether it merely grants you an additional right to cheap pizza."

At one point, Thomas also remarked, perhaps somewhat bitterly, "Most people simply do not care, and I'd say most do not understand what licensing means in the first place. But at the same time, a lot of people do not care at all what they are doing."

And there you have it. ■■■

Choosing a storage snapshot tool

Clone It!

Experts agree that you should keep a copy of your data, but restoring from incremental backups takes time and sometimes doesn't work as expected. Alternatively, you can capture your data in a snapshot. Read on for a review of some leading Linux snapshot tools. *By Erik Bärwaldt*

Modern operating systems often include more than 100,000 files. In the event of system crash, bringing a system back online by restoring from incremental backups can be a complex and time consuming task. Data may even get lost along the way, if you forgot to back up certain directories or if you only created incremental backups without performing the occasional full backup.

Snapshots are a fast alternative to incremental backups. A snapshot is a copy of the dataset frozen at a point in time. This article compares several snapshot tools, including Clonezilla [1], CYA [2], Partimage [3], qt-fsarchiver [4], and Snapper [5]. See the "Not Considered" box for a few tools that did not make the list.

Requirements

Snapshot software differs greatly from backup software. A snapshot usually includes a drive partition. Unlike conventional backup sets, this unit cannot be supplemented and accelerated by differential or incremental partial backups.

Admins can thus reconstruct the complete system in a single pass in case of a crash. Unlike a backup solution, a snapshot image does not require importing several backup layers. One snapshot disadvantage is the increased time required for the application.

In contrast to backup applications, which are typically implemented in desktop environments and back up the current system, universally deployable snapshot programs have to be able to handle various filesystems. They also must be able to cope with heterogeneous IT infrastructures, as well as support common boot managers and the Master Boot Record (MBR) and GUID Partition Table (GPT) [9] layouts. In order to function in heterogeneous IT environments, the cloning programs also need to harmonize with older systems that still rely on conventional BIOS.

For systems with larger storage requirements, the applications also need to compress the generated images on the fly. In larger organizations with centralized data storage, it also helps if the generated system images can be stored on an NFS or SMB system via FTP transfer or an external storage option with SSH access if one exists. Network use is also

Not Considered

This article does not include doClone [6], because the last version was released four years ago, and the tool is not in the software repositories of the most popular Linux distributions.

The Duplicacy [7] command-line tool and rsnapshot [8] are also missing. Duplicity focuses on backups in the cloud; rsnapshot is similar in its functionality to more conventional incremental backup programs.

good for rollouts, where the admin can fuel a large number of identically configured computer systems with the same operating system in a short time.

Clonezilla

The best known example of cloning software is the free live system Clonezilla [1]. Developed in Taiwan, Clonezilla is based on Debian Sid or alternatively on Ubuntu 18.04.1 LTS and has no graphical user interface (GUI). You control the complete system via ncurses menus.

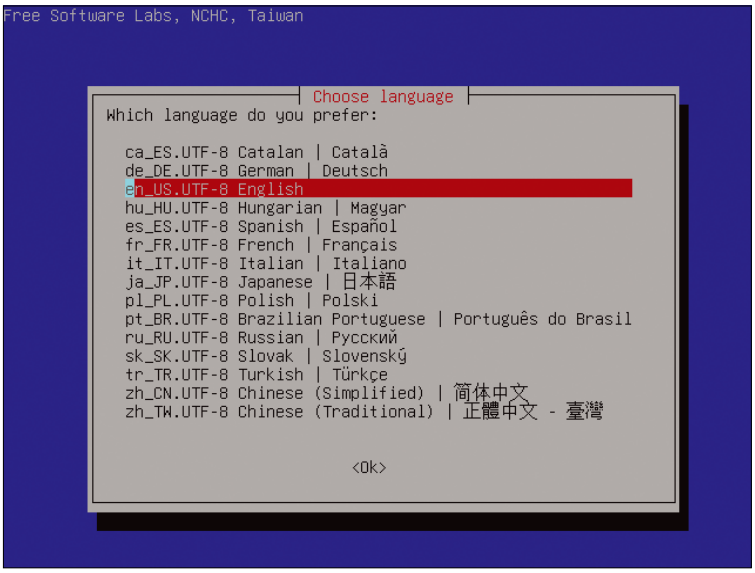


Figure 1: Clonezilla uses an ncurses interface.

Clonezilla is available in several variants. While the alternative Ubuntu version exclusively serves 64-bit systems, the Debian-based version also offers two 32-bit versions, one of which supports processors with the Physical Address Extension (PAE) [10].

The ISO images each are less than 300MB and fit easily on CD-ROMs, allowing the tool to run on very old computer systems [11]. The software is available as a hybrid image, which can therefore be launched from a USB-based storage medium. The developers stipulate a minimum of 196MB RAM for operation.

Clonezilla is also suitable for heterogeneous infrastructures: In addition to the filesystems commonly used under Linux, it also supports Windows, Apple's HFS+, and several BSD derivative filesystems. Also, Clonezilla can reinstall the current 2.*x* version of the GRUB bootloader (or the legacy version if required) and can cope with common older and newer BIOS versions.

On booting the Clonezilla disc, several startup options await the user in the GRUB boot manager. If there are problems with the screen resolution, Clonezilla also offers a generic VGA driver along with legacy SVGA and XGA resolutions. It is also possible to load the entire system into the computer's RAM, which noticeably accelerates work on older systems in particular.

When you select one of the start options, the system branches to an ncurses screen, where you can define the locale in the first step (Figure 1). Then, the actual Clonezilla tool starts. In the first window, it prompts you for the operating mode. This is where you say whether you want to clone a single partition or a complete disk.

The software either copies the source to an identical target or creates an image of the source disk or partition. As a further alternative, there are also possibilities for cloning on the network. The network option is intended for large-scale software deployment, while the local option is more suitable for individual cloning operations or for creating snapshots for data backup.

In just a few steps, Clonezilla guides the user to the finished image. To do this, it first queries the target for saving. You can choose from a list, whether this is a local medium; an SSH, Samba, or NFS server; or whether Clonezilla should park the image in the Amazon cloud. Another possibility is a path in RAM, but its volatility can cause problems.

If you go for the local option, you can specify the target disk in the next step. For external options, you can then connect to the source computer. The system automatically detects a plugged-in USB data carrier. Next, choose the source; Clonezilla lists the different partitions on the local disk.

The following ncurses screen allows a filesystem check to ensure the filesystem's integrity. Clonezilla also offers to check an image's recoverability after it has been created. In the last dialog, the user can also select encryption according to the AES-256 standard.

After a final confirmation prompt, Clonezilla creates the image. When the software is finished, you can shut down the system, restart it, or create another image (Figure 2).

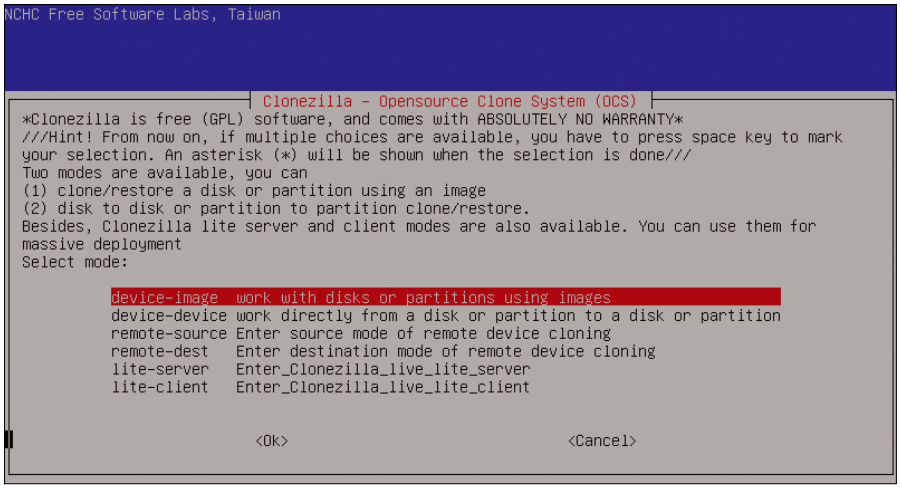


Figure 2: Clonezilla offers numerous snapshot possibilities.

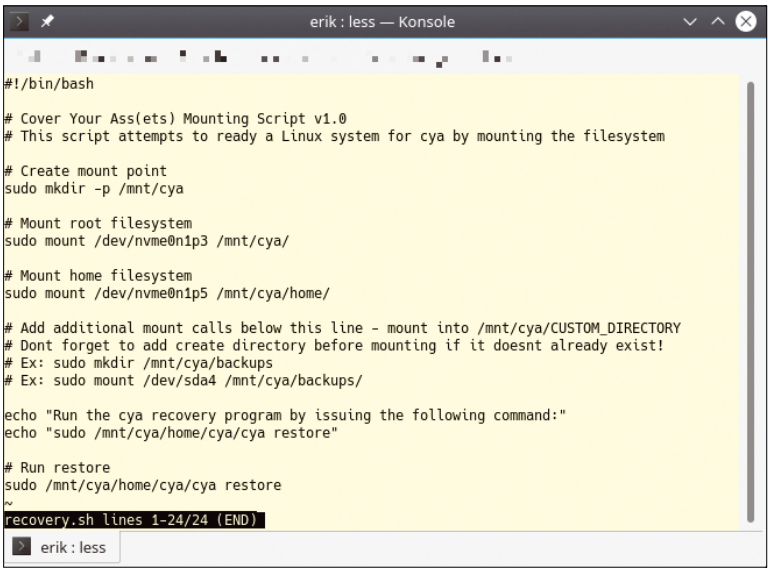


Figure 3: CYA generates a recovery script before creating an image.

If you use Clonezilla, you must make sure that the target partition is at least as big as the source partition when cloning a disk. In addition, the source drive must be unmounted while the cloning process is running or creating an image. Cloning does not work with the drive mounted.

Since version 2.5.2-31, the developers have also integrated a Clonezilla Lite server into the live system. The server is used to enable the rapid deployment of operating systems in a network environment; it can clone 40 plus computers simultaneously. The server functions and the corresponding dialogs are included in the previous routine's ncurses menus, so you don't need any additional training.

If you want to restore an image, just reverse the steps. After starting the live system, again set the locale. In Clonezilla, define the source medium where the images you want to transfer are located. In the next step, select the image repository in which the desired image is stored. In the last dialog, Clonezilla uses the `restoredisk` option to transfer the clone or image to the target disk.

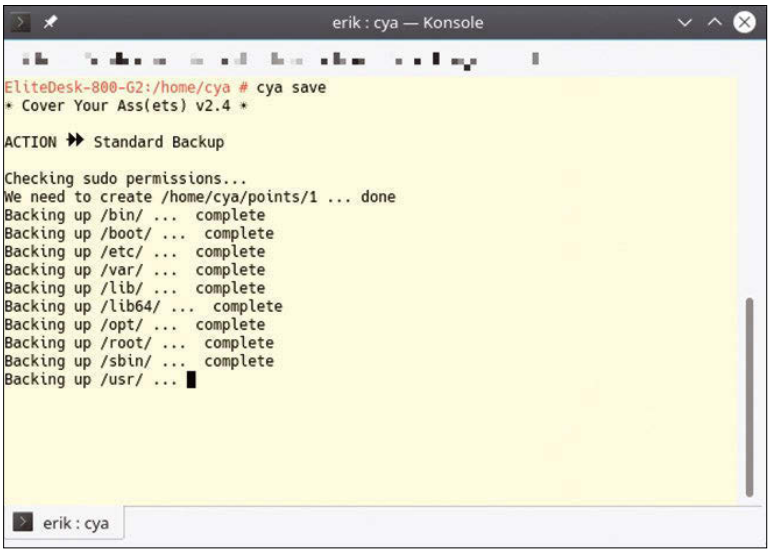


Figure 4: Because CYA works at the command line, the progress indicator is fairly simple.

CYA

US-based Cyberweb Solutions' Cover Your Assets (CYA) [2] is designed as an easy-to-use command-line program. It only takes single snapshots. Licensed under BSD 3 as free software, CYA is a bash script based on rsync [12].

To use the program, first install the Git version control system. Then download CYA with the following command:

```
git clone https://github.com/cleverwise/cya.git
```

In a final step, copy the application using the following command

```
sudo cp cya/cya <target directory>
```

to a directory of your choice. You can then launch it in a terminal.

In contrast to conventional backup solutions, CYA generates a system snapshot to restore a defective operating system. It does not back up user data. However, users can define their own file paths to exclude or include certain directories. For example, you can exclude the `/var/log` directory from the system snapshot. CYA is universally deployable, because it supports numerous filesystems, including ext2/3/4, XFS, UFS, JFS, ReiserFS, Btrfs, and ZFS. CYA will start from a USB stick if desired, which is recommended for mobile use.

CYA works with a recovery script that you need to generate before creating a snapshot. To do this, simply enter the `cya script` command at the prompt. The routine mounts drives and creates a `chroot` environment. Then the script ends up in the `/home/cya` directory (Figure 3). You will want to store the script on a USB memory stick if you are planning a mobile application.

If you want to include your own file directories, it makes sense to set up profiles, which you store in `/home/cya/cya.conf`. The file, which is empty at the beginning, is only used to define any number of profiles for backup purposes. Calling a profile by its name starts the backup.

In the configuration file, define the individual files you want to include in your snapshot. Using the `EXCLUDE_` parameter, you can specify additional paths to files that you do not want to see in your snapshot. This parameter also belongs in the `/home/cya/cya.conf` file. The CYA documentation [13] provides a useful guide.

By default, you start the program with the `cya save` command. CYA then executes the script and stores the data in the `/home/cya/points` directory. CYA shows the progress in a terminal window (Figure 4).

To create a snapshot with a name of your choosing, enter

```
cya keep name <Name>
```

at the prompt. This backup cannot be overwritten. If later on you want to overwrite the existing backup

with a newer version using the same name, you have to call the command with the `overwrite` parameter.

To archive a snapshot, enter

```
cya keep name <Name> archive
```

The command now creates, compresses, and archives the snapshot, saving the data in `/home/cya/archives`.

Restoring a snapshot is as easy as creating it. First, start the computer, which is no longer working with a live system. This should be as similar as possible to the installed system. Then mount the USB stick with the `recovery.sh` file on the running live system. The following command:

```
sudo /mnt/cya/home/cya/cya restore
```

starts a recovery, and CYA rebuilds the system.

If you want a computer to regularly create snapshots with CYA, you need to define a cronjob. Currently, CYA does not yet have the option of starting scheduled automated backups.

Partimage

Partimage [3] lets you convert partitions into images. If necessary, Partimage can restore these images at any time. The software is available in the repositories of almost all popular Linux distributions, and you can support it on any supported Linux system using the appropriate package management tools.

Partimage is also included with SystemRescueCD [14], a live system that is also suitable for 32-bit computers. You can use Partimage from SystemRescueCD without the need to install.

Partimage not only saves and restores images locally, but also across the network. The software supports many popular Linux filesystems, including ext2/3, XFS, UFS, JFS, and ReiserFS. Partimage also supports FAT16/32, HPFS, HFS+, and limited NTFS systems used by other platforms, which makes the software particularly suitable for use in heterogeneous environments. Unfortunately, Partimage lacks support for modern filesystems such as ext4 and Btrfs, so its suitability for Linux partitions is limited.

To use the ncurses program, enter the `partimage` command at the prompt of a root terminal window. If you use SystemRescueCD, you can activate the program at the command line using the same command or call it on the Xfce desktop via *System* | *Partimage*.

Partimage's ncurses screen then appears. The text window groups all the partitions found on the system; you can then choose the one to use for creating an image. Below the window, enter a name for the image in the input line.

The partition to be backed up has to be unmounted, because Partimage cannot back up mounted partitions. If you forget to do this, the software displays an error message and cancels the action.

Once you have an image name, you can then decide whether to save the partition locally in a file or on a network server. If required, you can specify the server IP and determine whether Partimage should transmit the data with SSL encryption (Figure 5).

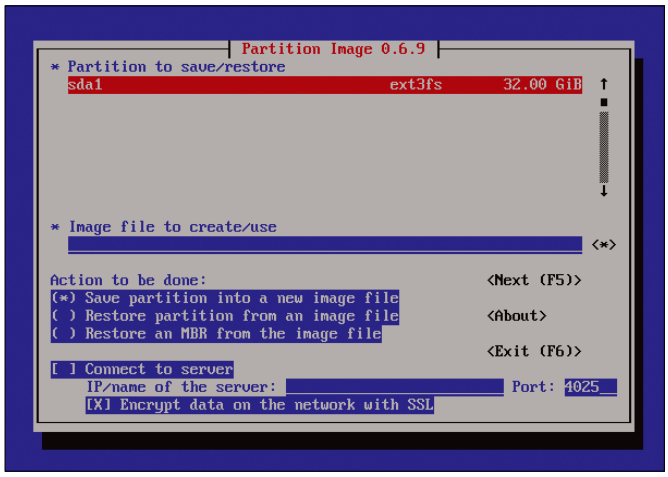


Figure 5: Partimage works with a classic ncurses interface.

Pressing F5 takes you to the next screen, where you decide whether Partimage should compress the data. Bzip2 and gzip are two more traditional compression formats. However, backing up a larger partition can take a long time, especially on slow computer systems with conventional hard disks. Uncompressed storage is a faster alternative, but it does not save space on the hard disk.

In the next dialog, you have the option to check the partition before saving it. By default, this option is enabled, because it protects data integrity. You can also add an image description at this point, if needed.

If you choose to enter a description, the program displays a new window with an input field. You will want to enable the *Prompt before overwriting* option; otherwise, the software will flatten existing images with the same name.

Under the *Image split mode* group, you can specify whether you want to split the image into different files. Either the software stops when the disk is full, or Partimage splits the image into several files of a fixed size, which you define in this dialog group.

The option *Wait after each volume change* interrupts the cloning process until the user has replaced a full removable disc. Finally, use the *If finished successfully* group to define an action that Partimage should start as soon as the software has

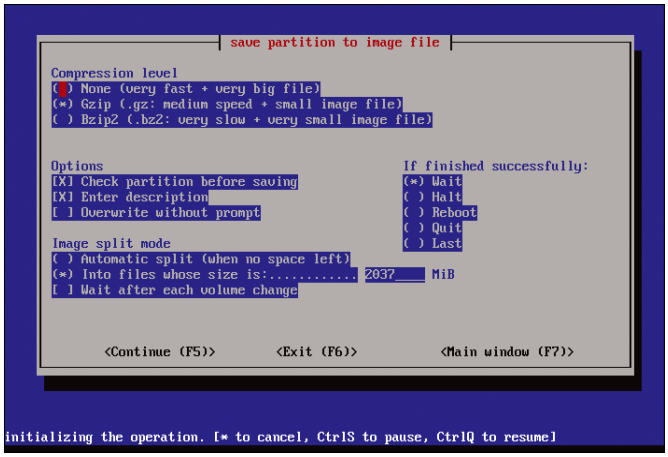


Figure 6: Partimage offers many options for creating images.

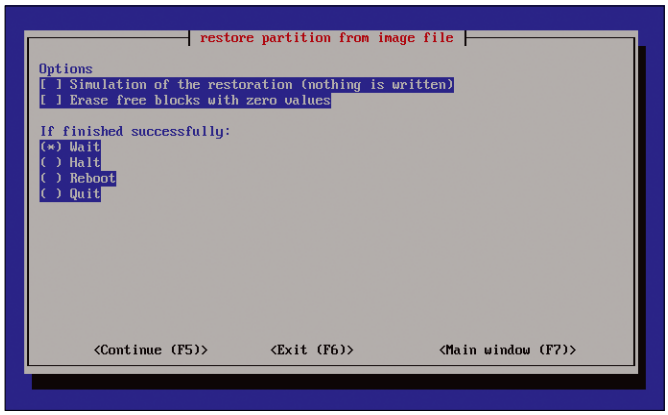


Figure 7: Partimage offers only a few restore options.

created the image. Pressing F5 displays a safety prompt, and Partimage then generates the image (Figure 6).

To restore an image, call the same initial dialog, but select the *Restore partition from image* option in the first window (Figure 5). You also need to specify the image's filename. Then press F5, which displays an Options window again. Under Options, you can also specify whether to simulate the restoration (nothing is written) and whether to delete free blocks with a zero value.

After setting the options and pressing F5, a status message and a confirmation prompt appear, before Partimage restores the partition. This terminates the program or shuts down the system depending on the settings (Figure 7).

qt-fsarchiver

The GUI for the FSArchiver [15] command-line program, qt-fsarchiver [4], groups the many FSArchiver parameters into an intuitive interface. This makes creating images and saving them child's play.

Currently, the software is only available in repositories of the Russian ALT Linux distribution, but there is also a Ubuntu PPA. On the project website, you will also find deb and RPM packages for recent distributions, and there is a live system based on Ubuntu 18.04 LTS [16].

Qt-fsarchiver backs up single partitions, as well as complete disks. Support is available for the ext2/3/4, JFS, XFS, Btrfs, VFAT, and NTFS filesystems. In addition, the software can archive a mass storage medium's MBR or GPT, including current NVMe SSDs. Integrated data compression saves space on the target disk. Among other methods, the software supports bzip2, gzip, qzip, and LZMA.

The program comes with a clear user interface that allows for immediate production use without training. Below a menu and buttonbar, qt-fsarchiver lists all the partitions available on the system on the right hand side, specifying the filesystem and the size in a table.

Below this, the currently mounted system's directory tree appears. You can specify where you want the software to store the image. The actual configuration options for creating and restoring images are available in the window on the left. First select a radio button to specify whether you want to create or restore an image. Below this, you will find some options for compression and encryption, as well as for potentially splitting the image into several files. In addition, you need to enter the image's name in an empty text field (Figure 8).

Pressing the *Save partition* button creates the image. Statistical information at the bottom of the dialog window lets you follow the process' progress.

A separate dialog is available for cloning a mass storage device (Figure 9). To reach the dialog, select *Actions* | *Clone Disk, Create, and Restore Image*. In a new window, choose your course of action by selecting a radio button in the *Planned action* area. Options include creating or restoring a hard disk or a partition's image. As a further option, you can elect to clone the hard disk.

Depending on your selection, different target and source media appear in the center part of the window. If you create a partition image, an additional directory tree appears to let you specify the target directory where you want to save the image. When you clone a hard disk, a second, preferably similar data carrier should be used as the target drive on the system.

On the dialog box's right side, choose a button to trigger an action or end a process. In the lower window segment, a progress indicator with absolute numbers and a progress bar indicate the task's progress.

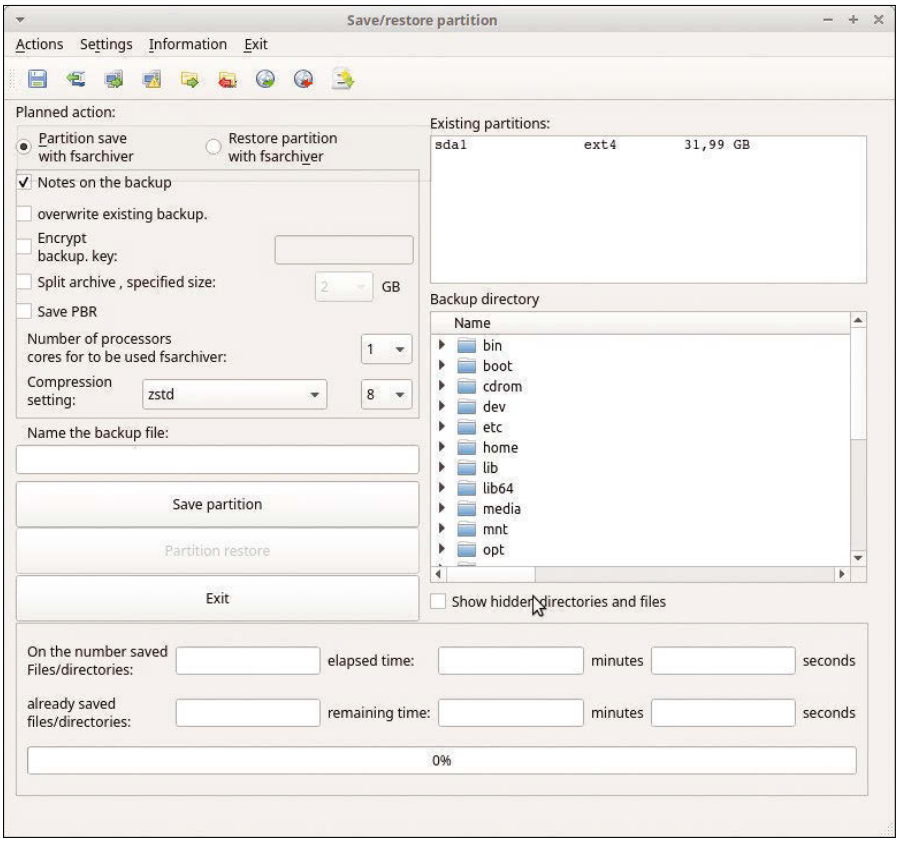


Figure 8: Qt-fsarchiver comes with a very clear user interface.

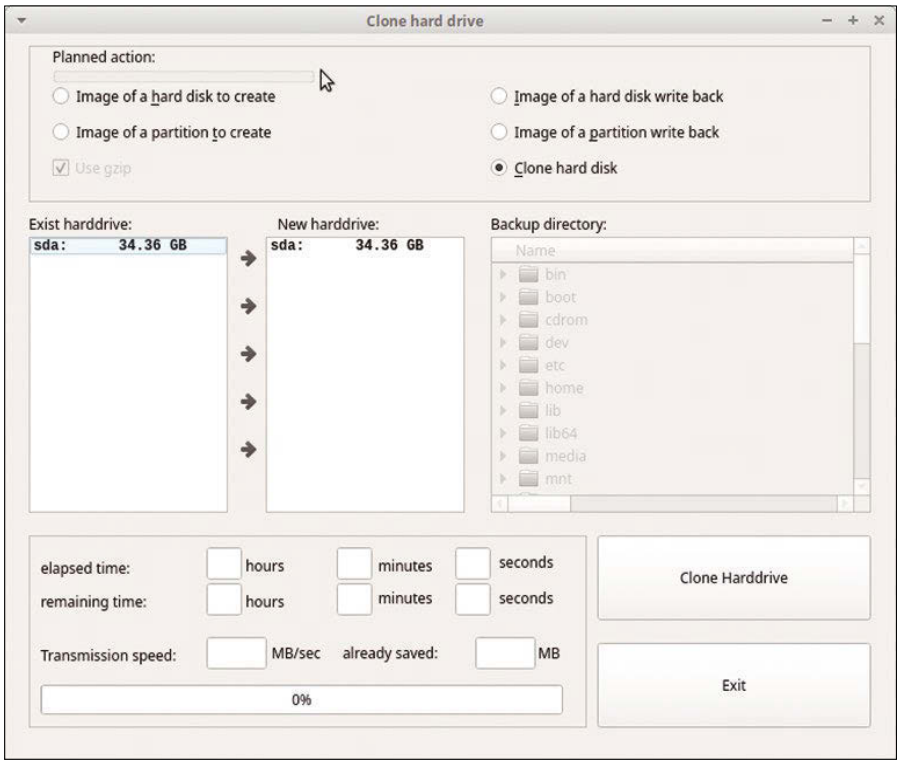


Figure 9: Both the cloning and image creation dialogs are intuitive.

Qt-fsarchiver also stores images on the network. Select *Save partitions on the network* in the Actions menu. This procedure displays a dialog if you open an SSH connection to a server. In another dialog, you then specify the partition to be backed up and the desired target directory. Then start the backup.

In the same way, you can restore images available on the network by selecting the *Restore a partition from the network* option in the Actions menu.

Snapper

Snapper [5] creates snapshots at the command line. The software is available for 32- and 64-bit hardware. Precompiled binary packages are available for numerous distributions [17], including Centos, Debian, Fedora, openSUSE, RHEL, Suse Enterprise Linux, Scientific Linux, and Ubuntu. In addition, there are software repositories that let you easily integrate Snapper into the package sources of various distributions.

Debian and Ubuntu users can install a GUI for Snapper [18] using the commands in Listing 1. On openSUSE, Snapper is available with a GUI in YaST 2 by default.

Snapper, which was created by the SUSE developers, orientates its functionality on SUSE's products. Not surprisingly, Snapper offers an optional module for SUSE's configuration interface YaST 2 and supports the Zypper package manager, both of which are used exclusively in SUSE Linux Enterprise and openSUSE.

Additionally, there is a module for the GRUB 2 bootloader that generates several boot menu entries optimized for Snapper. Snapper relies on the Btrfs filesystem, which is the standard operating system in openSUSE.

Snapper cannot be used reliably with other filesystems, which is why I would strongly advise against using it with the popular ext3 or ext4 filesystems.

The tool not only creates and configures snapshots and reconstructs them on demand, but it also compares them against each other and modifies them. Note that the YaST 2 module, which helps to compare and modify snapshots, cannot modify the Snapper configuration. To modify the configuration, you'll need a terminal.

It is important to configure the software before first use so that it generates the appropriate snapshots. Otherwise, Snapper always creates a snapshot of the main directory in the Btrfs filesystem, and ignores all other sub-volumes. Even the user's home partition with the XFS filesystem preinstalled with openSUSE is not included in the snapshot.

Sub-volumes such as the home partition can be manually added to a Snapper snapshot at any time. However, the sub-volume must use Btrfs. In openSUSE, ext3 and ext4 partitions can be converted to Btrfs partitions, but not the default XFS partition.

By default, Snapper stores a configuration file for each sub-volume in the `/etc/snapper/configs/` directory. These files can be freely edited. You will want to adjust these settings on desktop systems, since the snapshot software creates an image whenever you use YaST and Zypper by default. In addition, you can define intervals at which Snapper automatically generates snapshots.

If used frequently, the software quickly takes up a large amount of space on your mass storage. Since openSUSE creates a 30GB root partition during the standard installation, you need to grow this partition to at least 200GB when using Snapper to avoid capacity problems.

You can configure the root directory in the `root` file, which you will find in the `/etc/snapper/configs/` directory. In the `limits for timeline cleanup` section, define how many snapshots you want to keep. The prerequisite for this setting is that the `create hourly snapshots` section contains a `TIMELINE _CREATE="yes"` option.

For desktop systems, it is recommended to reset the exaggerated default values to a maximum of two to three snapshots in the `TIMELINE_LIMIT_HOURLY` and `TIMELINE_LIMIT_DAILY` options. You will also want to reduce the values for the `TIMELINE_LIMIT_MONTHLY` and `TIMELINE_LIMIT_YEARLY` options to a low single-digit number (Figure 10).

Listing 1: Installing Snapper on Debian and Ubuntu

```
sudo apt install python3 libgtksourceview-3.0-1 python3 python3-dbus
  python3-setuptools git python3-dbus python3-setuptools git
git clone --depth=1 https://github.com/ricardomv/snapper-gui.git
sudo python3 setup.py install
snapper-gui
```

If you want to save further sub-volumes with Snapper, you again need to include them at the command line. To also include the user's home partition set up by openSUSE in Snapper, enter

```
sudo snapper -c home create-config /home
```

at the prompt. As mentioned previously, the home directory must be formatted with Btrfs for Snapper. If the home directory uses XFS, you cannot change it retroactively for integration with the snapshot program.

A configuration file lets you adjust the default settings for taking snapshots of the home sub-volume. The `sudo snapper list-configs` command checks whether Snapper has created the configuration file correctly, while `sudo snapper list` lists all snapshots saved by Snapper in the default configuration. If you want to avoid restoring certain files from snapshots, you can list them in the `/etc/snapper/filters` file.

If you prefer not to use the command line for controlling the program, simply call up the GUI interface. To open the GUI, select *Snapper* from the *Miscellaneous* group in YaST. The interface window that then appears shows the available snapshots grouped by sub-volume. If you modify the system by installing new software with the help of YaST and Zypper, for example, the list shows before and after entries. In this case, two IDs appear (Figure 11).

If you want to define a sub-volume area in which there are changes, select the corresponding snapshot pairs with the

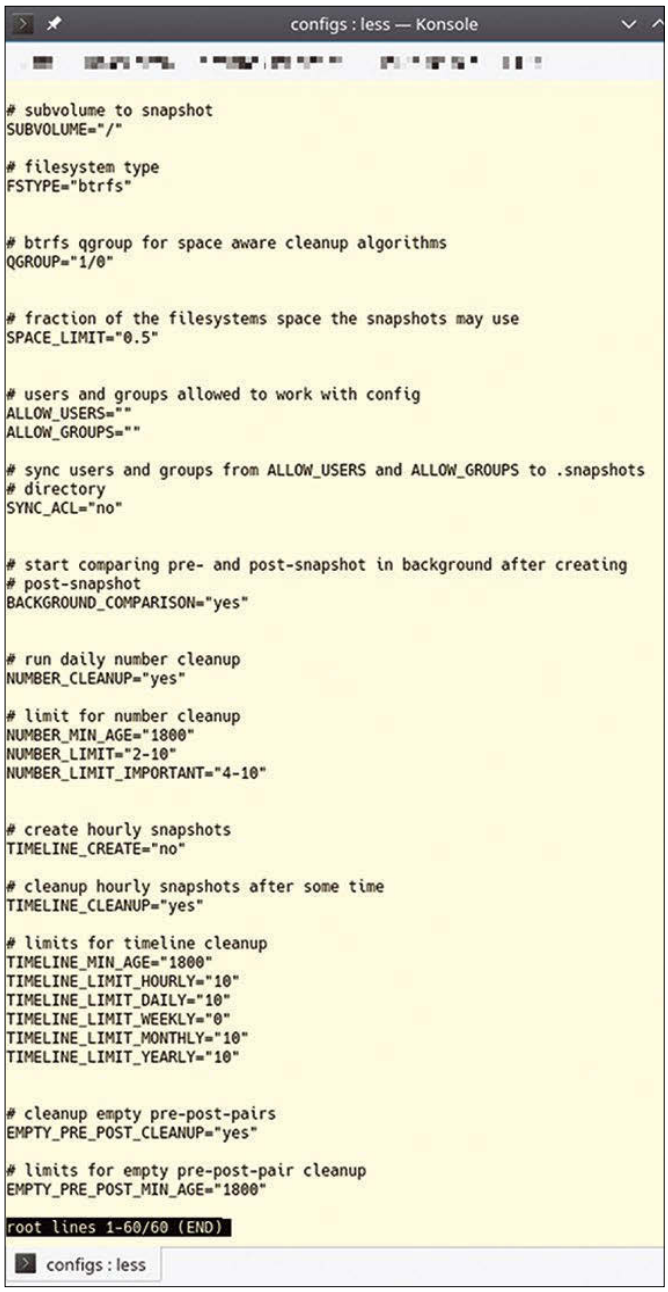


Figure 10: The configuration can be changed in the terminal if necessary.

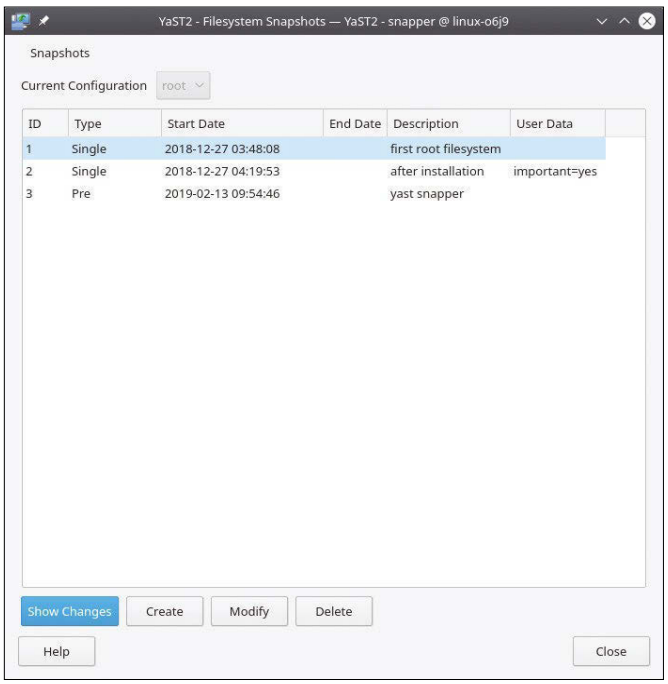


Figure 11: Snapper's GUI clearly lists the existing snapshots.

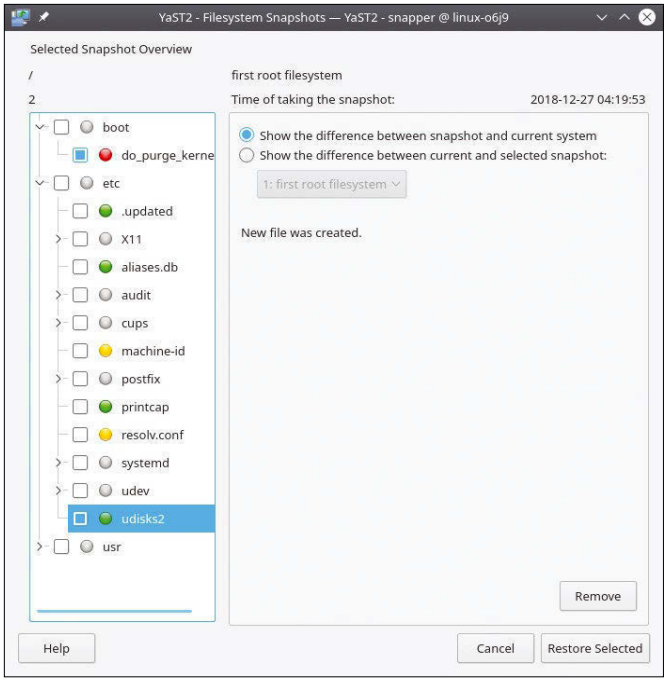


Figure 12: Snapper can also compare snapshots.

Table 1: Cloning Software

	Clonezilla	CYA	Partimage	qt-fsarchiver	Snapper
License	GNU GPL	BSD	GNU GPL	GNU GPL	GNU GPL
Supported filesystems	ext2/3/4, XFS, JFS, Btrfs, HFS+, FAT16/32, NTFS, UFS	ext2/3/4, XFS, UFS, ZFS, Btrfs	ext2/3, XFS, ReiserFS, JFS, UFS, HPFS, NTFS, FAT16/32	ext2/3/4, XFS, JFS, FAT16/32, NTFS, Btrfs	Btrfs
Profiles	No	Yes	No	No	Yes
Archiving	No	Yes	No	No	Yes
Live system	Yes	No	Yes (SystemRescueCD)	Yes	No
Scheduled	No	Yes	No	No	Yes
Intranet connection	Yes	No	Yes	Yes	No
Image comparison	No	No	No	No	Yes

mouse and click on *Show Changes*. A tree view on the left side lists the directories and files in question. On the right side of the window, there are also radio buttons for selecting changes, as well as the creation date and time when the two snapshots were created (Figure 12).

To manually create a snapshot, select the desired sub-volume from the list at the top of the interface's main window. After clicking on *Create*, enter the desired data for the snapshot in another window. You need to pay attention to whether you are creating a one-off snapshot or before-and-after snapshots. In the case of before-and-after snapshots, you need to link the new snapshot with an existing one.

If there are problems with the current sub-volumes requiring you to restore a snapshot, first switch to the overview page. The overview page shows the changes made to the selected snapshots. You can click on *Restore Selected* to restore the previous snapshot. In a separate confirmation window, the list of changed files appears (Figure 13).

Conclusions

The software solutions presented in this article are designed to create images and clones of data carriers – all of these tasks are performed reliably and without errors. (See Table 1 for a side-by-side comparison of the tools described in this article.) Differences arise in the application scenarios.

If you only want to create images under openSUSE – or another system with the Btrfs filesystem – it is best to use Snapper as a fast solution. For servers that lack a GUI, Partimage and CYA are better suited. However, due to the lack of filesystem support for ext4 and Btrfs, Partimage is only of limited use in Linux-only environments.

Qt-fsarchiver is the best fit for systems with a GUI. Thanks to the optionally available Live DVD, it can also act as a first-class alternative to the visually antiquated Clonezilla in heterogeneous IT infrastructures. Due to the intuitive user interface, qt-fsarchiver hardly requires any training. Thus, nothing stands in the way of an enterprise backup strategy that also includes images and cloned data media. ■■■

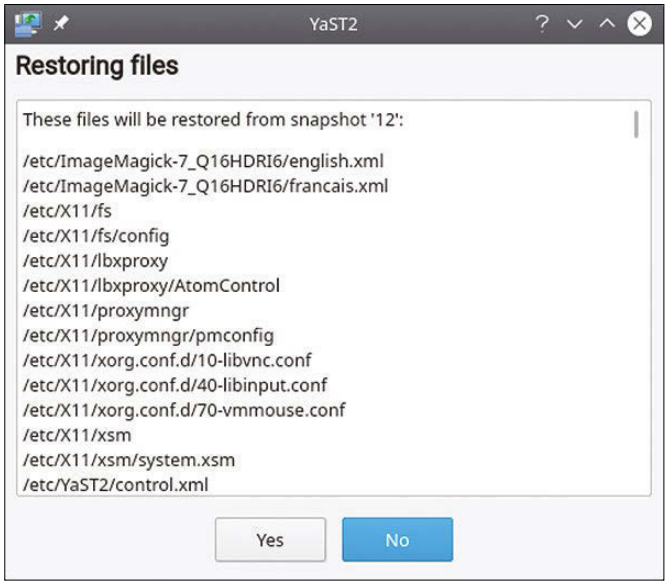


Figure 13: Snapper lists the files to be restored separately.

Info

[1] Clonezilla: *http://www.clonezilla.org*
[2] CYA: *https://www.cyberws.com/bash/cya/*
[3] Partimage: *http://www.partimage.org*
[4] qt-fsarchiver: *https://sourceforge.net/projects/qt-fsarchiver/*
[5] Snapper: *http://snapper.io*
[6] doClone: *http://doclone.nongnu.org*
[7] Duplicacy: *https://duplicacy.com*
[8] rsnapshot: *https://rsnapshot.org*
[9] GPT and operating system support: *http://www.rodsbooks.com/gdisk/*
[10] PAE: *https://de.wikipedia.org/wiki/Physical_Address_Extension*
[11] Clonezilla download: *http://www.clonezilla.org/downloads.php*
[12] rsync: *https://rsync.samba.org*
[13] CYA documentation: *https://github.com/cleverwise/cya*
[14] SystemRescueCD: *http://www.system-rescue-cd.org*
[15] FSArchiver: *http://www.fsarchiver.org*
[16] qt-fsarchiver download: *https://sourceforge.net/projects/qt-fsarchiver/files/*
[17] Snapper download: *https://software.opensuse.org/download/package?project=filesystems:snapper&package=snapper*
[18] Snapper GUI: *https://github.com/ricardomv/snapper-gui*

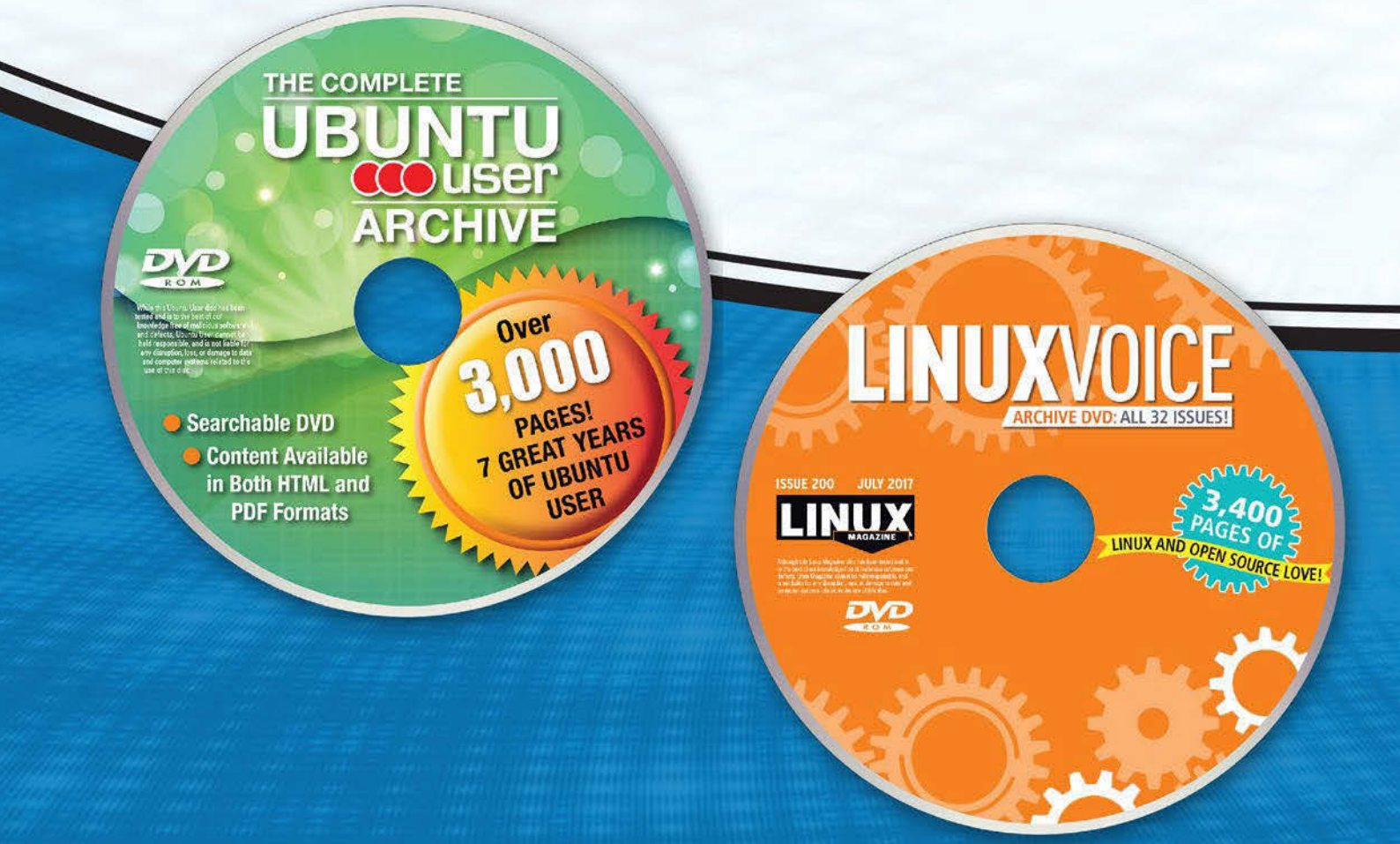
Complete Your Open Source Library with Archive DVDs!

Each fully-searchable archive DVD includes past articles so you can find the content you need quickly.

Save hundreds off the print and digital copy rate with a convenient archive DVD!

Order Your DVD Now!

shop.linuxnewmedia.com



The best Plasma ever

Red Hot

KDE Plasma 5.14 brings some highly visible improvements for KDE users. *By Ferdinand Thommes*

The two big Linux desktop environments are moving further apart. Gnome stands for simplicity, hiding many of its functions and making other functions accessible only through extensions. Increasingly, functionality is being removed from Gnome, such as lately the ability to place icons for directories or apps on the desktop itself. The Gnome desktop's operating concept requires that the user adapt to the software instead of the other way around.

Moving in the opposite direction, KDE developers are deliberately seeking to expand the desktop's capabilities. Recent releases have seen many continuous, consistent, and meaningful improvements to the Plasma desktop.

In Version 5, the KDE project modularized the KDE Software Compilation (KDE SC) [1] and divided it into three parts. This approach allows more independent development and publication of the individual parts, which allows for fast delivery of new functions.

KDE Frameworks 5.52 contains over 70 libraries, which are based on the Qt GUI toolkit [2] currently in version 5.11, and form the basis for the other two components. KDE Applications groups the applications that belong to the KDE core software [3]. The current version 18.08.2 includes applications such as Kate, Konsole, Gwenview, Dolphin, and Okular. This review puts the focus on the third component: the KDE Plasma 5.14 desktop, which was released in early October 2018.

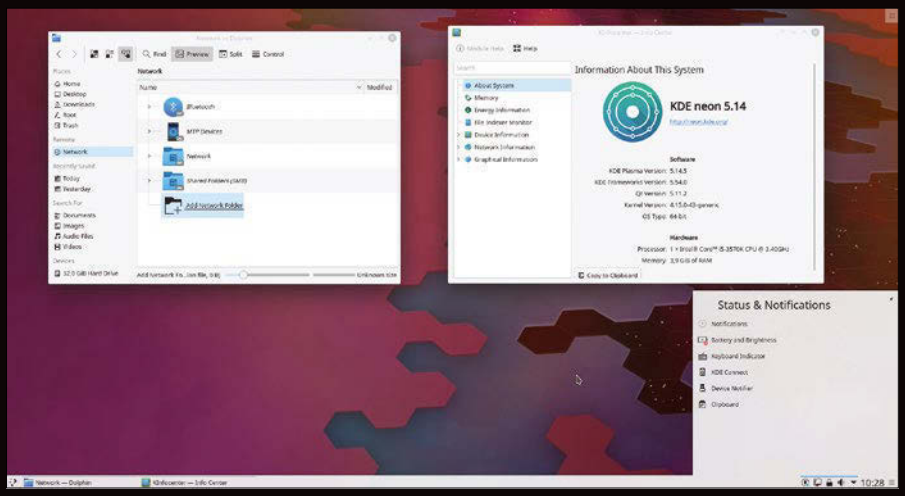


Figure 1: KDE neon constantly offers the latest developments. The version counter is currently at Plasma 5.14.5.

All Good Things

Plasma's development model provides for three releases per year, with each release seeing five minor versions for bug fixes. As of this issue's editorial deadline, Plasma 5.14.5 was the latest; when you read this article, version 5.15.2 will be the current version. In addition, versions with long-term support (LTS) have been introduced with Plasma 5.8; they offer up to 18 months of support and a correspondingly higher number of minor versions. Currently, Plasma 5.12 has LTS status, the next LTS version is not yet fixed.

In order to use a Plasma desktop with the latest KDE developments, we installed KDE neon User Edition (Figure 1), with a download size of 1.7GB [4]. Right after the installation, we checked how much main memory a KDE desktop currently needs. By splitting the libraries, the KDE desktop's fifth generation uses resources more sparingly than its predecessors (Figure 2).

Economical

While Gnome 3.30 on the current Fedora 29

Lead Image © Gino Santa Maria, Fotolia.com

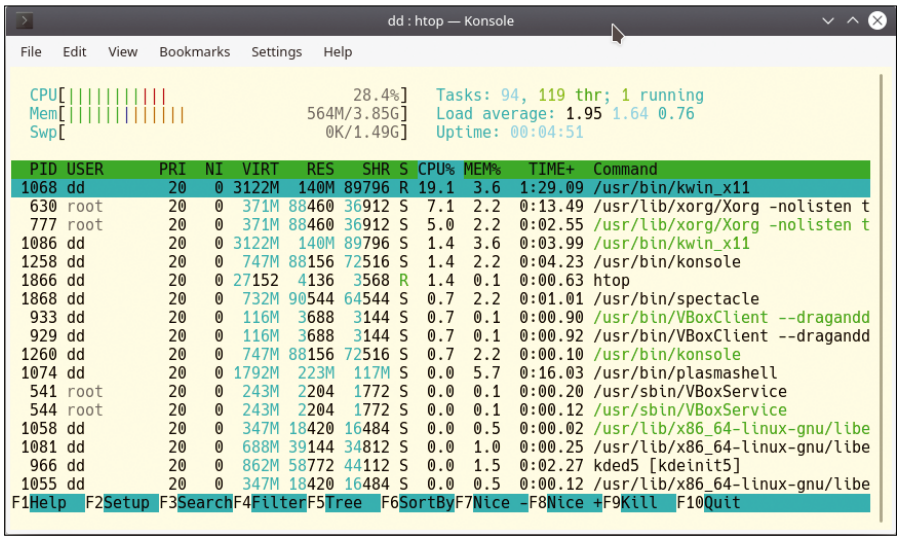


Figure 2: The fifth generation Plasma desktop uses a computer's resources far more sparingly than before due to increasing modularization.

grabs about 620MB RAM immediately after start-up, Plasma 5.14 is content with 446MB. For comparison: Xfce occupies about 350MB main memory under Siduction. Gnome and Cinnamon are therefore resource-hungry desktops, while Plasma and Xfce are in the midrange. Only LXQt and LXDE get by with even less RAM.

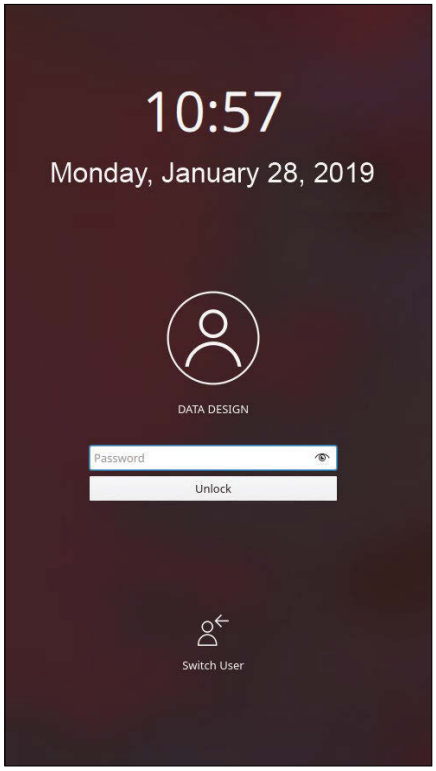


Figure 3: Changing user accounts is faster thanks to the *Switch user* button in Plasma 5.14; logging out the previous user is no longer required.

Shortly after installing Plasma 5.14, we noticed a first, very useful change for multiuser systems: The lock screen now lets you directly change the user where you had to log out in past versions (Figure 3). If you try to shut the system down and other users are logged on, a warning is output. These changes improve comfort and increase safety at the same time.

Discovering Discover

In 2018, one important application received a huge amount of attention across the different Plasma versions: Discover. The graphical software manager had a life of its own as Muon Discover before it assumed a leading role on the Plasma desktop.

In early 2018, Discover was still slow and buggy: it froze at run time, showed inconsistent results, and provided illogical user guidance. In addition, it offered no advantage over the command line. Since then, Discover has matured with every issue and is now at least on a par with its counterpart Gnome Software.

The developers not only perfected Discover visually, but they also extended it with essential functions. Today, Discover not only installs and updates applications, but also handles Plasma extensions known as plasmoids or widgets.

In Plasma 5.14, Discover has learned how to update the computer's firmware. In the background, the application uses Linux Vendor Firmware Service (LVFS) [5] developed by Fedora. Participating manufacturers enter their firmware updates into this database, which the system then detects and runs updates if required.

Whether it's updating Logitech's Unifying Receiver for wireless devices or UEFI on a Dell laptop, Discover informs Plasma 5.14 users about upcoming updates and can perform updates on demand (Figure 4). Until now, Plasma

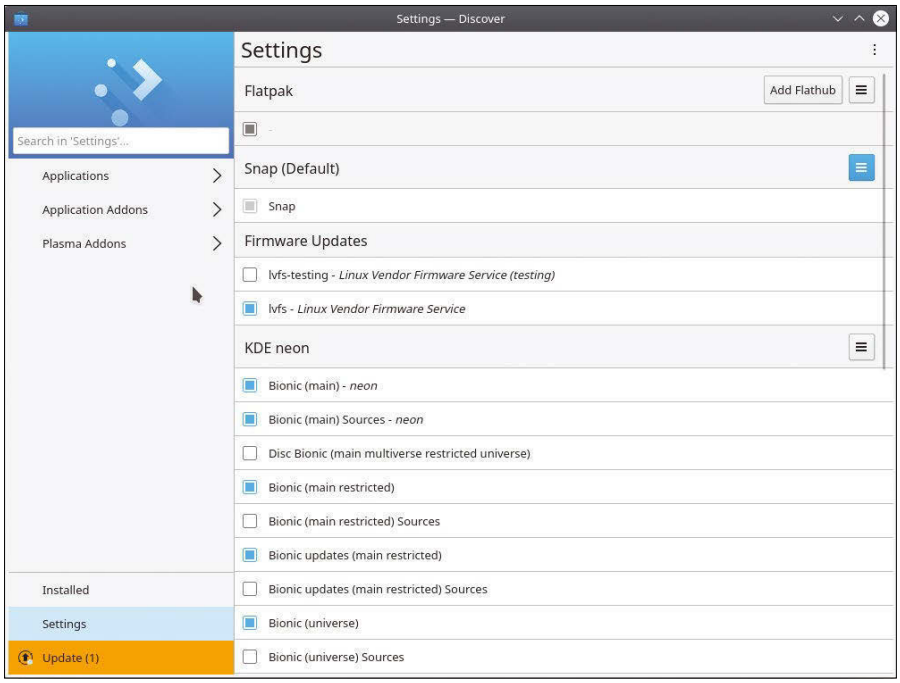


Figure 4: Discover now supports LVFS, which can be used to update the computer's or accessories' firmware.

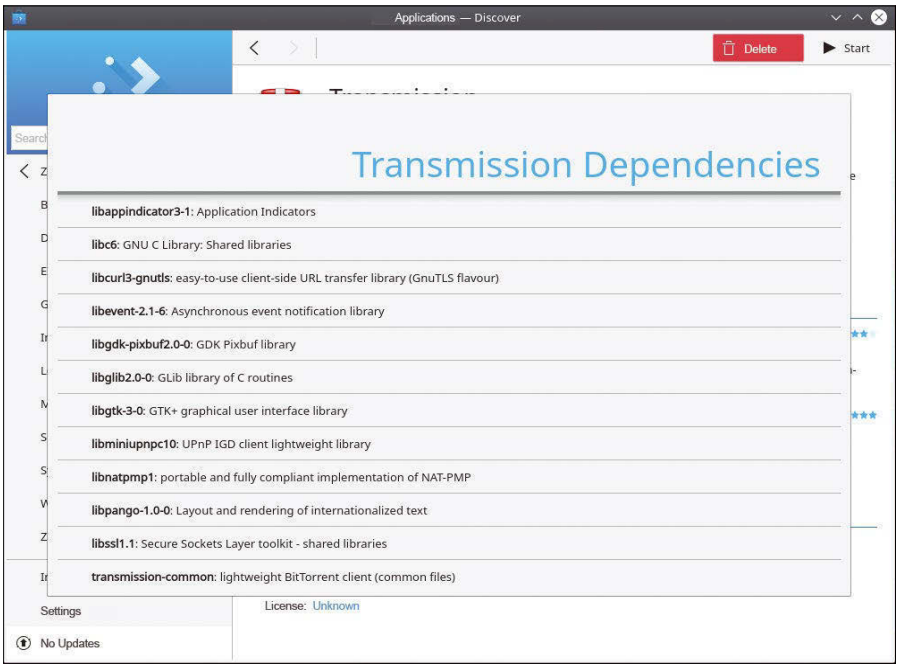


Figure 5: Clicking on *Show Dependencies* reveals the selected package's dependencies.

users had to use the `fwupdmgr` command in Konsole for updates.

Package Formats

The alternative package distribution systems Snap, Flatpak, and AppImage, which have been much discussed in recent times, also benefit from in-depth support in Discover.

The tool for Canonical's Snap format now supports Snap channels, which let you switch between different versions of an application installed as a Snap. If, for example, the developer offers alpha or beta versions in addition to the stable release, Discover now lets you seamlessly switch between the variants.

With Flatpack, Discover now notifies you when setting up Flatpak if the associated back end is missing and offers to install it. The handling of AppImages has been simplified, as thumbnails for this package format now automatically appear in the Dolphin file manager. For DEB packages, Discover now notifies you if the dependencies for a package change, packages are replaced, or they simply no longer exist.

You can find out about a package's dependencies by clicking on the application in Discover and selecting *Show Dependencies* (Figure 5). One of the new features is that applications can be sorted by publication date.

Wayland Making Progress

While the integration of the designated X11 successor, Wayland, has not yet progressed as far as at Gnome, the Plasma camp has shown some amazing progress in recent months. For example, in Plasma 5.12 LTS or 5.13, YouTube videos would not run, or support was very poor, using the new display protocol in our lab. In version 15.4.4, this now works smoothly.

In addition, Plasma 5.14 now supports the two Wayland protocols, XDG-Shell and XDG-Output. They provide for better application integration into the desktop and help handle multiple displays under Wayland. The developers also fixed problems with the clipboard between GTK + and Qt apps. Many effects of the KWin window manager were revised to make animations smoother under Wayland.

Change Displays

A new Display Configuration widget makes it easier to handle changes in multi-display environments. You can now quickly make changes to the displays' arrangement (including projectors) at the push of a button, without having to open System Settings.

In addition, you can prevent a computer or projector from going to sleep abruptly by selecting *Enable Presentation Mode*. The *Advanced Display Settings* button takes you to the System Settings display section (Figure 6) for further adjustments.

Importing Vaults

The Audio Volume widget now offers a speaker test that was previously only available in the Multimedia section of System Settings. The Network widget can now be used to set up SSH VPN tunnels again.

At directory and file level, the Plasma Vaults widget is supposed to allow the import of existing vaults with version 5.14; however, this option is currently unavailable in the GUI. To use this option, enter the command from Listing 1 as root at the command line. A graphical wizard will then guide you through the import (Fig-

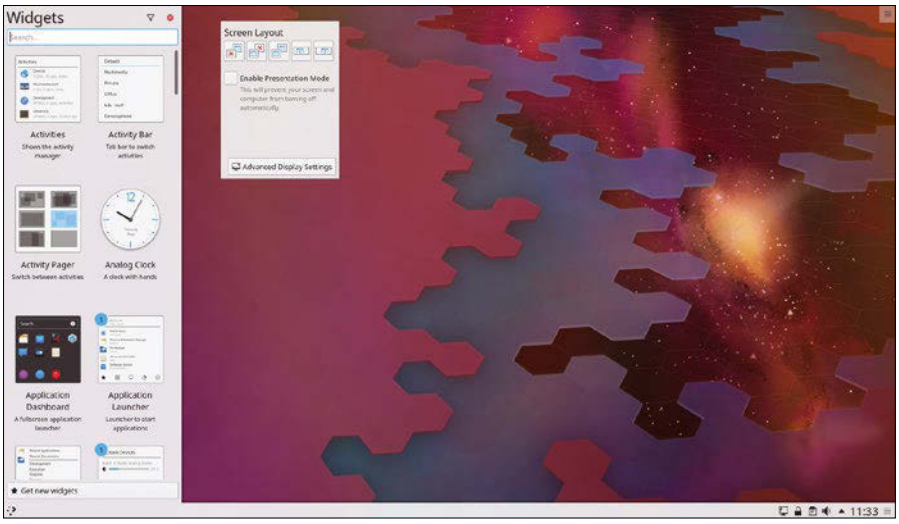


Figure 6: This widget helps you configure multi-display environments. Presentation mode prevents the screen from darkening in the middle of a presentation.

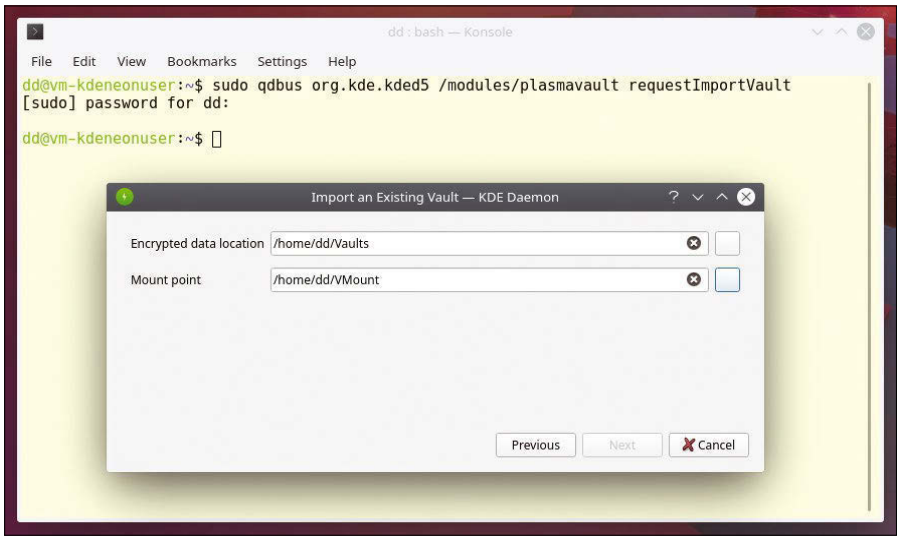


Figure 7: In the Vaults GUI, a button for importing existing vaults on other computers was obviously forgotten. However, the function is available via a D-Bus string (Listing 1).

ure 7). In addition, Plasma Vaults now supports an offline mode for handling particularly critical data.

A controversial change was reversed in KDE Applications 18.08: The former KWin maintainer Martin Flöser removed the option to start Dolphin, Kate, and other applications as root, because he thought it was too insecure. Users promptly launched protests against this deactivation. With Plasma 5.14, KDE has reverted the change; you can now run KDE apps with administrative rights again.

Listing 1: Importing an Existing Vault

```
# qdbus org.kde.kded5 /modules/plasmavault requestImportVault
```

Kill Window

The System Monitor (aka KSysGuard) has been given a new menu with the latest updates; they can be used to launch further applications.

Under the Tools drop-down menu, you will find – depending on which applications are installed on the system – a selection of related tools. On our test system, these were Filelight, Sweeper, KMag, and Htop in addition to the ubiquitous Konsole. The *Kill a Window* option lets you kill applications that are no longer responding (Figure 8).

The Spectacle screenshot tool was also upgraded. It has been given a more logical layout, which is now closer to its predecessor KSnapshot. Thus, the *Save as* function, which was previously hidden

Shop the Shop ⇒ shop.linuxnewmedia.com

Discover the past and invest in a new year of IT solutions at Linux New Media's online store.

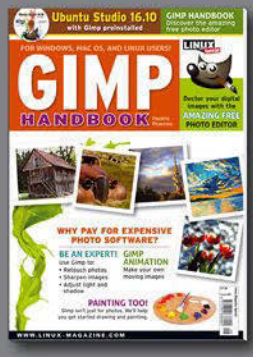
Want to subscribe?

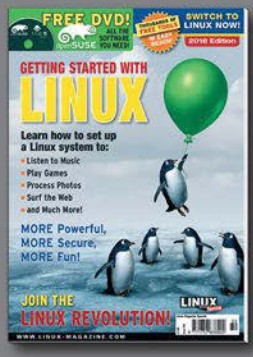
Searching for that back issue you really wish you'd picked up at the newsstand?

shop.linuxnewmedia.com

DIGITAL & PRINT SUBSCRIPTIONS

SPECIAL EDITIONS





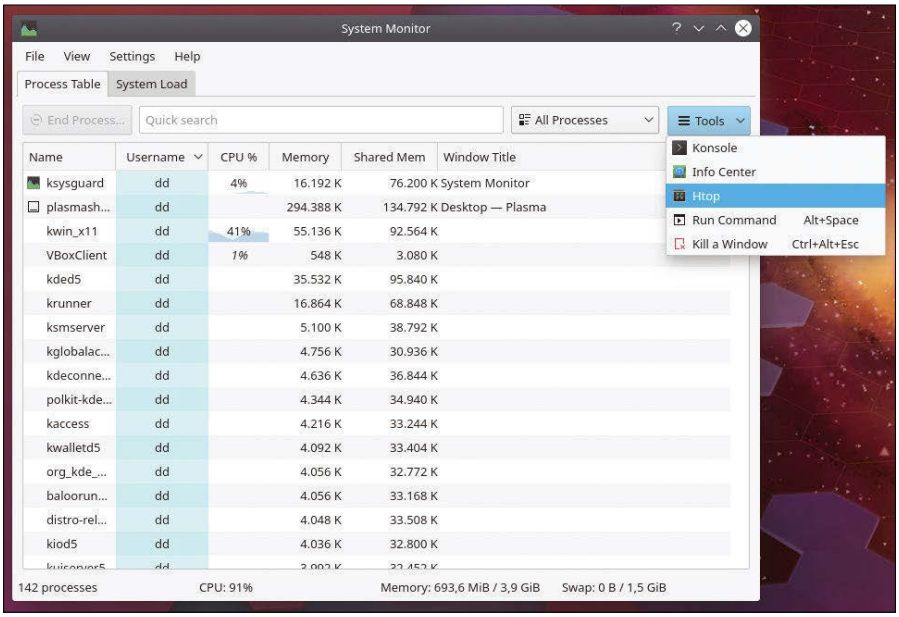


Figure 8: Thanks to a new menu, the System Monitor now serves as a starting point for further actions based on the displayed data.

in a submenu, has been given its own button. A new *Window Under Cursor* capture mode is now available.

The *Global Menu* (introduced with Plasma 5.9), which you activate via a widget, now can handle GTK+ applications. If you switch from the Kickoff menu to the full-page application overview, you can now view widgets in addition to applications and documents. This view sorts the widgets into categories, providing a good overview (Figure 9).

Conclusions

Plasma 5.14 runs smoothly, responds quickly to user actions, and saves resources. Overall, the new version is a huge leap in the right direction. However, there is still work to be done in terms of look and functionality. Some windows are too small when they open up in the default setting. For example, the next Plasma version will stretch the KInfoCenter window so that no scrollbars appear.

If you want to use the current Plasma version, but have an LTS distribution that only contains an older Plasma version, Kubuntu offers a Personal Package Archive (PPA). Distributions such as Arch Linux, Manjaro, KDE neon (used here), and KaOS, on the other hand, closely follow KDE development and offer Live images of the Plasma desktop.

To keep up with the latest improvements, follow Nate Graham's blog *Adventures in Linux and KDE* [6], where Graham, a KDE developer, reveals his co-workers' latest developments every weekend. For example, he has already announced support for the WireGuard VPN client in the Network Manager applet for Plasma 5.15 which was released in mid-February. ■■■

Info

[1] KDE SC: *https://en.wikipedia.org/wiki/KDE_Software_Compilation*
[2] Qt: *https://en.wikipedia.org/wiki/Qt_(Library)*
[3] KDE Applications: *https://en.wikipedia.org/wiki/KDE_Applications*
[4] KDE neon: *https://files.kde.org/neon/images/neon-useredition/current/*
[5] LVFS: *https://fwupd.org/*
[6] Nate Graham's blog: *https://pointieststick.wordpress.com/*

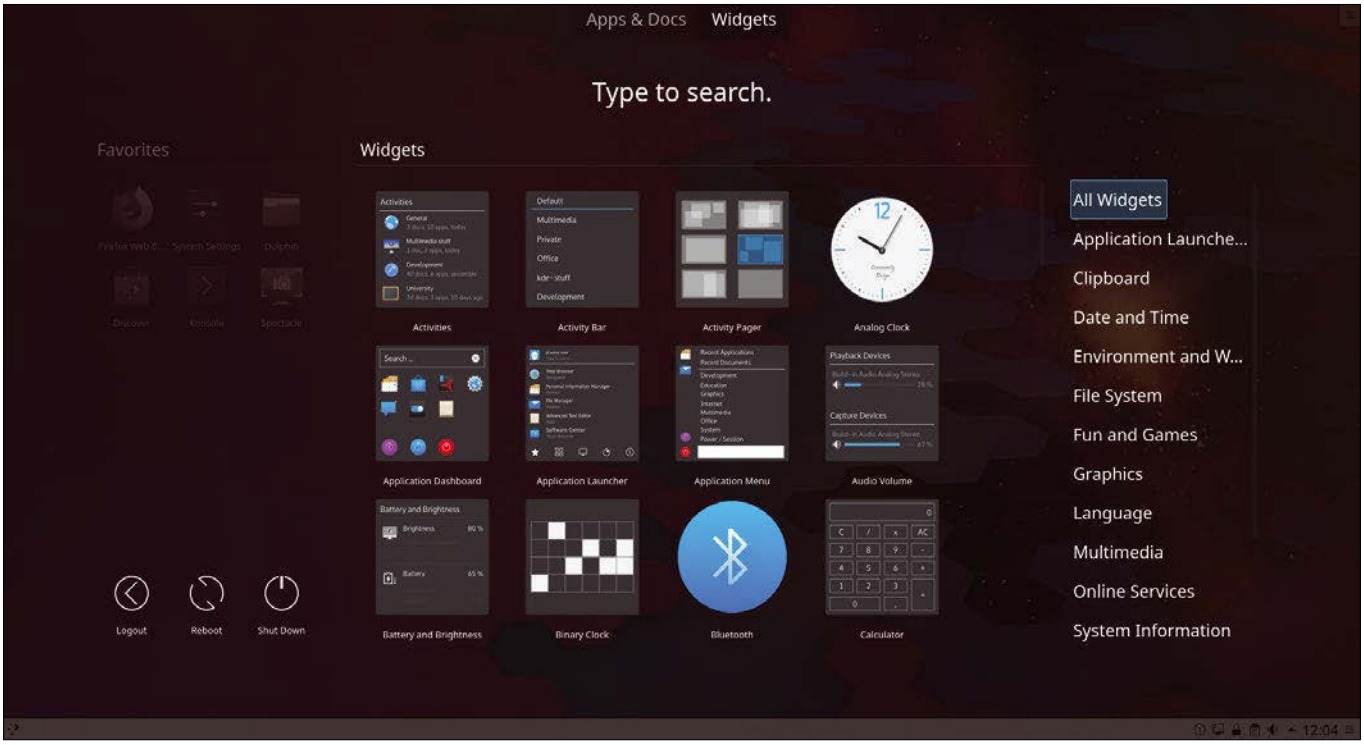


Figure 9: The full-screen application overview now also provides access to widgets.

DrupalCon
MINNEAPOLIS 2020
MAY 18-22

Minneapolis Convention Center
events.drupal.org/minneapolis2020

Solidify your plans.
Registration will open in **September of 2019**!

More than 3,000 of the top digital minds will convene in Minneapolis to share knowledge, create solutions, build relationships, and shape the future.

Network, learn and be inspired.

Register for one of the tracks below to optimize your onsite experience.

CONTENT & DIGITAL MARKETING

BUILDER

AGENCY LEADERSHIP

Redundant Internet connections on consumer hardware with OpenWrt and Mwan3

Double Track

Redundant Internet uplinks are standard on big corporate networks, but they are still the exception for the home and small-office sector. If you're ready to experiment, you can set up redundant Internet connections on your own home router with OpenWrt and the powerful Mwan3 extension.

By Martin Loschwitz

A router with two Internet connections provides many advantages, including better network performance and continued operation if one link fails. Redundant Internet connections have long been standard for data centers and enterprise networks, but they are relatively rare in homes and small offices.

One reason for the absence of redundant connections on small networks is cost, but even when cost is not an issue, home users and small-time admins are often intimidated by the technical issues of supporting two Internet connections through the same inexpensive router.

The ever-resourceful open source universe, however, does provide an inexpensive solution for running redundant Internet connections through an inexpensive home router. The solution begins with OpenWrt, a Linux distribution designed to run on routers and other embedded devices. If you are adventurous enough to install OpenWrt on your router device, you will discover you have much more control over the device than you ever had with the standard router firmware interface. In particular, the Mwan3 extension available for OpenWrt provides support for redundant connections. Mwan3 dynamically manages access via two or more uplinks, uses Ping to ensure that the links are working and, when necessary, adds NAT support and dynamic IPtables rules.

This article describes how to set up a redundant connection using OpenWrt [1] and Mwan3. Of course, if you try this configuration yourself, you'll need to subscribe to multiple Internet connections. OpenWrt supports DSL and PPPoE out the box; if you want to use LTE, the easiest way is through an OpenWrt-supported USB modem. OpenWrt can manage most legacy devices without any problems.

What Hardware?

At least as important as choosing the appropriate Internet access method is choosing the hardware. With OpenWrt as a router, no other router is allowed to block the line. If you already have DSL or a cable gateway, it is best to check the manual to see whether the device supports bridge mode. Bridge mode means that the router demotes itself to a simple modem and passes the incoming network signal on to another device – in this case, the OpenWrt-based router.

Many routers offer a bridge mode, but the way the user activates it differs. With cable providers, you can usually enable bridge mode via a web interface in the customer center. With DSL, you might find the option in the router's configuration interface.

If you have multiple working Internet connections and equip them with physical modems or routers downgraded to modems, you still have to worry about finding the right hardware for an OpenWrt router device (see box entitled "Hardware with More Power"). Although OpenWrt runs on a large number of recent devices, the quality of the OpenWrt implementation differs greatly for different devices.

The compatibility of the router device is also related to the manufacturers of the chipsets that are used in SOHO routers. On one hand is Qualcomm, with its Atheros series, which is one of the better-supported chips on Linux. On the other hand is industry leader Broadcom, with its various model ranges. Broadcom's Linux support varies in quality.

The OpenWrt developers recommend the Qualcomm variant, which is sometimes difficult to find. Devices by the usual suspects (Asus, Netgear, Linksys) typically use Broadcom chips. TP-Link routers rely on Qualcomm. The example described in this article will use the TP-Link AC1750 router (also known as the Archer C7), which has a street price of around EUR 90 (Figure 2).

If you prefer to search for other devices, you don't necessarily have to look for devices with multiple WAN ports. This is because the WAN ports are usually connected to the same switches in

Lead Image © sermax55, 123RF.com

Hardware with More Power

Support for Qualcomm hardware in OpenWrt is not perfect – in the majority of cases, users won't notice it, but a little background information is helpful in case you run into problems.

Modern network chipsets for LAN and WLAN come with a variety of special functions that enable performance gains. One such optimization feature is offloading, in which the device's network chip assumes tasks that would otherwise place load on the host CPU. Transferring functions such as packet filtering or connection tracking to the network chip can bring a significant performance boost.

The problem is that even well-supported Qualcomm drivers don't support every offloading function.

Snapshot versions of OpenWrt offer access to a generic offloading feature of the Linux kernel, but this feature was unstable during operation. The last stable OpenWrt version achieved 240 MBit/s in our lab – but it then reached the end of its tether and all purported tuning measures failed.

If you want to use OpenWrt and multiple uplink configurations behind high bandwidth connections – as described in this article – you might want to check out Omnia Turris [2], from the Czech domain administrators NIC.CZ (Figure 1). Omnia Turris is an OpenWrt-based router on open source hardware. The router achieves the promised GBit/s performance in the downstream without any problems. You can use Mwan3 with Omnia Turris, but the OpenWrt fork that Omnia maintains is prehistoric – it is based on OpenWrt 15.05, which was released in 2015.

Mwan3 is also very old, and some tips and configuration files described in this article cannot be used on an Omnia Turris, but the Omnia Turris solution is usually acceptable in small office environments. The Omnia Turris forum offers instructions for configuring Mwan3 with multiple WANs [3].

Figure 1: The open-hardware Omnia Turris from NIC.CZ saturates a GBit link without any problems.

the device as all other ports. It is therefore easy to convert one of the existing switch ports to a second WAN port at a later date.

Prepare Setup

The example described in this article assumes a redundant Internet uplink consisting of a DSL connection and a cable connection. Many revisions of the TP-Link AC1750 (Archer C7) router are available on the market. OpenWrt supports them all, but you need an OpenWrt image that matches the revision. It is worth visiting the Table of Hardware on the OpenWrt Wiki [4] to discover the details of the firmware image. If you are trying this procedure on a different device, the steps will be different – consult the manual for your own router.

Figure 2: Devices such as the TP-Link AC1750 are well suited as home routers for multiple uplinks.

You'll need an up-to-date version of OpenWrt to follow this article. How OpenWrt is installed on your router depends on the hardware. In the case of the Archer C7, the device must first be connected to a computer or an existing network using an RJ45 cable.

To install the firmware, set up the computer to run a TFTP server that is accessible on the IP address 192.168.0.66. The folder offered via TFTP must contain the factory image of OpenWrt 18.06.1 for the Archer C7v4; the file should be named `ArcherC7v4_tp_recovery.bin`.

Then turn off the AC1750 with the power button, hold down the reset button with a pointed object, switch the power button on again, and hold down the reset button for about 4 seconds. The LEDs on the device flicker briefly, then the OpenWrt image is downloaded via TFTP. Please note: The reset gives you factory settings on the device; if you previously configured a different IP address or special passwords, you will have to configure these customized settings again.

If your personal computer is on the same network as the AC1750, reboot, and you will receive an IP address from DHCP that will allow you to log in via SSH or the web interface without a password. The IP address for the router is 192.168.0.1. If you log in via the web interface, Luci forces the admin to set the password immediately so that an SSH login is no longer possible.

Configuration

Once you have installed OpenWrt on your router device, the next step is the basic configuration. A large part of the

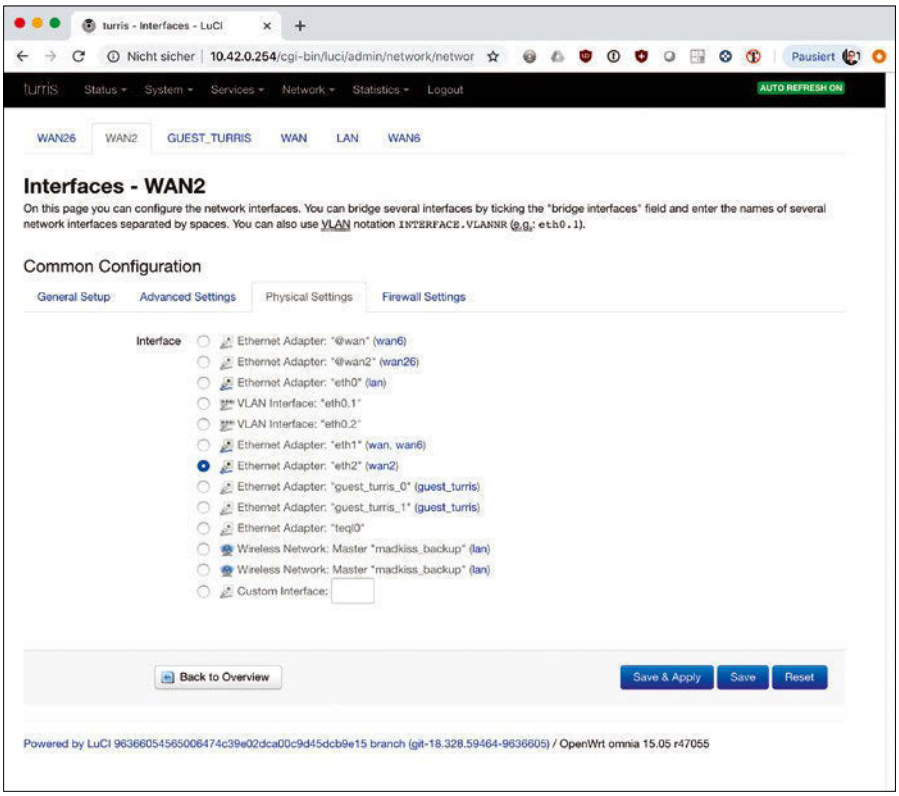


Figure 3: The additional WAN interface in OpenWrt must be connected to a physical interface (eth2 on the Omnia Turris).

configuration is the network settings. If you want a special DHCP configuration or a static IP address assigned to DHCP clients, enter the desired options in the Luci web interface under `Network | DHCP & DNS`. You can also use Luci to set up any special DNS forwarding servers. The OpenWrt router still does not have an Internet connection. The following sections describe how to set up the Internet uplink connections.

Uplink 1

The first uplink uses the router's WAN port. It makes sense to combine the WAN port with the faster uplink of the available lines. Plug one end of the RJ45 cable into the cable modem and the other end into the WAN port of the AC1750 on the other. The DSL connection delivers 65 MBit/s downstream and 20 MBit/s upstream; the coaxial cable connection is more powerful at 500/50 and thus the standard connection.

It may be necessary to restart the cable modem after connecting to the AC1750. Using Luci, log into OpenWrt and configure the `LAN` port for `DHCP`. OpenWrt automatically enables IPv6 in parallel if a genuine dual stack is available from the Internet provider.

If everything works out, the WAN connection will have an IPv4 and an IPv6 address at the end of the process. In preparation for the dual WAN setup, it makes sense to click on `Edit` for the WAN interface in Luci and to type `10` in the `Router Metric` box. OpenWrt needs this metric later to prioritize the uplinks.

Uplink 2

Uplink 2 is a bit trickier, because OpenWrt doesn't have a port for it yet. To correct this problem, click on `Network | Switch` in Luci to display the configuration of the built-in switch.

In the case of the AC1750, two VLANs are configured, with ports 1 to 4 on VLAN 0 and the `WAN` port on VLAN 1. After pressing `Add`, you can add a VLAN 3. Then ensure that the port named `Port 3` is only untagged on `VLAN 3`; for all other VLANs, however, the value in the table must be `off`. Finally, click on `Save & Apply`.

The other half of the configuration for uplink 2 consists of defining an interface in OpenWrt that uses the new VLAN. If you connect port 4 of the Archer C7 to your other uplink – such as the DSL modem, an Ethernet connection is established. But PPPoE is still missing for logging in on the provider side and thus establishing the connection.

Select `Network | Interfaces. Add New Interface` calls the required dialog. In `Protocol of the new interface`, select PPPoE or another suitable protocol. When it comes to selecting the interface, `eth0.3` is the right choice for the VLAN 3 you just created. You have now created a logical connection between the VLAN and a PPPoE connection on the router. The name for the interface is `WAN2` (Figure 3).

Next, select `Network | Interfaces` and choose `WAN2 Edit` in the interface. In the following dialog, define the PPPoE connection parameters. Important: A routing metric must also be defined for `WAN2`, say, `20` in this example.

After you save the settings, OpenWrt establishes the connection automatically.

Setting Up Mwan3

Now that the connections are configured, the next step is to configure Mwan3 to switch between the two active connections. All in all, this arrangement works very well for IPv4, but IPv6 can cause complications (see the box entitled "Trouble with IPv6").

To configure Mwan3 on OpenWrt, log into your router and run the `opkg update` command, which updates the OpenWrt package lists. Then run `opkg install luci-app-mwan3`, which installs both the GUI part of Mwan3 and the main Mwan3 application.

Once Mwan3 is set up, you can check its status and configuration via the web interface (Figure 4). Most admins, however, prefer editing the configuration file.

The `/etc/config/mwan3` file is the center for Mwan3 configuration. See the OpenWrt website [5] for more on editing the

Trouble with IPv6

Mwan3 uses NAT and Iptables rules to implement load balancing. For IPv4, this approach works very well, but if you use Mwan3 with IPv6, it is less effective.

The main reason Mwan3 doesn't work well with IPv6 is because it uses NAT. IPv6 has a totally different way of defining private local networks, and the support for IPv6 in Mwan3 appears to have problems with address assignment using the RADVD protocol, which is a kind of like DHCP for IPv6.

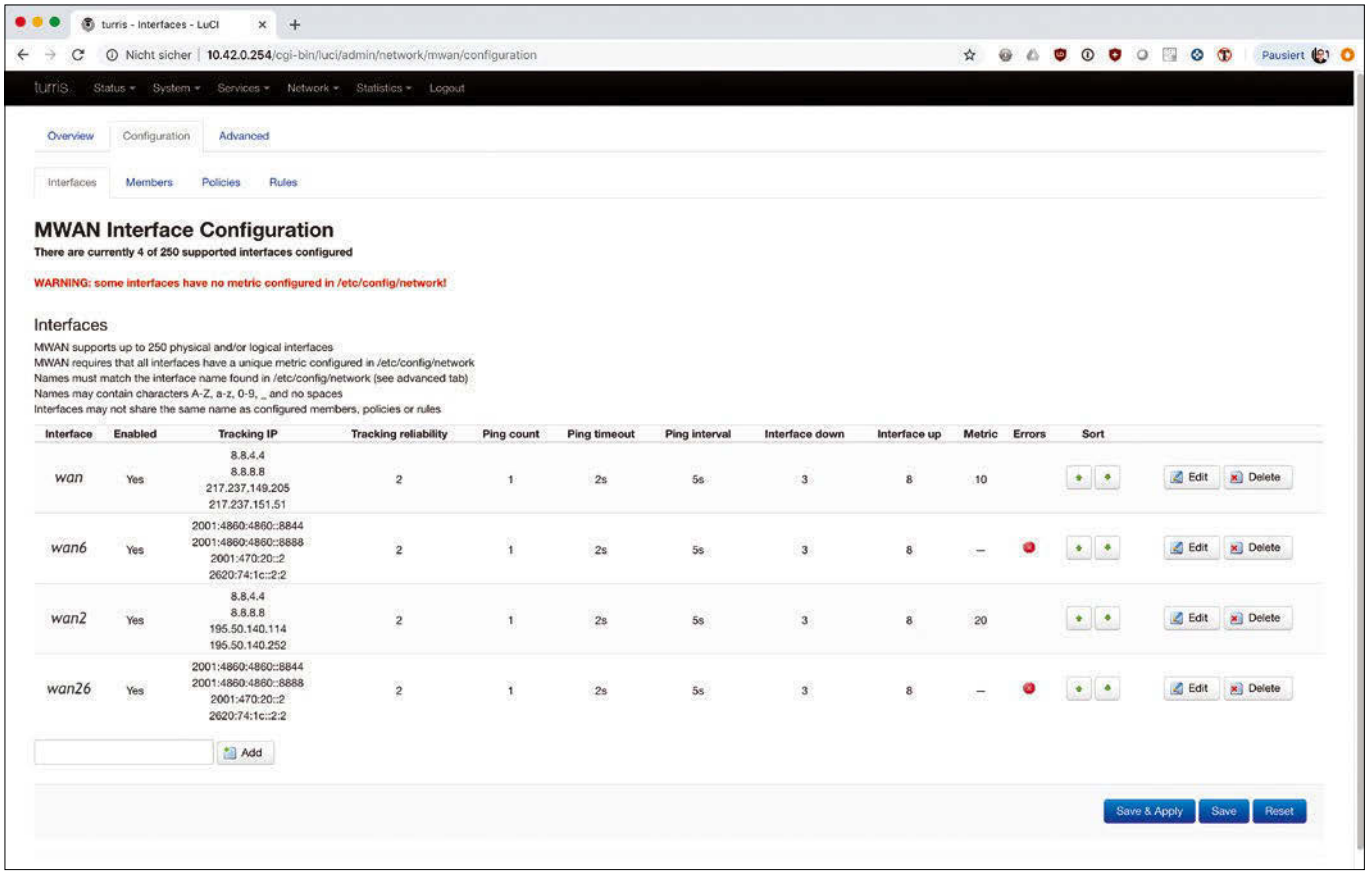


Figure 4: Once Mwan3 is configured, your can edit the configuration using the web interface.

Mwan3 configuration file. The first block in the file specifies the source address for packets generated on the router. This section is followed by four definitions for the interfaces present in the system: two for IPv4 and two for IPv6. Using the `list track_ip` directive, the admin can determine which hosts Mwan3 `ping`s to find out whether the line is online or offline. The other values are default parameters and can be adopted without changes.

```
        list use_member 'wan2_m1_w2'
        list use_member 'wan2_6_m1_w2'

config policy 'balanced'
        list use_member 'wan_m1_w3'
        list use_member 'wan2_m1_w2'
        list use_member 'wan6_m1_w3'
        list use_member 'wan2_6_m1_w2'

config policy 'wan_wan2'
        list use_member 'wan_m1_w3'
        list use_member 'wan2_m2_w2'
        list use_member 'wan6_m1_w3'
        list use_member 'wan2_6_m2_w2'

config policy 'wan2_wan'
        list use_member 'wan_m2_w3'
        list use_member 'wan2_m1_w2'
        list use_member 'wan6_m2_w3'
        list use_member 'wan2_6_m1_w2'

config rule 'https'
        option sticky '1'
        option dest_port '443'
        option proto 'tcp'
        option use_policy 'wan_wan2'

config rule 'default_rule'
        option dest_ip '0.0.0.0/0'
        option use_policy 'wan_wan2'
```

Figure 5: The rule entries in the Mwan3 configuration determine how Mwan3 routes packets.

The `member` paragraphs in the Mwan3 configuration assign metric values and weights to the NICs. However, these entries only become meaningful in the context of the following policy definitions. You need these policies to bundle several interfaces and the associated metrics and weights to create a complete package.

At the bottom are the `rule` paragraphs, which define concrete rules for certain types of traffic from certain sources or with certain destinations. The `https` rule in the example specifies that the same connection must always be used for HTTPS connections. `sticky` specifies that connections must not switch between interfaces.

See the Mwan3 documentation for more on configuring Mwan3 settings.

Load Balancing is Standard

In configurations with multiple network interfaces, Mwan3's default operating mode is to perform load balancing. If you don't want Mwan3 to perform load balancing, replace the value `balanced` with `WAN_WAN2` in the `rule` paragraphs at the end of the `/etc/config/mwan3` file (Figure 5). The `WAN_WAN2` setting means Mwan3 will only use WAN2 in cases when the first interface WAN1 is offline.

Conclusion

If you think you would benefit from the load balancing and redundancy of a dual uplink configuration, or even if you just want to experiment with new tricks on an old home router, download OpenWrt and get started with the Mwan3 extension. ■■■

Info

[1] OpenWrt: *https://openwrt.org/*

[2] Omnia Turris: *https://omnia.turris.cz/en/*

[3] Mwan3 on the Omnia Turris: *https://forum.turris.cz/t/mwan3-multiwan-lte-setup/1331*

[4] OpenWrt Table of Hardware: *https://openwrt.org/toh/start*

[5] Mwan3 Documentation: *https://openwrt.org/docs/guide-user/network/wan/multiwan/mwan3*

Simplifying and improving standard commands

New Commands for Old Purposes

As the standard core Linux commands have become more complex due to revisions, new commands have risen to take their place. Bruce looks at seven of these modern commands, plus a terminal emulator. *By Bruce Byfield*

The more basic a command is, the more likely it is to predate Linux. The long history of commands is an advantage in that core commands have had more than four decades to get the bugs out and to make revisions as the expectations and needs of Linux users have evolved. However, more options can also make mastering commands more difficult. At times, all the revisions can make learning and remembering commands absolutely cumbersome.

Rather than completely restructure basic commands, Linux programmers have often responded by writing completely new commands that are either rationalizations or simplified front ends for older commands. Being as set in my ways as most people, initially, I have often been skeptical of these new commands. Yet when I tally them up, I find that I have replaced standard commands with newer ones in a surprising number of cases. Here are seven leading examples.

Author

Bruce Byfield is a computer journalist and a freelance writer and editor specializing in free and open source software. In addition to his writing projects, he also teaches live and e-learning courses. In his spare time, Bruce writes about Northwest coast art. You can read more of his work at *http://brucebyfield.wordpress.com*

tree

Few commands are as basic as `ls`. When working from the command line, you can use it a dozen times in an hour to list the contents of a directory.

`tree` (Figure 1), which replaces `ls`, shares many of the same options, but with one important difference: `tree` displays the contents of a directory in a tree view, similar to the one in a file manager. That may sound like a minor difference, but a visualization is always welcome, especially in these desktop-oriented days.

By default, `tree` displays the current directory. However, you can also specify a top directory after the command options. Usefully, archived files are treated as directories, so their content can be viewed directly and without another command.

If you want more information, you can set `tree` to list file attributes and include hidden files. If you want less, you

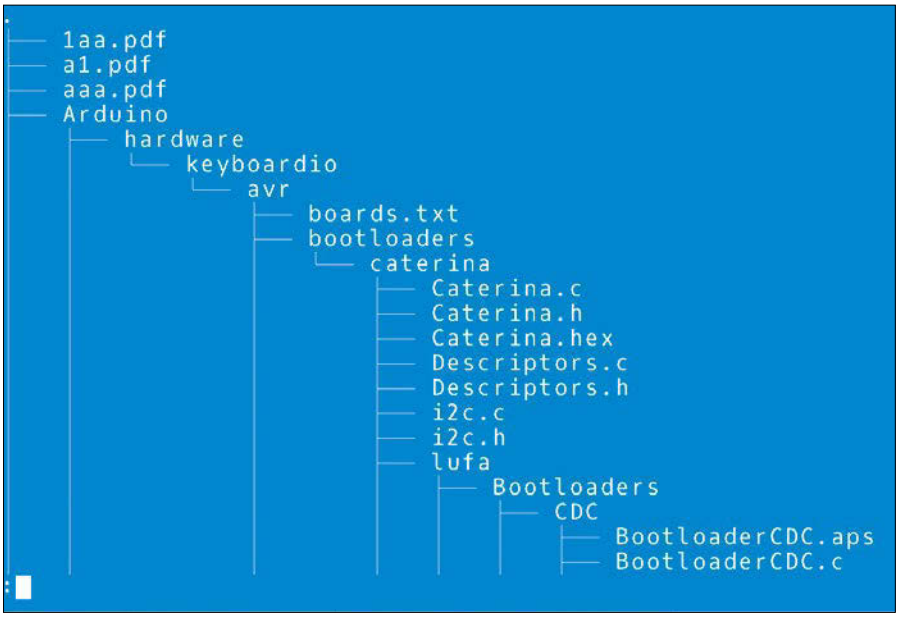


Figure 1: `tree` **improves on** `ls` **by offering a graphical view of the directory tree.**

Lead Image by Andras Vas on Unsplash

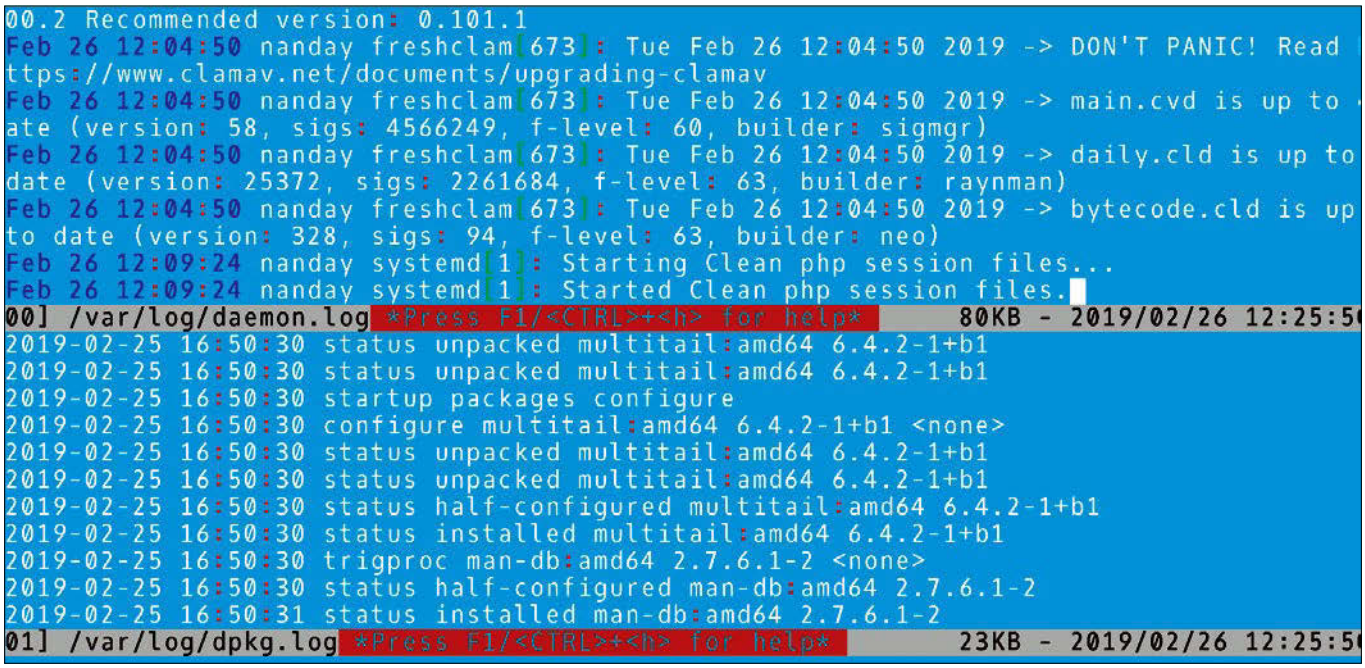


Figure 2: `multitail` **(Figure 2) displays the bottom output from multiple files.**

can show only directories and omit blank directories. Essentially, `tree` does everything `ls` does – even color-coding by setting color-coding variables – while also providing an additional bag of small enhancements.

multitail

`tail` is a standard command used by administrators to keep an eye on the most recent lines in logs and other files that change frequently. As the name implies, its replacement `multitail` (Figure 2) allows multiple files to be monitored with the same command, saving both effort and system resources.

Like `tail`, `multitail` can set the number of lines to display, as well as the interval between samples. In addition, `multitail` can be set to mark set intervals in which nothing happened with an *x*, or to replace repetitions of the same line with a message stating how many times it has been repeated. Similarly, `multitail` includes options for which windows present the output, including the option to merge all output from all files in the same window. Windows that are inactive for a defined interval can be set to close automatically, and output can be sent to a file or to another command.

One especially useful feature is the ability to use regular expressions. When regular expressions are defined, an entire directory can be monitored at the same time, with messages appearing only when a file has changed. Color and column formatting options are also available – all of which is considerably more sophisticated than `tail` with its handful of commands.

pax

Over the years, Linux has accumulated a bewildering set of formats for file compression (`bzip2`, `gzip`, `tar`, etc.). Each format has its own options, although some are more closely related than others. The `pax` command cuts through this confusion by offering a single interface for over half a dozen formats (Figure 3). The main difficulty is that while some formats like `tar` use the option `-x FORMAT`, other formats like `bzip2` have their own options, probably because support for them was added afterwards. Most users, too, will be surprised to learn that there are three different `tar` formats from which to chose. However, because format is only one option among others, `pax` spares users having to scan man pages when dealing with unfamiliar formats.

`pax`'s other options are not as complete as those for single format commands, but they are adequate enough for most purposes. Files can be appended to existing archives, and the contents of an archive can be defined

```
bb@nanday:~/fonts/Raleway$ pax -v -w -j -f Raleway.bzip2 ./Raleway*
./Raleway-Bold.ttf
./Raleway-ExtraBold.ttf
./Raleway-ExtraLight.ttf
./Raleway-Heavy.ttf
./Raleway-Light.ttf
./Raleway-Medium.ttf
./Raleway-Regular.ttf
./Raleway-SemiBold.ttf
./Raleway-Thin.ttf
pax: ustar vol 1, 9 files, 0 bytes read, 1167360 bytes written.
```

Figure 3: `pax` **provides a front end for multiple archive formats.**

```
bb@nanday:~$ fd ./vim*
vim.rc
vim.rc~
```

Figure 4: `fd` **is an alternative to the standard** `find` **command.**

by several criteria, such as file or block sizes, or a range of dates to include or exclude. Other options specify what file attributes to use. Overall, `pax` is a major simplification of archiving options.

fd

`fd` (packaged as `fd-find` in some distributions) is a simplification of the `find` command. `fd` (Figure 4) has less than half the options of `find` and does not have obscure distinctions like global and positional options that can confuse occasional users.

What `fd` does have is a selection of the most commonly used options. It supports regular expressions – which a search command must do to be any use at all – as well as options to include hidden files, to choose patterns to ignore, to perform case-sensitive searches, or to filter by file types. Probably, it would take a sophisticated user to find `fd` lacking.

most

The Bash shell is rich in pagers for viewing files. Although each has more functionality than its predecessors, all remain available in major distributions.

The newest pager is `most` (Figure 5). Its name continues a running joke; `less` is more than the `more` command, while `most` is more than `less`.

```
New York Bagels

2 cups warm (not boiling) water
2-3 TBSP active dry yeast
3-4 TBSP brown sugar
2 TEASP salt
6 cups of flour
poppy or sesame seeds

        Follow these instructions:

1. Combine water, yeast and 3TBSP of sugar in a bowl. Stir and let stand until foamy,$

2. Add 4 cups of flour and mix thoroughly.

3. Add an additional 2 cups of flour, a little at a time, stirring with wooden spoon $

4.Grease a large bowl with oil and place the dough in the bowl, turning it over a cou$

5. Cover bowl with a clean, dry towel and let stand in a warm, draft-free spot for 45$

6. Remove dough from the bowl and punch down.
-- MOST: bagel-recipe.txt                                          (1,1) 0%
Press `Q' to quit, `H' for help, and SPACE to scroll.
```

Figure 5: `most` **is an enhancement of** `less` **and** `more`**.**

```
root@nanday:~# apt install xchat
Reading package lists... Done
Building dependency tree
Reading state information... Done
Recommended packages:
  xchat-indicator
The following NEW packages will be installed:
  xchat
0 upgraded, 1 newly installed, 0 to remove and 112 not upgraded.
Need to get 0 B/381 kB of archives.
After this operation, 1,052 kB of additional disk space will be used.
Retrieving bug reports... Done
Parsing Found/Fixed information... Done
Selecting previously unselected package xchat.
(Reading database ... 521217 files and directories currently installed.)
Preparing to unpack .../xchat_2.8.8-15~bpo9+1_amd64.deb ...
Unpacking xchat (2.8.8-15~bpo9+1) ...
Processing triggers for mime-support (3.60) ...
Processing triggers for desktop-file-utils (0.23-1) ...
Setting up xchat (2.8.8-15~bpo9+1) ...
Processing triggers for man-db (2.7.6.1-2) ...
Processing triggers for hicolor-icon-theme (0.15-1) ...
```

Figure 6: `apt` **is an improved front end for Debian's** `dpkg` **and related utilities.**

```
root@nanday:/home/bb/projects/creative/raven-winter# rsync -v ./*.odt /home/bb/
Aglachads-trial-karas-read..odt
Raven-Winter-chapter11-with voices.odt
Raven-Winter-chapter15.odt
chapter-template.odt
chapter15-notes.odt
chapter16-notes.odt
chapter3-revisions.odt
interlude2.odt
raven-winter-general-notes.odt

sent 159,284 bytes  received 187 bytes  318,942.00 bytes/sec
total size is 158,553  speedup is 0.99
```

Figure 7: `rsync` is as useful for copying local files as it is for remote ones.

The advantages of `most` begin with multiple display options. It continues with the ability to display multiple files and to navigate between them using keyboard shortcuts. If you are frequently opening man pages, you might also want to set `most` as the default pages with commands, options, and other standard items color-coded.

apt

`apt-get` is the front end for the `dpkg` command in Debian and its derivatives. Over two decades or so, `apt-get` has grown both in complexity and in the number of related utilities – most of which include `apt` in their names. The result is an immensely powerful yet immensely confusing collection of tools.

First developed by Ubuntu, `apt` (Figure 6) is a replacement for the most common uses of `apt-get` and its utilities. Not only does it drop the last four characters of the command, but the same basic command is used for functions like `search` that were originally in a separate command. At the same time, the same functions found in `apt-get`, such as `install` and `update`, are also used, making `apt` easy to learn for experienced users. As an added bonus, `apt` includes a progress bar that is easier to read at a glance or from a distance than `apt-get`'s percentage completion counter.

`apt` is not nearly as comprehensive as `apt-get` and its utilities, so for advanced users it is not a replacement. However, for maybe eighty percent of standard package management, it makes maintenance simpler and more convenient.

rsync

`rsync` (Figure 7) was originally written for transferring files remotely over SSH. However, for many users, its speed and versatility makes `rsync` well-suited for local file copying as well, functioning as a replacement for `scp` and `cp`. Part of its appeal is its wide breadth of options, but much of its appeal is its delta transfer – that is, its ability to simplify a transfer by copying only files whose attributes or sizes have changed. This feature makes `rsync` useful for updating backups as well.

`rsync` is powerful by itself, but it becomes even more powerful when used via `Unison`, a front end that makes its options easier to navigate.

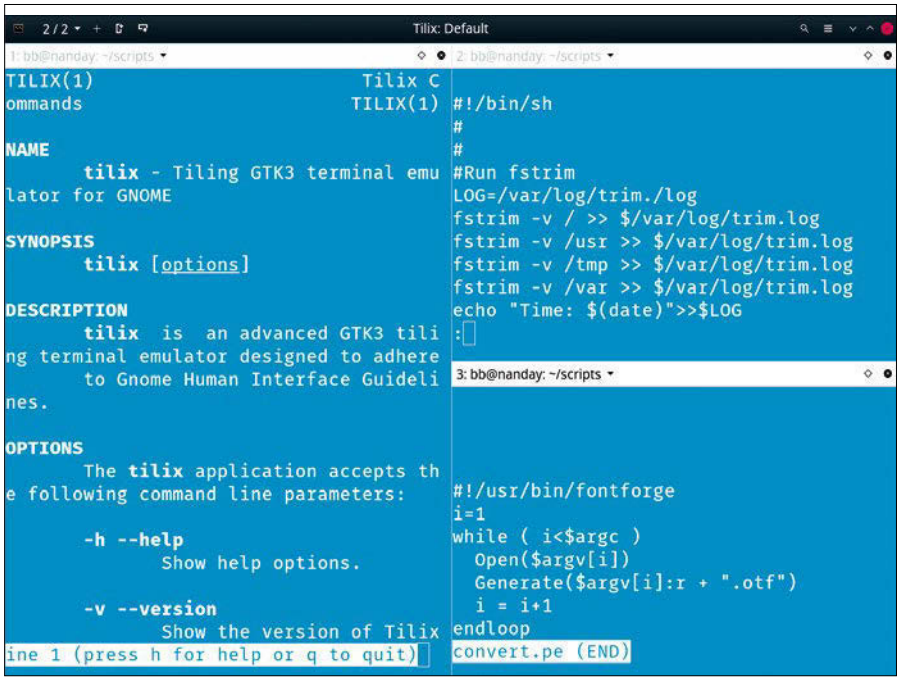


Figure 8: Tilix is a sophisticated virtual terminal.

Tilix

The terminals used to enter these commands and others like them have changed little over the history of Linux. Recently, though, the virtual terminal has also been updated. The most comprehensive of these efforts is Tilix, which is written in GTK3 + .

Tilix (Figure 8) is especially ideal for those who run several virtual terminals at the same time. It not only provides an index pane or tabs for easy navigation, but also the horizontal or vertical splitting of the current window.

Once multiple terminals are started, they can be synced, set to read-only, password-protected, and given bookmarks and hyperlinks. Notifications can be set to signal the end of a process, and triggers to start a process. One especially useful feature is a paste dialog to streamline copying.

None of these features is unique in itself, but their combination transforms the terminal and makes standard terminals seem stodgy and old-fashioned. Like the commands listed above, Tilix can make you feel that you are using a thoroughly modern tool and wonder how you ever got along without it. ■■■


LINUX MAGAZINE
INTERNET OF THINGS
Tools for home automation
Bluetooth Tricks
Linux on Old Hardware
Tablets on Linux
Raspberry Pi 3 B+
Python Gambling
Logism
MYCROFT
Can an open source personal assistant compete with Alexa?
Scanners and Linux
Peppermint OS
Meltdown & Spectre
Knight's Tour
Node-RED
LINUXVOICE
IS FIREFOX BACK?
We compare Mozilla's reborn browser with recent versions of Chrome
Kernel Self-Protection
Font Tricks
Write a script that changes your YouTube metadata
Lakka
NoMachine
Tkinter Library
4 COLLABORATIVE OFFICE SUITES: Keep your team in sync
LIBREBOOT and the liberation of firmware
MAPPING TOOLS
Create and edit digital maps with QGIS and QMapShack
4 Open Source Microblogging Tools
Kubuntu Up Close
MakerSpace
Caddy
ODROID-C2
Ogg Vorbis
A Python Script
REE BOOT!
boot and the liberation of firmware
Bash Tricks
FEDORA 27
We test drive the newest Fedora
TERMINAL TUNING
RMINAL TUNING
Eelo
GIT WORKSHOP
Up close with the classic Linux version control system
INNOVATIVE DISTROS
QUBES OS
Timers
WILL SYSTEMD REPLACE CRON?
ET CORE
.NET
INNOVATIVE DISTROS
Explore these hidden gems:
Systemd
Getting more from the elusive new Linux init system
10 Top PDF Readers
Arduino Programming for Open Hardware Projects
Homegrown Facial Recognition
Build facial recognition into your own Python scripts
Maker Tricks
Cover Your Ass(ets)
PRIVACY
Stop snoopers and protect your identity
Rockbox
RSS Readers
PiXtend
Rasp Pi Sailboat
Ready to Go?
DVD INSIDE! ALL 214 ISSUES ON A SEARCHABLE DISC!
HUGE SAVINGS! €29.90 VALUE!
Upribox 2
Git Workshop
Purism Libr
/proc Filesystem
cheat.sh
Counting Pennies
Electron Fra
Remote Pair Prog
FOSSPicks
Tutorials

Linux Magazine **is your guide to the world of Linux. Look inside for advanced technical information you won't find anywhere else!**

Expand your Linux skills with:

- In-depth articles on trending topics, including Bitcoin, ransomware, cloud computing, and more!
- How-tos and tutorials on useful tools that will save you time and protect your data
- Troubleshooting and optimization tips
- Insightful news on crucial developments in the world of open source
- Cool projects for Raspberry Pi, Arduino, and other maker-board systems

If you want to go farther and do more with Linux, subscribe today and never miss another issue!

Subscribe now!

shop.linuxnewmedia.com/subs

GET IT NOW!
FAST DELIVERY WITH OUR PDF EDITION

The sys admin's daily grind: Adminer

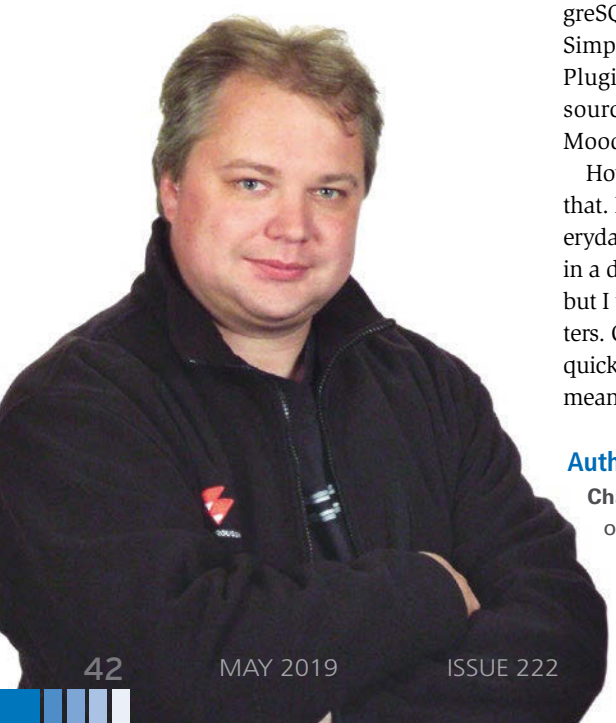
At the First Click

Sys admin columnist Charly freely admits that he doesn't like SQL and phpMyAdmin any more than he does COBOL. Instead, meet his new best friend; the slim, attractive database tool known as Adminer. *By Charly Kühnast*

When I was young and AIX 4 was in its infancy, we had a full-time database administrator at the data center who could not only recite the whole SQL command set forwards, but could also type it backwards. Behind his back, the trainees called him `GRANT` [1] father. Fortunately, I didn't inherit his job, because on my popularity scale, SQL ranks at about the same level as COBOL – I learned both once, and I try hard not to use either.

But unlike Cobol, SQL often sticks to the sole of my shoe like a piece of old chewing gum. So, I was looking for a tool that makes the most common SQL operations easier for me and isn't called phpMyAdmin, because I don't like that much either. I hit the bullseye with the tool I found: Adminder [2]! To my mind, this tool gets just about everything right.

First, there is the size. The variant with more than 40 languages gets by with about 470KB. If I limit myself to German or English for the program interface, this drops to hardly more than 300KB. The next surprise: I didn't have to unpack anything, because Adminer consists of a single PHP file. I copied it to a management server on the internal LAN. Adminer users need to go through LDAP authentication just once. Its maker, Jakub Vrána, does not recommend installing the tool directly on the database server.



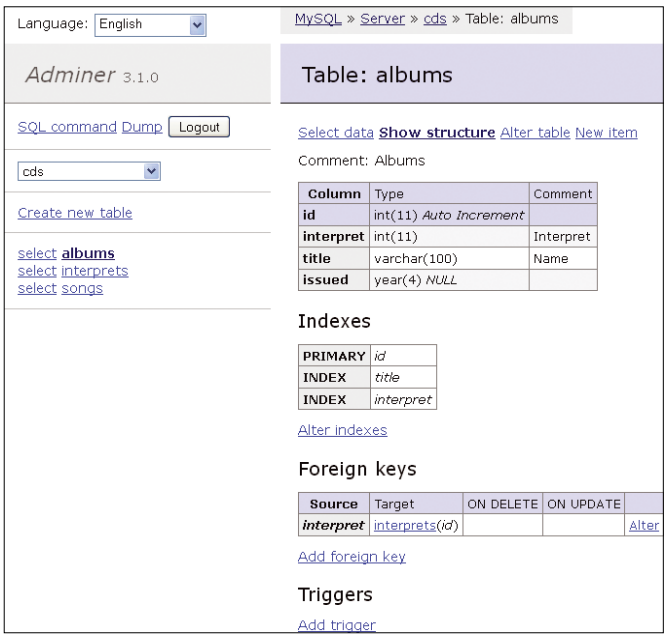


Figure 1: Adminer looking at the structure of a database table

Minor Interventions

The selection of databases that Adminer understands should be sufficient for most admins: MySQL, MariaDB, PostgreSQL, SQlite, MS SQL, Oracle, Firebird, SimpleDB, Elasticsearch, and MongoDB. Plugins are available from third-party sources, for example for managing Moodle or WordPress databases.

However, I don't need anything like that. I'm usually just doing annoying everyday repairs. For example, when a field in a database is limited to 128 characters, but I want to accommodate 200 characters. Or another common task: I need to quickly add another field to a table. This means that I am only using two percent of Adminer's feature set. But it is simply a pleasure to work with Adminer, and the pleasantly simple design maximizes the fun (Figure 1).

I could even create functions and stored procedures with Adminer – but I prefer to take a more hands-off approach than that. When I'm faced with difficult tasks, I prefer to ask one of the trainees from back in the day, the one they now call `GRANT` father behind his back. ■■■

Info

[1] SQL-GRANT-Kommando: *https://dev.mysql.com/doc/refman/8.0/en/grant.html*

[2] Adminer: *https://www.adminer.org*

Author

Charly Kühnast manages Unix systems in the data center in the Lower Rhine region of Germany. His responsibilities include ensuring the security and availability of firewalls and the DMZ.

MOBILE USERS

search for us today at your digital newsstand!

Only a swipe away!

Download our convenient digital editions for your iPad, iPhone, or Android device.

Visit our apps page for more information:
shop.linuxnewmedia.com/us/apps

Gimp image optimization with Python plugins

Photo Studio

Performing the same Gimp image processing steps again and again is tiresome and error prone. Mike Schilli assigns this task to a Python script via a home grown new menu entry. *By Mike Schilli*

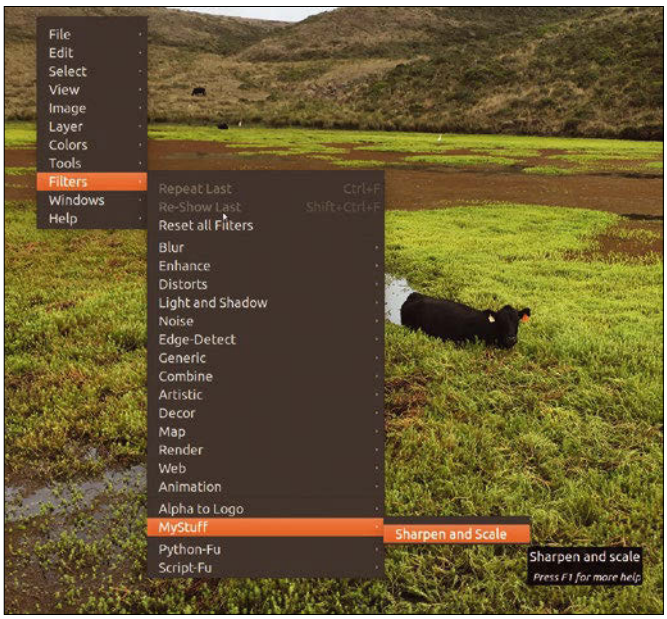
Today's cellphone cameras record images in giant formats that are hardly suitable for blogging or sending through narrow data pipes. I tend to scale all of these photos down to 2000x1000 pixels, maybe sharpen them a bit, and perform white balancing on each one. The Gimp image editor has been my tool of choice for many years, but it would be nice if it would help me out by performing these repetitive steps automatically.

In One Fell Swoop

Fortunately, Python DIY scripts can be easily integrated into Gimp. The Ubuntu installation of the *gimp* package already contains all the ingredients for home-made commands. Listing 1 [1] initializes a new custom plugin for scaling and sharpening a photo. Saved in the `~/.gimp-2.8/plug-ins/` directory with execution rights, Gimp finds the file on startup and adds its new menu entry as requested under *Filters* | *MyStuff* | *Sharpen and Scale* (Figure 1). The entry is then displayed both in Gimp's context and drop-down menus.

The main plugin script in Listing 1 calls the program logic for image processing in line 21 with `sharp_scale()`. This function was previously imported in line 3 from the `sharpscale.py` file (Listing 2) using an `import` command. Gimp also finds the second script in the plugin directory where the user previously installed it. Since it is only used as a library, no execution rights are required.

Listing 1 fetches Gimp-specific Python constructs like the `register()` function for integrating the plugin or constants like `PF_IMAGE` from the module `gimpfu` in line 2; scripts located in Gimp's plugin directory find it without additional import paths. The array starting in line 14 of Listing 1 defines two parameters for calling the plugin function, an image in Gimp's internal `PF_IMAGE` format and its top layer as a drawable; the latter is then used by the function later on for sharpening.



Lead Image courtesy of Mike Schilli

Figure 1: The menu entry for scaling and sharpening in Gimp, set up with a Python script.

Author

Mike Schilli works as a software engineer in the San Francisco Bay area, California. Each month in his column, which has been running since 1997, he researches practical applications of various programming languages. If you email him at *mschilli@perlmeister.com* he will gladly answer any questions.

Figure 2: Users see a progress bar below the image during the resizing process, which can take some time for larger images.

New Menu

If the user clicks on the newly added menu item as shown in Figure 1, the plugin framework jumps to the `sharp_scale()` function starting in line 6 of Listing 2 and passes the displayed image's descriptor and a reference to the drawable to it. The `scale_coords()` function computes the dimensions of the reduced image starting in line 30. It takes the current width and height of the image from line 8 and determines whether it is in landscape or portrait format.

It calculates the scaling factor `scale` by dividing the length of the longest side by the maximum size defined in the constant `SIZE_MAX`. The scaled image's dimensions are obtained by multiplying the width and height of the original image by the scaling factor in line 39.

Since scaling can take time with large images, and impatient users expect immediate feedback after pushing a button, lines 12 to 14 use `gimp_progress_init()` to output a status message, which Gimp displays as a progress bar (Figure 2).

When compressing the image, Gimp has to combine several pixels into one, which requires smoothing to make the image look natural afterwards. Line 17 therefore sets Lanczos resampling [2] as the interpolation function.

Well-Documented

Gimp documents the parameter values for calling internal functions directly in the application in the dialog box that appears when navigating to *Help* | *Procedure Browser* (Figure 3). There you can search for a keyword like *scale* or *sharpen* and, after some trial and error, typically find the right internal function and its parameters. If something goes wrong, for example, if the plugin script quits with an error, Gimp writes an error message to the console. It is therefore recommended to start Gimp from the command line during the debug phase so that the messages in the terminal allow conclusions to be drawn about anything that might have gone wrong. After making any changes to the Python code in the plugins, Gimp should be restarted to guarantee that the changes take effect.

For retroactive sharpening, the built-in `plug_in_sharpen()` function grabs the

Listing 1: sharpscaleplugin.py

```
01 #!/usr/bin/env python
02 from gimpfu import *
03 from sharpscale import sharp_scale
04
05 register(
06   "python-fu-sharp-scale",
07   "Sharpen and scale",
08   "First sharpens, then scales",
09   "Mike Schilli",
10   "Mike Schilli",
11   "2019",
12   "_Sharpen and Scale",
13   "RGB",
14   [
15     (PF_IMAGE,
16         "image", "Input image", None),
17     (PF_DRAWABLE,
18         "drawable", "Input drawable", None)
19   ],
20   [],
21   sharp_scale,
22   menu="<Image>/Filters/MyStuff"
23 )
24
25 main()
```

Listing 2: sharpscale.py

```
01 from gimpfu import *
02
03 SIZE_MAX=2000
04 SHARPEN=10
05
06 def sharp_scale(img, layer):
07   w, h = scale_coords(
08     img.width, img.height)
09
10   pdb.gimp_image_undo_group_start(img)
11
12   pdb.gimp_progress_init(
13     "Scaling Image to %dx%d ..." % (w, h),
14     None)
15
16   pdb.gimp_context_set_interpolation(
17     INTERPOLATION_LANCZOS)
18
19   pdb.gimp_image_scale(img, int(w), int(h))
20
21   pdb.gimp_progress_init(
22     "Sharpening Image with %d ..." %
23       SHARPEN, None)
24   pdb.plug_in_sharpen(img, layer, SHARPEN)
25
26   pdb.gimp_levels_stretch(layer)
27
28   pdb.gimp_image_undo_group_end(img)
29
30 def scale_coords(w,h):
31   if max(w,h) < SIZE_MAX:
32     return w, h
33
34   if w<=h:
35     scale=SIZE_MAX/float(h)
36   else:
37     scale=SIZE_MAX/float(w)
38
39   return int(w*scale), int(h*scale)
```

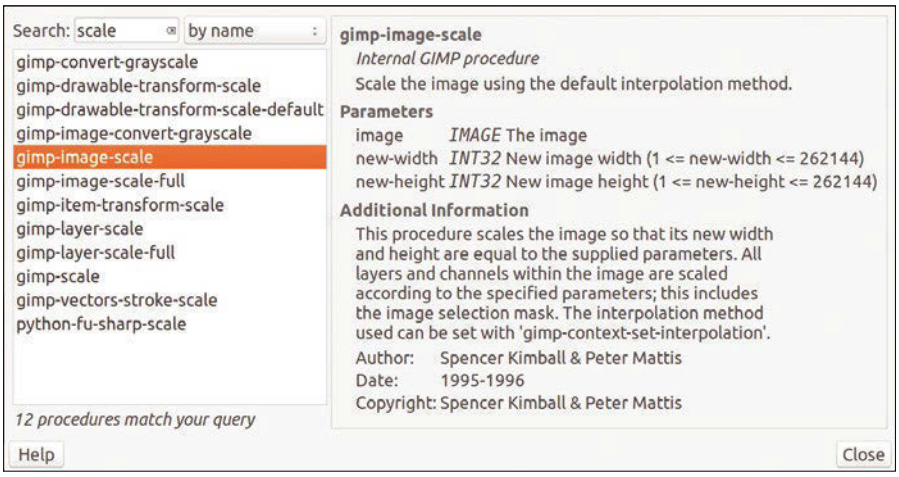


Figure 3: The Procedure Browser reveals the parameters of popular Gimp functions.

current image in line 24. The desired sharpness value is defined with a value of `10` by the `SHARPEN` constant in line 4; line 24 passes it on to the Sharpen plugin (a standard Gimp plugin already included). According to the manual, the function requires not only the Gimp image but also its premier layer as a drawable, which, as mentioned above, the plugin registration passed to the `sharp_scale()` call right from the outset.

Show Your Colors

As the third correction of raw images, I often adjust the white balance. If a digital photo does not exhaust the full range of possible pixel values from 0 to 255 for each of the three channels (red, green, blue), images often lack vibrancy and appear flat (Figure 4).

If you prefer brighter colors and higher contrast, you can fire up a useful tool in Gimp under *Colors* | *Auto* | *White Balance*; this adjusts the pixel values of the three channels so that their distribution in the histogram covers the entire intensity bandwidth. This doesn't always look good, but it sometimes gives boring pictures the necessary pizzazz (Figure 5).

However, my initial search for the corresponding Gimp function in the Procedure Browser and in the Plugin Browser was unsuccessful. But luckily, Gimp's source code on GitHub is visible to everyone (Figure 6), and a short search for the `Automatic white balance` string in the repository showed that the corresponding Gimp procedure goes by the name of `drawable_levels_stretch()`.

This function is again documented in the Procedure Browser and only requires the drawable of the image as a parameter. I quickly added the call at line 26 of Listing 2, and it became part of the image processing process.

One Step Forward, One Step Back

Gimp naturally treats the three actions of scaling, sharpening, and color correction of the image as three separate actions. If a user changes their mind afterwards

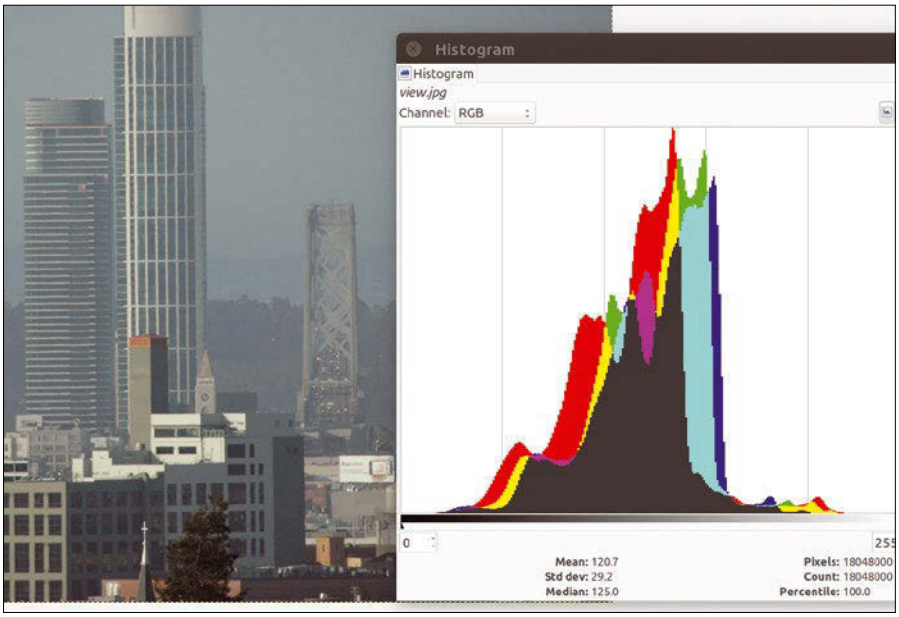


Figure 4: Before the white balance correction, the histogram shows narrowly distributed values and the image appears flat.

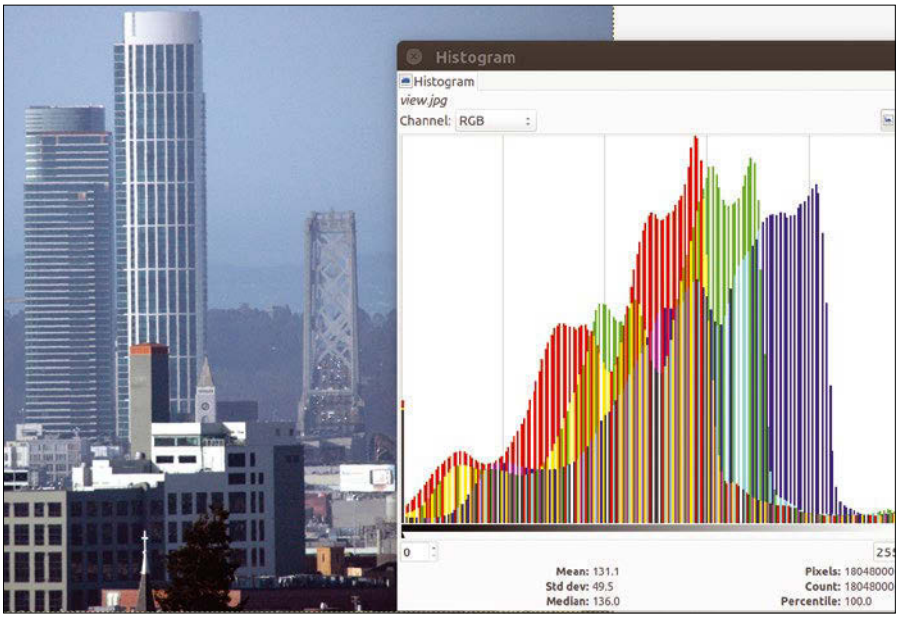


Figure 5: White balance makes the histogram wider and the image more vivid.

```
42  static const GimpActionEntry drawable_actions[] =
43  {
44    { "drawable-equalize", NULL,
45      NC_("drawable-action", "_Equalize"), NULL,
46      NC_("drawable-action", "Automatic contrast enhancement"),
47      G_CALLBACK (drawable_equalize_cmd_callback),
48      GIMP_HELP_LAYER_EQUALIZE },
49
50    { "drawable-levels-stretch", NULL,
51      NC_("drawable-action", "_White Balance"), NULL,
52      NC_("drawable-action", "Automatic white balance correction"),
53      G_CALLBACK (drawable_levels_stretch_cmd_callback),
54      GIMP_HELP_LAYER_WHITE_BALANCE },
```

Figure 6: A look at the Gimp source code on GitHub reveals that the function for white balance is called `drawable_levels_stretch`.

and clicks on the `Undo` menu entry, they would have to do this three times in a row, which is annoying, because they actually only initiated a single combined action via a mouse click in the menu. The solution is to call `gimp_image_undo_group_start()` in line 10 and the corresponding `..._end()` in line 28.

They define all three intermediate actions as a single undo unit, so that Gimp will roll them back atomically, just like it will roll them forward as one step when asked to do so. Figure 7 shows that even Gimp's Edit menu now shows the last action not as the white balance (which was last to run), but as *Sharpen and Scale*, which is the name of the combined action we used in the plugin code for the whole enchilada to happen in one go.

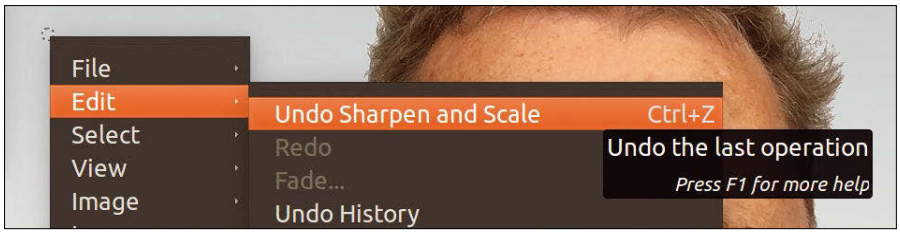


Figure 7: Thanks to `undo_group_start()` and `...end()`, Gimp can reverse the actions in one go.

I hope this issue's examples have shown that Gimp's Python API has access to all internal functions and that it is easy to bundle workflows consisting of many small steps, attach them to a menu and run them in one fell swoop. This eliminates drudgery for the user, saves time, and eliminates errors humans inevitably make. ■■■

Info

[1] Listings for this article: *ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/222/*

[2] Lanczos resampling: *https://en.wikipedia.org/wiki/Lanczos_resampling*

Emergency shell access with tmate

Over Your Shoulder

Tmate lets you access the shell session of a remote computer – even through a router.

By Ferdinand Thommes

Your parents or friends have replaced an unmanageable Windows installation with an easy-to-use Linux variant, and they have designated you as their remote admin. To access the system from afar, you resort to a GUI-based tool such as TeamViewer or AnyDesk. All is well until the graphical user interface (GUI) for the computer you need to access breaks. Suddenly, your GUI-based remote access tool won't work anymore, because you don't have access to an X server.

Tmate [1], which stands for "team mate," offers a convenient solution for remote access in situations when you can't depend on the graphical tools. This fork of the popular tmux [2] terminal multiplexer has significantly more functions than the original. Tmux is functionally similar to Screen [3] in some ways. You can use tmate to create and manage remote virtual sessions within a terminal or terminal emulator.

Self Hosted

Tmate is not only suitable for external help, but also lets developers collaborate on software. Technically, the tool uses Secure Shell (SSH) to establish a secure connection to the tmate.io server, which is under the control of the tmate developers.

Alternatively, you can run tmate on your own server. The resource requirements are kept within such narrow limits that a recent Raspberry Pi or a similarly equipped single-board computer is sufficient. The machine only generates an SSH ID and some URLs for different types of sessions – SSH or web sessions in read-only mode or with full access.

Installing tmate is easy, because the program is available from the package sources of all common distributions. For Arch Linux, use the AUR user archive;

```
ft@ft-N14xWU:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ft/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ft/.ssh/id_rsa.
Your public key has been saved in /home/ft/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:ZJfeirJSgEaYvyqs9zF3BgB6sSJYj82Jk3pQPLMtzr8 ft@ft-N14xWU
The key's randomart image is:
+---[RSA 4096]----+
| +=              |
|++=@ .     .     |
|*oB**   o o      |
|.=*.o. o o .     |
|.+.o .. S . .    |
| .+   .. . .     |
|.. .o.o + .      |
|o.. o+ =         |
|+. .Eo.          |
+----[SHA256]-----+
```

Figure 1: An SSH key pair is quickly created. In contrast to normal use, the public key remains on the computer. The computers only exchange randomly created IDs or URLs via the software at startup.

Lead Image © Cathy Yeulet, 123RF.com

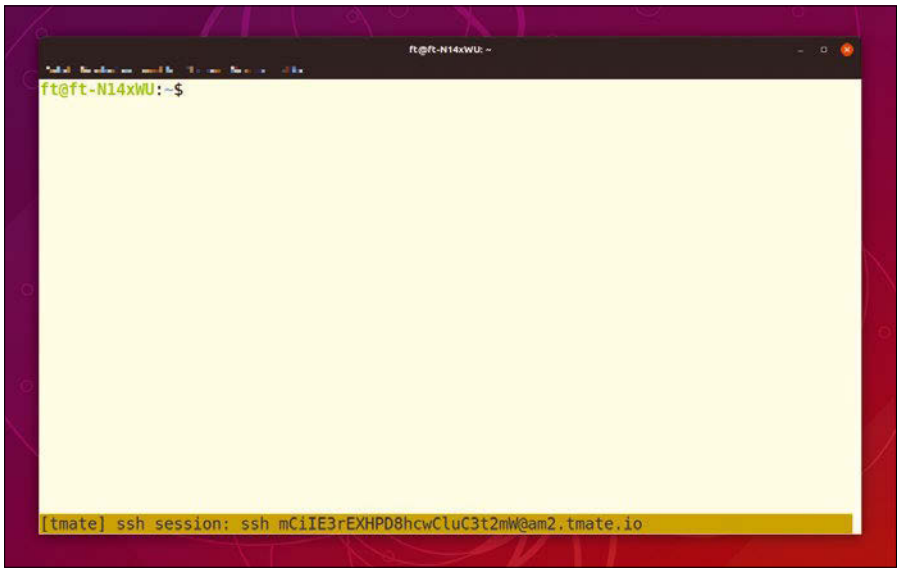


Figure 2: After the application starts, it displays a randomly generated SSH ID at the bottom of the terminal.

for Debian and its derivatives including Ubuntu, just type:

```
sudo apt install tmate
```

On Fedora, this translates to:

```
sudo dnf install tmate
```

Creating Keys

After the installation, which occupies only a few kilobytes on your disk, you need to create a key pair for SSH (if it does not already exist). As shown in Figure 1, this is done by typing:

```
ssh-keygen -t rsa -b 4096
```

Answer the individual prompts by pressing the Enter key. You can assign a password for the key, but in our example this is not necessary for home use.

Tmate uses the RSA cryptographic procedure with a secure key length of 4096 bits. By default, the keys are located in the hidden `.ssh/` directory in your home directory; tmate will now find them automatically when you use the tool. You need to generate these keys on each computer involved if a key pair does not already exist there.

When you open a connection, the program creates a 150-bit session token. It then launches a tmux server in a sandbox without a filesystem and user rights, but with its own namespace to isolate the server from other processes. To make this possible, the software opens all the files it needs for running the tmux server before they end up in the sandbox.

These measures are intended to limit the leverage of potential exploits targeting the tmux server. An attacker would not be able to access other sessions. The technical principles are explained in detail in a 10-page paper [4].

SSH ID or URL

After starting tmate as a user for the first time, a bar at the bottom of the terminal window shows a random SSH ID for establishing an SSH session in Read/Write mode (Figure 2). It is best to copy this line to the clipboard, because it disappears after a short time.

If you did not react fast enough, you can get the data including the other IDs or URLs back on the screen with the `tmate show-messages` command. This command is also used for logging during a session (Figure 3).

While the SSH ID displayed at the start of tmate creates a read-write session in a terminal, `tmate show-messages` also shows an ID for a terminal session that supports read-only access. In addition, the program provides two URLs that support the same procedure in the browser (Figure 4).

Sending Keys

Now you only have to send the SSH ID or URL to the other participants of the session. The best way to do this is to use chat and mail or share via a service such as Nextcloud. The final option you could

THE COMPLETE
LINUXVOICE
3,400 PAGES OF LINUX AND OPEN SOURCE LOVE!
LINUXVOICE
ARCHIVE DVD: ALL 32 ISSUES!
Since April 2014, Linux Voice has showcased the very best that Free Software has to offer. Now you can get it all on one searchable DVD.
Includes all 32 issues
in EPUB, PDF, and HTML format!
Order now!
shop.linuxnewmedia.com

```
ft@ft-N14xWU:~$ tmate show-messages
Thu Nov 22 08:49:06 2018 [tmate] Connecting to ssh.tmate.io...
Thu Nov 22 08:49:07 2018 [tmate] Note: clear your terminal before sharing readonly access
Thu Nov 22 08:49:07 2018 [tmate] web session read only: https://tmate.io/t/ro-VdXGZbJ2iwXSyIYCCZG2CF76m
Thu Nov 22 08:49:07 2018 [tmate] ssh session read only: ssh ro-VdXGZbJ2iwXSyIYCCZG2CF76m@am2.tmate.io
Thu Nov 22 08:49:07 2018 [tmate] web session: https://tmate.io/t/mCiIE3rEXHPD8hcwCluC3t2mW
Thu Nov 22 08:49:07 2018 [tmate] ssh session: ssh mCiIE3rEXHPD8hcwCluC3t2mW@am2.tmate.io
Thu Nov 22 09:01:34 2018 [tmate] A mate has joined (89.249.64.155) -- 1 client currently connected
Thu Nov 22 09:06:16 2018 [tmate] A mate has joined (95.168.145.55) -- 2 clients currently connected
```

Figure 3: Using the `tmate show-messages` command, the program displays the SSH IDs and URLs (only valid for this session), as well as the running application's logs.

consider is an error-prone transmission by telephone.

The participants you invited and entrusted with an ID or URL do not need to install tmate or tmux. You don't need to use the same distribution or even use Linux: FreeBSD, OpenBSD, NetBSD, and Mac OS X can also serve as a basis – only Windows is left out in the cold.

When the other party now enters the transmitted SSH ID/URL, the terminal shows what the person seeking help is seeing in the terminal themselves. Commands and their results run back and forth between the computers in fractions of a second. In the session with read and write access, all parties can enter data themselves; in the restricted session, only the host can write; the participants remain restricted to reading.

If you no longer need the split terminal, it is advisable to end the session by entering `exit`. If you just close the terminal, the other side will still have access to the computer – a potential security risk.

Conclusions and Outlook

If the GUI fails to launch, remote session support with tools such as TeamViewer or AnyDesk falls flat. Among other things, tmate plays to its strengths on such occasions and makes it possible to access the remote computer via a shell without great effort.

In addition, tmate offers all the benefits of tmux, but eliminates the need to manually set up an SSH connection when sharing the terminal. This makes it suitable not only for supporting remote computers, but also for pair programming, for example. A developer types code into a terminal, while a colleague reads it to validate the input. This way of working detects errors at an early stage and ensures high-quality code.

The developers are currently thinking about making it possible to transfer files according to the SCP [5] principle. So far, you have to use a service like transfer.sh [6] within tmate. Session logging is also on the developers' roadmap. ■■■

Info

[1] tmate: *https://tmate.io*

[2] tmux: *https://github.com/tmux/tmux/wiki*

[3] Screen: *https://www.gnu.org/software/screen/*

[4] Technical principles: *https://viennot.com/tmate.pdf*

[5] SCP: *https://en.wikipedia.org/wiki/Secure_copy*

[6] transfer.sh: *https://transfer.sh*

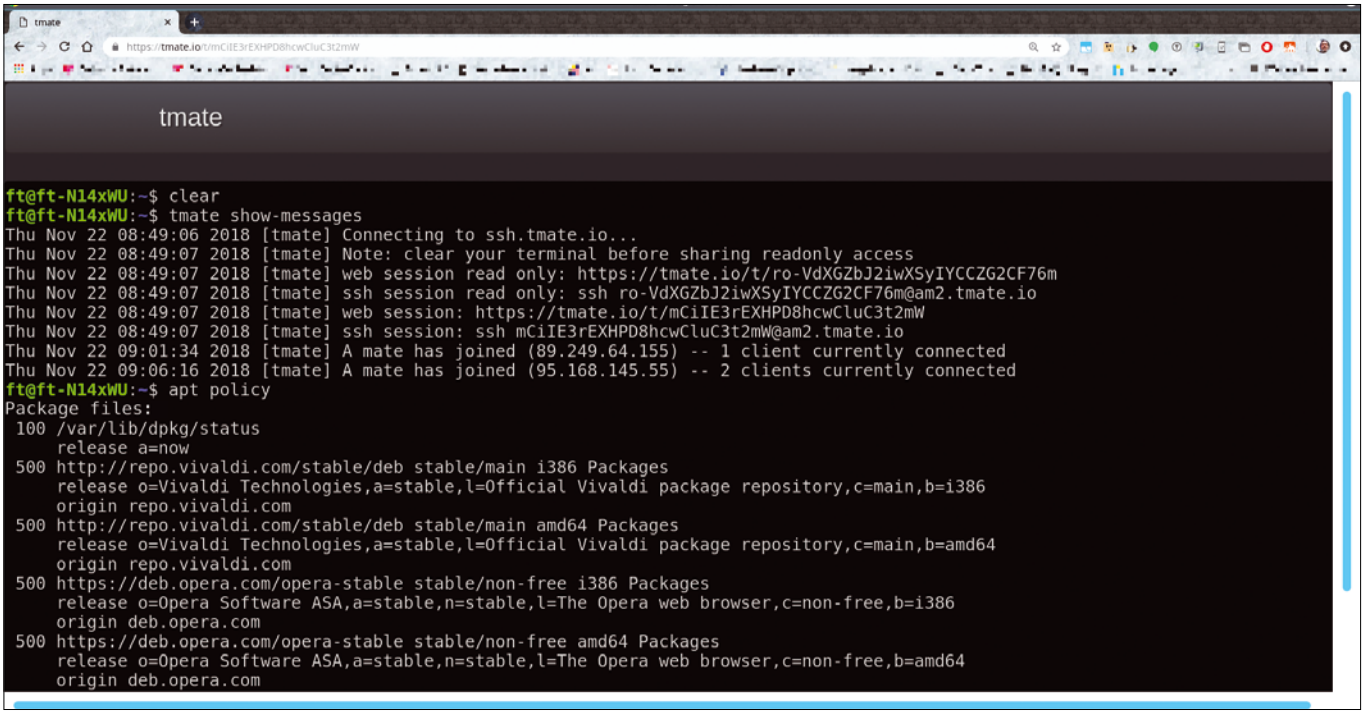


Figure 4: Tmate not only works in the terminal, but also in the web browser. This also applies if you host tmate yourself.

Available Now

* 2018 EDITION *

Linux Magazine Archive DVD

Searchable Linux Archive: 19,000 Pages of Practical Know-How

214 issues of Linux Magazine on one handy DVD!

ORDER NOW!

shop.linuxnewmedia.com

PDF types in LibreOffice

Custom Export

With LibreOffice, you can customize the way you export PDFs. *By Bruce Byfield*

LibreOffice and related apps have supported the PDF format for over a decade. If nothing else, LibreOffice is a free alternative to the $15 monthly fee required to license Adobe Acrobat tools.

Most people exporting a PDF from LibreOffice just use the default settings, exporting to PDF as quickly as possible. However, those who are more hands-on know that they can do more than accept the default. If users choose, they can customize everything from the pages to export to the look of the exported file in a reader and its initial view from *File* | *Export as* | *Export as PDF*.... By clicking on the *General* tab, you can see that those options include four PDF format variants: Hybrid, Archive, Tagged, and PDF forms (Figure 1). Each of these PDF types requires different preparation.

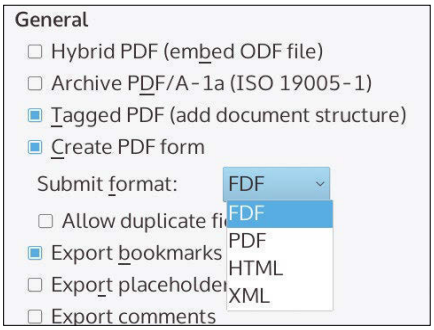


Figure 1: The four PDFs that LibreOffice supports.

Hybrid PDFs

Hybrid PDFs are two files in one: a PDF file and an Open Document Format (ODF) file, which is LibreOffice's default format. A Hybrid PDF acts like a normal PDF file in a reader, but it can open in the appropriate LibreOffice module – Writer, Calc, or Impress.

Why would you want a Hybrid PDF? The reason is simple. LibreOffice has supported editing of PDF files in Draw for years. Draw is a useful PDF viewer, because it supports multiple pages and includes a pane for jumping from one page to another, just like a dedicated PDF reader.

However, your ability to edit is limited. Images can be edited freely, but text in a PDF is structured line by line (Figure 2). You can change words within a line. If there is enough white space, you can even add a line. However, any changes longer than a line require editing the line below as well. In no time at all, you can have cascading changes and endless lines to edit. More substantial edits, such as adding or deleting a paragraph – let alone a page – can be almost impossible.

By contrast, the ODF file in a Hybrid PDF opens in the appropriate LibreOffice module and can be edited as easily as the original file. When you are finished, the changes are passed on to the Hybrid PDF as you save. What might take an hour to edit in an ordinary PDF can be reduced to a couple of minutes in a Hybrid PDF.

As a side benefit, you can store both formats in the same document, reducing the number of files you need to track.

One drawback to a Hybrid PDF is that it is two files in one, and therefore about 50 percent larger than an ordinary PDF file. Another drawback is that there is no way to identify a Hybrid PDF in a file manager. You can only open the Hybrid PDF in LibreOffice and see in which LibreOffice module it opens. Yet even that may not help, because Tagged PDFs, ordinary PDFs, and Hybrid PDFs created in Draw will all open in Draw. The easiest way to identify Hybrid PDFs is to save them to a directory called Hybrid PDFs.

Archive PDFs

Changing formats – especially proprietary ones – means that the software needed to read and edit documents written today may not be available at some point in the future. Open source formats are generally considered the best hedge against this possibility, but Adobe has released an open standards solution in the form of PDF/A (ISO 19005-1). In the short term, the format is useful for ensuring that a PDF can be read on any system, regardless of its resources.

Photo by naomi tamar on Unsplash

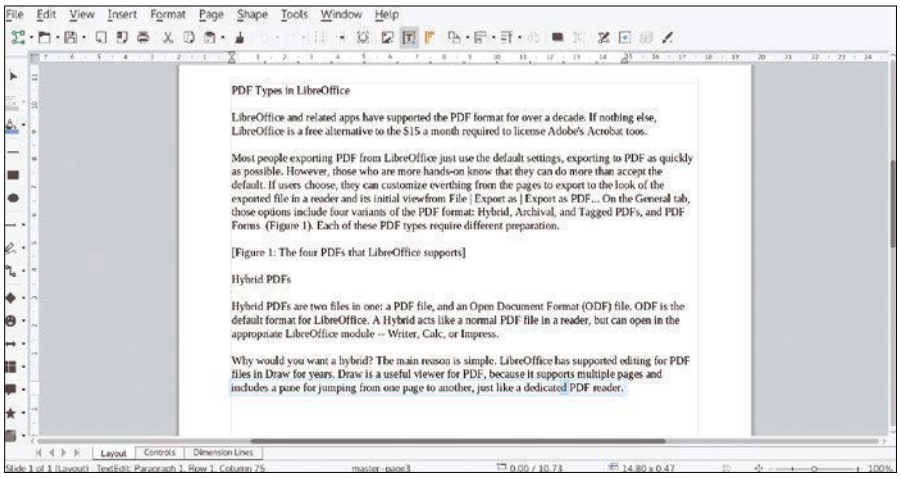


Figure 2: LibreOffice supports editing PDFs in Draw, but the PDF's structure means that editing is line by line.

A PDF/A file is a PDF that contains everything necessary to display the file; it embeds fonts, images, and anything else that the document requires. In addition, when creating an archive file, you might want to fill out the fields in *Tools* | Options |*Personal Data*, as well as the metadata fields in *Files* | *Properties*. In particular, the *Description* and *Custom Properties* tabs can be used. Include any information that you think someone else using the file might need to know.

Note that Archive PDF files can be huge, especially if a document contains several different fonts. To give you an idea, an Archive PDF file with no images using a single font is almost half as big again as the same document in an ordinary PDF.

Tagged PDFs

When you open *File* | *Export As* | *Export as PDF*, you'll find *Tagged PDF (add document structure)* already selected. The reason for this default is that Tagged PDFs have several advantages over ordinary PDFs while adding little to the file size. In fact, online, you often find recommendations that everyone should be using Tagged PDFs.

A Tagged PDF contains meta-information about the structure and layout of the document, adding far more details than are contained in an ordinary PDF, which is formatted by the position of lines on a page. This information is unseen in PDF readers, but has a number of advantages.

To start with, a Tagged PDF opens quicker. For another, when all or part of a file is copied to another application, it displays as a single graphic file and does not need to be reformatted, which makes recovery of information easier, as well as making reading easier for other applications, such as accessibility tools, to import and manipulate the file. Other tags can also be added, such as geographic information. In general, a Tagged PDF makes the PDF format easier to work with, although it opens in Draw in LibreOffice.

PDF Forms

PDF forms can be filled in by users and emailed from a web page with a simple script (examples of which can be found online for different programming languages, such as JavaScript or Python). To make a form, open a new Writer file and select *View* | *Toolbars* | *Form Controls*. If you are doing a long, complicated form, you might also select the Form Design toolbar, which contains such useful tools as the Form Navigator, as well as buttons for toggling the design grid. Otherwise, you can get by with just the Form Controls toolbar.

To begin designing a form, toggle on *Design Mode* in the Form Design toolbar. You can add all the usual widgets to the form, defining their size and position with the mouse. Available widgets include input fields, option buttons, checkboxes, and combo boxes. Each widget can be further defined by clicking on it and making selections from the Properties Checkbox dialog (Figure 3).

When you are finished, save the form and select *File* | *Export As* | *Export as PDF...* | *General* | *Create PDF form*. You will also need to make a choice from the *Submit format* combo box to choose the format in which completed forms will be sent: FDF, PDF, HTML, or XML. You can check the results by opening the resulting PDF and trying to write in a field or use a widget.

Choices

To some extent, these options can be combined. You can combine a Hybrid and Archive PDF without difficulty or use either with a PDF form. However, there would be no point in using a PDF form or an Archive PDF with a Tagged PDF, because their purposes are more or less opposing ones.

And what if you choose *File* | *Export As* | *Export As PDF* and simply accept LibreOffice's default settings? Then your only concern is to name your file. I have not found a complete list of PDF export defaults, but presumably the settings are those that are active when you open the PDF Options dialog, including *JPEG compression* set at *90%*, *Tagged PDF*, and *Export bookmarks*. These defaults work extremely well for many purposes, but it can be useful to remember that they are not the only options when you create PDFs with LibreOffice. ■■■

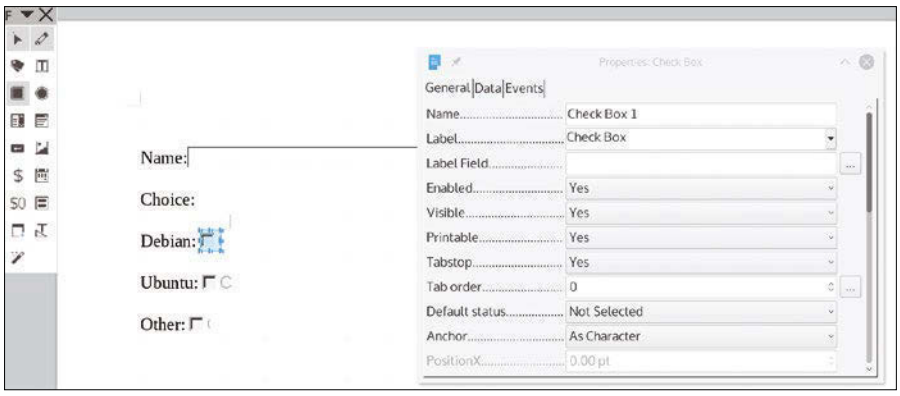


Figure 3: The tools for creating a PDF form.

MakerSpace

MicroPython and ESP8266 Airboat

Water Python

An airboat project uses a standalone WiFi web server for remote control. *By Pete Metcalfe*

My daughters and I have built a number of airboat projects, but this time I thought that I'd use MicroPython [1], a lean and efficient implementation of the Python 3 programming language that includes a small subset of the Python standard library and is optimized to run on microcontrollers.

For this project, my goal was to create a MicroPython standalone WiFi access point and use it as a small web server for controlling the airboat fans.

Hardware

The hardware for this project included:

- 1 ESP8266 module (WiFi microchip)
- 2 L9110 fans
- 1 Arduino Uno proto shield (or use a small breadboard)
- 9V battery or portable phone charger
- K'Nex blocks
- 1 small plastic box
- 2 plastic water bottles
- Duct tape

MicroPython is supported on a number of WiFi-enabled ESP32 and ESP8266 modules. These modules are well priced, with the NodeMCU module starting around $5. Most of the ESP8266 modules also support Lua and Arduino C/C++ programming.

For this project, I used a WeMos ESP8266 module, which comes in a Arduino Uno form factor (Figure 1). I like the WeMos modules, because I can use my Uno proto shields. Also, these modules support 5V DC, as opposed to the typical MicroPython modules, which only support 3.3V. The fans will work with 3.3V, but they generate a lot more wind power at 5V.

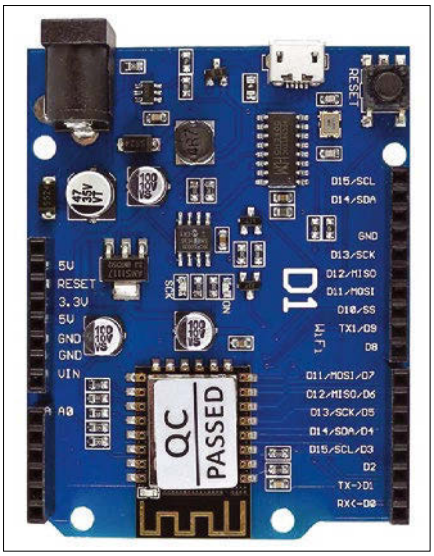


Figure 1: WeMos ESP8266 with an Arduino Uno form factor.

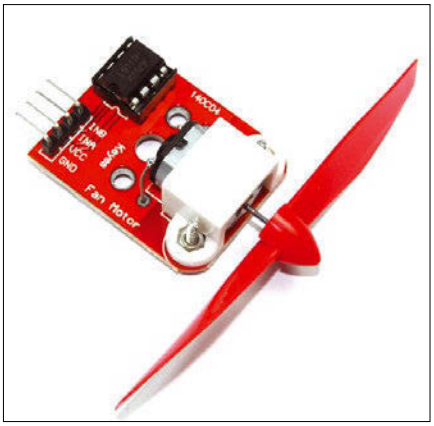


Figure 2: L9110 fans.

Lead Image © shamain, 123RF.com

Author

You can investigate more neat projects by Pete Metcalfe and his daughters at *https://funprojects.blog*.

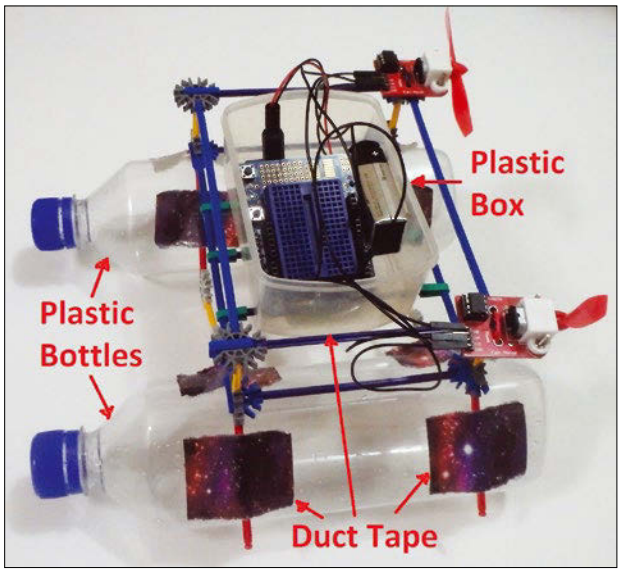


Figure 3: Airboat construction.

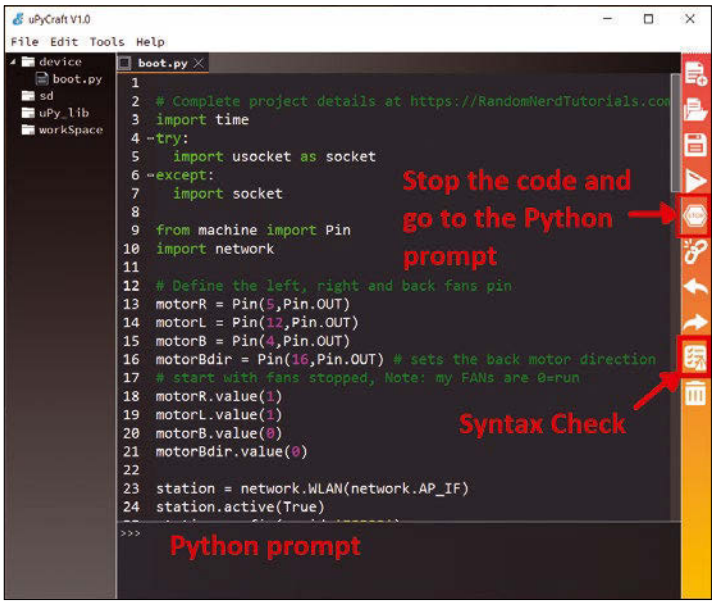


Figure 4: uPyCraft MicroPython IDE.

The L9110 fans (Figure 2) designed for Arduino projects only cost about $5. They have four pins: VCC, GND, INA (direction), and INB (on/off). For this project, I only used forward-spinning fans, so I only needed to use the INB pin.

Boat Construction

For the boat frame, I used K'Nex pieces because they are lightweight and sturdy (Figure 3); however, other construction kits could be used, as well. Water bottles provided flotation, and duct tape secured the bottles to the frame. To help protect the electronics, I used a small plastic snack container. The fans were wire-wrapped to the boat frame.

MicroPython Setup

Among the available MicroPython development environments, the uPyCraft IDE [2] that I used for this project is well-integrated (Figure 4) and has some excellent tutorials to help you get started.

Typically for MicroPython projects, you create two applications: `boot.py` and `main.py`. I like to equate this arrangement to Arduino, which has a `setup()` function (`boot.py` in Python) and a `loop()` function (`main.py` in Python). For this example, I wanted to keep the documentation simple, so I put everything in the `boot.py` file; however, a `main.py` file is recommended for larger projects.

Creating an Access Point

Figure 5 shows how to set up an access point in just a few lines of code. For this example, the access point is called *ESP32* with the password *12345678*. When the code runs, I will be able to see when a remote user connects and disconnects to the access point.

MicroPython Web Server

After getting the access point working, the next step is to create a web server. For this simple web server project, I embedded the HTML content into my Python code. For a more complex application, I would definitely have web pages that are files independent of the code.

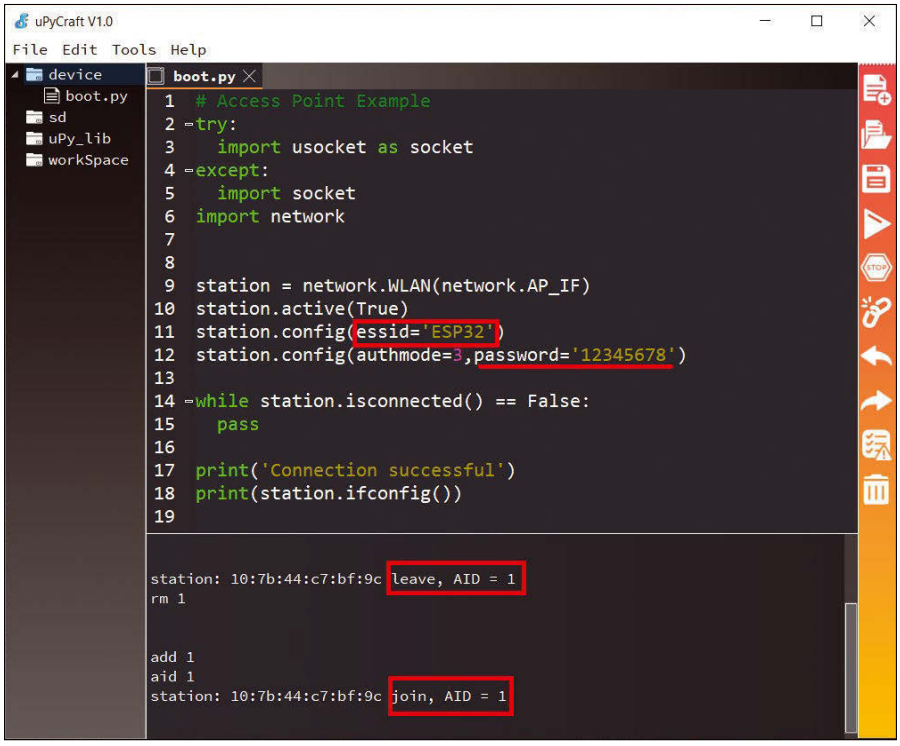


Figure 5: MicroPython access point.

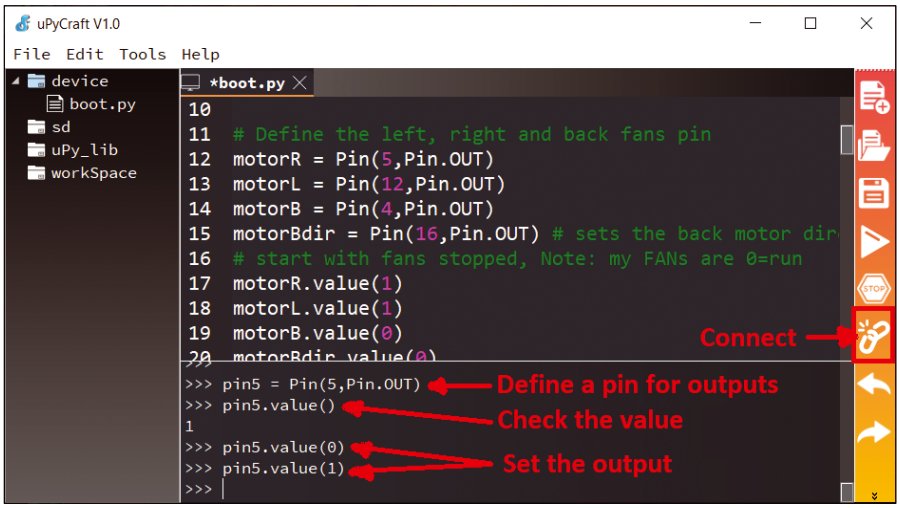


Figure 6: MicroPython I/O test on the command line.

Listing 1: Access Point Web Server

```
01 # MicroPython boot.py - Access Point Web Server
02
03 try:
04   import usocket as socket
05 except:
06   import socket
07
08 import network
09
10 station = network.WLAN(network.AP_IF)
11 station.active(True)
12 station.config(essid='ESP32')
13 station.config(authmode=3,password='12345678')
14
15 while station.isconnected() == False:
16   pass
17
18 print('Connection successful')
19 print(station.ifconfig())
20
21 # Function to read HTTP Request and Send Web Page
22 def web_page(request):
23
24   fans_state = ""
25   if request.find('/?forward') > 0:
26     fans_state="Going Forward"
27   if request.find('/?Stopped') > 0:
28     fans_state="Stopped"
29
30 # Web Page
31   html = """<html><head> <title>Boat Web Server</title>
32   <meta name="viewport" content="width=device-width, initial-scale=1">
33   <link rel="icon" href="data:,"> <style>
34   html{font-family: Helvetica; display:inline-block; margin: 0px auto;
         text-align: center;}
35   h1{color: #0F3376; padding: 2vh;}p{font-size: 1.5rem;}
36   .button{display: inline-block; background-color: #e7bd3b; border: none;
37   border-radius: 4px; color: white; text-decoration: none; font-size: 30px;
                  width:100%}
38   </style></head>
```

For the web server example application (Listing 1), I start with the access point connection code (lines 10-19) and then set up a socket on port 80 (lines 48-50). The HTTP request/response sequence is passed through a function called `web_page(request)` (line 22), which looks for keywords in the HREF request. The embedded HTML code passes keywords by anchor tags (e.g., `<a href='/?forward'>`; line 41).

Often, mobile frameworks like Bootstrap are used to help with formatting; however, because I'm running a standalone access point, I need to define all style codes manually.

The MicroPython default web server address is 192.168.4.1, which you can change in the access point setup. When the web server is running, I should be able to toggle the stop/forward states.

Writing Outputs

The MicroPython command-line interface (CLI) is a good way to test the hardware setup and outputs. To access it, press the *Connect* icon (Figure 6) and enter Ctrl + C.

The *machine* library lets you manage hardware pins:

```
>>from machine import Pin
```

Pin objects can be either inputs or outputs:

```
Pin14 = Pin(14, Pin.IN)
Pin5 = Pin(5, Pin.OUT)
```

The value of a pin is read by `pin<object>.value()` and is set by `pin<object>.value(<thevalue>)`.

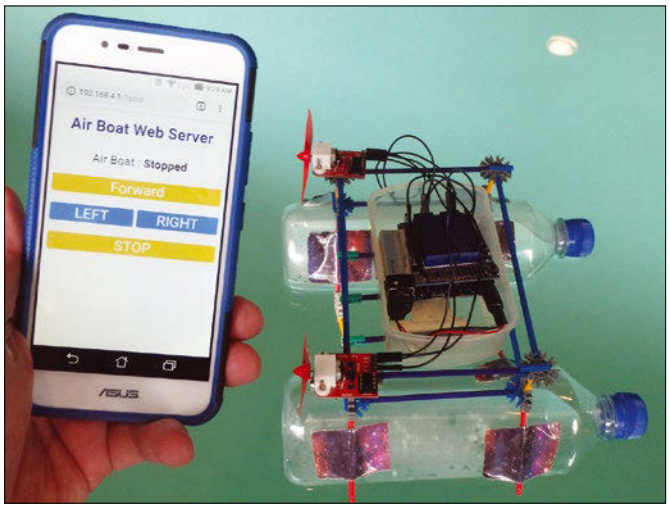


Figure 7: Airboat web controls.

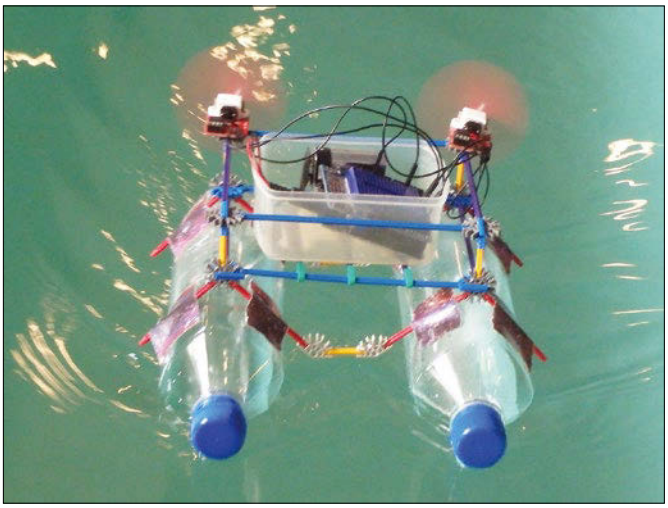


Figure 8: Airboat in action.

Final Application

Now that all the pieces are working independently, I can pull it all together. For the final code (Listing 2), I've defined two fans (`motorR` and `motorL`); a fan control function (`fancontrol(left,right)`, line 31) is called from the web requests (lines 35-49). Figure 7 shows the airboat

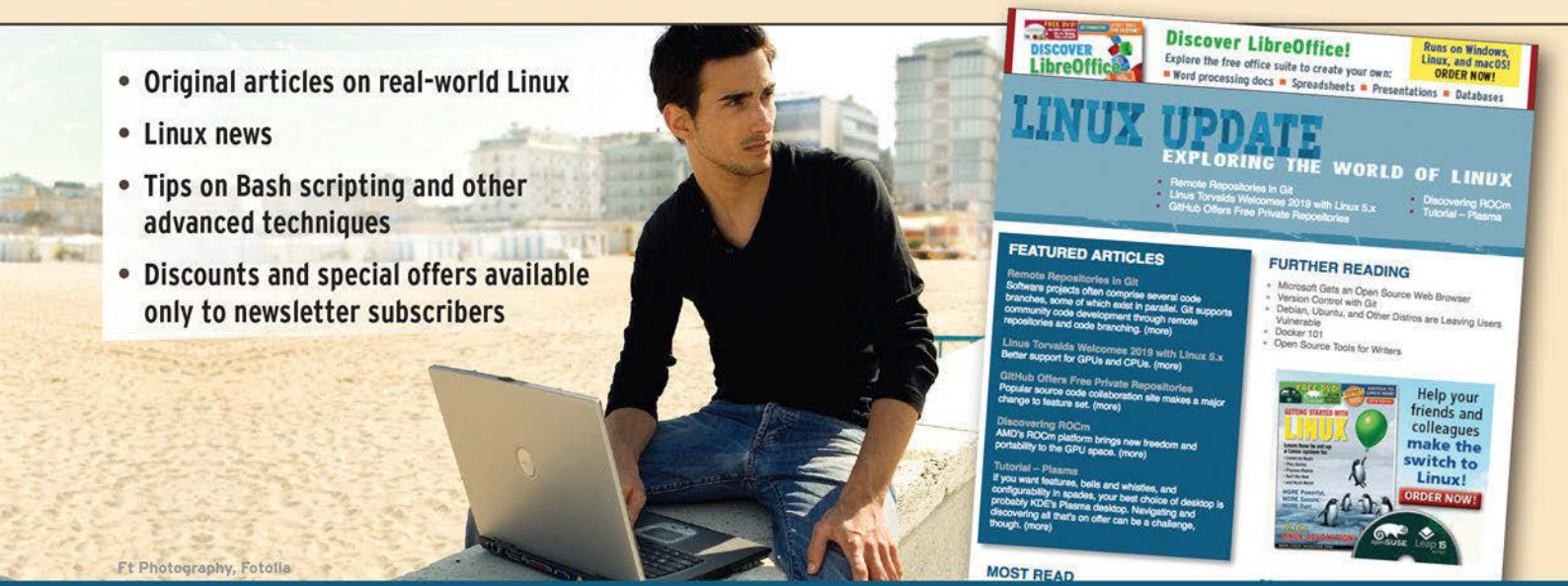
Listing 1: Access Point Web Server (continued)

```
39   <body> <h1>Boat Web Server</h1>
40   <p>Airboat : <strong>""" + fans_state + """</strong></p>
41   <p><a href='/?forward'>
     <button class="button">Forward</button></a></p>
42   <p><a href='/?stop'>
     <button class="button button">STOP</button></a></p>
43
44   </body></html>"""
45
46   return html
47
48 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
49 s.bind(('', 80))
50 s.listen(5)
51
52 while True:
53   conn, addr = s.accept()
54   print('Got a connection from %s' % str(addr))
55   request = conn.recv(1024)
56   request = str(request)
57   print('The Content = %s' % request)
58   response = web_page(request)
59   conn.send(response)
60   conn.close()
```

LINUX UPDATE

Need more Linux? Our free Linux Update newsletter delivers insightful articles and tech tips to your mailbox twice a month.

- Original articles on real-world Linux
- Linux news
- Tips on Bash scripting and other advanced techniques
- Discounts and special offers available only to newsletter subscribers



Ft Photography, Fotolia

https://bit.ly/Linux-Update

web server, and Figure 8 shows the airboat in action.

Summary

You'll be quite surprised how fast two fans will move the airboat. To ensure the boat runs straight forward, balancing the direction of the fans or adding a simple rudder might be necessary.

I've done the same project on the exact same hardware in Arduino C++. The response speeds seem similar, but the Python code might be slightly leaner. Although the MicroPython IDE wasn't as robust as the Arduino IDE, I enjoyed testing the Python code interactively from a command prompt.

I won't be giving up on Arduino C++, but I can definitely see a place for MicroPython, especially for projects with lots of string manipulation. ■■■

Info

[1] MicroPython: *https://micropython.org/*

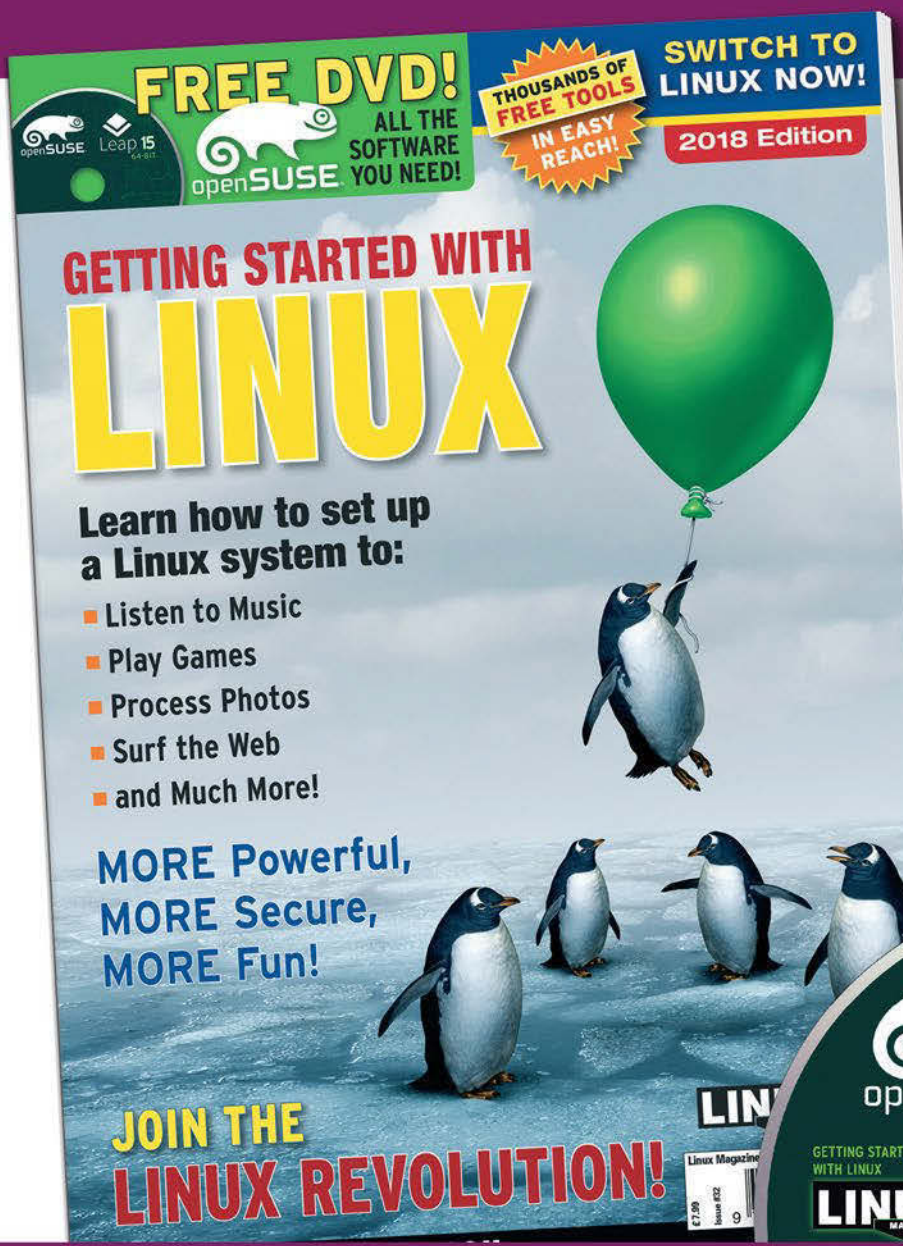
[2] uPyCraft: *https://github.com/DFRobot/uPyCraft*

Listing 2: Access Point and Web Controls

```
01 # MicroPython boot.py - Access Point and Airboat Web
     Controls
02 import time
03 try:
04   import usocket as socket
05 except:
06   import socket
07
08 from machine import Pin
09 import network
10
11 # Define the left, right, and back fan pins
12 motorR = Pin(12,Pin.OUT)
13 motorL = Pin(4,Pin.OUT)
14
15 # start with fans stopped, Note: my FANs are 0=run
16 motorR.value(1)
17 motorL.value(1)
18
19
20 station = network.WLAN(network.AP_IF)
21 station.active(True)
22 station.config(essid='ESP32')
23 station.config(authmode=3,password='12345678')
24
25 while station.isconnected() == False:
26   pass
27
28 print('Connection successful')
29 print(station.ifconfig())
30
31 def fancontrol(left,right):
32   motorL.value(left)
33   motorR.value(right)
34
35 def web_page(request):
36
37   fans_state = "Stopped"
38   if request.find('/?forward') > 0:
39     fans_state="Going Forward"
40     fancontrol(0,0)
41   if request.find('/?left') > 0:
42     fans_state="Going Left"
43     fancontrol(1,0)
44   if request.find('/?right') > 0:
45     fans_state="Going Right"
46     fancontrol(0,1)
47   if request.find('/?stop') > 0:
48     fans_state="Stopped"
49     fancontrol(1,1)
50
51   html = """<html><head> <title>Airboat Web Server</title>
52   <meta name="viewport" content="width=device-width,
     initial-scale=1">
53   <link rel="icon" href="data:,"> <style>
54   html{font-family: Helvetica; display:inline-block;
     margin: 0px auto; text-align: center;}
55   h1{color: #0F3376; padding: 2vh;}p{font-size: 1.5rem;}
56   .button{display: inline-block; background-color:
     #e7bd3b; border: none;
57   border-radius: 4px; color: white; text-decoration: none;
     font-size: 30px; width:100%}
58   .button2{background-color: #4286f4; width:49%}
59   </style></head>
60   <body> <h1>Airboat Web Server</h1>
61   <p>Airboat : <strong>""" + fans_state + """</strong></p>
62   <p><a href='/?forward'>
     <button class="button">Forward</button></a></p>
63   <p><a href='/?left'>
     <button class="button button2">LEFT</button></a>
64   <a href='/?right'>
     <button class="button button2" >RIGHT</button></a></p>
65   <p><a href='/?stop'>
     <button class="button button">STOP</button></a></p>
66
67   </body></html>"""
68
69   return html
70
71 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
72 s.bind(('', 80))
73 s.listen(5)
74
75 while True:
76   conn, addr = s.accept()
77   print('Got a connection from %s' % str(addr))
78   request = conn.recv(1024)
79   request = str(request)
80   print('The Content = %s' % request)
81   response = web_page(request)
82   conn.send(response)
83   conn.close()
```

Want your friends and colleagues to make the switch to Linux?

This single issue shows beginners how to:



- Install Linux
- Download and install free software for your Linux system
- Create documents and spreadsheets
- Play games
- Surf the web
- Process photos
- Play music and videos
- and much more!

HELP OTHERS
JOIN THE LINUX REVOLUTION!

ORDER ONLINE:
shop.linuxnewmedia.com/specials

MakerSpace

Choosing the right board

Decisions, Decisions

When it comes to open hardware projects, the choice of an Arduino or a Raspberry Pi board can determine success or failure. Read on for guidance in selecting the best board for your specific needs. *By Bruce Byfield*

Open hardware would not be where it is today without Arduino and Raspberry Pi boards. Compact, powerful, and mostly open source, both Arduinos and Rasp Pis offer a level of functionality in a small footprint that simply did not exist a decade ago. To casual observers, the two may seem interchangeable, but when the time comes to choose which to develop a project around, your choice can make the difference between success and failure. While some overlap exists, at the very least matching the hardware to the project and your intentions can determine how complicated your device can become.

Many do-it-yourselfers make this choice on the advice of friends – which is fine, so long as their friends' knowledge is as extensive as it sounds. The problem is, people tend to advocate what they know. For this reason, researching the differences between Arduinos and Rasp Pis is generally the safer and most efficient approach. What follows is a general overview of the capacity of both and the purposes for which they are most suited.

SBCs vs. Microcontrollers

While both Arduinos and Rasp Pis are sometimes described as nano computers, only Rasp Pis are computers in the same sense as a laptop or a workstation. Technically speaking, Rasp Pis are single-board computers (SBCs) [1], offering basic computer functionality on a single circuit board. Just what that functionality is has shifted since the first Rasp Pi was released in 2009 and still depends partly on the model. Currently, the fastest Rasp Pi has a 1.4GHz processor, slightly less than that of entry-level Chromebooks, but many times faster than any Arduino. Depending on the model, a Rasp Pi's peripherals can include an Ethernet port, aluminum heat sinks, and LAN and Bluetooth support. Memory is either on-board or provided by an online microSD card. Additional functionality can be added using Hardware Attached on Top (HATs) [2] for purposes like high-resolution displays and sound cards – a development that technically plays fast and loose with the definition of an SBC.

On the other hand, Arduino boards lack the versatility of a full computer. Instead, they are microcontrollers [3] with limited functionality. Rather than an operating system that runs an application, an Arduino runs a limited set of instructions in its firmware. The absence of an operating system reduces the resources

Lead Image © donatas1205, 123RF.com

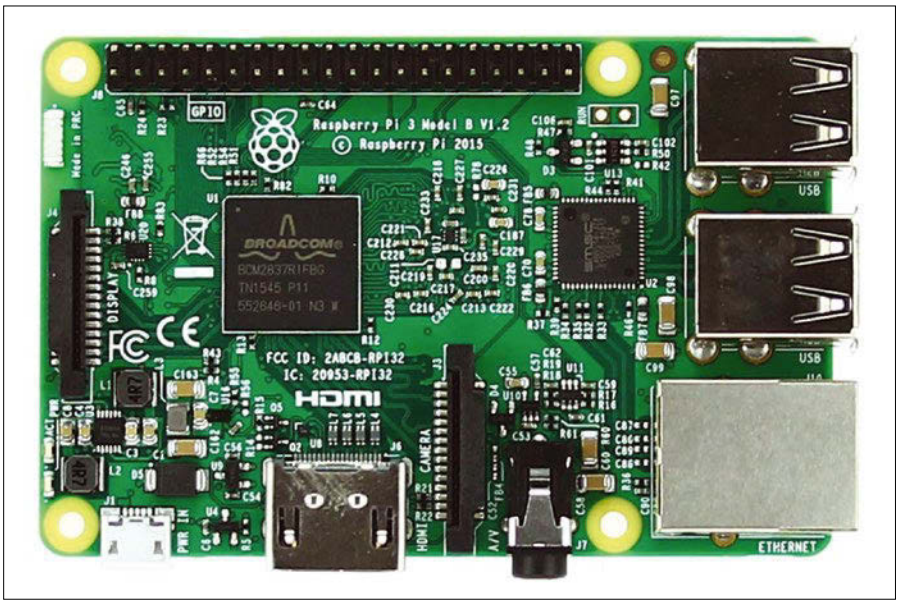


Figure 1: Along with the RPi3 Model A, the RPi3 Model B is the fastest and most fully-featured Rasp Pi.

to run an Arduino and usually reduces boot time, as well as simplifying firmware modification. Just like Rasp Pis with HATs, Arduinos can extend their functionality with shields [4] – peripherals like sound cards and sensors – although generally they have far fewer features than any Rasp Pi.

These distinctions are murkier than they were a decade ago, but still remain broadly true. You might just manage to give an Arduino the functionality of a Rasp Pi, but the main reason to do so would be just to prove you could. After all, the whole point of both Rasp Pis and Arduinos is to provide off-the-shelf components.

Models

Both Rasp Pis and Arduinos have been around long enough to have a number of different models, almost all of which are under $60 for the basic board. However, you may want to buy a starter kit with some basic instructions and cables or other hardware, which can easily double the price. Alternatively, because both are strongly oriented to education and do-it-yourself applications, you can find no shortage of instructions online. You can also buy hardware separately from online stores such as Adafruit [5].

You can study the Rasp Pi specifications online [6]. Here, it is worth noting that the Raspberry Pi 1 (RPi1) line has been a standard with various updates for seven years. It is suitable for many basic operations. The Raspberry Pi 3 (RPi3) line offers the most advanced features, including 64-bit computing, a 1.2GHz quad-core processor, WiFi, and USB boot capabilities (Figure 1). Another possible option is the Pi Zero, which is smaller than other Rasp Pis physically and has reduced input/output, plus the Pi Zero W has the ability to add HATs without soldering (Figure 2).

Arduinos come in over two dozen models [7], with several new ones added in most years. One or two are distributed only in Europe, and some of the more popular models may be sold out at any given time. Although a thorough description of the available Arduinos would be a magazine issue in itself, hobbyists and embedded developers might start by looking at the Arduino Zero (Figure 3), which seems to have replaced the UNO as the entry-level Arduino [8]. A more powerful alternative for open hardware ranging from keyboards to prosthetics is the Arduino Mega 2560 [9] (Figure 4). However, once you have chosen to use a microcontroller, spend a few hours browsing the descriptions in the Arduino online store to find which board best suits your needs.

With both Rasp Pis and Arduinos, you can also find third-party clones at a somewhat cheaper price. However, many of these clones are proprietary and require drivers to use. Moreover, while I have never used a clone, rumors of quality problems are common when discussing some clone brands. Unless you have a compelling technical reason to use a clone, I suspect that avoiding them would be sensible.

Licensing

In many ways, the Raspberry Pi Foundation acts like an open source project. The Foundation contributes to several open source projects, including the kernel, and uses free licenses for most of its own software. In addition, Raspbian, its main operating system, is a derivative of Debian, and most of its other supported operating systems are based on Linux distributions as well. However, its CPU's firmware is proprietary and developed by Broadcom.

In comparison, Arduino describes itself as "an open source electronics platform" [10]. It uses version 3.0 of the Creative Commons Attribution ShareAlike (CC BY-AS) license [11]. Under this license, designs and docu-

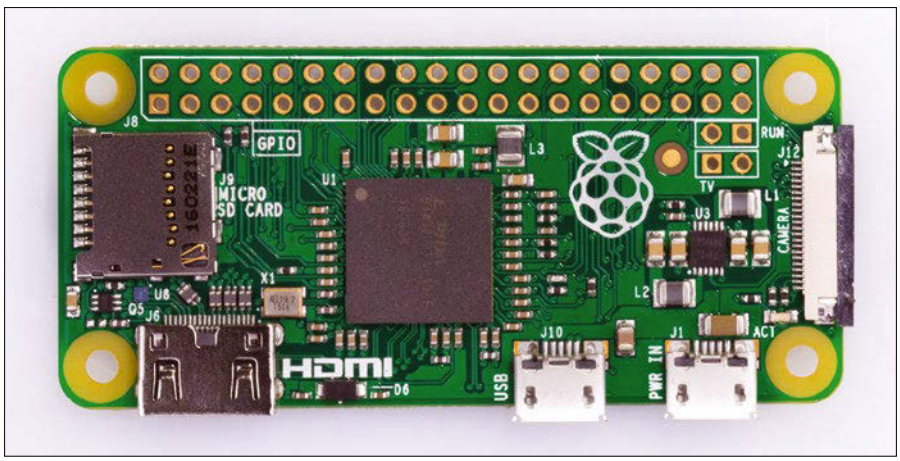


Figure 2: The Pi Zero is the smallest and most stripped down Rasp Pi.

Figure 3: The Arduino Zero is a popular entry-level Arduino.

mentation derived from Arduino properties must credit Arduino and use the same license.

Making a Choice

On the one hand, anything you can do with an Arduino board should be possible with a Rasp Pi. However, in many cases, a Rasp Pi would be a waste of resources. Using a Rasp Pi can also cause unnecessary complications, such as the need for a continuous power supply and a much longer boot time. On the other hand, while you might be able to use several Arduinos to do some procedures more suitable to a single Rasp Pi, that is likely to be cumbersome and inefficient. Either of these extremes should be avoided whenever possible, if only because they both increase the possibility of something going wrong.

Sometimes, the choice may be simply a matter of the features you want.

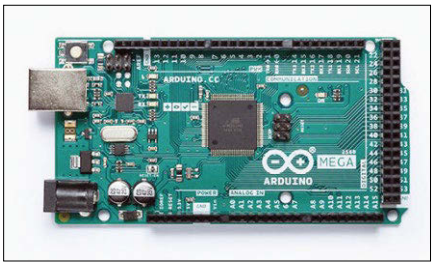


Figure 4: The Arduino Mega 2560 is often the choice for commercial open hardware devices.

For example, if you need an Ethernet port, a Rasp Pi is probably a more straightforward choice. Similarly, if you want users to be able to customize by flashing firmware, an Arduino is designed to make that easy. At other times, your own expertise may be a deciding influence. For example, if you are familiar with standard programming languages, a Rasp Pi may be a better environment for you, while if your programming is limited mostly to scripts or your expertise lies in electronics, you might find Arduino C more within your competence.

However, most of the time, the most important factor is the intended purpose. An Arduino is a specialty device, designed to do a few simple tasks over and over. It makes simple logical choices, such as taking a sensor reading, either for the sake of the reading or as a signal to start or stop another operation. At other times, it defines pieces of hardware, such as with the Keyboardio Model 01, which uses Arduinos to assign keys on a keyboard, or defines a piece of hardware's purpose. Once you have debugged the firmware, an Arduino can generally be counted on to run on its own, or in the background, like a printer that waits for input to begin functioning. If you are unsure whether an Arduino is suitable for a project, do an Internet search for "what you can do with an Arduino" and see if any of the uses mentioned in the results resembles yours.

By contrast, consider a Rasp Pi if you want any of the functionality of a personal computer – anything from a graphical display or an input device to Internet access, multitasking, or in-depth calculations. Compared to other computers, a Rasp Pi is a relatively low-powered device, but it is still much more versatile than an Arduino. Doing a search for "what you can do with a Raspberry Pi" may help to clarify your choice.

Perhaps the best selection criteria appeared a few years ago in *Make:* magazine: "Think about what you want your project to do. If you can describe it with less than two 'and's, get an Arduino. If you need more than two 'and's, get a Raspberry Pi" [12]. Regardless of your choice, you will need to further limit your final decision by deciding which model Arduino or Rasp Pi you need, and whether you will need any hardware add-ons, but at least you will know where to begin. ■■■

Info

[1] SBCs: *https://en.wikipedia.org/wiki/Single-board_computer*

[2] HATs: *https://www.raspberrypi.org/blog/introducing-raspberry-pi-hats/*

[3] Microcontroller: *https://en.wikipedia.org/wiki/Microcontroller*

[4] Shields: *https://www.progressiveautomations.com/collections/shields-and-sensors*

[5] Adafruit: *https://www.adafruit.com/*

[6] Rasp Pi models: *https://en.wikipedia.org/wiki/Raspberry_Pi#Generations_of_released_models*

[7] Arduino boards: *https://store.arduino.cc/usa/arduino/boards-modules*

[8] Arduino Zero: *https://store.arduino.cc/usa/arduino-zero*

[9] Arduino Mega 2560: *https://store.arduino.cc/usa/arduino-mega-2560-rev3*

[10] Arduino licensing: *https://www.arduino.cc/en/Guide/Introduction*

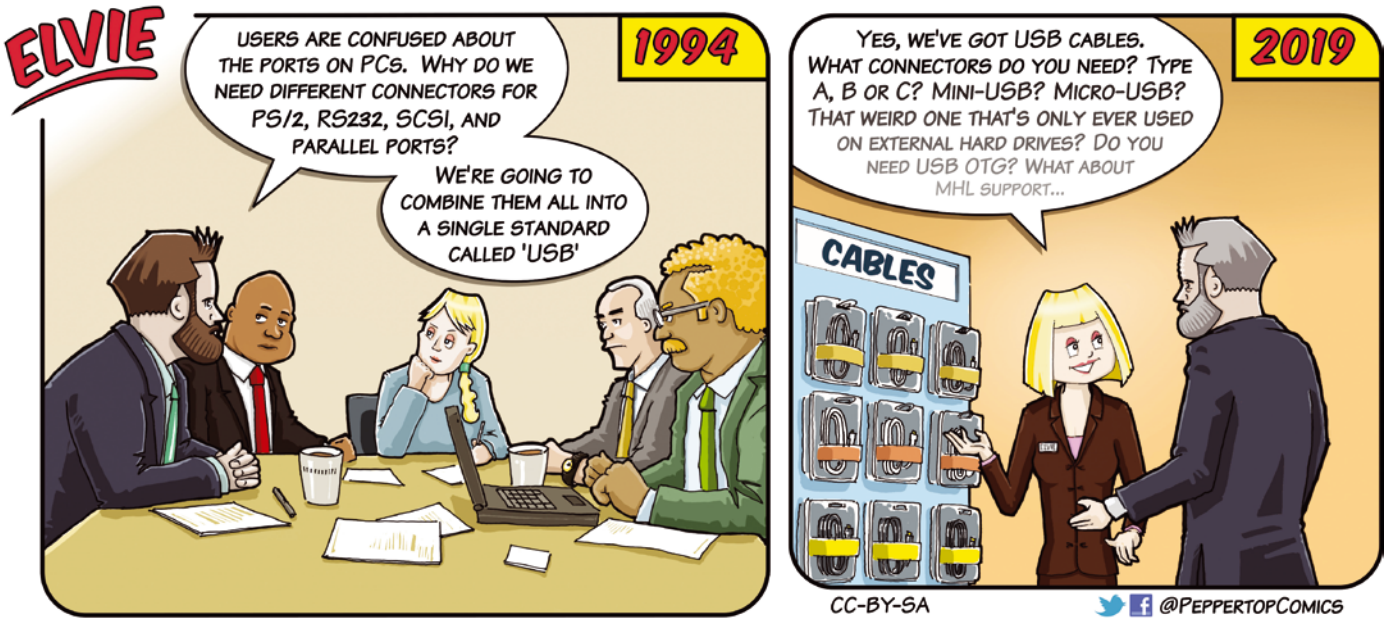
[11] CC BY-SA 3.0: *https://creativecommons.org/licenses/by-sa/3.0/*

[12] A simple guide: *https://makezine.com/2015/12/04/admittedly-simplistic-guide-raspberry-pi-vs-arduino/*

In the old days, computer users had to train their brains to think like a computer. A new generation of computer tools is dedicated to organizing and formatting information in a way that is most useful to users. This month, we show you how to collect your thoughts with a pair of useful outline tools: CherryTree and Piggydb. We also continue our series on Bash scripting with a look at test conditions and exit codes, and we show you how to build a 3D model with OpenSCAD.

Image © Olexandr Moroz, 123RF.com

LINUXVOICE



Jon "maddog" Hall is an author, educator, computer scientist, and free software pioneer who has been a passionate advocate for Linux since 1994 when he first met Linus Torvalds and facilitated the port of Linux to a 64-bit system. He serves as president of Linux International®.

MADDOG'S DOGHOUSE

After 25 years of waiting for open hardware, Maddog predicts the era of closed, proprietary chips as the only option is drawing to an end. BY JON "MADDOG" HALL

A promise of open hardware

For close to 25 years, I have been dealing with trying to develop and maintain computer systems over long periods of time. In dealing with the longevity issues, I have been suffering through the use of closed source firmware inherent in GPUs, BIOS, WiFi, Bluetooth, etc. in particular, and closed source in general.

For the first 20 years, I was very patient. I realized that companies had business plans, and they did not know or trust Free and Open Source Software and Hardware (FOSSH) business plans. Some had contracts that they had signed as they bought and sold intellectual property (IP). Some companies could not expose the sources of the software they bought because of the contracts they had signed with the software or hardware providers. Who could possibly have the expertise to maintain the software provided with the hardware or produce software better than the manufacturer, right? Read that last sentence with sarcasm turned fully on.

In the late 1980s, Digital Equipment Corporation (DEC) had a world-class X Window System server that worked with "dumb frame buffers." The engineer who wrote the code refused to support shipping the source code due to the IP that was inherent in the X server. DEC was losing millions of dollars of sales, because we did not make our X server code available.

I flew to California from the East Coast to talk with the engineer. I asked him to go over each and every part that he felt was unique and tell me if any of our competitors (who all allowed customers access to their X servers) had that particular IP. It finally turned out that the only IP that was not in our competitor's X servers was a routine that was hardly used by anyone. When asked if this little-used piece of code was worth losing millions of dollars in sales, the engineer agreed to ship the source code.

Over the past five years, I have been working with a few hardware projects such as Caninos Loucos [1]. I have also been looking at lots of single-board computers (SBCs) from various organizations. I have a large plastic bin full of them, most of them using ARM architecture, while some use Intel or AMD. Some of the companies that made these SBCs are out of business. Some are still in business, but they have moved on from creating updates to their drivers that would allow these boards to work with more modern versions of the kernels.

These SBCs will become useless over time as the various kernels are updated with new features, new security fixes, and new device support, simply because the source code for these pieces of firmware are not available. Personally, this is an annoyance. However, for people who bought these systems hoping that they would be useful for a reasonable period of time, it could mean huge losses of money, because they were not buying the units just for evaluation (as I did); they wanted to use them in education, or manufacturing, and therefore purchased hundreds or thousands of them.

One small company I know sold 60,000 of their units to various people with version 3.x of the Linux kernel running on them. Now, about a year later, version 4.x is the mainstream kernel, and the binary device drivers that came with the 3.x-based system are close to useless. This includes the firmware for the GPU, the WiFi driver, and the USB driver, as well as some of the bootloader.

Most of these hardware devices are still viable for use, but because the devices themselves are "end of life," the manufacturers are not interested in upgrading the binary drivers for the newer kernel. Even if they were, the same issue would probably come up as the hardware moves through newer versions of the kernels.

On the other hand, the vendors could "open up" the hardware interfaces of the different devices by telling FOSS device driver writers how the registers and chips actually work. The FOSS community could maintain these components as long as the community desires.

The computer industry has changed over time. Significantly powerful, cheap computers allowed groups like the Linux kernel developers in the 1990s to develop a full-fledged kernel without having to buy a multi-hundred-thousand-dollar computer to do the development.

Today there are groups of people, such as the University of California's RISC-V project, who are designing and manufacturing full-scale open CPUs, with other groups developing GPUs and other sophisticated processing units, producing them in open FABs.

These groups understand that opening up their hardware interfaces to FOSSH programmers will allow a greater number of sales. Users of GPUs, WiFi chips, USB devices, and SBCs will now stop using closed, proprietary chips in their products in lieu of ones that can be supported properly.

This is not a threat, but a promise. ■■■

Info

[1] Caninos Loucos: *http://caninosloucos.org/en/*

What?!

I can get my issues SOONER?

Available anywhere, anytime!

Sign up for a digital subscription and enjoy the latest articles on trending topics, reviews, cool projects and more…

shop.linuxnewmedia.com/digisub

Comparing the CherryTree and Piggydb outline editors

Sort Your Thoughts

Organize your thoughts with the CherryTree desktop app and an innovative web application called Piggydb.

BY PETER KREUßEL

Anyone who has ever tried to organize complicated topics in a meaningful way understands that the information rarely maps to a simple structure. Ideas overlap, and concepts that fall in one area might easily reappear somewhere else. Outline editors help you unravel this kind of complex information. The outline tools populating the Linux landscape come in several different forms. We decided to compare a couple of popular options: CherryTree and Piggydb. These tools illustrate different approaches to the time-honored task of organizing human thought: CherryTree uses tree structures, while Piggydb relies on mesh graphs.

CherryTree

CherryTree [1] identifies each node in the topic tree (Figure 1, left panel) with a cherry symbol and assigns to it a text file that supports simple formatting, images, and tables. External files can also be embedded.

The software packages the whole thing into a SQLite database, with the option of encryption. Alternatively, CherryTree can export to XML, which is only useful for speed if you need access outside of CherryTree.

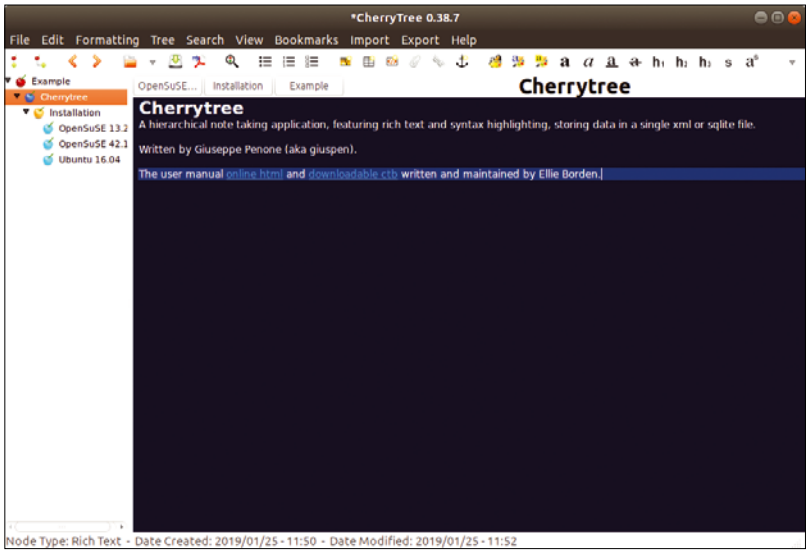


Figure 1: CherryTree assigns a note to each symbol in the "cherry tree" on the left.

The program features practical keyboard shortcuts: The Alt+Left Arrow and Alt+Right Arrow shortcuts let you navigate the tree, while the normal arrow keys move the cursor through the text in the usual way. Ctrl+N inserts a node at the same level as the currently selected node; Shift+Ctrl+N inserts a child node.

Shift+Up Arrow and Shift+Down Arrow let you move the node in the tree; Shift+Right Arrow and Shift+Left Arrow change its hierarchy level. Shift+Ctrl+Right Arrow opens a dialog in which you can use the arrow keys to select a new higher-level node for the active element. CherryTree also sorts nodes alphabetically on request. To prevent accidental changes, you can assign write protection to important nodes.

The notes assigned to the nodes support simple formatting such as bold (Ctrl+B), italic (Ctrl+I), underline (Ctrl+U), and strikethrough (Ctrl+E). The Format menu lists all possible formats and their matching keyboard shortcuts. The dark blue text background can be easily changed to *Light background, dark text* under *Settings | Rich Text | Theme*.

In addition to embedded images and tables, CherryTree supports enumerations and bullet lists, as well as *Task lists* with checkboxes. Links can point to web pages, external files, directories, or other nodes in the CherryTree document. If you insert an *Anchor* in the text, it can be referenced by a link like in HTML code and thus point to a specific place in a CherryTree note.

In addition to the tree structure's fundamental ordering principle, CherryTree also provides a search function that searches either the text of all or certain nodes, as well as the node names. The search can handle regular expressions and, if desired, will also filter by node change or creation date.

CherryTree lets you import numerous file formats. In addition to text and HTML, they also include formats from other notebook programs such as Tomboy, RedNotebook, or TuxCards. The program exports the complete tree structure or parts of it as text, PDF, or HTML. In addition, partial

branches can be exported as CherryTree files and inserted into another CherryTree document at any point in the outline.

Piggydb

Piggydb [2] as a Java program does not need any installation in the typical sense. You just need to unpack the `piggydb-standalone-7.0.zip` [3] zip archive for the desktop version. Then launch the program by typing:

```
java -jar piggydb-standalone.jar
```

In the standalone version, an icon appears in the taskbar; it can be used to exit the program. For the server variant, `piggydb-all-in-one-7.0.zip`, this does not exist. A script named `run.sh` is used to start the program here; more details can be found in the `README.txt` file and in Piggydb's documentation [4].

After starting Piggydb for the first time and logging in as *owner* (which also happens to be the default password), click on the green plus button in the titlebar (Figure 2, item 4) to open the Fragment Editor to create a new text fragment (Figure 3). For already existing fragments, you will also see a plus button below the text content (Figure 2, item 4), which creates index cards linked to this fragment.

Piggydb has only one element type: the text fragment. You can add outline items such as previously composed text by filling in either the *Title* field or the larger main text field (Figure 3). Because the Piggydb author believes that using predefined outlines has a negative effect on creativity, he recommends that you first create short, unlinked text fragments in a brainstorming phase and then integrate them at a later stage.

Piggydb Editor

In the Piggydb editor, you can format the text with wiki code, like used by Wikipedia (Figure 3): `'''bold'''`, `''italic''`, and `__strikethrough__` provide highlighting; `-Element` and `+Element` create enumerations and bullet lists. Piggydb also supports blocks in typewriter font, tables, and embedded external files. You can find out more by clicking on the blue question mark button in the editor.

Formatting switches let you skip the wiki formatting. If you mouse over the switches, you will

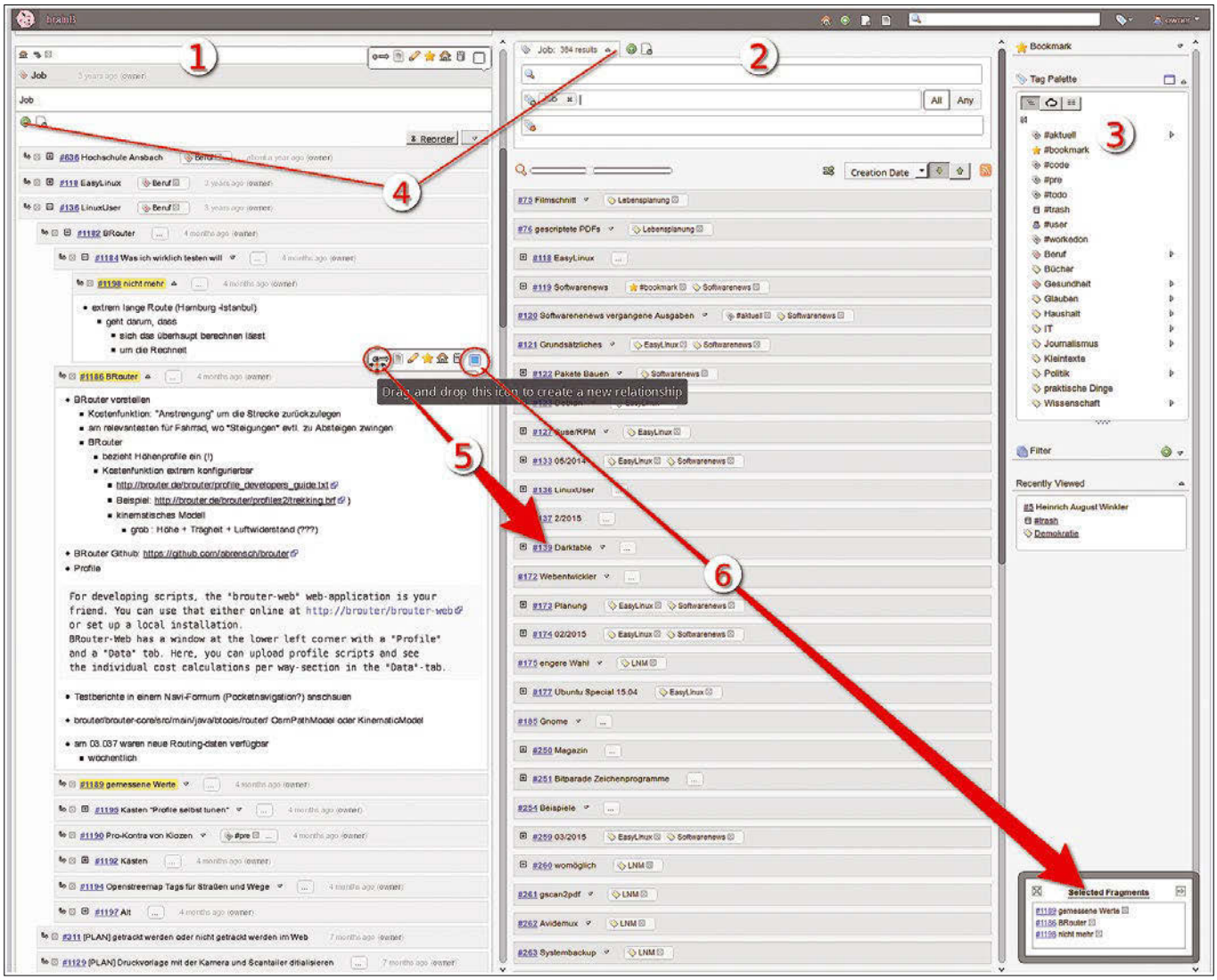


Figure 2: 1. Piggydb shows linked tabs in a hierarchical view. 2. The middle column contains the search result list. 3. Keywording is also available. 4. Click on the green plus button to open the Fragment Editor. 5. Create links by dragging the corresponding symbol onto other index cards. 6. You can also create links by selecting multiple elements.

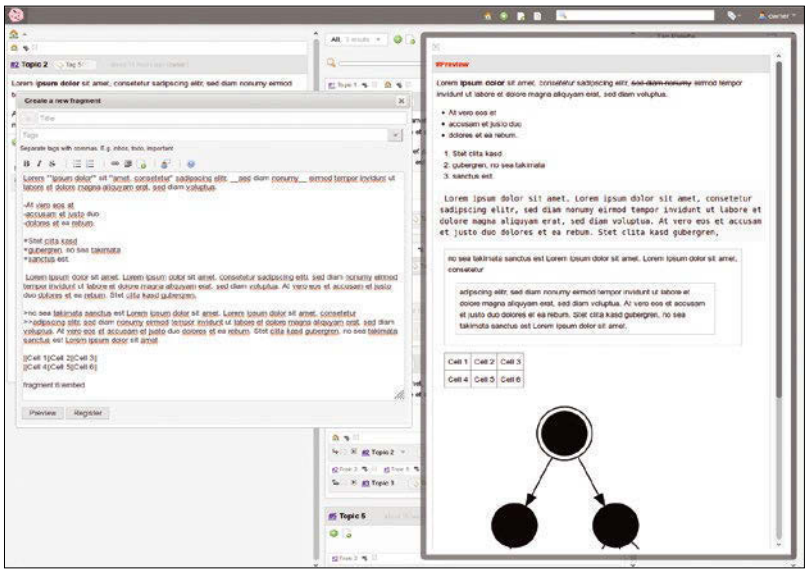


Figure 3: In the Fragment Editor, you can format the text with wiki code. A click on the *Preview* button shows the formatted result (right).

see the corresponding keyboard shortcuts. You can edit existing fragments by mousing over the gray titlebar and clicking on the pencil icon in the pop-up that appears.

A click on the *Preview* button shows the results of the wiki formatting on the right. The tool shows your images as thumbnails and other embedded documents as icons that reflect the file type. A click on *Register* saves the document.

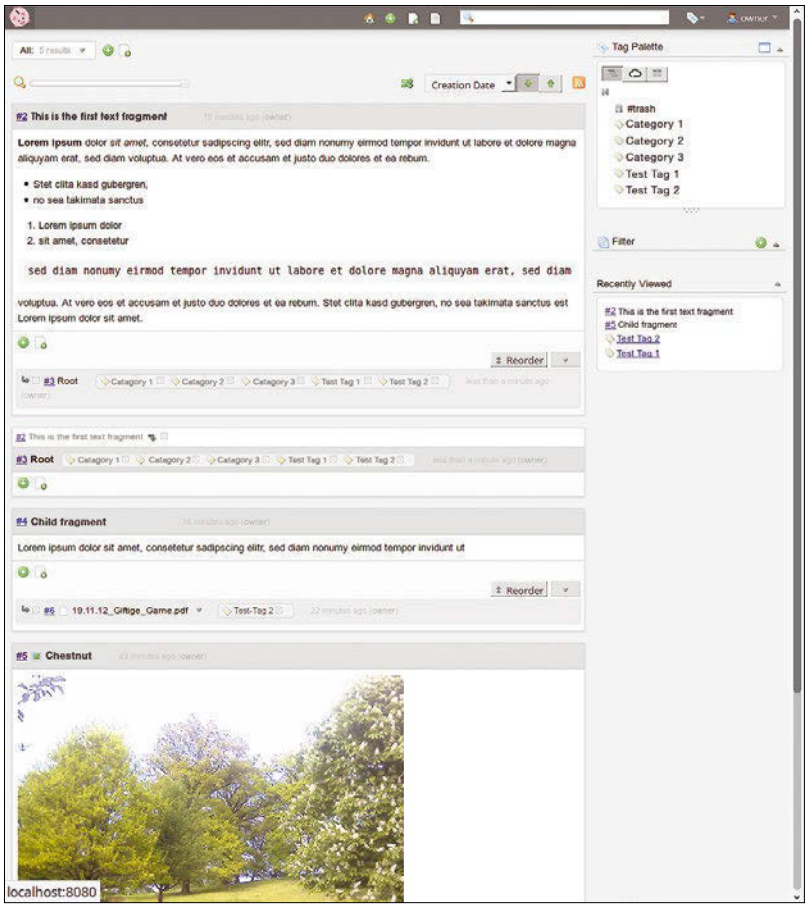


Figure 4: The new Piggydb database after creating some index cards: Since none of the elements has been marked yet as a root, the left column with the overall outline view is missing.

Links

To link fragments, mouse over a fragment's gray header bar and drag the plus and arrow symbol (far left) to another fragment's header (Figure 2, item 5). By default, the dragged element now becomes the parent element. In the pop-up that appears after dragging and dropping, however, the link direction can be reversed.

So far, Piggydb shows you the previously created entries in a two-column view on the left and a keyword palette and a list of the last fragments you visited (Figure 4) on the right. This changes when you click on one of the blue hashtag links (e.g., *#3*) to open one of the fragments. The left column then shows you the selected fragment including all its sub-fragments. From the second level onwards, the fragments can be expanded using the small plus icons to the left of the fragment titles (Figure 5, item 1).

In a middle column below the item detail view, you can still find the list of the newest items. However, if the open fragment contains tags (Figure 5, item 1), only those that also contain at least one of the tags of the opened element (Figure 5, item 2) will appear, rather than all recently changed fragments.

The middle column is actually the search function's result list. You can change the search filter by clicking on the down arrow to the right of the hit list. Three search fields are then unfolded. The top one is used for a keyword search. The middle one limits the search to tags – *All* and *Any* give you AND or OR link options. The lower field excludes fragments with specific tags.

If you drag the detail slider located below the hit list to the far right, 10 fully expanded index cards appear below. Move it all the way to the left, and you'll see 200 hits instead. There are several intermediate steps between these extremes. If you mouse over the title of an element that is not completely visible, a detailed preview appears in the window's lower right corner.

A drop-down menu with function buttons, which appears as soon as you mouse over an element's top bar, allows you to select as many fragments as you like using the checkbox on the far right (Figure 2, item 6); the fragments then appear in the Selected Fragments sub-window. You can link them all at once by dragging another element's *Create a new relationship* icon onto this sub-window.

The left outline column appears as soon as you mark one or more index cards in the button drop-down menu using the *#home* tag (by clicking the home button, Figure 6).

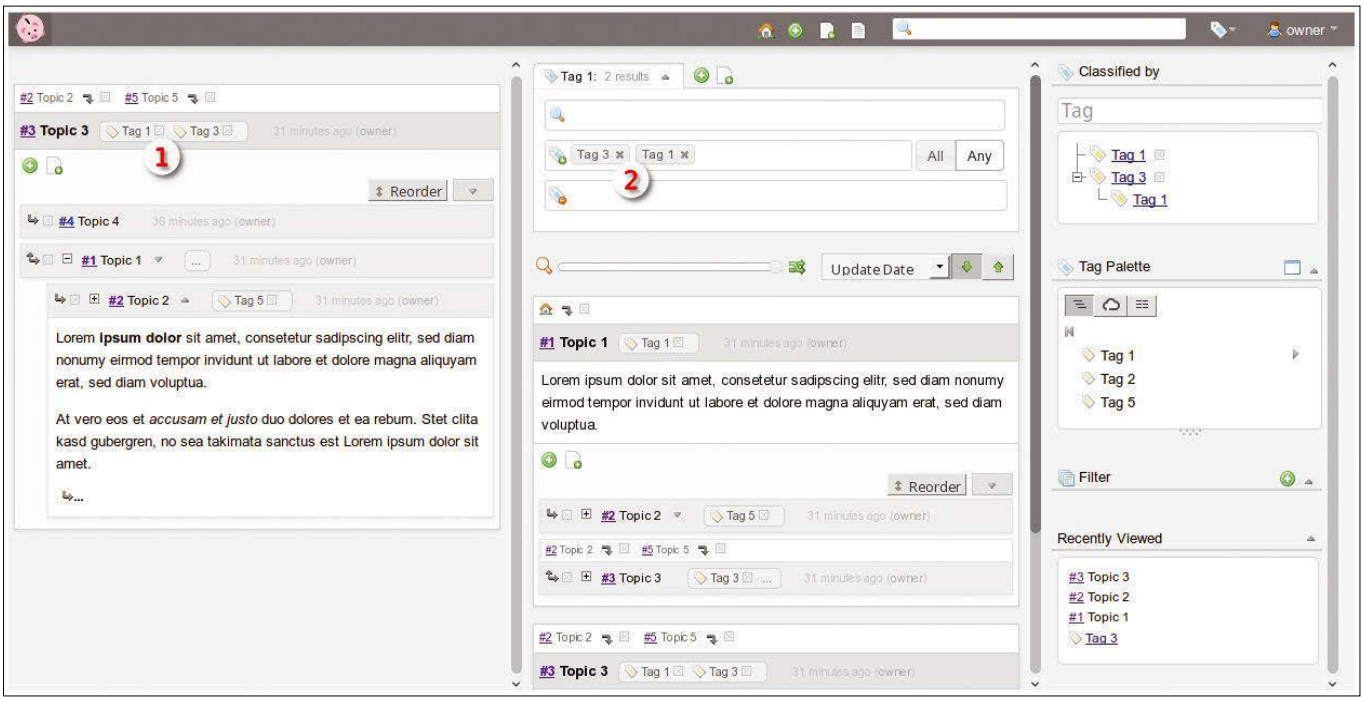


Figure 5: Clicking on the blue hashtag link (***#3***) opens the corresponding element as the root in the left column. In addition to the text it contains, the symbol for a linked parent element and all subordinate index cards are located at the top.

Piggydb: Mesh not Tree

At first glance, like CherryTree, Piggydb seems to use the linear-hierarchical outline principle (Figure 2, item 1). However, this is only true for the view level.

In Piggydb, each node can be linked to any other node. The network structure is not subject to the restrictions of a hierarchical tree structure (Figure 7, left). In a tree, each node is assigned to exactly one parent element; the branches only spread outwards from the trunk, and the software does not allow back references.

Unrestricted meshing (Figure 7, right) is helpful when structuring a knowledge base [5]. Not every index card can always be clearly assigned to a single topic (Figure 8). It is often more appropriate to place the card in several places at the same time.

As an example, imagine creating a Piggydb outline for the necessary components to assemble a PC. First, you create a file card for each component as a child element of *Market overview* and then structure it by hardware type. At some point, you create a *Parts list* fragment and drag the desired component into it. The elements combined to form the *Parts list* still belong to the *Market overview*. As soon as you discover new information about a component, you can edit the corresponding index card in one of the two sections – the other is automatically synchronized with it.

If you mouse over a file card's header, a pop-up with function buttons appears. Click

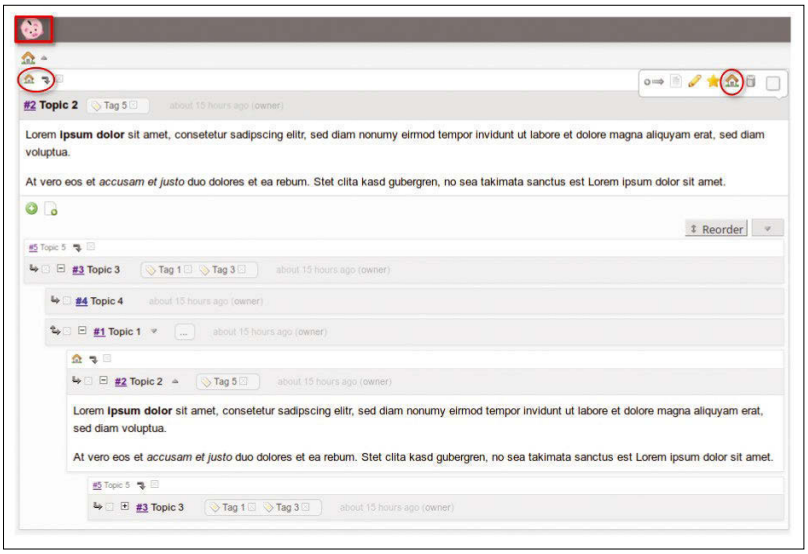


Figure 6: In contrast to linear outlines, Piggydb does not have an "Item 1" where the outline begins. Therefore, you can select any number of root elements using the ***#home*** tag; Piggydb then displays them on the main page with all linked elements as subordinate drop-down index cards.

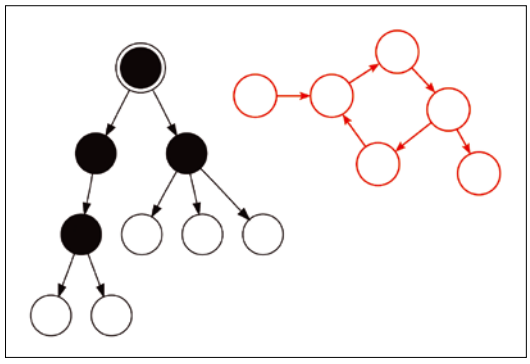


Figure 7: Tree graphs (black) branch away from the root, while mesh graphs (red) allow an arbitrary relationship of links.

on the document symbol (second from left) to open a static text page in a new web browser tab with the index card titles appearing as main headings (Figure 9). At the top of the page, you will also see a list of the index card titles that you can copy to a word processer for a printable parts list.

Tagged

In addition to element links, Piggydb supports keywords (tags) as a secondary ordering principle. However, Piggydb goes beyond the usual range of functions by allowing tags to be linked hierarchically. Subordinate tags are transitive. For example, let's imagine you define a tag named *Computer hardware* and assign to it the subordinate tags *Graphics cards*, *CPUs*, and *Motherboards*. If you now select a fragment with *Graphics cards*, it automatically belongs to the higher-level category *Computer hardware*.

The Tag Palette on the right in the Piggydb window (Figure 10) initially displays only the first keyword level. Given an intelligent hierarchical tag arrangement, everything remains clear even if you have a large number of keywords. Simply clicking on the arrow on the right shows you a tag's subordinate categories. The arrow above and directly to the left of the tags takes you back to the higher-order category.

Alternatively, you use the middle button with the cloud symbol to switch to a tag cloud display. It scales the categories to reflect the number of assignments. The flat view (right button) shows the tags of all layers.

You can create new keywords by typing them into the second field in the Fragment Editor, as a comma-separated list. Alternatively, click on the arrow to the right of the input box for a keyword palette. To view all the tagged fragments, click on the desired term in the palette. The middle column then lists the elements in the detail level specified by the detail-level slider.

In the left column, you can now see the tag editor. This is where you can rename the keyword and assign subordinate or superordinate keywords to it. The left column, marked by an arrow pointing upwards, represents the higher-order categories; the right column, marked by an arrow pointing downwards, represents the lower-order categories. In a column's text input boxes, you can either type keywords or select a keyword by clicking on the small arrow to the right of the input box and then clicking *Add*.

For maximum flexibility, Piggydb combines previously structured headings and longer text blocks into one object type, the text fragment. This tendency towards standardization goes even further. In Piggydb, each keyword optionally functions as a text fragment, and each text fragment also functions as a tag. However, you first need to create the other object type for assignment explicitly.

If you want to assign a text fragment to an existing tag and use the tag's name for the fragment, first click on the tag in the keyword palette. In the tag editor in the left column, click on the pen icon top right; this opens the Fragment Editor. After clicking on *Register*, the text fragment then becomes available as an initially unlinked element, and you can integrate it into your knowledge base like any other element.

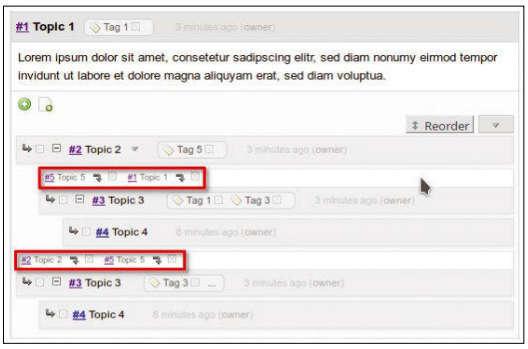
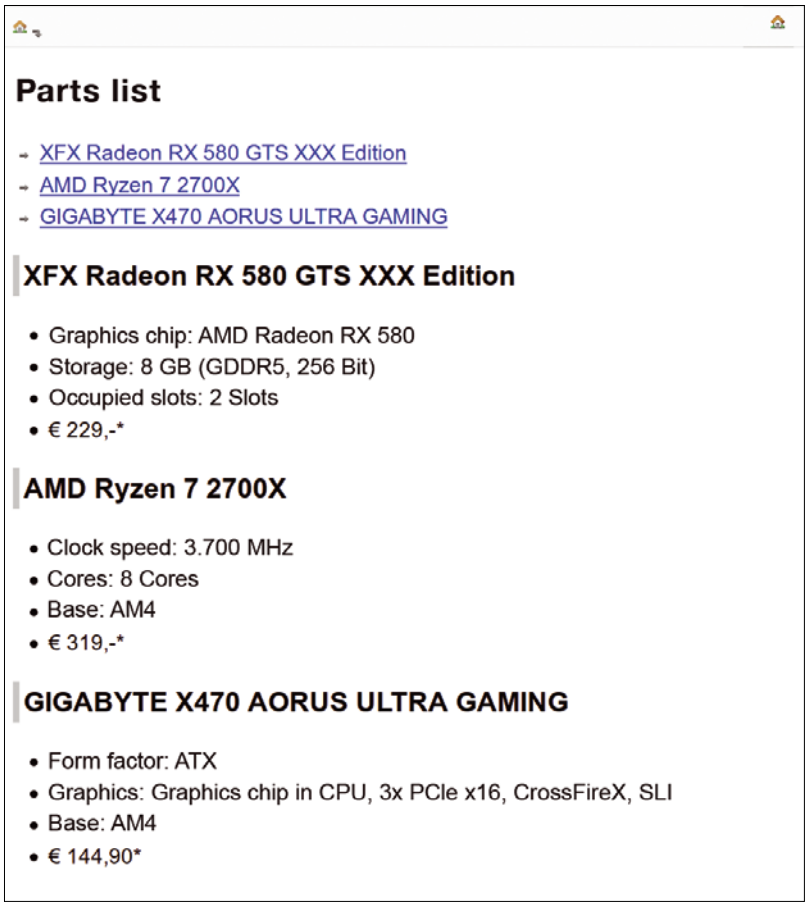


Figure 8: Behind the seemingly linear-hierarchical arrangement of the index cards in Piggydb, there is actually a mesh of relationships that – unlike tree graphs – allows several parent elements per node (outlined in red).



Parts list

- XFX Radeon RX 580 GTS XXX Edition
- AMD Ryzen 7 2700X
- GIGABYTE X470 AORUS ULTRA GAMING

XFX Radeon RX 580 GTS XXX Edition

- Graphics chip: AMD Radeon RX 580
- Storage: 8 GB (GDDR5, 256 Bit)
- Occupied slots: 2 Slots
- € 229,-*

AMD Ryzen 7 2700X

- Clock speed: 3.700 MHz
- Cores: 8 Cores
- Base: AM4
- € 319,-*

GIGABYTE X470 AORUS ULTRA GAMING

- Form factor: ATX
- Graphics: Graphics chip in CPU, 3x PCIe x16, CrossFireX, SLI
- Base: AM4
- € 144,90*

Figure 9: The document view, which combines the index cards' title and text content on a static web page, is suitable as a summary for knowledge bases and for copying the text to a word processor.

If you created the text fragment first and want to assign it a new tag with the same name, click on the tag symbol to the left of the title field in the Fragment Editor. If you now open the text fragment in the detail view or click on the keyword of the same name in the palette, the same thing happens in both cases. In the left column, you see the text fragment together with its child elements, while all the text cards marked with the tag appear in the right-hand column.

The advantages of this synchronization of text fragment and keyword may not be obvious at first glance. However, it is often helpful to explain the tag, which is limited to a short keyword, in more detail in the linked text. If, on the other hand, you notice that you want to link a text fragment with many more fragments, it is useful to create an associated tag. You can then link new fragments directly in the text editor by typing the keyword into the *Tag* field.

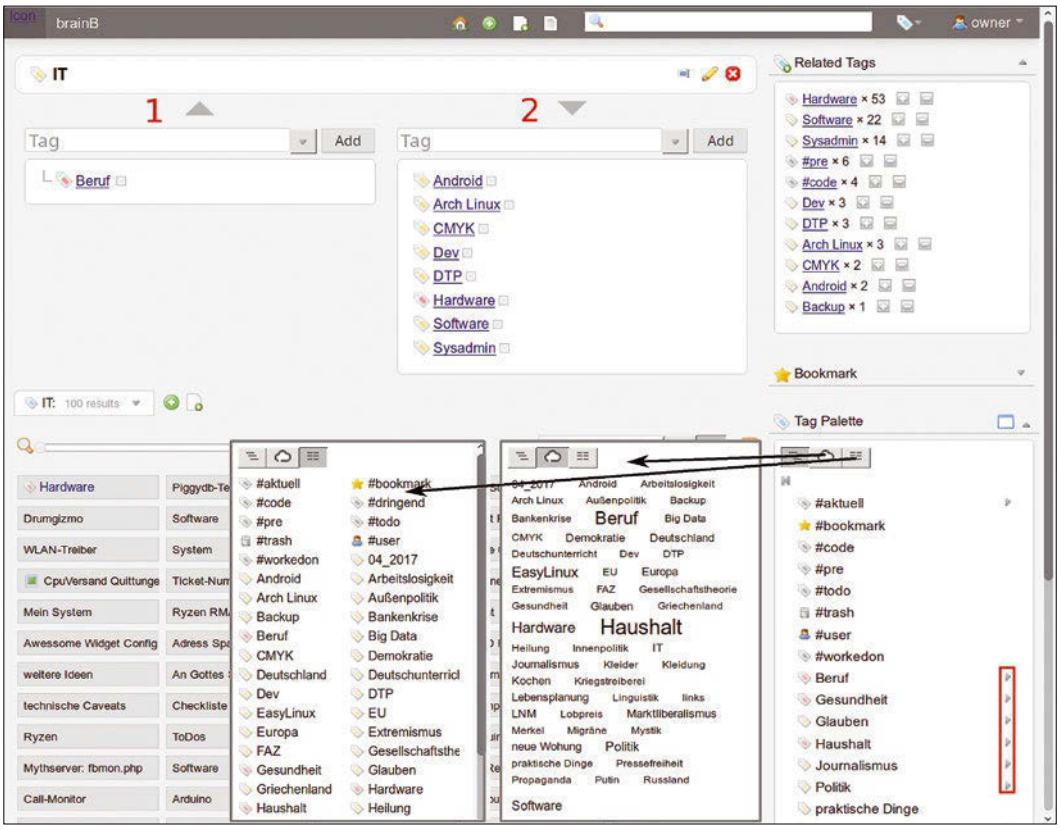


Figure 10: In Piggydb, you can assign parent (1) and child (2) tags to each keyword. The standard Tag Palette displays only first-order keywords with arrows to navigate through the hierarchy. However, there is also a tag cloud view and a flat tag list (both shown to the left of the Tag Palette).

Current Status

Piggydb is currently in a dormant state with the last release (a bug fix) in 2016 [6]. The author is currently working on an open source successor, Cotoami [7], which is not yet available as a stable version. Civilizer [8], a web application strongly based on Piggydb, was last updated in May 2018.

Conclusions

CherryTree is a free, practical, and mature outlining program. Without daring to experiment, it offers a complete and clear-cut scope. Sensibly selected keyboard shortcuts help users to work quickly and conveniently. If you like to work with classic linear structures, CherryTree is the right choice for you.

Piggydb cannot compete with CherryTree in terms of ease of use, but it will run on a centralized server, making the knowledge base available on all computers on a network without installing additional software. In contrast to CherryTree with its linear structure where each entry belongs to exactly one parent element, Piggydb supports multiple relations between the elements, which better matches the structure of human knowledge.

In addition, Piggydb's web application is well suited to freely formulating ideas in a brainstorming phase and then condensing them later into an outline. Piggydb's most practical advantage is the ability to declare any number of elements as root elements that appear on the start page at the press of a button. All fragments linked in this way appear with collapsible child elements like a file manager's folder view. ■■■

Info

[1] CherryTree: *https://www.giuspen.com/cherrytree/*

[2] Piggydb: *https://piggydb.net/about/*

[3] Piggydb download: *https://sourceforge.net/projects/piggydb/files/Piggydb/*

[4] Piggydb manual: *http://piggydb.jp/en/*

[5] TED talk about knowledge visualization: *https://www.youtube.com/watch?v=BQZKs75RMqM*

[6] Piggydb's 10th anniversary: *https://piggydb.net/2018/10/07/the-10th-anniversary-of-piggydb/*

[7] Cotoami: *https://github.com/cotoami*

[8] Civilizer: *http://suewonjp.github.io/civilizer/*

shop.linuxnewmedia.com

REAL SOLUTIONS *for* REAL NETWORKS

ADMIN - your source for technical solutions to real-world problems.

It isn't all Windows anymore - and it isn't all Linux. A router is more than a router.

A storage device is more than a disk. And the potential intruder who is looking for a way around your security system might have some tricks that even you don't know. Keep your network tuned and ready for the challenges with the one magazine that is all for admins.

Each issue delivers technical solutions to the real-world problems you face every day. Learn the latest techniques for better:

- **network security**
- **system management**
- **troubleshooting**
- **performance tuning**
- **virtualization**
- **cloud computing**

on Windows, Linux, and popular varieties of Unix.

SUBSCRIBE NOW
SAVE 30%

ADMIN
Network & Security

shop.linuxnewmedia.com

Identify mounted filesystems

Transparency

Tools like mount and findmnt reveal which partitions and filesystems are mounted on your computer.

BY FRANK HOFMANN

When creating a boot medium or expanding a system, you often need to create a new partition or add a filesystem to an existing partition. Selecting the wrong device can cost you valuable data.

Tools like `partx`, `fdisk/cfdisk`, `gdisk/cgdisk`, `mount`, `lsblk`, and `findmnt` help you discover which partitions and filesystems are present on the system and where they are currently mounted. We take you on a quick tour of some of these partition and filesystem tools, which are found on most of the standard Linux distributions.

Disk Tools

To make sure that you pass the correct parameters to the commands, you first need an overview of the volumes that exist on the system. The `partx` program will help you here. The `--show` switch tells it to list the individual partitions (Figure 1).

Figure 1: `partx` lets you quickly inventory which partitions exist on the system.

```
user@debian95: ~
File  Edit  View  Search  Terminal  Help
root@debian95:/home/user# partx --show /dev/sda
NR     START       END   SECTORS  SIZE NAME UUID
 1      2048   5896191   5894144  2.8G       1c248227-01
 2   5898238  31455231  25556994 12.2G       1c248227-02
 5   5898240   8511487   2613248  1.3G       1c248227-05
 6   8513536  14954495   6440960  3.1G       1c248227-06
 7  14956544  15564799    608256  297M       1c248227-07
 8  15566848  31455231  15888384  7.6G       1c248227-08
```

Figure 2: Using `fdisk`, you can determine the partition type, if required. Among other things, the software displays this information when you call up the list of partitions.

```
user@debian95: ~
File  Edit  View  Search  Terminal  Help
root@debian95:/home/user# fdisk -l
Disk /dev/sda: 15 GiB, 16106127360 bytes, 31457280 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x1c248227

Device     Boot    Start      End  Sectors  Size Id Type
/dev/sda1  *        2048  5896191  5894144  2.8G 83 Linux
/dev/sda2        5898238 31455231 25556994 12.2G  5 Extended
/dev/sda5        5898240  8511487  2613248  1.3G 83 Linux
/dev/sda6        8513536 14954495  6440960  3.1G 82 Linux swap / Solaris
/dev/sda7       14956544 15564799   608256  297M 83 Linux
/dev/sda8       15566848 31455231 15888384  7.6G 83 Linux
```

The output includes six columns with the partition number, start and end blocks, number of blocks, size, name, and UUID of the partition. The example shows six partitions `/dev/sda1`, `/dev/sda2`, and `/dev/sda5` to `/dev/sda8`.

This output already gives you a brief overview. However, it does not yet show you how the individual partitions are structured. `fdisk`, which outputs the individual partitions if you pass in the `-l` option (Figure 2), provides a little more information. However, the program only works for disks with a capacity of up to 2TBe, whereas the `gfdisk` and Parted tools can analyze larger disks.

Next up is `cfdisk`. As you can see in Figure 3, `cfdisk` provides even more information. This information includes the fact that `/dev/sda1` is a primary partition, `/dev/sda2` is an extended partition, and `/dev/sda5` through `/dev/sda8` contain logical partitions. In addition, it shows you the partition type – `/dev/sda6` turns out to be a swap partition.

`cfdisk` has a counterpart named `cgdisk`, which works the same way as `fdisk/gdisk`. When it comes to partition tables, `fdisk` and `cfdisk` only support Master Boot Record (MBR) format. MBR is a boot sector format that was popular with BIOS-based computers for many years. On the other hand, `gdisk` and `cgdisk` also support the GUID Partition Table (GPT) format. GPT is the successor to MBR for partition tables on hard disks and part of

Table 1: lsblk

NAME	Device name
MAJ:MIN	Main and extension numbers of the device
RM	*0* for fixed and *1* for removable devices (like CD/DVD drives)
SIZE	Storage capacity of the device in human readable form
TYPE	Type of device (see Table 2)
MOUNTPOINT	Directory in which the device is mounted

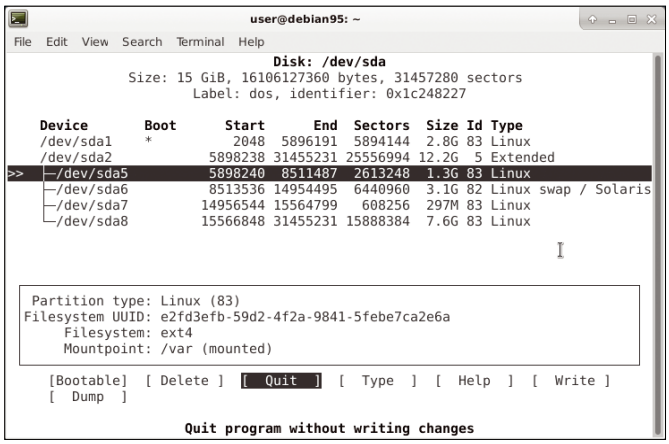


Figure 3: `cfdisk` shows you more information about the disks installed in the computer and the partitions on them.

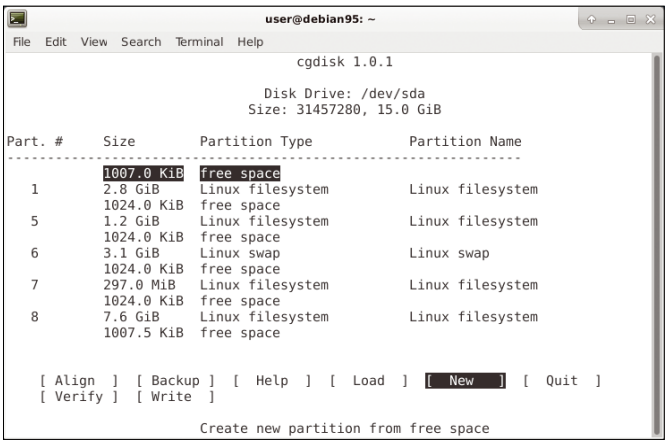


Figure 4: With `cgdisk` you can view the parameters of the partitions. The program reads the data from a GPT.

the UEFI standard. FigFigure 4 shows the `cgdisk` menu after calling `cgdisk -l /dev/sda`.

Table 2: lsblk Types

disk	Hard disk
loop	Loop device
lvm	Logic Volume Manager (LVM)[1] partition
part	Partition on a storage medium
rom	CD/DVD drive

1 LVM is the abstract partitioning schema across disks, partitions, and filesystems. LVM enables dynamically changeable partitions (logical volumes) across multiple hard disks.

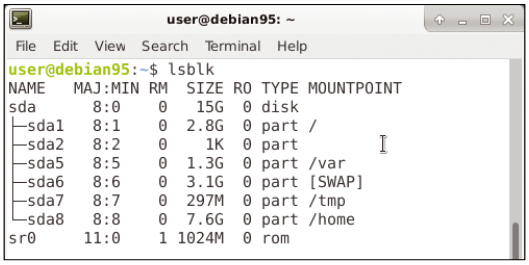
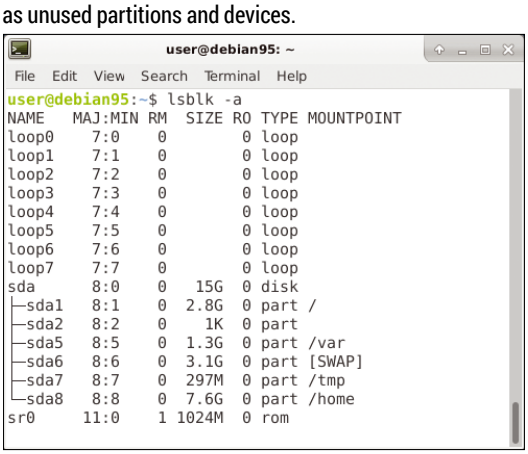


Figure 5: Using `lsblk`, you can discover details of the devices and partitions on the computer.

Figure 6: The `-a` parameter lists the active devices, as well as unused partitions and devices.



However, all these programs do not help to detect whether the listed partitions are mounted in the filesystem or where they sit in the directory tree. If you want answers to these questions, the `lsblk` program comes into play.

Mounted Disks

`lsblk` (the abbreviation stands for "list block devices") shows all block devices. The default view in `lsblk` includes only the devices or partitions currently in use (Figure 5). In the right-hand column, you will find the mount point (i.e., the position at which the device is mounted in the directory tree).

Figure 7: The standard output from `mount` tells you about the mounted filesystems and their mount options.

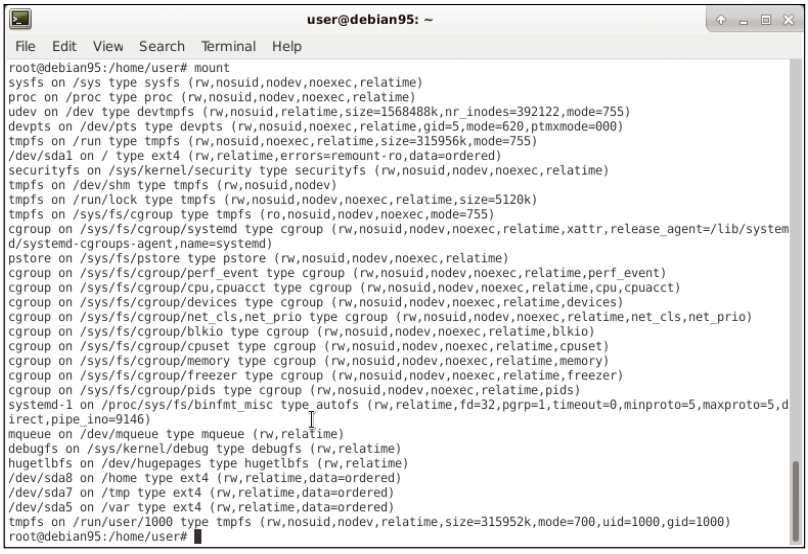
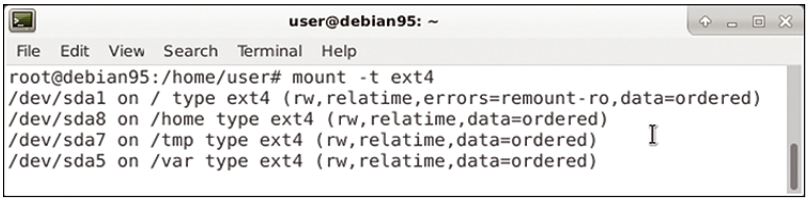


Figure 8: If you are looking for information about selected partitions only, use the `-t` option in `mount` to display them.



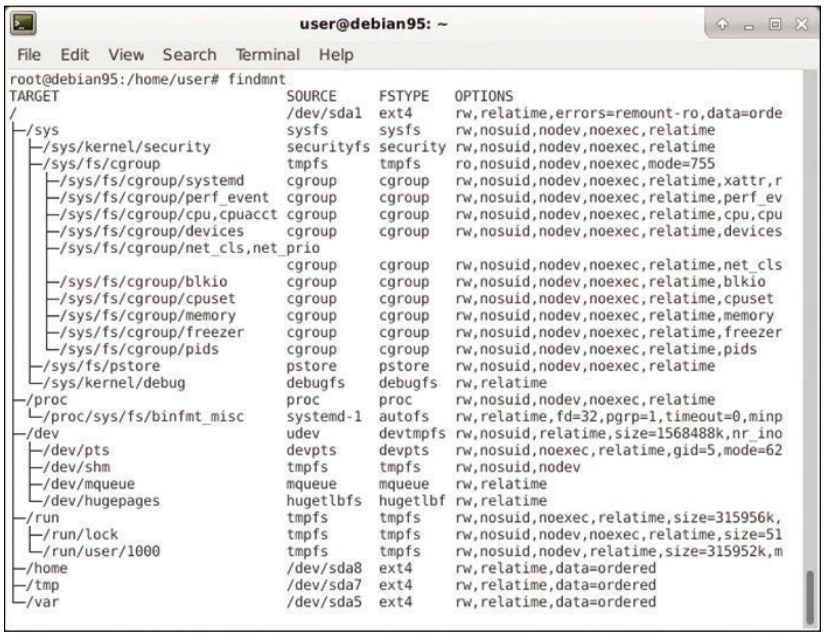


Figure 9: `findmnt` provides an overview of the mounted filesystems in the form of a tree view, including further information on each entry.

The output of `lsblk` has six columns. Table 1 gives an overview of the information you get from this list.

Using the `-a` switch helps `lsblk` bring more devices to light by including devices that are not currently in use. For example, Figure 6 shows a series of loop devices (virtual block devices that do not map to a physical device, but use a file for underlying storage) from `/dev/loop0` to `/dev/loop7`.

Filesystems

The `-a` switch tells you which partitions exist and under which mount points they are mounted in the filesystem. So far, however, you do not know which options were set when mounting the partition. The `mount` command shows these kinds of details (Figure 7).

The output from the command is quite extensive and often includes numerous virtual devices managed by the kernel and system programs. In many cases, however, only the physical mass storage devices and their mounted partitions are of interest.

Since Linux partitions mostly have the ext filesystem (today mostly ext4), you could filter the output of `mount` with `grep` accordingly. However, this detour is not actually necessary. Mount offers a `-t` (short for `--type`) switch that outputs filesystems of a certain type only. For example, the command in Figure 8 only returns partitions formatted with ext4.

`mount` provides so much information that clarity can suffer. The output of `findmnt` from the same software package offers a better structure. When called without further options, `findmnt` shows a tree view (Figure 9) from which you can see how the individual filesystems are mounted. The `findmnt` command also identifies the origin and type of the filesystem and the options with which it was mounted.

`findmnt` also supports the `-t` option, with which you can filter for ext4 filesystems, for example (Figure 10). Unlike `mount`, `findmnt` isn't limited to a single filesystem parameter but lets you enter several if required.

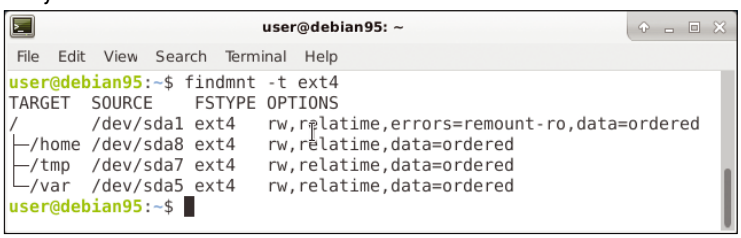
Conclusions

The tools presented in this article let you track down any filesystem. You can easily see what is mounted and where it can be found in the directory tree. This knowledge prevents you from accidentally deleting important data when performing an action with a hard disk because you are using the wrong parameters. ■■■

The Author

The digital nomad Frank Hofmann works from Berlin, Geneva, and Cape Town as a developer, LPI-certified trainer, and author. He is coauthor of *Debian Package Management* (*http://www.dpmb.org*).

Figure 10: Like `mount`, `findmnt` also offers the option to filter the output by filesystems for clarification.

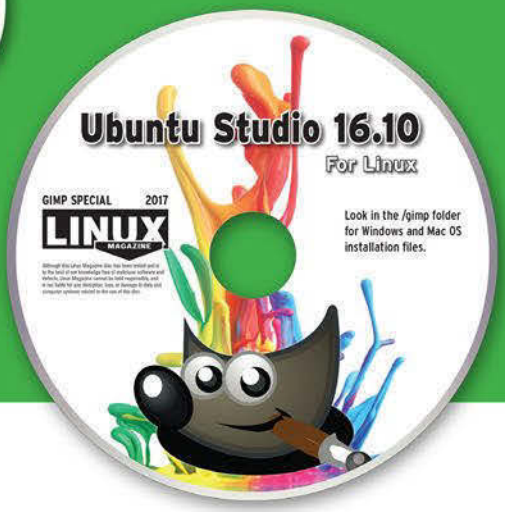


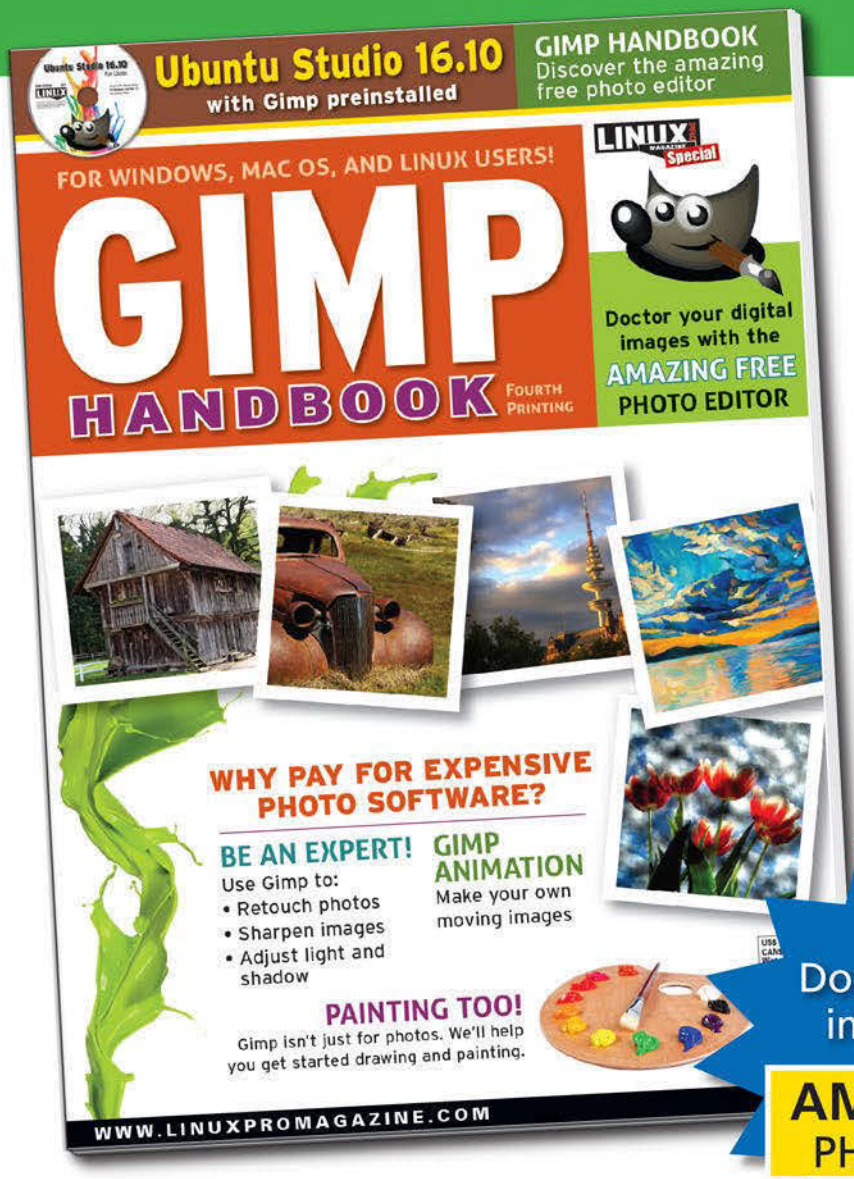
Acknowledgements

The author would like to thank Axel Beckert for his criticism and suggestions regarding this article.

Shop the Shop shop.linuxnewmedia.com

GIMP HANDBOOK





SURE YOU KNOW LINUX...

but do you know Gimp?

- Fix your digital photos
- Create animations
- Build posters, signs, and logos

Order now and become an expert in one of the most important and practical open source tools!

Gimp
Doctor your digital images with the
AMAZING FREE PHOTO EDITOR!

Order online:
shop.linuxnewmedia.com/specials

FOR WINDOWS, MAC OS, AND LINUX USERS!

FOSSPicks

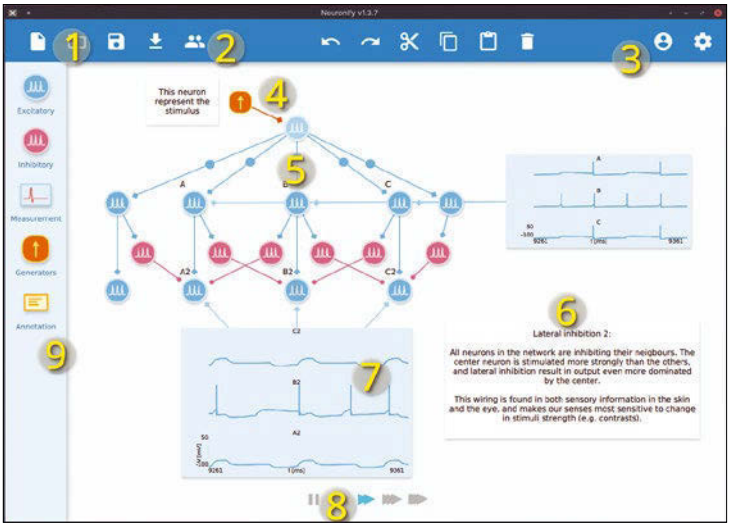
Sparkling gems and new releases from the world of Free and Open Source Software

FOSSPicks was very nearly late this issue, because Graham couldn't tear himself away from playing the wonderful Ascii Patrol in his terminal. BY GRAHAM MORRISON

Neural network simulator

Neuronify

This may be the first time we've ever looked at a piece of software that attempts to simplify neural networks. It may even be the first time we've looked at any software dealing with neural networks. This is because neural networks are complex, and without academic imperative, they're not something you can easily understand. But that's exactly what Neuronify is trying to do – help beginners explore and begin to understand neural networks. A neural network (in the computing sense) models the behavior of neurons in the brain in an attempt to learn things from datasets that would ordinarily be difficult to discover without specific and exhaustive analysis. Thanks to big datasets being created by companies like Google and Amazon, neural networks have become a huge field of research in software engineering and could hold the key to the future of vital services, such as health care and transportation.



1. File operations: Load networks, including many examples, or start a new project. 2. Community: Other users have shared their creations, which can be opened from here. 3. Account: Create an account to join the community and upload your own networks. 4. Stimulus: Every network has a stimulus, much like every circuit has a power source. 5. Signal: Watch the signal travel from the stimulus and through the exciters and inhibitors in your network. 6. Annotations: Help others understand your network (and research) by adding text. 7. Voltmeter: Measure the potential value as signals are received from the network. 8. Playback: You can adjust the speed of forward playback. 9. Tools palette: Create your network from these components.

Neuronify makes a bold claim – it wants to make it possible for you to work on neural simulations without prior computational experience. It does this by allowing you to build "circuits" in a graphical interface that are always live and running, much like an electrical circuit. These circuits produce feedback for you to see exactly what's happening. When you launch the application, easily installed from a snap, there's a simple tutorial to guide you through the key elements. Each stage of the tutorial is itself a working circuit that's active so you can see what's happening. This starts with the nodes you'll be using in the circuit: a current source, a "leaky neuron" that fires a signal when its potential passes a threshold, and a voltmeter that displays the value of that potential.

The application itself works much like a software modular synthesizer. You add elements from the palette on the left and connect these together using nodes. It's easy to work with but difficult to interpret any results without further study. Fortunately, there are many brilliant examples included that go much further than the simple tutorial. Because they're all annotated, you can learn a great deal simply by reading the accompanying text and studying the output. Also check out the button hidden at the top of the toolbar labelled *community*. Click on this, and you'll be able to explore simulations created and shared by other Neuronify users. There are currently only a handful, but they're often complex and show where that application is being used and studied.

Thanks to the beautifully implemented user interface (UI) and animation, it's fascinating to watch and play with the speed controls or the connections, even when you have no greater context or understanding. That may be the whole motivation behind making it so easy to get into neural networks: Because even when you're playing, you're reinforcing the ideas and patterns in your brain, so you can have a better sense of what neural networks are and how they might be implemented.

Project Website
http://ovilab.net/neuronify/

CPU tinkering

Undervolt GUI

Give your laptop better battery performance, with less heat, by undervolting its Intel hardware.

There's a fantastic command-line tool called `intel-undervolt` that gives you a fine level of control over how your Intel CPU is powered. It's the opposite of what you might typically do to facilitate overclocking your CPU, and it's useful for reducing power consumption and heat generation at the cost of performance. It can be a very effective way to extend battery life, especially if you're simply web browsing or typing up a document. As with overclocking, there is a risk you can damage your hardware, and it only works with Haswell and newer Intel CPUs. The `intel-undervolt` command can be a little intimidating, requiring a specific configuration file and command-line arguments, which is where `linux-intel-undervolt-gui` can help; it's a graphical wrapper around a Python script that is itself a wrapper around the voltage control provided by `intel-undervolt`.

The GUI requires your root password, so you may want to test this on a non-critical machine first. With those credentials it can then read your current voltages with a single click. There are four sliders: core/cache voltage (a single slider, because these need to be the same), GPU voltage, uncore voltage, and analog I/O voltage. Start with the CPU/cache voltage, and see how your system performs. However, only make small changes, because it can take a while for problems to become apparent. One change too many is likely to cause your computer to hang, especially when dealing with the CPU. This happened when we tested values over -100mv, for example. The GPU voltage for your graphics hardware seems to be a little more forgiving and less likely to cause crashes. Values less than -120mv caused graphical glitches for us, but we were able to restore the slider to a more modest value without a crash.

> useful for reducing power consumption and heat generation

Project Website
https://github.com/lukechadwick/linux-intel-undervolt-gui

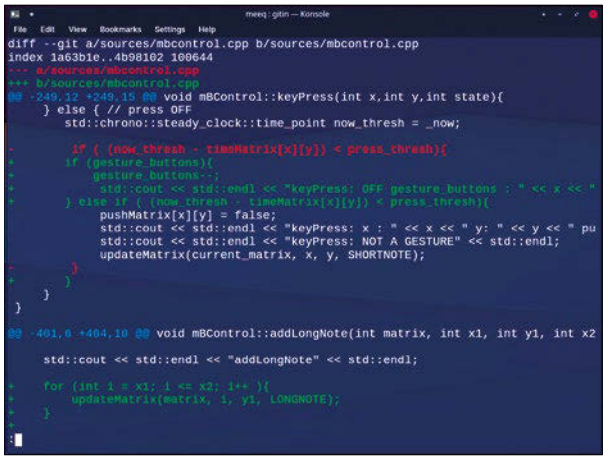
Git explorer

gitin

Tools for navigating `git` repositories, their history, and their branches are becoming almost as common as CPU monitors. But like CPU monitors, they also happen to be very useful. Having more options to choose from means you're more likely to find a tool to fit your exact requirements. One fully fledged option is `grv`, which runs from the command line and features several different view modes for diving into the details of any `git` repository. At the other end of the feature spectrum, `gitin` is another great option. Unlike `grv`, `gitin` isn't an application in which you spend time in; instead, it works best as a form of wrapper around the `git` commands you might use anyway.

Its principle functions allow you to work with branches, check the status of a repository, and view the logs, which are all you need in a tool since you don't want to duplicate what `git` already does so well.

Type `gitin log` within a repository, for instance, and you see a list of the most recent commits. Using the cursor keys to scroll through them updates details on each commit, and pressing return dives into which files have changed. Select a file, and you get the traditional `diff` view of what's changed within that file. It's minimal and effective and, most importantly, doesn't detract from the work you're likely trying to do in the same terminal. It just lets you see the same details `git` can provide in a more convenient and interactive way that's also quicker than trying to work out the commands using `git` alone. The use of color is also very effective and helps especially in the `diff` view where the red of a remove line and the green of a new line is far easier to see than the `-` and the `+`.



If you need a simple, quick, and easy way to view vital git repo information, gitin is an excellent choice.

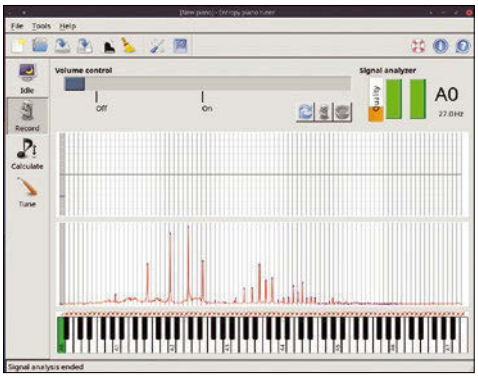
Project Website
https://github.com/isacikgoz/gitin

Piano tuner

Entropy Piano Tuner

If you're lucky enough to have a real physical piano made of wood and strings, you'll know that it typically needs regular maintenance. In particular, a piano needs to be tuned, and unlike a guitar, tuning a piano isn't a simple task. There are 88 keys, and each key has multiple strings that need to be tuned to a specific tension, all the while keeping a keen ear for any extra harmonics or vibrations from other keys, strings, and the frame within the instrument. It normally takes a skilled piano tuner an hour or two to keep your piano in the best shape, and this can be expensive. Entropy Piano Tuner isn't a replacement for those skills, but it is a huge help if you have one or two keys that go out of tune before the others, or if you want to learn the dark art of piano tuning yourself.

The process starts by creating a new file for your specific piano. With that done, the application defaults to an idle mode that constantly listens to your microphone input, automatically detecting the pitch of any notes you play. If the levels seem good for this pitch detection, you then switch to *Record* mode. Now you can patiently record an audio signal from every key on your piano, which takes around 30 minutes. As you record, a spectrogram of frequencies is produced for each key, and a little tick on the GUI shows its successful status. With that done, you press *Calculate*, and the application will generate a tuning curve for your piano. A nice feature here is that you can connect a MIDI keyboard and play your tuned piano samples to make sure the recommended curve sounds accurate. If it does, the final mode is *Tune*. In this mode, you play a key and Entropy Piano Tuner shows you both the current pitch, and the pitch it needs to be, which you adjust with your piano tuning tool. Do that for every key, and you have a tuned piano and extra money in your pocket!



As you record each key from your piano, a small tick mark is placed above the keyboard so you can see which keys you've yet to record.

Project Website

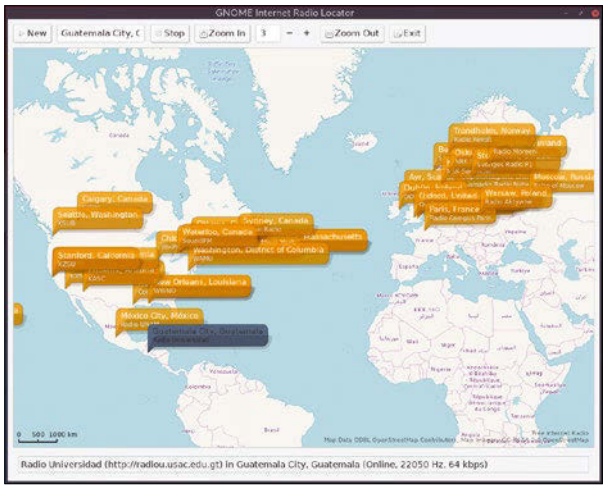
http://piano-tuner.org/

Internet radio

Gnome Internet Radio Locator

Back when cheap pervasive Internet was still young, one of the best things you could waste your bandwidth on was Internet radio. It was a little like hiding under your childhood bedcovers trying to listen to some esoteric foreign radio station on a shortwave radio. But because this was the Internet, the only geographical limitation was whether a station was broadcasted or not, or whether your ADSL connection could handle the data. In the modern era, esoteric radio is simply a voice command away, but it's still wonderful. And one of the best ways to explore Internet radio is from an actual representation of the globe so you can see roughly where the radio is being created and published. This is exactly what Gnome Internet Radio Locator does.

Thanks to the magic of OpenStreetMap, the main view is an explorable map that operates just like an online map. Zoom in and zoom out or scroll around to focus on a specific area. The stations held in the local database are visible as pins and flags on the map, so you can easily see which stations are broadcast from Mexico City, for example, or Europe. Double clicking on one of these starts the playback, and you can then see details such as the bitrate and the sampling rate, as well as the source URL for the station's website. You can also add your own stations using the *New* button, which makes this a great place to store your own discoveries and, potentially, share them back via the source code for other listeners to enjoy. In fact, the developers openly request you do this by filing bug reports for the stations you want to see, which may just be one of the easiest ways to make your first open source contribution.



Discover a whole world of Internet radio with a little help from OpenStreetMap and Gnome.

Project Website

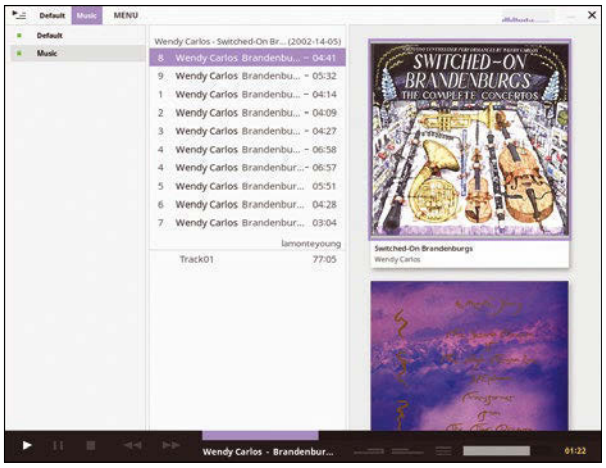
https://gitlab.gnome.org/GNOME/gnome-internet-radio-locator

Music player

Tauon Music Box

Yes, there are many music players, and yet people keep developing them. It's almost like writing a music player is the equivalent to writing "hello world" as a way to make sure your tool chains and libraries are in place. Except, they require much more effort than writing "hello world," so it must be because the developer wants to explore some new concept or idea. This is broadly true of Tauon Music Box. It has been designed to be "simple and streamlined while putting the user in control of their music collection," and it does a very good job. The streamlined UI is particularly impressive as the design is both simple and very adaptable. For example, the main view takes a traditional approach, with the source on the left, the playlist in the middle, and album art and details on the right, but it's always responsive and easy to modify.

Most elements can be quickly and easily changed, and the flexibility of the UI is brilliant. A click on the icon on the top right will remove the source pane, and clicking on the *Menu* button opens a drop-down list of thumbnails for various layouts. Updates happen almost instantaneously, and the audio controls and real-time visualization are equally responsive – vital for a music player. From the settings pane, you can even flick between several well chosen light and dark themes and set options for transcoding the audio for output and linking with both Last.fm and ListenBrainz. The only slight hitch is that the default audio engine uses the BASS audio library, which is proprietary, but this can be changed to GStreamer if you prefer. If the developer could just add the ability to source a music folder from a Samba share, rather than having to add that share manually to `fstab`, you wouldn't need any other player.



Tauon Music Box will also work with Picard, if installed, for easy tag editing across your entire collection.

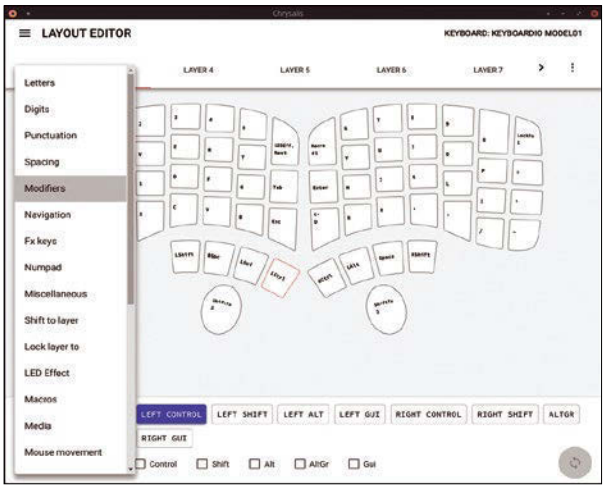
Project Website
https://github.com/Taiko2k/TauonMusicBox

Keyboard configuration

Chrysalis

There are many Kickstarter projects that have hit their goal only to flounder during that insane period after they get the money. This is for a variety of reasons, but perhaps the most common is that a project fails to anticipate the monumental effort required to take a product from the prototype to the production line. The wooden "heirloom grade" keyboard, called keyboard.io, could have so easily been one of those failures. It was two years late, but during the course of those many extra months, its creators – Jesse and Kaia – wrote a wonderful series of blog posts on dealing with the minutia of small-scale manufacturing and its horrendous complexity, from the USA to Chinese New Year in Shenzhen. All this time later, the keyboard is a reality, and Bruce Byfield wrote a wonderful and in depth look at the hardware in issue 211 (June 2018) of this very magazine.

At the heart of keyboard.io is an open source philosophy that not only touches the hardware and design, but also the software. Its Arduino-based firmware is open source and so too is Chrysalis, a new desktop application used to configure the keys. Chrysalis is not just limited to keyboard.io hardware. Chrysalis can be used with any Kaleidoscope-powered keyboard, which has become an open standard for general Arduino powered keyboards, including the Model 01, the ErgoDox EZ (and any other keyboard wired like the original ErgoDox), the Atreus, and Dygma's Raise (preliminary, at the moment). This means you can buy one of these or build your own, and you don't have to worry about the firmware or the configuration. The software saves you from editing Arduino source files and works in real time with your keyboard. Adjust the keys for each layer, and adjust the shortcuts that work across keys and layers to create the ultimate configuration for your work.



If your keyboard runs an open source firmware, chances are you can run Chrysalis to change its colors and behavior.

Project Website
https://github.com/keyboardio/Chrysalis

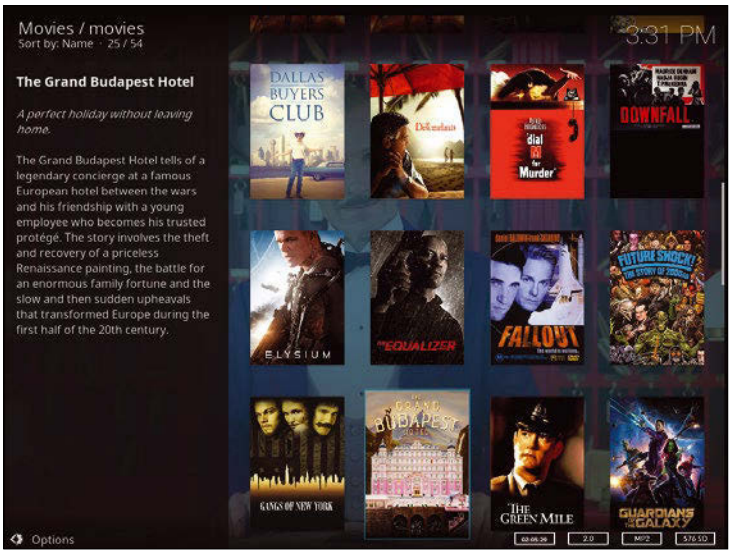
Media player

Kodi Leia

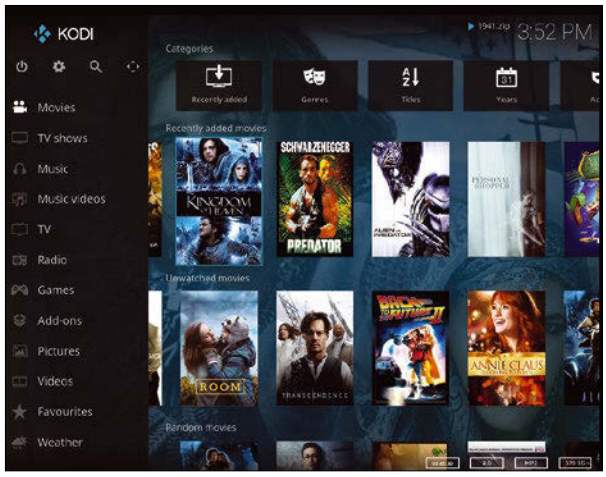
Kodi, alongside the likes of Firefox and VLC, is a flagship open source project. It has been at the cutting edge of media applications for years, fearlessly pushing updates, themes, add-ons, plugins, back ends, front ends, platforms, and clients for a variety of different architectures, from the Raspberry Pi to Android. Of course, this is because it's not your average media player, but something that works best when it takes over your machine. That's why it's perfect on a Raspberry Pi, or side-loaded as an Android Debug Bridge (ADB) onto something like Amazon's Fire TV. We love it when there's a major update, because Kodi's amazing development team manages to cram in so many (media) life-changing features.

Version 18 comes two years after the previous major release, and it's difficult to choose which of its major new features to cover. Perhaps the most interesting is RetroPlayer, an embedded games emulator that supports many different kinds of old systems and their software. This allows you to launch, play, and quit back to the menu all from the same interface. It's based on libretro, which mean you can play almost anything, including games for the Commodore 64, Nintendo Game Boy/64/SNES, and Sega Dreamcast/Genesis/Saturn, plus many old MS-DOS games. File management is still very basic, and you obviously need to have legal access to the ROMs, but there's also a brilliant add-on that will download games directly from the Internet Archive (*https://archive.org*), where many are being saved from oblivion as abandonware. This lets you choose between hundreds of games being stored online, sorted by year and by platform, and you can easily waste hours reliving the wasted hours of your youth.

Add-ons themselves have also become simpler, with the requesters for entering repository source details and the add-on viewer much closer in the menu hierarchy, plus there's much improved dependency management. A new button in the add-on selector lists the dependencies required by a selected add-on along with details about which dependencies are installed and which are not. Seeing this information in one place is much more convenient than manually trying to figure out missing dependencies when an add-on doesn't work. Kodi is now far more compatible with DRM-protected content, and in particular, both Netflix and Amazon Prime. Historically, these two services have been able to work with add-ons, but service updates often broke compatibility. With the new release, you can stream full HD from these sources, much like any other source, and you don't have to worry about your device defaulting to a low quality fallback mode. Plus, using Kodi is a big improvement over Amazon's sprawling interface of adverts on its Fire TV box, for instance. And finally, "Live" TV no longer feels like a second-class citizen. Thanks to an overhauled UI and new PVR-centric add-ons, Kodi is now just as good at EPG viewing and scheduling as more traditional PVRs, such as MythTV. You will still need a back end controlling the hardware, but the Kodi front end at least can almost replace a satellite or terrestrial STB, or even IPTV, to make Kodi the best open source one-stop solution you can install.



While its main UI has changed very little, Kodi has seen many changes beneath the surface over the last two years.

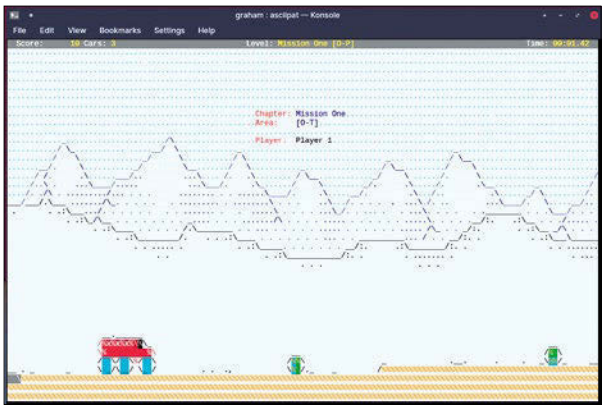


Because you can see an add-on's dependencies, it's now easy to install any of the hundreds of add-ons available for Kodi.

Project Website
https://kodi.tv

Arcade conversion

Ascii Patrol:



With a title that's a brilliant pun and great replayability, Ascii Patrol has become our go-to play over SSH while you rsync your game.

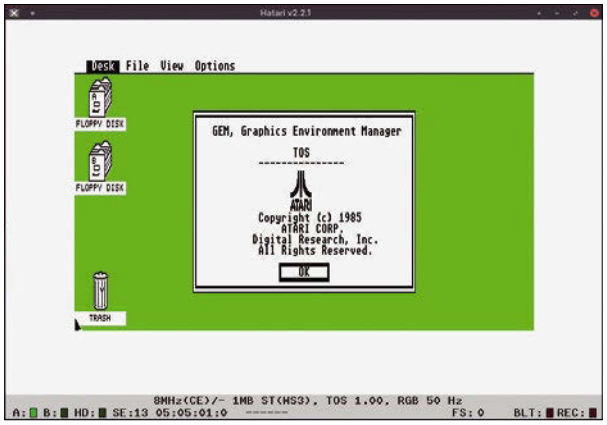
When you think of games made with ASCII characters, you typically start with Snake and end with NetHack, both of which are brilliant in their own way. You don't typically think of something with refined arcade quality playability and design. And yet that exactly describes Ascii Patrol, an interpretation of the arcade classic, Moon Patrol, recreated in text characters to run on a console. Outside of 80s-era home computer arcade conversions, it's one of the few modern recreations of an old game built atop an even more limited graphics engine than the original. However, in this case, it makes the graphics utterly enchanting. You'll be hooked from the moment you launch the game and the moon buggy scrolls smoothly from one side of the screen to the other.

Pressing any key from the title screen takes you to an overview screen, which includes a player profile and editable avatar, a campaign overview complete with a map, and an online leaderboard, plus there's a handy controls overview. It's like an AAA game on Xbox Live. Press the spacebar, and you're playing the game. As with the original, the action comes from a side-scrolling obstacle course where you accelerate and decelerate your buggy, jump, and shoot at things while avoiding rocks and crevasses. Your score is based on how far you get. It's deceptively tricky and very addictive, and you soon forget the entire game is being rendered using only ASCII characters. You need to complete each level to unlock the next, and this is what makes you want to get over the learning curve, just like in the olden days. It plays brilliantly, and it's just as addictive and as playable as the original, with the new/old graphics being part of the charm, rather than a compromise. Play it over Telnet today!

Project Website
http://ascii-patrol.com/

Atari ST emulator

Hatari



Play games and even use MIDI equipment with this brilliant recreation of an Atari ST.

When it comes to emulation, the Atari ST is often overlooked in favor of the Commodore Amiga. This could be because Commodore's story is the more romantic: an underdog with superior hardware under the custodianship of an owner incapable of building on its advantage. The Atari ST was also hugely successful in the 80s, to such an extent that it became the bane of every Amiga owner as they suffered substandard game ports from its rival, the lowest common denominator. But it's still a huge chunk of many childhoods, and as a result, great fun to play with via an emulator like Hatari. What's also great about Hatari is that it has very few dependencies, it's cross-platform, and you can easily build it yourself. All you need to get rolling is a legal copy of Atari's ROM or a copy of the free EmuTOS ROM.

If the emulator locates your ROM, it will launch and present you with the old GEM desktop in all its wonderful green and gray glory. Hatari's UI is equally austere, looking like something built in the Unix/X windows/Motif era, but also totally fitting with the late 1980s. As with the Amiga emulator, UAE, you press F12 to bring up the options overlay. From here, you can insert floppy disk images, hard disks, joysticks, and upgrade the memory and CPU with a click of a checkbox, although this may break compatibility. And compatibility is brilliant. We raced through lots of old games, from Captain Blood to Midwinter, and everything seems to work faultlessly. The emulator also handles MIDI for controlling keyboards and synthesizers. This means that Hatari could be used for more than nostalgia as there's a growing trend for modern musicians to go back to old software for its simplicity and durability.

Project Website
http://hatari.tuxfamily.org

Shell test conditions and exit codes

Decision Making Scripts

The Bash shell uses different criteria to make decisions. Learn how to teach your shell scripts to make the right choice.

BY MARCO FIORETTI

In the previous installment of this series [1], I described how to add different possible courses of action to a script, so that the script can choose by itself which action to execute while running. In this issue, I will explain how to teach a script to choose which of the available actions to execute.

Often, the real decision-making challenge lies not in figuring out whether your shell script needs a `while` loop or some nested `if` statements but rather in determining the *conditions* that will tell your script when it should stop that loop or which branch of that "if" statement to execute. The main Bash tools for this purpose are a big set of test operators (see the descriptions online [2] [3] [4]) and their corresponding syntax, which can evaluate whether some condition is true or false. By contrast, exit status codes [5] are the traces that built-in commands, or whole scripts, leave behind to communicate their achievements.

In this article you will learn through examples and working code:

- Which kinds of testing operators are available
- The syntax of those operators and their related Bash keywords
- How to retrieve (or provide for further testing) a single command or an entire script's exact outcome via exit codes

The Author

Marco Fioretti (*http://mfioretti.com*) is a freelance author, trainer, and researcher based in Rome, Italy. He has been working with free/open source software since 1995 and on open digital standards since 2005. Marco also is a Board Member of the Free Knowledge Institute (*http://freeknowledge.eu*).

Test Conditions and Operators

Shell scripts can check if a condition is verified or not in three main ways: values of numbers, content and structure of text strings, and file properties. In and of themselves, the syntax and operators are not really difficult. They are just very picky and hard to memorize, because they are numerous and as hard to distinguish as they are powerful and useful in practice. For this reason, I highly recommend that you print or save cheat sheets of the resources referenced in this article [2], [3], [4], [5].

When I talk about "test conditions," I mean the code that follows Bash keywords like `if`, `while`, or `until`. There are four different ways to write these test conditions; all are relatively simple, but very picky. The devil really is in the details.

The first and perhaps most common way of writing test conditions is the single-bracket syntax, as follows:

```
if [ ! -f somefile ]
   then
   # do something
```

This code means "do something" *only* if `somefile` is *not* a regular file: the `-f` operator asks if it is true that `somefile` is a regular file. The exclamation mark before `-f` negates the statement (i.e., inverts the answer). Of course, the condition inside the brackets may be much more complex, as I will show in a moment.

The other main syntax for Bash test conditions uses two brackets per side and behaves in slightly different ways, which you really want to know in order to exploit the syntax instead of enduring much frustration.

As you already know, before using a variable, the Bash interpreter splits any variable containing spaces into the several words defined by those spaces. This is why this check

```
TEST_VARIABLE='Hello World'

if [ $TEST_VARIABLE == 'Hello World' ]
then
echo "TEST succeeded!"
fi
```

will fail complaining "`[ too many arguments`": `$TEST_VARIABLE` was split into the two separate terms "Hello" and "World", and the string comparison operator `==` only accepts *one* term per side. However, just wrap that condition into another pair of brackets as follows

```
TEST_VARIABLE='Hello World'

if [[ $TEST_VARIABLE == 'Hello World' ]]
then
echo "TEST succeeded!"
fi
```

and it will merrily print "`TEST succeeded!`": The first effect of double brackets is to disable word splitting on the left term of a condition.

On the right term of the same condition, the double bracket syntax has the opposite effect. Put asterisks into that term

```
TEST_VARIABLE='somestring'

if [[ $TEST_VARIABLE == so*string* ]]
then
echo "TEST succeeded!"
fi
```

and the test will succeed. Remove the outer brackets, and it will fail, unless you set `$TEST_VARIABLE` to be *exactly* '`so*string*`' (*without* the quotes). The reason is that only the double brackets enable "globbing" on the right term; they make Bash decode any asterisk inside that term not as an ordinary character, but as a wildcard meaning "there could be anything here!" In other words, without globbing, a string like '`so*string*`' matches only that exact sequence of 10 characters. With globbing, it includes *every* string that starts with `so` and contains the sequence `string`, with zero or multiple random characters before and after `string`.

Double brackets also support Perl-style regular expressions to match patterns inside strings:

```
TEST_VARIABLE='so*string*'

if [[ $TEST_VARIABLE =~ ^so ]]
then
echo "TEST succeeded!"
fi
```

The code above will print "`TEST succeeded!`", because the right half of the condition above means "match any string that *starts* with `s` and `o`."

Another thing that the two main bracket syntaxes treat differently is filenames. Inside single brackets, `*.txt` expands to be "*all* the files in this folder ending with the `.txt` extension," because the asterisk is interpreted as "zero or multiple characters." Inside double brackets, instead, the asterisk would be taken literally, and `*.txt` would mean "a file with the `.txt` extension and one asterisk as name."

The final difference between single and double brackets is that only the single bracket form accepts the `-a` and `-o` formats of the logical `AND` and `OR` operators, whereas double brackets allow the use of `&&` and `||` (more on this later). In general, double brackets are trickier, but they are more flexible and better suited to handle text comparisons.

Alternative Syntaxes

In addition to the single and double bracket constructs, you also can check test conditions in Bash with the built-in `test` command and normal parentheses:

```
if [ $COUNTER -ge 100 ] ; then...
if test $COUNTER -ge 100; then...
if (( $COUNTER >= 100)); then...
```

Those three conditions mean the same thing: "If `$COUNTER` is equal or greater to 100, then do something."

The `test` option is more compact, but I personally find it a bit less readable than the brackets. On the other hand, the parentheses are very clear but only work on numerical conditions, not with strings or filenames.

Combining Test Conditions

Test conditions would be of very limited use if they could not be combined. Regardless of syntax (single bracket, double bracket, parentheses, or the `test` command), you can use the `-a` (`AND`) operator to denote that two conditions must be satisfied. This is how you tell Bash to do something only if `$COUNTER` is equal or greater to 100, and `$NAME` is equal to "Mark":

```
if [ $COUNTER -ge 100 ➤
  -a $NAME == 'Mark' ]; then....
```

The `test` and brackets constructs can, as already mentioned, also use the `&&` and `||` versions of the `AND` and `OR` operators, while parentheses only support the `-a` form.

Whatever format of `AND` and `OR` you use, remember that unless you use parentheses, `AND` always takes precedence over `OR`.

Last but not least, whatever syntax you use or whatever condition you test, never forget that:

- Quoting variables makes the test work even if there are spaces or newlines inside them.
- Spaces between brackets and the variables and operators they contain are necessary.

Number and String Operators

As mentioned earlier, there are three main types of operators, each for a class of objects: numbers, text strings, and files.

As far as numbers are concerned, the next installment in this series will be specifically devoted to doing math with the Bash shell. Therefore, I am only going to briefly mentions the main numerical operators `-gt`, `-ge`, `-lt`, and `-eq` (greater, greater or equal, less, and less or equal, respectively). These operators let you write "keep doing something, until `$COUNTER` becomes greater than 100":

```
while ( $COUNTER -le 100)
    do something
```

The operators that compare strings are equally intuitive, as you can see in this small list:

```
$A  = $B True if the $A and $B strings are equal
$A != $B True if they are different
-n $A    True if length of $A is greater than 0
-z $A    True if $A if empty or uninitialized
```

File Test Operators

Real gurus know that "in Unix, everything is a file." This means that something like `/home/marco/somefile` may *not* be what we normally mean by the word "file," but instead could be a physical device (like a serial or parallel port) or

Listing 1: file-analyzer.sh

```
01 #! /bin/bash
02
03 TARGET=$1
04 MAXSIZE=$2
05
06 cd $TARGET
07
08 IFS=$(echo -en "\n\b")
09
10 for FF in `find . | sort`
11 do
12 ###############################################
13   if [ -d "$FF" ]
14   then
15   continue
16   fi
17
18   ###############################################
19
20   if [[ $FF =~ [[:space:]] ]]
21   then
22   printf "%-25.25s : %s\n" "SPACES in file name (1)" $FF
23   fi
24
25   if [[ "$FF" = *" "* ]]
26   then
27   printf "%-25.25s : %s\n" "SPACES in file name (1)" $FF
28   continue
29   fi
30   ###############################################
31
32   if [ -x "$FF" -a ! -O "$FF" ]
33   then
34     printf "%-25.25s : %s\n" "NON-owned executable" $FF
35   continue
36   fi
37   ###############################################
38
39   EXT=`echo $FF | awk -F . '{print $NF}'`
40   if [ -x "$FF" -a "$EXT" = "pdf" ]
41   then
42   printf "%-25.25s : %s\n" "EXECUTABLE PDF" $FF
43   continue
44   fi
45   ###############################################
46
47   if [[ "$FF" == */drafts/* ]]
48   then
49   printf "%-25.25s : %s\n" "DRAFT, remove?" $FF
50   continue
51   fi
52   ###############################################
53
54   if [ -L "$FF" ]
55   then
56   if [ -e "$FF" ]
57     then LINKSTATUS='LINK'
58     else LINKSTATUS='BROKEN LINK'
59     fi
60     printf "%-25.25s : %s\n" "$LINKSTATUS" $FF
61   continue
62   fi
63   ###############################################
64
65   SIZE=`ls -s $FF | sed -e 's/ .*//'`
66   if [ "$SIZE" -eq "0" ]
67   then
68     printf "%-25.25s : %s\n" "EMPTY FILE" $FF
69   continue
70   elif [ "$SIZE" -ge "$MAXSIZE" ]
71   then
72     printf "FILE TOO BIG: %-10.10s : %s\n" $SIZE $FF
73   fi
74 done
```

even pipes or sockets (i.e., objects that connect running programs so they can directly exchange data). You can reveal the exact nature of what *looks* like a normal file with these operators:

```
-e $F : $F exists
-p $F : $F exists and is a named pipe
-S $F : $F exists and is a socket
-f $F : $F exists and is a regular file
-h $F : $F exists and is a symbolik link
  (equivalent form: -L
-d $F : $F exists and is a directory
```

Other operators test file permisions and ownerships:

```
-r $F : $F exists and is readable
 (similarly, -w = writable, -x = executable)
-O $F : $F exists and is owned by the user
  running the test
-G $F : $F exists and owner by the user's group
```

The following two operators, instead, compare file time stamps

```
$A -nt $B
$A -ot $B
```

to tell the script whether file `$A` is newer (`-nt`) or older (`-ot`) than file `$B`. See the Linux Documentation Project for the entire list of file test operators [4].

Real World Example

A small company, which I will not name, spent a few years reshuffling everything (business plan, product catalog, staff, and junior consultants) every quarter. That chaos gave people (or perhaps forced on them) the "freedom" to store and (re)organize all the files and software they used in whatever way they wanted. To make a long story short, the dubious honor of sorting that mess eventually fell on me.

In practice, I had to locate, inside hundred thousands of files of all sorts, the files that were broken links, too big, potential security risks due to wrong permissions or other reasons, or symptoms of other problems. Of course, to solve that problem, I used the power of the Bash test operators. The result was a Bash file analyzer (see Listing 1 for a simplified version).

The file analyzer takes as input the directory that it should scan and a file size expressed in bytes. It then looks at all the files in that directory, reporting all those that are bigger than that size or violate any other rules embedded in the code, as shown in Listing 2.

After moving to the target directory (Listing 1, line 6), the script finds and sorts all the files and folders it contains, in order to examine them one at a time (line 10). The command in line 8, right before the loop, sets the Bash Internal Field Separator (IFS) to newline (`\n`), backspace (`\b`), and nothing else. Without it, files or folder names containing spaces would be split by those spaces, and the computer would therefore load into `$FF` the names of files or folders that do not exist, causing a long stream of useless warnings.

The body of the loop consists of several, independent file tests, each looking for a specific condition or combination of conditions. Whenever one of those tests matches, the script reports what it found and then moves, thanks to the `continue` keyword, to the next file in the pipe. The way a `for` loop can work on a dynamic list of files, as well as the behavior of the `continue` keyword, are described in detail in the previous installment of this series [1]. In this article, I will just focus on the test operators.

I am only interested in files, so the first thing I do (lines 13 to 16) is to just jump to the next item in the list whenever I find a directory. At first glance, it may seem that a better way to skip directories would be to pass the `-f` ("files only") option to the `find` command in line 10. That option, however, would make the loop also skip links and other non-regular files, which I do need to find instead. This is why I make the script "find" everything and then ignore the directories.

The two `if` blocks in lines 20 to 29 do the same thing twice, just to show you two different methods (regular expressions and plain comparison): They report files with spaces in their names (or in the names of the folders containing them). In the original script, this check existed because such spaces were not allowed due to some legacy custom software unable to handle them properly. Here, I have left it, because knowing several ways to check if a string contains spaces or other special characters is always useful.

The next check (lines 32 to 36) reports all files that are executable (`-x`) but do *not* (`!`) belong to the user that is running the script (`-O`). Then, in lines 39 to 44, I first save into `$EXT` the extension of the current file and issue a warning if that file is an

Listing 2: File Analyzer Output

```
BROKEN LINK           : /home/testing/link-to-non-existing-file
DRAFT, remove?        : /home/testing/drafts/biography.md
EMPTY FILE            : /home/testing/just-an-empty-file.txt
EXECUTABLE PDF        : /home/testing/pdfs/executable.pdf
FILE TOO BIG          : /home/testing/archives/a-really-big-file
NON-owned executable  : /home/testing/someweirdfile
SPACES in file name : /home/testing/filename with spaces
```

Exit Codes to Avoid

Certain exit codes have reserved meanings, which you must not use them unless you really want to break or confuse whoever runs your code (other scripts, other users, or even you). You can find these reserved exit codes listed online [5]. To give you an idea of the kind of messages reserved exit codes carry, here are two examples:

- 126: Command invoked cannot execute (not an executable program or permission problem)
- 127: Command not found (possible problem with `$PATH` or a typo)

executable PDF. (In general, every file should only have the minimal set of permissions it actually needs to be used.)

Another thing I had to do was to find files that may no longer be needed in order to free disk space. This is the task shown in line 47, which detects all files inside subfolders named `drafts`.

Links to other files are found with the `-L` operator in line 54. If they point to existing files (line 56), they are just listed. Otherwise, a warning is issued.

The pipe in line 65 saves into `$SIZE` the size in bytes of the current file (to understand how it works, execute that pipe on any file at your command prompt, one piece at a time). The `if` statements that follow print proper warnings if that number is either zero or above the threshold passed as the second argument to the script.

The script in Listing 1 works, but there are many things it does not do. The first is to actually check the format of a file, instead of blindly trusting a filename extension that may be wrong or missing. The way to fix this shortcoming is to use the `file` command; this is left as exercise for the reader.

Exit Codes

Many times a script can only decide what to do next by looking at the exact outcome of some command or other program. Traditionally, this information is provided through exit status codes, also known as return statuses. Exit codes are integer numbers that commands or scripts output when they end. By historical convention, an exit status equal to zero means that the execution was successful, and any non-zero value indicates that something bad, or at least unexpected, happened. In the Bash environment, the exit status of the last instruction that was executed is stored in the special variable `$?`.

Every well-written program (including your scripts) should output defined exit codes, one for each way in which the script could end. As a matter of fact, all Bash functions, pipes, and scripts always return an exit code even if their author did not set it explicitly. That number is the exit code of the last command executed by the function, pipe, or script.

However, it is good practice to make a script end with an informative exit code (see the box "Exit Codes to Avoid"). Doing so couldn't be simpler: Just use the `exit` command, followed by the number designated as exit status:

```
exit 400 #
```

Another thing you should remember about exit status codes is that every time your script interprets a line, it produces a number that you may examine with numeric test operators to diagnose problems or decide what to do next. This means, among other things, that exit codes also provide the basis to quickly implement test operators for files or strings that may be more verbose, but also more sophisticated, than the built-in operators.

A Final Example

To learn another way to check if a filename or any other string contains spaces, type the following commands at the prompt:

```
#> TESTSPACE='Hello World'
#> echo $TESTSPACE | egrep -q "[[:space:]]"
#> echo $?
#> TESTSPACE='HelloWorld'
#> echo $TESTSPACE | egrep -q "[[:space:]]"
#> echo $?
```

Info

[1] "Tutorials – Shell Flow Control" by Marco Fioretti, *Linux Magazine*, issue 221, April 2019, pp. 86-91

[2] Bash test constructs: *https://www.tldp.org/LDP/abs/html/testconstructs.html*

[3] Bash "if" statement conditions: *https://linuxacademy.com/blog/linux/conditions-in-bash-scripting-if-statements/*

[4] Bash file testing operators: *www.tldp.org/LDP/abs/html/fto.html*

[5] Bash reserved exit codes: *www.tldp.org/LDP/abs/html/exitcodes.html#EXITCODESREF*

COMPLETE YOUR LIBRARY

Order a digital archive bundle and save at least 50% off the digisub rate!

ORDER YOURS TODAY!
shop.linuxnewmedia.com

You get an **entire year of your favorite magazines** in PDF format that you can access at any time from any device!

Designing your own stuff with OpenSCAD

3D Designer

Ah! What a joy your first 3D printer … but once you have printed your first benchy, where do you go from there? To building your own pieces, of course!

BY PAUL BROWN

If you do a superficial search online, it would seem that there are two things you can do with your new 3D printer: the first is print benchies (castles and skulls with which to adorn your bookshelf); the second is to create the bits of your next Comic Con cosplay costume you can't make with a sewing machine.

Then there is a subgroup of people who seem to think that 3D printing is an aim in itself, and that the best use of their filament is to build things that further enhance their machines. I am one of them.

Printer Pimping

Truth be said, I have not been this excited over a gadget in years. I was in awe of my first prints and realized this was a thing I wanted to keep in good working order for ever. I found a post on Thingiverse [1] that explained that my printer model had a design flaw: The vent over the electronics was facing upwards; hence dust and little bits of plastic from the prints could fall in and damage the fan. The solution was to print an exhaust vent cover that would propel the flow of air towards the front.

That led to a down-the-rabbit-hole situation in which I ended up printing a drawer that would hold all the printer's bric-a-brac (spare nozzles, screws, bolts, and belts), a tool holder, and an improved filament feeder.

What I could not find was a decent camera holder. You see, continuing with the topic of "3D printer as an aim in itself," a lot of people attach cameras to their printers and then turn the videos into time-lapses, giving birth to that genre of videos on YouTube that show objects seemingly growing from nothing on a printer's hotbed.

There is also a practical reason for training a lens on your prints: Some print managers like Atelier [2] allow you to monitor your prints from afar. If the camera shows the print going wrong, you can abort it before things get really messed up.

None of the available camera holders were any good, either because they didn't fit my cheapo Chinese sports cam, or they didn't fit my printer.

So, I decided to make my own.

CAD Applications

This meant getting up to speed with CAD software for Linux. Blender is great for making artistic stuff for printing, but it is massive overkill just to print cubes, cylinders, and spheres. Because, as I soon discovered, most mechanical pieces are made out of those parts, you don't have to wrap your head around Blender for that.

Besides, you want a simple way of measuring precise lengths, widths, heights, and angles and applying constraints to each and all of them. So far, the best programs I have been able to find for that are OpenSCAD [3] and FreeCAD [4].

In this issue, we'll be looking at OpenSCAD. We'll tackle FreeCAD further down the road in a separate tutorial.

OpenSCAD is interesting because you use a programming language to describe all the objects that make up your pieces. This may sounds scary, but it is not. And, believe me, FreeCAD is much more mind-bending.

Interface

OpenSCAD's relative straightforwardness starts with its interface. The main window is divided into roughly three panes (Figure 1). On the left, you have the script editor, which is where you write your programs. The editor has syntax highlighting and auto-indents lines when you use the JavaScript-like code formatting style (see Listing 1).

On the right, at the top, you have the preview pane that shows your objects. Use the left mouse button to rotate the view, the wheel on your mouse zooms in and out, and holding down the right mouse button and dragging pans the view.

Finally, on the right under the preview, you have the console. OpenSCAD shows messages here as it renders your objects. It will also show error messages if it encounters syntax errors or has problems with the objects.

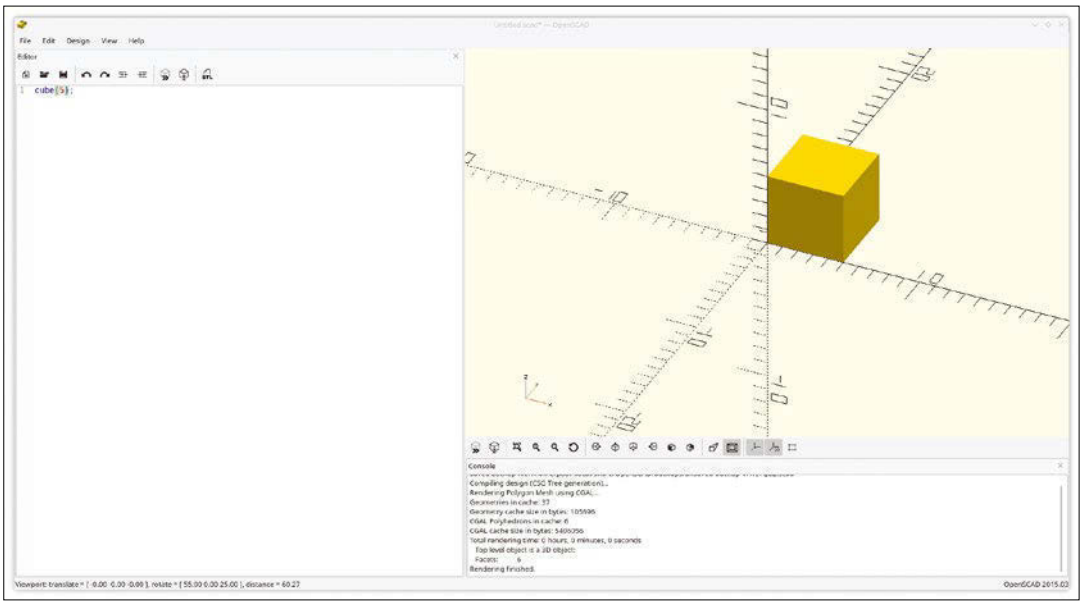


Figure 1: OpenSCAD uses a programming language to describe objects and their location to the renderer.

Although everything is important, the *File | Examples* option in the menu across the top of the window is especially worth exploring. The projects in there will help you get a better grasp of how OpenSCAD works.

Also check out the View menu, since it contains options for updating and changing what you see in the preview, along with keyboard shortcuts to make your workflow more agile. Some of the options available from View are also available in the toolbar under the preview pane.

This covers the essential interface bits. Things get really interesting when it's time to write scripts.

Language Basics

Here too the basics are, well, basic:

```
cube (5);
```

Listing 1: Modifying a Cube

```
// Method 1
translate ([5, 5, 5]) rotate([45, 45, 45])
                        cube (5);

// Method 2
translate ([5, 5, 5]) {
 rotate([45, 45, 45]) {
  cube (5);
 }
}

// Method 3
translate ([5, 5, 5])
  rotate([45, 45, 45])
    cube (5);
```

prints a cube with sides measuring 5 units. In my case, that translates to millimeters when I send it to my printer. The cube's lower left-hand corner on its front face is at the *[0, 0, 0]* position (Figure 1 again), so the cube is drawn to the right, up, and back from the viewer's initial point of view.

Although this is the default, you can add another parameter to the cube as follows:

```
cube (5, center = true);
```

to make the center of the cube coincide with *[0, 0, 0]*, and then the cube is drawn around its central reference point.

You can also create in a similar way spheres, cylinders, and polyhedrons, the latter being customized 3D shapes.

To make the object appear in the preview pane, press the render button in the toolbar at the top of the script editor pane (second from right), or in the

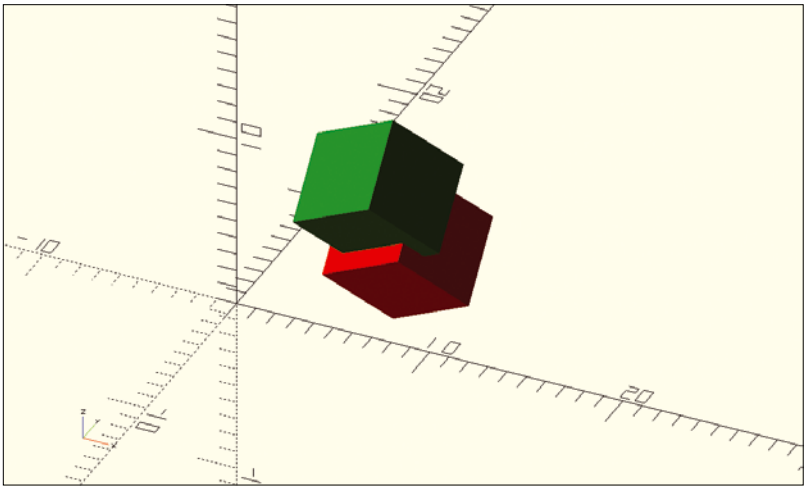


Figure 2: The order in which you apply modifiers to your objects can change where they are located.

toolbar under the preview pane (second from left). You can also hit the F6 key.

You can move things around with the `translate ()` function and rotate them with the `rotate()` function. To make this work, you embed the shape you want to modify within the function that modifies it. Listing 1 shows several ways you can do that.

- In *Method 1*, you simply put all the commands on one line, with the inner-most nested element last and moving outwards as you move backwards to the beginning of the line.
- *Method 2* is a bit clearer, as it uses a notation similar to JavaScript or C/C++, where you user braces (`{}`) to enclose nested blocks of code. As we are on the topic of C/C++, notice how the OpenSCAD's language uses C/C++ style comments, with `//` to indicate a line comment and `/* ... */` for multiple lines of comments.
- *Method 3* is similar to Python, as you use indentation at the beginning of the line to indicate which block of code is nested within which container.

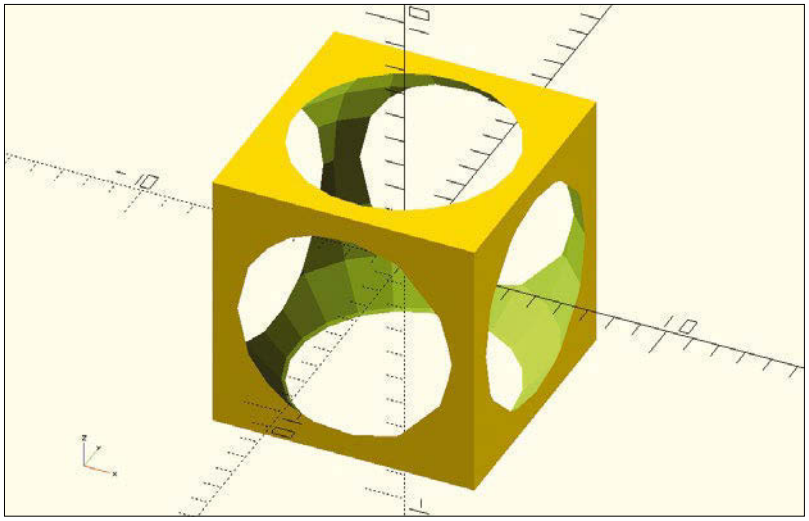


Figure 3: A cube that has had a sphere subtracted from it.

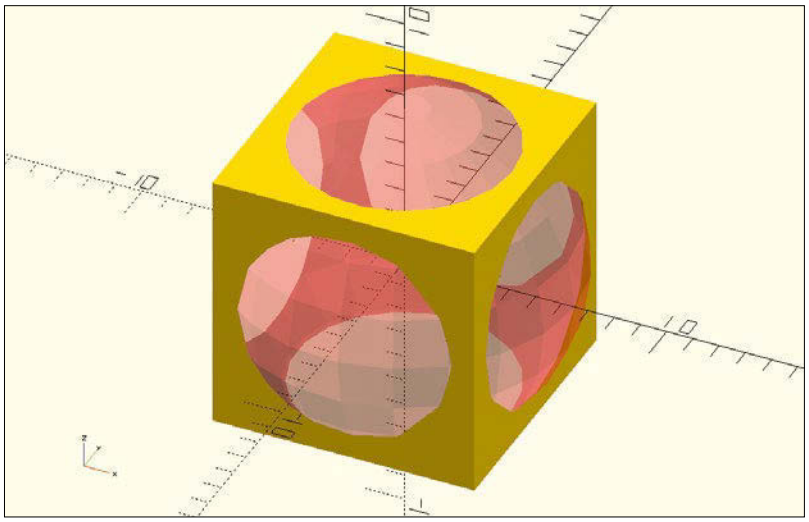


Figure 4: You can show subtracted pieces as translucent bodies by using the `#` in front of the commands that create them.

In our example, you may be tempted to think that there is some sort of invisible 3D cursor at work. When you write

```
translate ([x, y, z])
```

it moves said cursor to coordinate *x*, *y*, and *z*. Then, when the renderer executes

```
rotate ([a, b, c])
```

it rotates the cursor in 3D space and then draws the cube.

However, that is not what happens. In fact, the renderer executes commands inner-most first and moves out from there. Think of it like this: The renderer first locates the object it has to actually draw. The operation nearest to the object gets applied first, then the next to nearest, and so on, until all operations have modified the object. In the example above, that would be the rotation is applied first and then the translation.

This is important, because the order in which modifications are applied to a body matter. In fact,

```
translate ([5, 5, 5])
  rotate([45, 45, 45]) cube (5);
```

will give you a quite different result to:

```
rotate([45, 45, 45])
  translate ([5, 5, 5]) cube (5);
```

Figure 2 shows both. The green cube shows what happens when you rotate first and then translate, and the red cube shows what happens when you translate and then rotate. They are in different positions because with the green cube, you rotate the cube while it is still located at *[0, 0, 0]* and then move it to where you want it to be (which is what you probably wanted to do all along). With the red cube, you move the object and then rotate, but the rotation takes the original point as its reference, so it moves the cube on a 45 degree arc measuring 5, with its origin at *[0, 0, 0]*.

Adding and Subtracting

There are several ways of creating complex parts in OpenSCAD. One of them is by adding, intersecting, and subtracting one object from another. List-

Listing 2: Cube Minus Sphere

```
difference () {
  cube (10, center = true);
  sphere (6.5, center = true);
}
```

ing 2 shows a cube that has had a sphere subtracted from it (Figure 3).

The way this works is that the second object (`sphere ()`) is subtracted from the first (`cube ()`). Notice that, due to OpenSCAD's scripting nature, objects never "go away" when you operate on them. In fact, you can still show subtracted pieces by typing a `#` in front of them and then hitting *Preview* (the F5 key). The "ghost" objects will show up as a translucent piece in the display pane (Figure 4).

Another way of showing all the bodies that make up a piece is by marking the checkbox at *View* | *Thrown together* in the menu.

The other two Boolean operations available are `intersection ()`, in which you are left with an object that shares the space common to the two original objects (Figure 5), and `union ()`, which just melds both objects together.

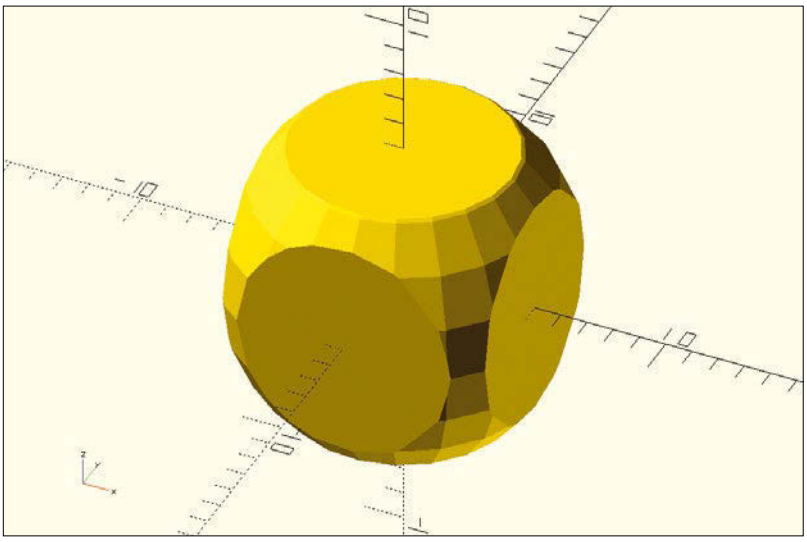


Figure 5: A cube intersected by a sphere leaves the space that both objects share.

Special Variables

You will notice that the sphere in the example above is looking a bit chunky. There are several inbuilt variables that can help with that. `$fn`, for example, sets the number of subdivisions in a curve. Use it like this

```
sphere (6.5, $fn=25);
```

to create a sphere with 25 subdivisions on each of its parallels.

In Figure 6, you can see what three spheres look like with, from left to right, the `$fn` variable set to `5`, `25`, and `50`. The "sphere" (to call it something) on the left has north and south poles and an equator made from pentagons. Meanwhile, the sphere with a `$fn` of `50` looks much more sphere-like.

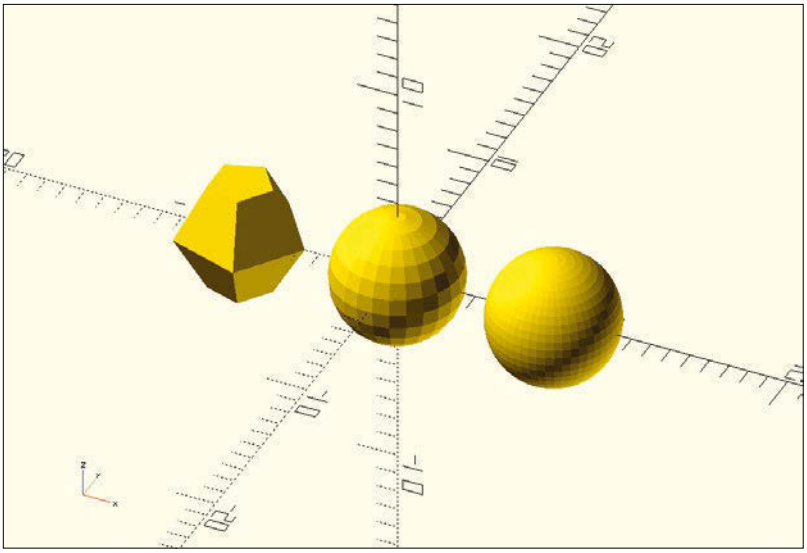


Figure 6: Use the `$fn` variable to make curved surfaces smoother.

You may be tempted to make `$fn` equal to `500` or `1000` to get a perfectly smooth sphere. Don't do that. Anything over `100` will eat up cycles from your CPU like candy, probably crashing your system. Besides, you are producing objects to print on a 3D printer. You'd need a crazily precise printer to be able to distinguish between a sphere with a `$fn` of `50` and another with a `$fn` of `500`.

Modules

At some point, things are going to get complex. You are going to have objects twisted around other objects, modifying other objects that are used in combination with yet more objects.

It helps then that openSCAD supports *modules* to organize stuff. Modules are simply a bunch of instructions that go together.

To illustrate how they work, let's make a funnel as shown in Figure 7.

A funnel is made up by two parts: a hollow, cut-off conical object that I am going to call the *hat* (I couldn't find formal names for the parts of a funnel, so I am improvising here), and a hollow cylinder I am calling the *pipe*.

Turns out both the hat and the pipe can be made entirely from cylinders, since the hat, for openSCAD, is a cylinder with two different diameters at each end. As you need it to be hollow, you make another cylinder with a slightly smaller radius at both ends and subtract that from the bigger one. The module for that would look like Listing 3.

Listing 3: Hat Module

```
module hat () {
  difference () {
    cylinder (r1=8, r2=2, h=10, center=true,
              $fn=50);
    cylinder (r1=7.5, r2=1.5, h=10,
center=true, $fn=50);
  }
}
```

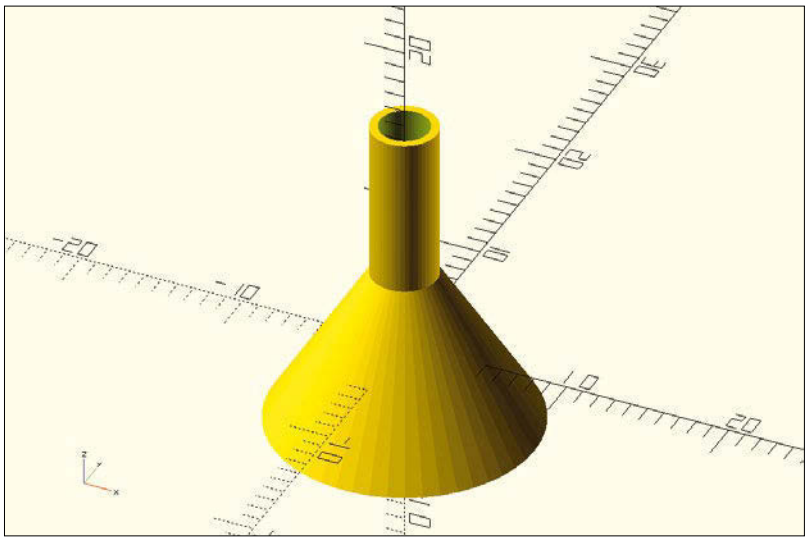


Figure 7: A funnel built using modules.

Listing 4: Pipe Module

```
01 module pipe () {
02   translate ([0,0,10]) {
03     difference () {
04       cylinder (r=2, h=10, center=true,
                   $fn=50);
05       cylinder (r=1.5, h=10, center=true,
                   $fn=50);
06     }
07   }
08 }
```

The pipe module is even simpler (Listing 4), but you do have to remember to translate the lower end of the pipe to the top of the hat (line 2).

With these two modules, you can create a `funnel ()` module that calls the other two (Listing 5, lines 17 to 20). Then call the `funnel ()` module to actually show something in the preview pane (Listing 5, line 22).

You can take this modularity further and save a set of modules into a separate *.scad* file altogether. You can then import them back into other files using either the `include` or the `use` methods.

The `include </path/to/file.scad>` method imports the whole thing, modules and commands that are not in modules, integrating the whole imported file into your current file. The `use </path/to/file.scad>` method, on the other hand, only imports the modules so you can use them as library modules with the code in your current file.

This means that, to get a funnel with `include`, all you have to do is:

```
include <funnel.scad>;
```

To do the same with `use`, you would also have to call the `funnel ()` method like this:

Listing 5: funnel.scad

```
01 module hat () {
02   difference () {
03     cylinder (r1=8, r2=2, h=10,
                 center=true, $fn=50);
04     cylinder (r1=7.5, r2=1.5, h=10,
                 center=true, $fn=50);
05   }
06 }
07
08 module pipe () {
09   translate ([0,0,10]) {
10     difference () {
11       cylinder (r=2, h=10, center=true,
                   $fn=50);
12       cylinder (r=1.5, h=10, center=true,
                   $fn=50);
13     }
14   }
15 }
16
17 module funnel () {
18   hat ();
19   pipe ();
20 }
21
22 funnel ();
```

```
use <funnel.scad>;
funnel ();
```

Conclusion

This article has covered the most essential bits and pieces of the OpenSCAD language. With this and the user manual's excellent documentation [5], you should be able to start creating your first 3D pieces.

That said, there is quite a lot more to learn regarding OpenSCAD. We'll be looking at more advanced things in next month's issue. We will also be creating a real piece to print and showing how you can go all the way from concept to real physical object.

Until then, have fun! ■■■

Info

[1] Thingiverse: *https://www.thingiverse.com/*

[2] Atelier: *https://atelier.kde.org/*

[3] OpenSCAD: *http://www.openscad.org*

[4] FreeCAD: *https://www.freecadweb.org/*

[5] OpenSCAD User Manual: *https://en.wikibooks.org/wiki/OpenSCAD_User_Manual*

Celebrating 25 Years of Linux!

ORDER NOW

Get 7 years of *Ubuntu User*

ON ONE DVD!

THE COMPLETE

UBUNTU user ARCHIVE

Over 3,000 PAGES! 7 GREAT YEARS OF UBUNTU USER

Searchable DVD!

All Content Available in Both HTML and PDF Formats

shop.linuxnewmedia.com

FEATURED EVENTS

Users, developers, and vendors meet at Linux events around the world. We at *Linux Magazine* are proud to sponsor the Featured Events shown here

For other events near you, check our extensive events calendar online at *http://linux-magazine.com/events.*

If you know of another Linux event you would like us to add to our calendar, please send a message with all the details to *events@linux-magazine.com*.

JAX DevOps Conference 2019

Date: May 14-17, 2019

Location: London, United Kingdom

Website: *https://devops.jaxlondon.com/*

JAX DevOps is a four-day conference for software experts featuring in-depth knowledge of the latest technologies and methodologies for lean businesses. Join the software delivery revolution for accelerated delivery cycles, faster changes in functionality, and increased quality in delivery.

openSUSE Conference 2019

Date: May 24-26, 2019

Location: Nuremberg, Germany

Website: *https://events.opensuse.org/conferences/oSC19*

The openSUSE Conference is the annual openSUSE community event that brings people from around the world together to meet and collaborate. The organized talks, workshops, and BoF sessions provide a framework around more casual meet ups and hack sessions.

ISC High Performance 2019

Date: June 16-20, 2019

Location: Frankfurt, Germany

Website: *https://www.isc-hpc.com/*

The ISC High Performance conference will bring together over 3,500 researchers and commercial users, and 160 exhibitors, ready to share their experiences with the latest technology and products of interest to the high performance computing (HPC) community.

Events

LinuxFest Northwest	April 26–28	Bellingham, Washington	https://lfnw.org/conferences/2019
Linux Storage, Filesystem & Memory Management Summit	April 30 – May 2	San Juan, Puerto Rico	https://events.linuxfoundation.org/events/linux-storage-filesystem-mm-summit-2019/
Red Hat Summit 2019	May 7–9	Boston, Massachusetts	https://www.redhat.com/en/summit/2019
Cloud Migration Summit	May 14	London, United Kingdom	https://www.cloudmigrationsummit.com/events/781-strategic-cloud-migration-summit
Open Source Data Center Conference (OSDC)	May 14–15	Berlin, Germany	https://osdc.de/
JAX DevOps Conference 2019	May 14–17	London, United Kingdom	https://devops.jaxlondon.com/
OSCamp Ansible	May 16	Berlin, Germany	https://opensourcecamp.de/
Linux Presentation Day 2019	May 18	Cities across Europe	http://www.l-p-d.org/
Cephalocon Barcelona	May 19–20	Barcelona, Spain	https://ceph.com/cephalocon/barcelona-2019/
2019 USENIX Conference on Operational Machine Learning	May 20	Santa Clara, California	https://www.usenix.org/conference/opml19
Kubecon + Cloud Native Con Europe 2019	May 20–23	Barcelona, Spain	https://events.linuxfoundation.org/events/kubecon-cloudnativecon-europe-2019/
openSUSE Conference 2019	May 24–26	Nuremberg, Germany	https://events.opensuse.org/conferences/oSC19
OSS 2019 (Open Source Systems)	May 26–27	Montreal, Canada	https://www.oss2019.org/
Secure Linux Administration Conference (SLAC)	May 27–29	Berlin, Germany	https://www.heinlein-support.de/secure-linux-administration-conference
ISC High Performance 2019	June 16–20	Frankfurt, Germany	https://www.isc-hpc.com/
KubeCon + CloudNativeCon China 2019	June 24–26	Shanghai, China	https://www.lfasiallc.com/events/kubecon-cloudnativecon-china-2019/

Images © Alex White, 123RF.com

CALL FOR PAPERS

We are always looking for good articles on Linux and the tools of the Linux environment. Although we will consider any topic, the following themes are of special interest:

- System administration
- Useful tips and tools
- Security, both news and techniques
- Product reviews, especially from real-world experience
- Community news and projects

If you have an idea, send a proposal with an outline, an estimate of the length, a description of your background, and contact information to *edit@linux-magazine.com*.

The technical level of the article should be consistent with what you normally read in *Linux Magazine*. Remember that *Linux Magazine* is read in many countries, and your article may be translated into one of our sister publications. Therefore, it is best to avoid using slang and idioms that might not be understood by all readers.

Be careful when referring to dates or events in the future. Many weeks could pass between your manuscript submission and the final copy reaching the reader's hands. When submitting proposals or manuscripts, please use a subject line in your email message that helps us identify your message as an article proposal. Screenshots and other supporting materials are always welcome.

Additional information is available at:
http://www.linux-magazine.com/contact/write_for_us.

NOW PRINTED ON recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

Authors

Contact Info

Editor in Chief
Joe Casad, jcasad@linux-magazine.com

Copy Editor
Amy Pettle

News Editor
Swapnil Bhartiya

Editor Emerita Nomadica
Rita L Sooby

Managing Editor
Lori White

Localization & Translation
Ian Travis

Layout
Dena Friesen, Lori White

Cover Design
Dena Friesen

Cover Image
© almoond and roystudio, 123RF.com

Advertising
Brian Osborn, bosborn@linuxnewmedia.com
phone +49 89 3090 5128

Marketing Communications
Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
2721 W 6th St, Ste D
Lawrence, KS 66049 USA

Publisher
Brian Osborn

Customer Service / Subscription
For USA and Canada:
Email: cs@linuxpromagazine.com
Phone: 1-866-247-2802
(Toll Free from the US and Canada)

For all other countries:
Email: subs@linux-magazine.com

www.linuxpromagazine.com – North America
www.linux-magazine.com – Worldwide

While every care has been taken in the content of the magazine, the publishers cannot be held responsible for the accuracy of the information contained within it or any consequences arising from the use of it. The use of the disc provided with the magazine or any material provided on it is at your own risk.

Copyright and Trademarks © 2019 Linux New Media USA, LLC.

No material may be reproduced in any form whatsoever in whole or in part without the written permission of the publishers. It is assumed that all correspondence sent, for example, letters, email, faxes, photographs, articles, drawings, are supplied for publication or license to third parties on a non-exclusive worldwide basis by Linux New Media USA, LLC, unless otherwise stated in writing.

Linux is a trademark of Linus Torvalds.

All brand or product names are trademarks of their respective owners. Contact us if we haven't credited your copyright; we will always correct any oversight.

Printed in Nuremberg, Germany by hofmann infocom GmbH on recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

Distributed by Seymour Distribution Ltd, United Kingdom

LINUX PRO MAGAZINE (ISSN 1752-9050) is published monthly by Linux New Media USA, LLC, 2721 W 6th St, Ste D, Lawrence, KS, 66049, USA. Periodicals Postage paid at Lawrence, KS and additional mailing offices. Ride-Along Enclosed. POSTMASTER: Please send address changes to Linux Pro Magazine, 2721 W 6th St, Ste D, Lawrence, KS 66049, USA.

Published monthly in Europe as Linux Magazine (ISSN 1471-5678) by: Sparkhaus Media GmbH, Zieblandstr. 1, 80799 Munich, Germany.

Approximate
On Sale Date

UK / Europe	May 04
USA / Canada	May 31
Australia	Jul 01

Issue 223 / June 2019

Git Tricks

The Git distributed version control system is a popular tool for open source developers around the world. Next month we highlight some tips and tricks for saving time and solving small problems with Git.

Image © Sergey Nivens, 123RF.com

Preview Newsletter

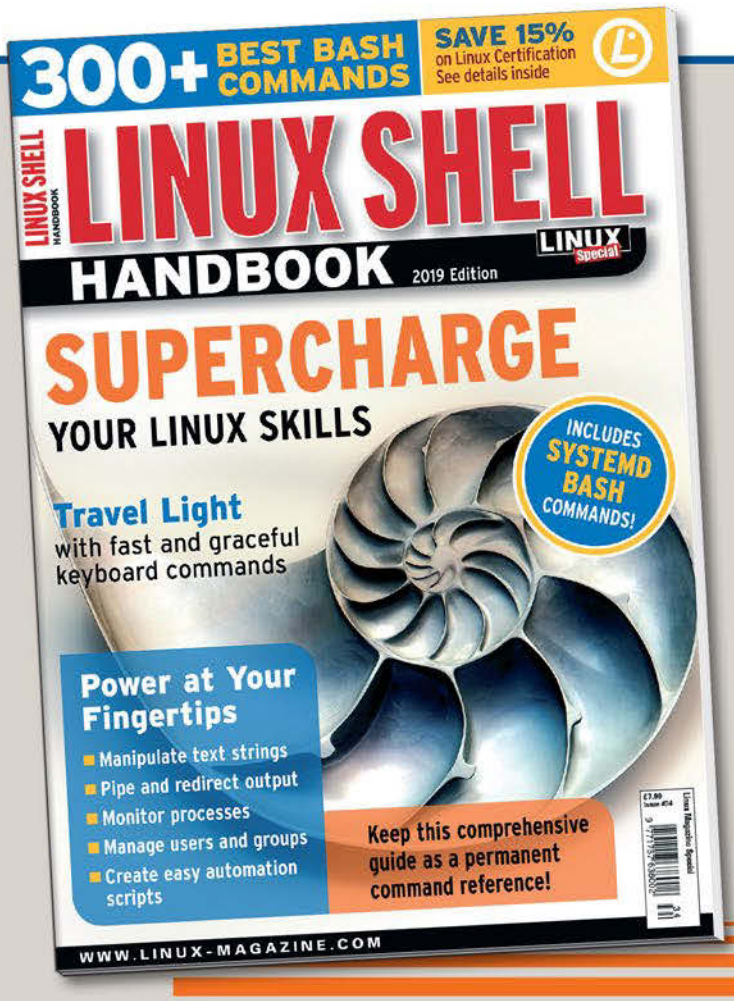
The Linux Magazine Preview is a monthly email newsletter that gives you a sneak peek at the next issue, including links to articles posted online.

Sign up at: ***www.linux-magazine.com/newsletter***

Shop the Shop shop.linuxnewmedia.com

EXPERT TOUCH

After selling out in 2018, the new *Linux Shell Handbook* is available now! The 2019 edition is packed with utilities for configuring and troubleshooting systems.



2019 Edition!

The *Shell Handbook* provides a comprehensive look at the world inside the terminal window. You'll learn to navigate, manipulate text, work with regular expressions, and customize your Bash settings.

Exclusive Bonus: Each issue includes a special discount saving you 15% off LPIC-1 certification from Linux Professional Institute (LPI). Look for the special code inside your copy!

Here's a look at just some of what you'll see in the new Shell Handbook:

- Customizing Bash
- Pipes and Redirection
- Regular Expressions
- Text Manipulation Tools
- Systemd
- Bash Scripting
- Networking Tools
- And much more!

Make the *Shell Handbook* your permanent desktop reference on the world of the terminal window.

ORDER ONLINE:
shop.linuxnewmedia.com/specials

SUPERMICRO

Save Money.
Save Mother Earth.

RESOURCE-SAVING SERVERS

Build your data center with technology that is
BIG on **SAVINGS** and **LOW** on environmental **IMPACT**
with Intel® Xeon® Scalable Processors

60% Space Savings | 50% Energy Savings | e-Waste Savings

intel XEON PLATINUM inside

Learn more at
www.supermicro.com/WeKeepITGreen

SUPERMICRO

© Super Micro Computer, Inc. Specifications subject to change without notice.
Intel, the Intel logo, Xeon, and Xeon Inside are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.
All other brands and names are the property of their respective owners.