

FREE  
DVD

FreeBSD 12.0  
64-bit

**Gnome Remote Desktop**  
Remote sharing for  
Wayland-based systems

**GIT TRICKS**  
VERSION CONTROL SECRETS  
FROM THE EXPERTS

LINUX  
MAGAZINE



# LINUX

## PRO MAGAZINE

JUNE 2019

# GIT TRICKS

Version control secrets from the experts

## Choose a Git GUI

### Bash 5.0

Long-awaited update for  
the classic hacker shell

### FreeBSD 12.0

Will this excellent  
server OS work as  
a desktop system?

### VolksPC

Run a Debian  
desktop over Android



### Google Firebase

Cloud storage for  
your IoT project

### Sudo Voodoo

Fine-tune system  
privileges

# LINUXVOICE

- Taking Notes with Joplin
- maddog: Freedom Zero
- Unforeseen Incidents:  
Point-and-Click Mysteries



## FOSSPicks

- Kdenlive
- Bookworm
- Plasma Pass

## Tutorials

- Shell Math
- More openSCAD

Issue 223  
June 2019  
US\$ 15.99  
CAN\$ 17.99

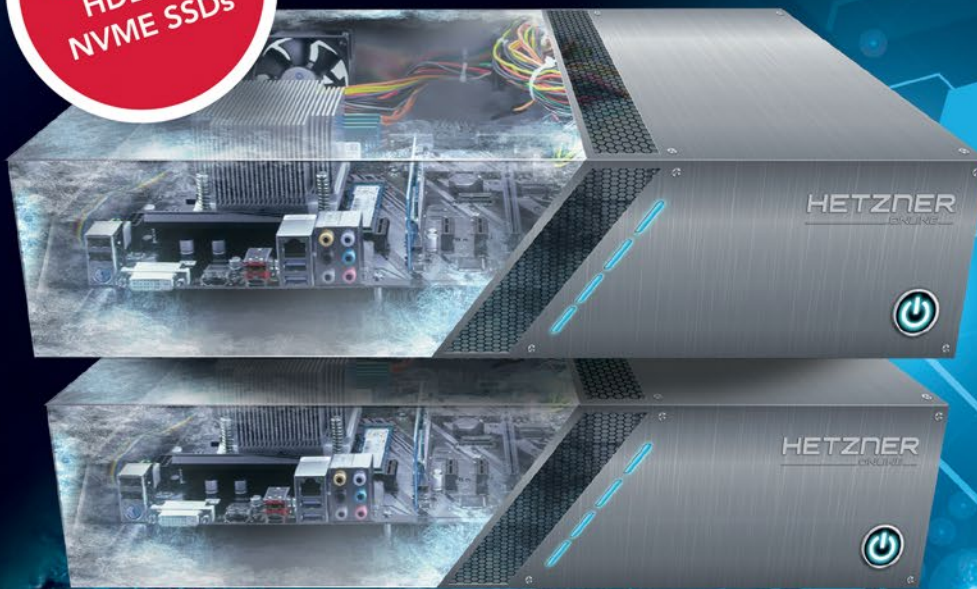


WWW.LINUXPROMAGAZINE.COM

# AN EXTREME REFRESH WITH OCTA-CORE POWER! DEDICATED ROOT SERVER EX62

High speed performance with the new  
Intel Core i9-9900K octa-core processor

ENTERPRISE  
HDDs or  
NVME SSDs



## Dedicated Root Server EX62

- ✓ Intel® Core™ i9-9900K Octa-Core incl. Hyper-Threading-Technology
- ✓ 64 GB DDR4 RAM
- ✓ 2 x 8 TB SATA Enterprise Hard Drive 7200 rpm
- ✓ 100 GB Backup Space
- ✓ Traffic unlimited
- ✓ Location Germany or Finland
- ✓ No minimum contract
- ✓ Setup Fee \$78

monthly from \$ **72.50**

## Dedicated Root Server EX62-NVMe

- ✓ Intel® Core™ i9-9900 Octa-Core 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 \$78

monthly from \$ **72.50**

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](http://www.hetzner.com)



# REACHING BACK

Dear Reader,

The Free Software community sometimes reminds me of those similar-but-mysteriously-not-similar parallel universes that turn up in science fiction novels. It looks just like the rest of the world, but occasionally you get a reminder that it really is a little different.

Linux users can download any tool they need for free. You're working along, and you think of some task you need to accomplish. Then you search for the best available tool and download it – it is yours. Or if you feel like trying a new Linux operating system, you look around, compare the features of the available systems, download an ISO, and start the installation.

Looking for an open source tool can feel a lot like shopping. If you're in a hurry (which many of us often are) or if you are in a less-than-mindful frame of mind (where many of us live), you might have the notion that downloading an operating system is similar to walking into a drugstore and buying some toothpaste, but in the Free Software universe, you need to operate with a little more care.

Linux Mint leader Clement Lefebvre posted a thoughtful message in the Mint blog recently to remind us that obtaining open source software isn't just a transaction – it is a two-way relationship that requires feedback in both directions [1].

The post points out that developers on Free Software projects like Mint aren't really in it for the money – what really motivates them is knowing that what they are doing matters and that it makes a difference for people. According to Clem, "developers need to feel like heroes."

He adds, "It's not always easy to achieve what we want; sometimes it's not even easy to define what we want to achieve. We can have doubts; we can work really hard on something for a while and then question it so much, we're not even sure we'll ship it. We can get demotivated, uncertain, depressed even by negative reactions or interactions, and it can lead to developers stepping away from the project, taking a break, or even leaving for good. And then sometimes simply seeing people enjoy what we did can boost an entire team, whether it's seeing happiness in an email/comment or getting a feeling of satisfaction after a constructive interaction, which leads to a fix or an implementation."

The next time you download some open source software, think about reaching back to the developers to let them know

what you think. By starting the conversation, you will be playing an active role in the development process. Show some appreciation, or at least, take them seriously enough to offer some constructive feedback.

As Clement Lefebvre puts it in his insightful post, "Feedback is something we should love, not something we should fear. It's what fuels our project and our development. When developers do things right, the changes they commit result in users being even more happy. When users do things right, the feedback they give results in developers being even more motivated."



Joe Casad,  
Editor in Chief



## Info

[1] Linux Mint Blog, March 2019: <https://blog.linuxmint.com/?p=3736>



## WHAT'S INSIDE

**This month** we share some advanced techniques for working with the famous Git remote version control system used by open source developers. Other highlights in this issue include:

- **Desktop FreeBSD** – is this highly secure and stable server operating system a worthy replacement for desktop Linux (page 30)?
- **Sudo Voodoo** – Sudo is a popular tool that lets you change user privilege levels on the fly. We show you some tricks for making sudo more powerful and more secure (page 38).

Look in MakerSpace for a study of Google's Firebase cloud platform for IoT, and check out LinuxVoice for a tutorial on Bash shell math.

## SERVICE

- 3 Comment
- 6 DVD
- 96 Featured Events
- 97 Call for Papers
- 98 Preview

## NEWS

### 08 News

- Software Freedom Conservancy Announces End to VMware Lawsuit
- Chef Goes All Open Source
- SUSE Spins off from Parent Company
- Gnome 3.32 Released
- NSA's Reverse Engineering Tool Released
- New Mirai Botnet Variant Discovered

### 11 Kernel News

- Enhancing KUnit
- Arguing over a Nonexistent Problem
- Cgroup Core Wars

### 14 SUSECON 2019

The SUSE community gathered in Nashville to explore the possibilities of SUSE's newfound independence.

## REVIEWS

### 30 Desktop FreeBSD

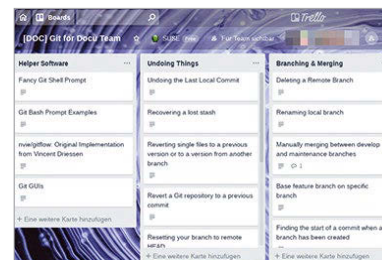
FreeBSD is a reliable and highly secure server operating system. We look at how FreeBSD fares as a desktop system.



## COVER STORIES

### 18 Git Tips and Tricks

The Git distributed version control system is a popular tool for managing open source development projects. If you know the basics of Git but are looking to learn the ways of the experts, read on for some useful Git tips and tricks.



### 24 Git GUIs

Complex Git projects sometimes require a better view of the dependencies and branches. Several tools offer GUI options for Git. We take a look at gitk, gitg, git-gui, and GitAhead.





## IN-DEPTH

### 34 Linux over Android

Linux desktop users can now use an estimated two million Android apps that were previously unavailable on Linux with VolksPC OS.

### 38 Sudo Voodoo

By taking the time to learn sudo's many options, you can make your system more secure.



### 40 Bash 5.0

It's a coincidence that the Linux kernel and Bash jumped to version 5.0 at about the same time. Linus assigns the numbers as he sees fit, but Bash changes its version to reflect major changes in the software. Here's what users can expect in Bash 5.0.

### 42 Bitnami

Setting up complex web-based services such as Drupal or Plone is never easy. Bitnami offers simplified installation with a preconfigured stack.

### 45 Charly's Column – Inav

During a long trek through the verbose syslog, really important warnings and errors are scattered along the path. Charly hires a tracker to help him search for clues: Log File Navigator.

### 46 Command Line – tail

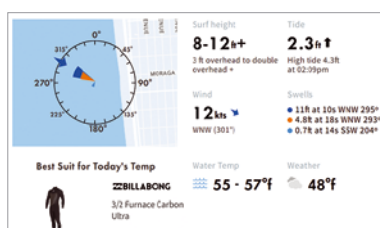
Tail's replacements, colortail and MultiTail, offer more sophisticated control over how your information is displayed.

### 50 Gnome Remote Desktop

In Fedora 29, you can enable Wayland remote desktop access in just a few mouse clicks.

### 54 Programming Snapshot – Colly

The Colly scraper helps Go developers collect data off the web.



## MAKERSPACE

### 58 Open Hardware – EOMA68 Laptop

Despite challenges, hardship, and delays, the EOMA68 laptop project is set to test its first PCBs.



### 62 Google Firebase

IoT projects on the Google Firebase platform promise future expandability and enhanced features.



SEE PAGE 6 FOR DETAILS

# LINUXVOICE

### 67 Welcome

This month in Linux Voice.

### 69 Doghouse – Freedom Zero

The GPL's "freedom zero" can be applied to more than just open-source software.

### 70 Game Review

In Unforeseen Incidents, a deadly virus and a spooky government quarantine are the prelude to an exciting point-and-click adventure for adults.



### 74 Joplin

If you need an open source alternative to the Evernote note taking tool, why not switch to Joplin?

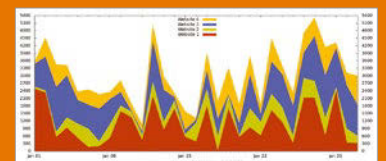
### 78 FOSSPicks

This month Graham looks at Eureka, Bookworm, Kdenlive 19.04, KStars 3.1, digiKam 6, CRRCSim, and more!



### 84 Tutorials – Shell Math

Although Bash is not the most advanced environment for doing and visualizing math, its power will surprise you. Learn how to calculate and display your results with shell scripts.



### 90 Tutorials – OpenSCAD

OpenSCAD lets you use simple scripts to build 3D images that you can send to your 3D printer.

# On the DVD

**FreeBSD 12.0**  
— 64-bit —

**ubuntu**  
MATE 18.10

ISSUE 223

ISSUE 223 JUNE 2019

**LINUX**  
MAGAZINE

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.

**TWO TERRIFIC  
DISTROS**

**DOUBLE-SIDED  
DVD!**

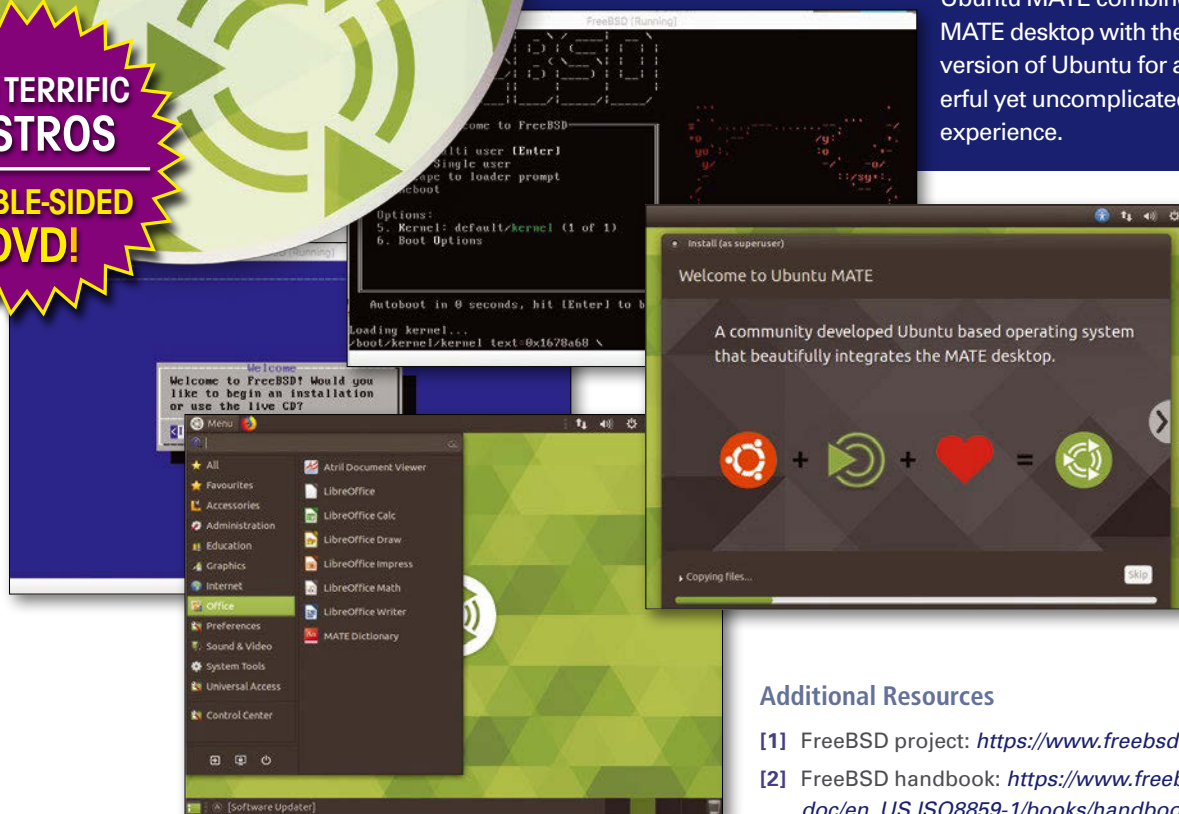
## FreeBSD 12.0

BSD (Berkeley Software Distribution) is a Unix-like operating system that has been in active development for more than 40 years. Today, several companies and community projects develop different versions of the BSD codebase. FreeBSD, a leading BSD alternative that has been around since 1993, is known for stability and security. FreeBSD is most frequently used as a server system. The system boots to a text-based user interface, but standard open source desktops are available through package repositories. (See the article on FreeBSD elsewhere in this issue.)

## Ubuntu MATE 18.10

The MATE desktop began as a fork of Gnome 2 developed for users who didn't like the changes brought in with the Gnome 3 shell. MATE is now a practical and easy-to-use desktop that competes with Gnome, KDE, Xfce, and other modern equivalents.

Ubuntu MATE combines the MATE desktop with the latest version of Ubuntu for a powerful yet uncomplicated user experience.



## Additional Resources

- [1] FreeBSD project: <https://www.freebsd.org/>
- [2] FreeBSD handbook: [https://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/](https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/)
- [3] FreeBSD forums: <https://forums.freebsd.org/>
- [4] Ubuntu MATE project: <https://ubuntu-mate.org/>
- [5] Ubuntu MATE tutorials: <https://ubuntu-mate.community/tips-tutorials-and-guides-index/14519>

Defective discs will be replaced. Please send email to [subs@linux-magazine.com](mailto: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.



# LISA<sup>19</sup>

October 28–30, 2019  
Portland, OR, USA  
[www.usenix.org/lisa19](http://www.usenix.org/lisa19)

LISA is the premier conference for operations professionals, where sysadmins, systems engineers, IT operations professionals, SRE practitioners, developers, IT managers, and academic researchers share real-world knowledge about designing, building, securing, and maintaining the critical systems of our interconnected world.

## Program Co-Chairs



Patrick Cable  
Threat Stack, Inc.



Mike Rembetsy  
Bloomberg

Sign up to receive updates at [www.usenix.org/lisa/linuxpro](http://www.usenix.org/lisa/linuxpro),  
and follow @LISAconference on Twitter.

# NEWS

Updates on technologies, trends, and tools

## THIS MONTH'S NEWS

08

- Software Freedom Conservancy Announces End to VMware Lawsuit
- Chef Goes All Open Source

09

- SUSE Spins off from Parent Company
- Gnome 3.32 Released
- More Online

10

- NSA's Reverse Engineering Tool Released
- New Mirai Botnet Variant Discovered

### Software Freedom Conservancy Announces End to VMware Lawsuit

Linux developer Christoph Hellwig has announced that he is discontinuing his lawsuit against VMware for non-compliance with the terms of the GPL. Hellwig and the Software Freedom Conservancy accused VMware of including GPLed code associated with vmklinux into VMware's proprietary vSphere product. A German appeals court dismissed the case on February 28. Hellwig and the Software Freedom Conservancy have decided they will not appeal the case further in German courts.

The judge appears to have decided the case on procedural grounds without taking on the larger questions related to the GPL and the power of the copyleft protection. The questions hinged around whether the plaintiffs had successfully proven that the code was present in VMware's code base and that the use of the code was non-compliant. VMware maintains that the vmklinux code is a separate component that does not force release of vSphere under the copyleft requirement.

Software Freedom Conservancy executive director Karen Sandler expressed disappointment, "VMware knew what they were doing was wrong but continued to generate revenue by infringing copyrights in Linux, while slowly working toward non-infringement."

The Free Software community has always been more focused on achieving compliance than on punishment or punitive damages. By that standard, the case appears to have succeeded despite the outcome. VMware announced that it will remove vmklinux from vSphere and hopes to accomplish this "...in an upcoming major release."



CC BY-SA 3.0

### Chef Goes All Open Source

The Chef automation tool, a popular solution for DevOps IT management scenarios, has announced that it will become a 100% open source platform. In the past, the basic Chef application was available in open source form, but the company also provided several enhancements and add-on tools with proprietary licenses. Rather than building proprietary tools around an open source core, Chef will now open source all of its software under an Apache 2.0 license.

According to Chef CEO Barry Crist, (<https://blog.chef.io/2019/04/02/chef-software-announces-the-enterprise-automation-stack/>) "Over the years we have experimented with and learned from a variety of different open source, community, and



commercial models, in search of the right balance. We believe that this change, and the way we have made it, best aligns the objectives of our communities with our own business objectives. Now we can focus all of our investment and energy on building the best possible products in the best possible way for our community without having to choose between what is “proprietary” and what is “in the commons.”

This move toward free software does not mean that Chef is changing its focus on commercial enterprise customers. Instead, the change underscores the modern reality that the enterprise is more about services than it is about code. The company has also announced a commercial version called Chef Enterprise Automation Stack (<https://www.chef.io/products/enterprise-automation-stack/>) that will combine the open-source software with enterprise-grade warranties, indemnifications, and support.

## SUSE Spins off from Parent Company

SUSE has completed its move from Micro Focus to EQT, a growth investor firm. As the focus is shifting towards moving up in the stack, towards the cloud, there is consolidation happening in the market. While Red Hat has become a unit of IBM, SUSE is heading towards becoming an independent entity.

Many would argue that post-IBM acquisition of Red Hat, SUSE has become the ‘biggest’ Linux vendor. While Linux is still the core of SUSE business, the company has built a massive portfolio of emerging technologies (<https://www.youtube.com/watch?v=YgaxoIQ2QLo&t=1s>) like cloud, containers, and IoT.

“Current IT trends make it clear that open source has become more important in the enterprise than ever before,” said SUSE CEO Nils Brauckmann. “Our genuinely open, open source solutions, flexible business practices, lack of enforced vendor lock-in and exceptional service are more critical to customer and partner organizations, and our independence coincides with our single-minded focus on delivering what is best for them.”

To continue its momentum, SUSE has expanded its executive team. Enrica Angelone has become the new chief financial officer, and Sander Huyts is SUSE’s new chief operations officer. Thomas Di Giacomo, formerly chief technology officer for SUSE, is now president of Engineering, Product and Innovation.

The company believes ([https://www.prnewswire.com/news-releases/suse-completes-move-to-independence-reaffirms-commitment-to-customers-partners-and-open-source-communities-as-industrys-largest-independent-open-source-company-300813231.html?tc=eml\\_cleartime](https://www.prnewswire.com/news-releases/suse-completes-move-to-independence-reaffirms-commitment-to-customers-partners-and-open-source-communities-as-industrys-largest-independent-open-source-company-300813231.html?tc=eml_cleartime)) that EQT’s backing and SUSE’s independent status will enable the company’s continued expansion as advanced innovation drives growth in SUSE’s core business as well as in emerging technologies, both organically and through add-on acquisitions.



## Gnome 3.32 Released

The Gnome community has announced (<https://www.gnome.org/news/2019/03/gnome-3-32-released/>) the release of Gnome 3.32. The release incorporates 26,438 changes (made by approximately 798 contributors), including new features improvements and performance enhancements.

Gnome users will be greeted with a refreshed visual style for the user interface and icons. Gnome 3.32 brings support for fractional scaling as an experimental option, which will satisfy those users who have HiDPI monitors.



## MORE ONLINE

### Linux Magazine

[www.linux-magazine.com](http://www.linux-magazine.com)

### Linux Administration Focus

<http://www.linux-magazine.com/tags/view/administration>

### Audacity: An Open Source Audio Editor and Recorder • Ken Hess

If you are looking for an audio editing and creation tool, Audacity lets you create and edit professional audio files at no cost, all while using your existing hardware.

### Managing Network Bandwidth with Trickle Mayank Sharma

The trickle command-line utility helps you shape network traffic.

### ADMIN HPC

<http://www.admin-magazine.com/HPC/>

### OpenMP • Jeff Layton

The HPC world is racing toward Exascale, resulting in systems with a very large number of cores and accelerators.

### ADMIN Online

<http://www.admin-magazine.com/>

### Cryptographic Key Access in the Cloud Thorsten Scherf

Cryptographic keys, usually available locally but not on remote computers, can be accessed for use in cloud environments.

### Open Source Customer Relationship Management Software • Rick Timmis

SuiteCRM 7 is a powerful open source enterprise customer relationship management solution.

### Validating Docker Containers Tim Schürmann

A new test tool by Google lets you peek inside Docker containers, so you can make sure they hold exactly what you expect.

To streamline the user experience across apps, Gnome has removed the Application menu and moved its contents to a primary menu located within the application window.

While there are popular Chromium and Firefox web browsers, Gnome also offers its own web browser called Web. Gnome's browser now comes with an automation mode, which allows users to control web applications using WebDriver. Speaking of control, Gnome has improved Touchpad support. The touchpad support is still not on par with MacOS, but users can swipe left or right to go back or forward through browsing history.

Gnome 3.32 has improved security. The Settings tools feature a new panel called Application that offers permissions control for various applications, including installed Flatpak packages. Gnome Software (applications manager for Gnome) has improved handling for apps available from multiple sources, including distribution repositories and Flatpak.

Arch and Gentoo users can already test Gnome 3.32; openSUSE Tumbleweed is expected to get the update soon.

## ■ NSA's Reverse Engineering Tool Released

The National Security Agency (NSA) has released the source code of its software reverse engineering tool, GHIDRA, on GitHub.

GHIDRA is NSA's classified, Java-based reverse engineering framework, which the agency uses to disassemble binaries of software to understand its functionality. It's a critical tool to reverse engineer malicious software such as malware. GHIDRA also features a GUI and can run on Linux, macOS, and Windows.

"With this release, developers will be able to collaborate by creating patches, and extending the tool to fit their cybersecurity needs," said the blog post.

According to NSA, the source code repository includes instructions to build on all supported platforms. GHIDRA source code includes a suite of software analysis tools. Some of its core capabilities include disassembly, assembly, decompilation, graphing, and scripting. It supports a wide variety of processor instruction sets and executable formats and can be run in both user-interactive and automated modes.

Security analysts can use the source code to develop their own GHIDRA plug-in components and/or scripts using the exposed APIs.

The source code is available for download at [ghidra-sre.org](https://ghidra-sre.org) along with the 9.0.2 patch.

## ■ New Mirai Botnet Variant Discovered

Cybersecurity experts at Unit 42 (<https://unit42.paloaltonetworks.com/mirai-compiled-for-new-processor-surfaces/>) have discovered a new variant of the Mirai botnet that targets Linux powered IoT devices. The botnet took a huge chunk of Internet down in 2016, including web hosting provider OVH and DNS provider Dyn.

The new variant targets embedded devices like routers, network storage devices, NVRs, and IP cameras. Unit 42 found this new variant targeting enterprise WePresent WiPG-1000 Wireless Presentation systems and have discovered it in LG Supersign TVs.

"This development indicates to us a potential shift to using Mirai to target enterprises. The previous instance where we observed the botnet targeting enterprise vulnerabilities was with the incorporation of exploits against Apache Struts and SonicWall," wrote Ruchna Nigam of Unit 62 in a blog post, "In addition to this newer targeting, this new variant of Mirai includes new exploits in its multi-exploit battery, as well as new credentials to use in brute force against devices."

This Mirai variant has added 11 new exploits, taking the total exploits to 27.

Enterprise customers need to focus on the security of their network and IoT devices running within their network. They should embrace some best practices, including changing the default password; be aware of what IoT devices are living within their network and also ensure they are fully patched.



# Zack's Kernel News



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

*By Zack Brown*

## Enhancing KUnit

Since its introduction by Brendan Higgins in October 2018, the KUnit unit testing framework has seen a lot of activity, and Brendan has implemented a lot of enhancements and documentation. Recently, he submitted some patches to support the bailing out of any given unit test, which he said was needed for implementing assertions. An assertion is a useful and cute debugging tool, where at a certain point in your code you claim that a particular value or condition is true, because at that point in the code you really expect it to be true. You're not branching execution based on a conditional; you're just saying, "I think the world looks like this." Then, if it doesn't look like "this," the program can crash or throw an exception or do whatever you told it to do in case an assertion turns out to be false.

You can see how unit tests and assertions might get in each other's way. If you write a unit test to feed bogus inputs into a function to make sure it handles that case properly, the bogus inputs could end up triggering the assertion and short circuiting your goal, which is to test the actual code, not the assertion.

Brendan wanted unit tests to abort in the event of triggering an assertion within the code. At that point, there would be no reason to continue that particular test. As he put it at some point deep in the conversation, "The idea with assertions is that you use them to state all the preconditions for your test. Logically speaking, these are the premises of the test case, so if a premise isn't true, there is no point in continuing the test case, because there are no conclusions that can be drawn without the premises."

Near the periphery of his goals, however, in the error-handling portion of his code, he tested to see whether the unit test had failed to actually abort as intended, in which case his patch called `BUG()`. To that, Frank Rowand remarked, "`BUG()`, which is a panic, which is crashing the system, is not acceptable in the Linux kernel. You will just annoy Linus if you submit this."

Brendan replied that he thought `BUG()` would be OK, since KUnit would never be used in a production kernel. Consequently, there was no risk of panicking any production system. Also, he said, if KUnit ever did get to this particular point of execution, everything would go kablooeey and trying to proceed normally would be a no-no.

But Frank replied that, in fact, even in a non-production kernel, it would be unacceptable to intentionally produce a panic unless it were the only way to avoid data corruption. He also added that it would be wrong to assume KUnit would not be compiled into production systems. It could very well happen, he said.

Even if the unit test failed, Frank also said that didn't mean it would be best to abort the test as a whole. He said, "The remainder of the unit tests can still run. There may be cascading failures, but that is OK."

Brendan said he didn't want to dispute Frank's point, but he did also reply for clarity that "the path where `BUG()` gets called only happens if there is a bug in KUnit itself, not just because a test case fails catastrophically." And he went on, "This should never actually happen, even if the assertion that called it was violated. `KUNIT_ASSERT_*` just says, 'this is a prerequisite property for this test case'; if it is violated, the test case should fail and bail, because the preconditions for the test case cannot be satisfied. Nevertheless, other tests cases will still run."

There was a little more discussion, but nothing conclusive. Several folks went back and forth on how best to handle various cases, like what null pointers traditionally meant in a testing context, and whatnot.

I love watching the Linux kernel ecosystem as it attracts more and more symbiotic entities to it. One of the greatest of these is of course the C compiler, but so are all aspects of revision control, debugging, and testing, including KUnit. It's clear this is a project that's still in the relatively early stages of what will be a long and elaborate life. For me, it's fun to watch KUnit's tendrils descending

## 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**.

into the various inner reaches of the kernel development process.

## Arguing over a Nonexistent Problem

A strange argument arose over a nonexistent problem in an inscrutable part of the kernel that turns out to be utterly relevant to all things. Andrea Parri started the whole debate with a patch to affect the ordering of memory reads during concurrent operation. Under a hypothetical circumstance, one read might depend on another, yet be performed in the reverse order. Andrea's patch was intended to fix this.

Paul McKenney had no objection and was all set to apply the patch and feed it up to Linus Torvalds, when Peter Zijlstra pointed out that, in fact, the particular hypothetical situation being fixed by Andrea's patch could never be part of an actual physical computer.

Andrea agreed wholeheartedly that there was no way that even the vilest and most depraved mind could ever construct a monstrous perversion of space and time so as to produce a computer capable of triggering the bug that he had fixed. By way of compromise, he said that in the future the bug could always be reinserted into the code if that's what the developers really wanted.

Peter was having none of this sophistry. He said flatly, "What I do object to is a model that's weaker than any possible sane hardware."

And in support of this, Will Deacon remarked, "Whilst I completely agree that relying on the ordering provided by `dep ; > rfi` is subtle and error prone, having it forbid the outcome above appeals to a hardware-based mindset of how memory ordering works. In the kernel community, I would posit that the majority of developers are writing code with the underlying hardware in mind and so allowing behaviors in the memory model, which are counter to how a real machine operates, is likely to make things more confusing, rather than simplifying them!"

Alan Stern suggested letting Andrea's impossible-to-hit bug fix go into the kernel, but to also retain the code comment for the behavior that was being replaced, in order to "explain that the operational model predicts this ordering, but it has been left out of the LKMM for practical (not technical) reasons."

Andrea backed off, saying, "I won't try to push this patch further." But he did feel that Peter in particular was being a bit of a stickler, always insisting that kernel features should only be implemented if there were actual users to use them.

Meanwhile, Paul veered into untapped realms of theory, saying that since Andrea's patch was primarily related to issues of concurrent code execution, and reading and writing:

*"We had some debates about this sort of thing at the C++ Standards Committee meeting last week.*

*"Pointer provenance and concurrent algorithms, though for once not affecting RCU! We might actually be on the road to a fix that preserves the relevant optimizations while still allowing most (if not all) existing concurrent C/C++ code to continue working correctly. (The current thought is that loads and stores involving inline assembly, C/C++ atomics, or volatile get their provenance stripped. There may need to be some other mechanisms for plain C-language loads and stores in some cases as well.)*

*"But if you know of any code in the Linux kernel that needs to compare pointers, one of which might be in the process of being freed, please do point me at it. I thought that the `__smp_call_function()` code fit, but it avoids the problem, because only the sending CPU is allowed to push onto the stack of pending `__smp_call_function()` invocations.*

*"That same concurrent linked stack pattern using `cmpxchg()` to atomically push and `xchg()` to atomically pop the full list would have this problem. The old pointer passed to `cmpxchg()` might reference an object that was freed between the time that the old pointer was loaded and the time that the `cmpxchg()` executed. One way to avoid this is to do the push operation in an RCU read-side critical section and use `kfree_rcu()` instead of `kfree()`. Of course, code in the idle loop or that might run on offline CPUs cannot use RCU, plus some data structures are not happy with `kfree_rcu()` delays, so..."*

Peter tried to wrap his head around what Paul was asking, but had no sort of luck until he found the scholarly article, "Exploring C Semantics and Pointer Provenance" [1], which begins with the lovely and encouraging pronouncement:

*"C values cannot be treated as either purely abstract or purely concrete entities:*

*The language exposes their representations, but compiler optimizations rely on analyses that reason about provenance and initialization status, not just runtime representations. The ISO WG14 standard leaves much of this unclear, and in some respects differs with de facto standard usage – which itself is difficult to investigate."*

Peter read this and the rest of that baleful article and then staggered back to the Linux Kernel Mailing List saying, "that's all massive bong-hits. That's utterly insane. Even the proposed semantics are crazy; they include UB and are therefore broken on principle."

Peter cast about desperately right and left, saying, "We need to find a GCC person to find/give us a knob to kill this entire class of nonsense. This is just horrible broken shit."

Paul calmly replied:

*"Yes, this all is a bit on the insane side from a kernel viewpoint. But the paper you found does not impose this; it has instead been there for about 20 years, back before C and C++ admitted to the existence of concurrency. But of course compilers are getting more aggressive, and yes, some of the problems show up in single-threaded code.*

*"The usual response is 'then cast the pointers to `intptr_t`!', but of course that breaks type checking.*

*"There is an effort to claw back the concurrency pieces, and I would be happy to run the resulting paper past you guys.*

*"I must confess to not being all that sympathetic to code that takes advantage of happenstance stack-frame layout. Is there some reason we need that?"*

Peter replied:

*"Not that I'm aware; but if it gets this 'obvious' case wrong, I worry what else it gets wrong.*

*"At the very least, we should get this fixed and compile a kernel with the fixed compiler to see what (if anything) changes in the generated code and analyze the changes (if any) to make sure we were OK (or not).*

*"I mean, yes that example is UB, but the result is also clearly batshit insane."*

So, there you have it – from fixing a nonexistent bug, to questioning the nature of reality and one's own place in the universe.

## Cgroup Core Wars

Andrey Ryabinin had a bone to pick with the cgroup code. In particular, he didn't



like the way Linux handled memory limits. Any time there was more than one virtual machine running, if anything at all hit any kind of memory limit, Linux would attempt to reclaim memory from all cgroups. Andrey didn't like this at all, because "the cgroup that allocates more often always wins. E.g., job that allocates a lot of clean rarely used page cache will push out of memory other jobs with active relatively small all in memory working set."

He pointed out that he knew about the `memcg` controls, designed to help avoid that very circumstance. But he said these were insufficient, because "memory cgroups are very hard to configure right, because it requires precise knowledge of the workload, which may vary during the execution. E.g., setting memory limit means that job won't be able to use all memory in the system for page cache even if the rest [of] the system is idle."

Essentially, as Andrey said, the current situation required painstakingly error-prone configuration of each and every cgroup.

He submitted a patch that he felt would do better. Instead of manually tweaking each cgroup, he wanted to fix the way memory was reclaimed in the first place. In particular, instead of Linux reclaiming RAM from all cgroups equally, it would reclaim only inactive memory pages – and only from cgroups that were hoarding a lot of inactive pages. Only when everyone's list of inactive pages was low would the kernel

then revert to its previous behavior, shrinking all cgroups equally.

Rik van Riel liked this whole concept, but had some technical suggestions for Andrey's patch. Johannes Weiner also strongly supported Andrey's patch and also offered some technical suggestions for improvements.

Roman Gushchin, however, had doubts. As he put it, "What's considered active and inactive depends on memory pressure inside a cgroup. Actually active pages in one cgroup (e.g., just deleted) can be colder than inactive pages in another (e.g., a memory-hungry cgroup with a tight memory, `max`)." He added, "Also a workload inside a cgroup can to some extent control what's going to the active LRU. So it opens a way to get more memory unfairly by artificially promoting more pages to the active LRU. So a cgroup can get an unfair advantage over other cgroups."

Andrey objected to this description of fairness. He said, "If fair means to put equal reclaim pressure on all cgroups, then, yes, the patch increases such unfairness, but such unfairness is a good thing." He presented, as an example, "Cgroup A with active job loading a big and active working set, which creates high memory pressure, and cgroup B – idle (no memory pressure) with a huge not used cache. It's definitely preferable to reclaim from B rather than from A."

Roman insisted that while Andrey was logically correct, there was still no clear

way to distinguish between active and inactive pages, and thus, no clear way to know that Andrey's implementation would do the right thing – or better than existing code – in any given instance. He concluded, "So it means that depending on memory usage pattern, some workloads will benefit from your change, and some will suffer."

There was no conclusion during the thread. And it's not at all clear whether Andrey's patch will ultimately make it into the kernel. Anytime something depends on average use-case patterns across the known universe, coupled with uncertain mechanisms to distinguish between the thing we want and the thing we don't want, we're probably looking at a long and controversial debate.

At the same time, Andrey's initial memory theft exploit may indeed be real – giving preference to the cgroup that allocates and wastes the most memory and pushing other cgroups out entirely. Depending on how you look at it, that could be considered a security hole. If so, there may start to be real pressure to address the problem one way or another, either with something along the lines of Andrey's patch, or something else that avoids the exploit. ■■■

## Info

[1] "Exploring C Semantics and Pointer Provenance":

<http://www.open-std.org/jtc1/sc22/wg14/www/docs/n2311.pdf>

# IT Highlights at a Glance



- Linux Update
- ADMIN Update
- ADMIN HPC

Keep your finger on the pulse of the IT industry.

**Too busy to wade through press releases and chatty tech news sites?**  
Let us deliver the most relevant news, technical articles, and tool tips – straight to your Inbox.

Admin and HPC: <https://bit.ly/HPC-ADMIN-Update>

Linux Update: <https://bit.ly/Linux-Update>

Photo by Andrew Neal on Unsplash

New energy at SUSE's annual SUSECON conference

# Chameleon Goes Country

The SUSE community pondered new challenges and the path ahead at its annual SUSECON convention in Nashville, Tennessee. *By Brian Osborn*



**T**he past 12 months have been a whirlwind of change for SUSE. Former parent company Micro Focus announced last July that it was spinning off SUSE as an independent company [1], and since then, the leading European enterprise Linux vendor has been in a flurry of reinvention. In addition to

embracing a new emphasis on growth, SUSE is also finding its way through the changes in the Linux space following IBM's acquisition of Red Hat.

SUSE's life as an independent company got off to a roaring start as the annual SUSECON kicked off in Nashville, USA just days after SUSE legally split from Micro Focus. The excitement of SUSE employees, partners, and customers energized the event, as visitors contemplated the possibilities of SUSE's newfound independence.

## Strategies and Trends

Future strategy was a major topic at this year's SUSECON. SUSE CEO Nils Brauckmann covered all the important facts in the opening press conference: SUSE had about \$400 million in revenue in 2018, which represents approximately 15% growth over 2017. This revenue growth was accompanied by strong growth in the size of the SUSE staff. The company added more than 300 employees in the last 12 months. Overall, SUSE now has over 1750 employees located in 34 countries and spanning 75 nationalities.

The company is preparing for continued strong growth in both revenue and employees over the next several years. EQT [2], the private equity firm that bought SUSE from Micro Focus for US\$2.54 Billion, is a growth investor, meaning that its focus is on enabling



### Author

Brian Osborn is the publisher of *Linux Magazine*.



companies to expand to meet their value potential. SUSE plans to grow organically with new and existing products, but the company also plans to pursue an aggressive acquisition strategy. Key acquisition targets are likely to be companies with strong technology offerings in and around SUSE's Kubernetes-based cloud and container solutions.

SUSE's strategy follows the mantra of digital transformation "From the Edge to the Core to the Cloud," focusing on containers and software-defined infrastructure, with Kubernetes being the most-mentioned technology at the event.

Interestingly, SUSE itself is facing many of the same challenges as its customers. Having just been spun out of a larger organization, SUSE runs a mix of legacy and more modern IT solutions, many of which might not be ideally suited to an independent IT company, and the IT team is now pursuing its own digital transformation.

Since independence, SUSE has also gone through some restructuring in an effort to prepare for the planned strong growth phase. According to Thomas Di Giacomo, President of Engineering, Product, and Innovation, SUSE has reorganized their product development teams to no longer be organized by task, but rather to follow the agile, DevOps model of locating development, QA, documentation, and product management in product-based teams.

## The Conference

SUSECON 2019 kept to the tried-and-tested mix of strategic overview, hardcore technical information, and good fun. SUSE parody videos [3], which started several years ago on a whim, have become serious business. These music videos served as pre-keynote entertainment every morning and were featured in many other spots throughout the event. The SUSE band (almost all members are employees) rocked the Wildhorse Saloon for the event party on Wednesday night [4], playing many of the parody songs as well as rock classics. How many employee bands can say they have played a 1000+ person venue in Music City? Probably no others, but this band deserves the honor – they are really that good.

Technology was still the centerpiece of the event, with over 150 sessions on everything from developer tools to ma-

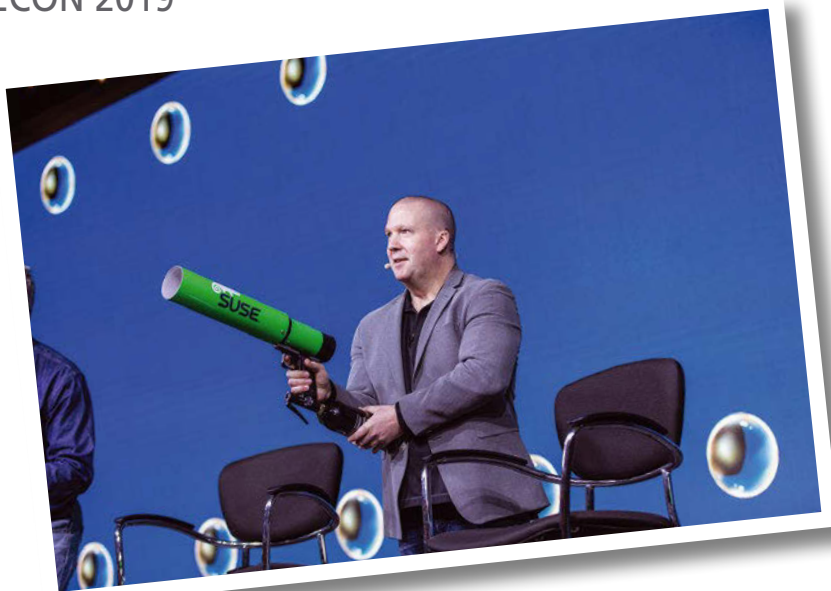
chine learning. Interestingly, many of the talks were submitted by SUSE partners or customers, giving the conference a strong base of real-life experience. The "hallway track" seemed to focus on the new opportunities SUSE will have as an independent company. Kubernetes was surely the hottest topic – everything is being containerized! The New Buzzword award goes to "multi-cloud," a term that refers to organizations purposefully spreading their cloud workloads over various cloud solutions to avoid being too dependent on one provider. As vendor lock-in has become something enterprises are looking to avoid at all costs, more organizations are moving to a multi-cloud strategy.

## Docs and Demopalooza

The closing session at SUSECON is Demopalooza, where SUSE engineers risk their reputations by







showing the power of some of the latest solutions through live demos in front of the entire crowd. The nerd factor was high as the lead-in to the evening featured actual concert footage of Kraftwerk's Computerworld! The electronic music theme continued throughout the presentation, as much of the demo was based on electronic music that was being spontaneously programmed on-stage. Despite technical challenges, such as two different power outages, the demos did work. In an odd way, they may have worked too well: the live container deployment necessary to support the musical endeavors on the other end of the stage occurred so quickly that some in the audience didn't even see it happen – "wait, what?"

The Friday after SUSECON featured a handful of community events. Most notably, the SUSE Documentation Team

hosted the first SUSE Doc Day, where team members and volunteers worked together to improve documentation on areas such as container technologies, OpenStack Cloud, and Open Build Service. The openSUSE community also hosted a summit on Friday and Saturday, offering a wide range of sessions, including openSUSE on ARM and setting up iSCSI.

### What's Next?

This year's SUSECON marked the switch from Fall to Spring; the tradition of alternating between

North America and Europe is said to continue, although the location of the next SUSECON has not yet been announced. The rumor mill was leaning heavily toward Barcelona or Dublin – we'll be there in any case! ■■■

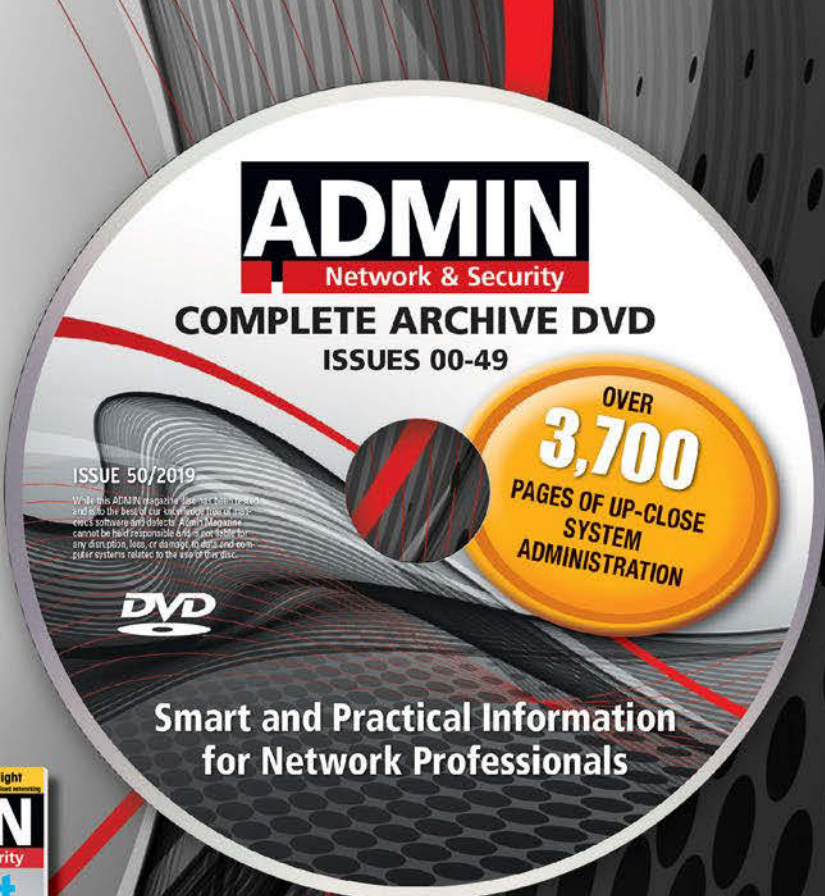
### Info

- [1] Press release on SUSE independence: <https://www.suse.com/c/news/suse-completes-move-to-independence-reaffirms-commitment-to-customers-partners-and-open-source-communities-as-industrys-largest-independent-open-source-company/>
- [2] Nils Brauckmann post on the EQT acquisition: <https://www.suse.com/c/the-future-of-suse-a-home-for-truly-open-open-source-solutions/>
- [3] Parody videos: <https://www.youtube.com/playlist?list=PL6sYHytyKN2-X93TurF3JptW8qSVm0DzA>
- [4] Wildhorse party with SUSE band: <https://www.youtube.com/watch?v=6lw5sRoXUpI>





# 9 Years of ADMIN on One DVD



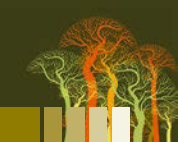
This searchable DVD gives you 50 issues of ADMIN, the #1 source for:

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

Clear off your bookshelf and complete your ADMIN library with this powerful DVD!

**ORDER NOW!**  
[shop.linuxnewmedia.com](http://shop.linuxnewmedia.com)





Tips from the experts on getting more from Git

# Git Tricks

The Git distributed version control system is a popular tool for managing open source development projects. If you know the basics of Git but are looking to learn the ways of the experts, read on for some useful Git tips and tricks.

By Markus Feilner

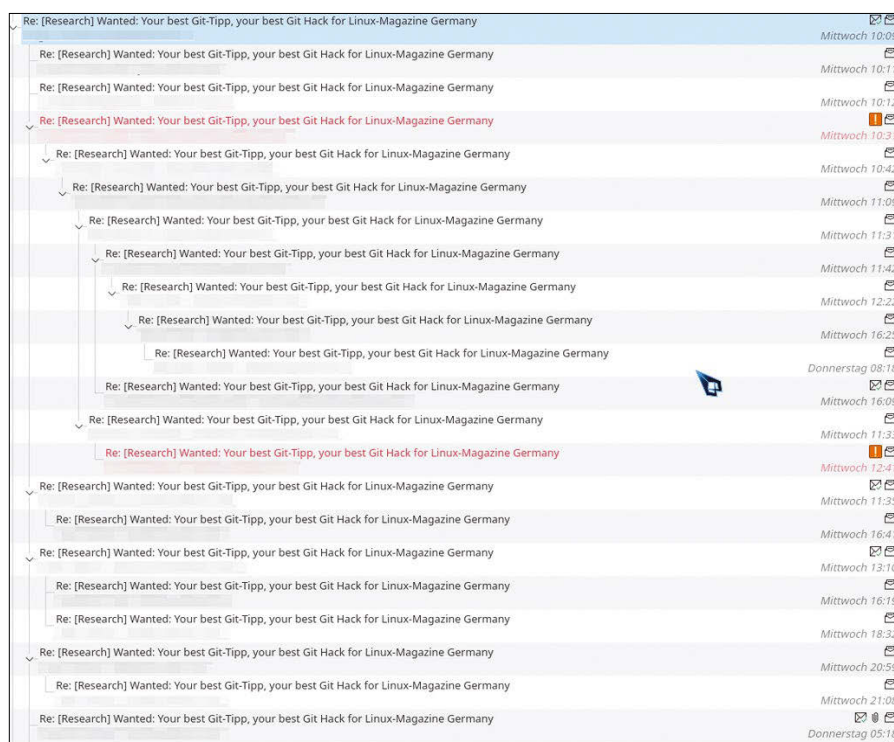
**“Y**ou must know a few Git experts,” an editor at *Linux Magazine* said to me. As the leader of the SUSE documentation team, I spend large portions of my day around veteran Linux developers. When I agreed to ask the team for some tips for using the Git version control system [1], I did not expect that I would trigger a thread that lasted several days (Figure 1).

The number and quality of the tips shows just how important Git is to SUSE developers. This article rounds up some of my favorite tips. If the tricks mentioned here are not enough for you, you will find many more online – aimed at beginners and connoisseurs alike. Check out the DZone site for a summary of useful Git commands [2]. The Fedora team has also assembled their own list of “pro tips” for Git developers [3]. And don’t forget the Git cheat sheet for an at-a-glance view of some important Git commands [4].

At SUSE, the developers aren’t the only ones who depend on Git. The documentation team uses Git every day, including the `gitflow` workflow feature, which we use for managing documentation projects [5]. The Trello board shown in Figure 2 is used

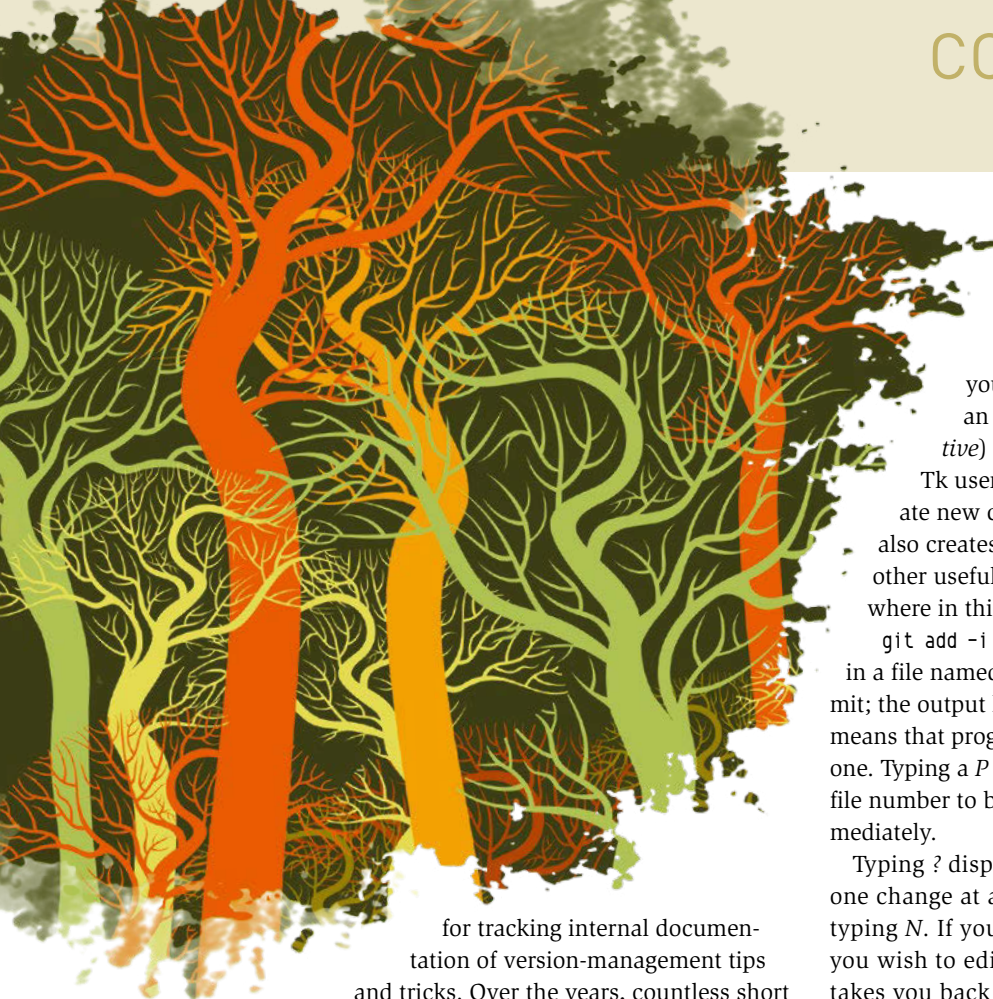
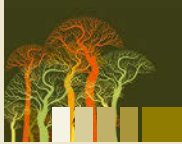
## Author

**Markus Feilner** is a seasoned Linux expert from Regensburg, Germany. He has been working with free and open source software since 1994 as a trainer, consultant, author, and journalist. He is currently employed as team lead of the SUSE Documentation Team in Nuremberg, Germany.



**Figure 1:** The thread with Git tips on the internal SUSE mailing list quickly grew.





for tracking internal documentation of version-management tips and tricks. Over the years, countless short

how-tos have accumulated, ranging from entry level to highly sophisticated solutions. The first five tips in this article – submitted by technical editor Thomas Schraitle – come from this pool.

## Tip 1: Getting Pretty

The Git Pretty flowchart from developer and blogger Justin Hileman [6] (shown in Figure 3) is extremely helpful for cleaning up after working on a project. The simple decision chains can help you decide how to respond to a mess, which is when you might need the help the most. At the end of each chain, you will mostly find simple Git commands. See the Git SCM site [7] for more on cleaning up in Git.

## Tip 2: Interactive, Please!

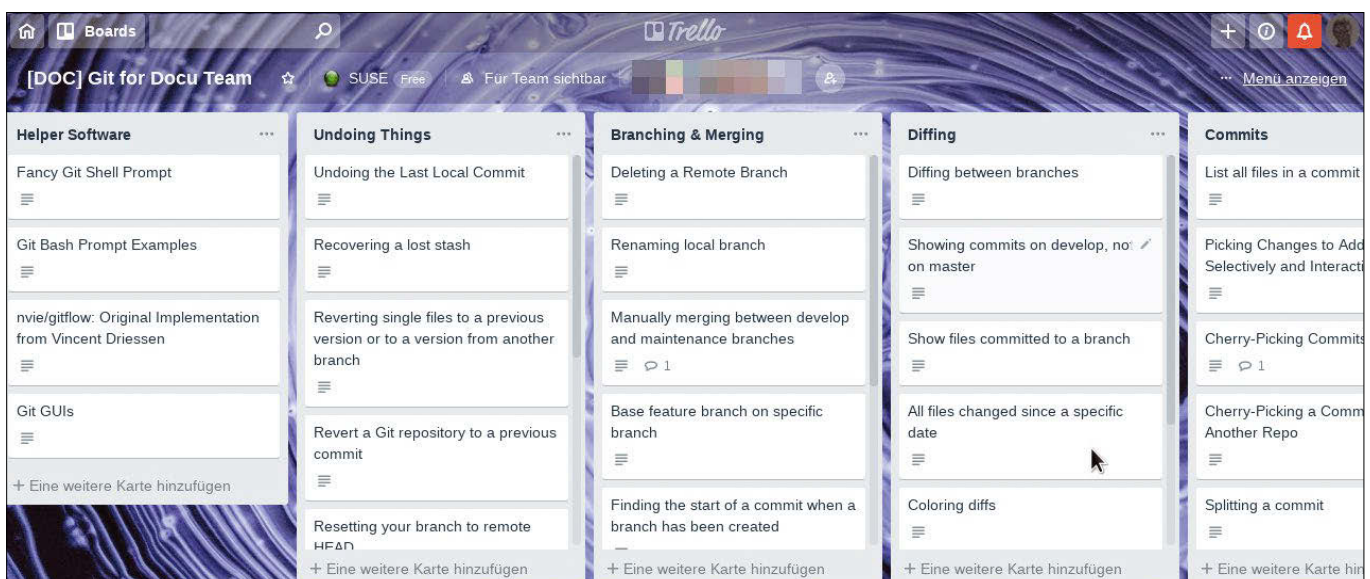
It happens time and time again: Users want to interactively decide what kind of code they will be adding and supplementing in their projects. If a `git add` is not enough for your needs, two options for working with Git in an interactive mode are `git add -i` (`-i` for *interactive*) or `git gui`. The `git gui` command opens a Tcl/Tk user interface, through which developers can create new commits or change existing settings. `git gui` also creates new branches, manages stages, and offers other useful features (see also the article on Git GUIs elsewhere in this issue).

`git add -i` is a little leaner. Listing 1 shows three changes in a file named `README.adoc` that are not yet ready for a commit; the output looks a bit like `git status`. The `+3/-1` in line 2 means that programmers have added three lines and removed one. Typing a `P` and Enter starts patching; Git then expects the file number to be selected (here 1) and shows the changes immediately.

Typing `?` displays the help. You will usually go through one change at a time and confirm it with a `Y` or reject it by typing `N`. If you press `D` followed by the number of the file you wish to edit, a diff appears. `Q` quits the viewer and takes you back to the shell, where you can complete the code in the usual way. Look online for more information on Git's interactive options [8].

### Listing 1: `git add -i`

```
01          staged   unstaged  path
02 1:    unchanged      +3/-1  README.adoc
03
04 *** Commands ***
05 1: status   2: update   3: revert   4: add untracked
06 5: patch    6: diff     7: quit     8: help
07 What now>
```



**Figure 2:** The SUSE documentation team uses Trello's Kanban boards for almost all tasks. The documentation specialists collect the best Git tips for new SUSE employees.



### Tip 3: Who Committed What?

The `git blame` command annotates a file with line-by-line revision information showing who made previous changes to each line. Thomas points out that many users don't realize the `-L` option accepts a start and an end value, which means you don't have to generate output for a whole file but can focus on a specific section of the code. GitHub also supports this function. The *Blame* button visualizes the output on the web and shows author, commit message, SHA sum, and the changed lines. If you like Emacs, you will be able to access the blame function via `M-x git blame-mode` if you install the `git-mode` extension.

SUSE developer Benjamin Poirier has set up a Vim command to show him the history of the code in the Git Gui – `git gui blame` (Listing 2). If you like this option, add the code

from Listing 2 to your own `.vimrc` file and enter `:GGBlame` (or `GG` and `Tab`) in a Git session in Vim.

### Tip 4: Just Ignore!

Git has at least three configuration files in which you can list files that you want Git to ignore:

- `.gitignore` – this file is the right place for anyone who wants to create an exclusion list for all users in their current repository.
- `.git/info/exclude` – the rules listed here for files to exclude only apply to your individual repository; they do not affect anyone else and only apply on the local machine.
- `~/.gitignore` – behaves in a similar way to `.git/info/exclude`, but applies to all repositories on the local machine.

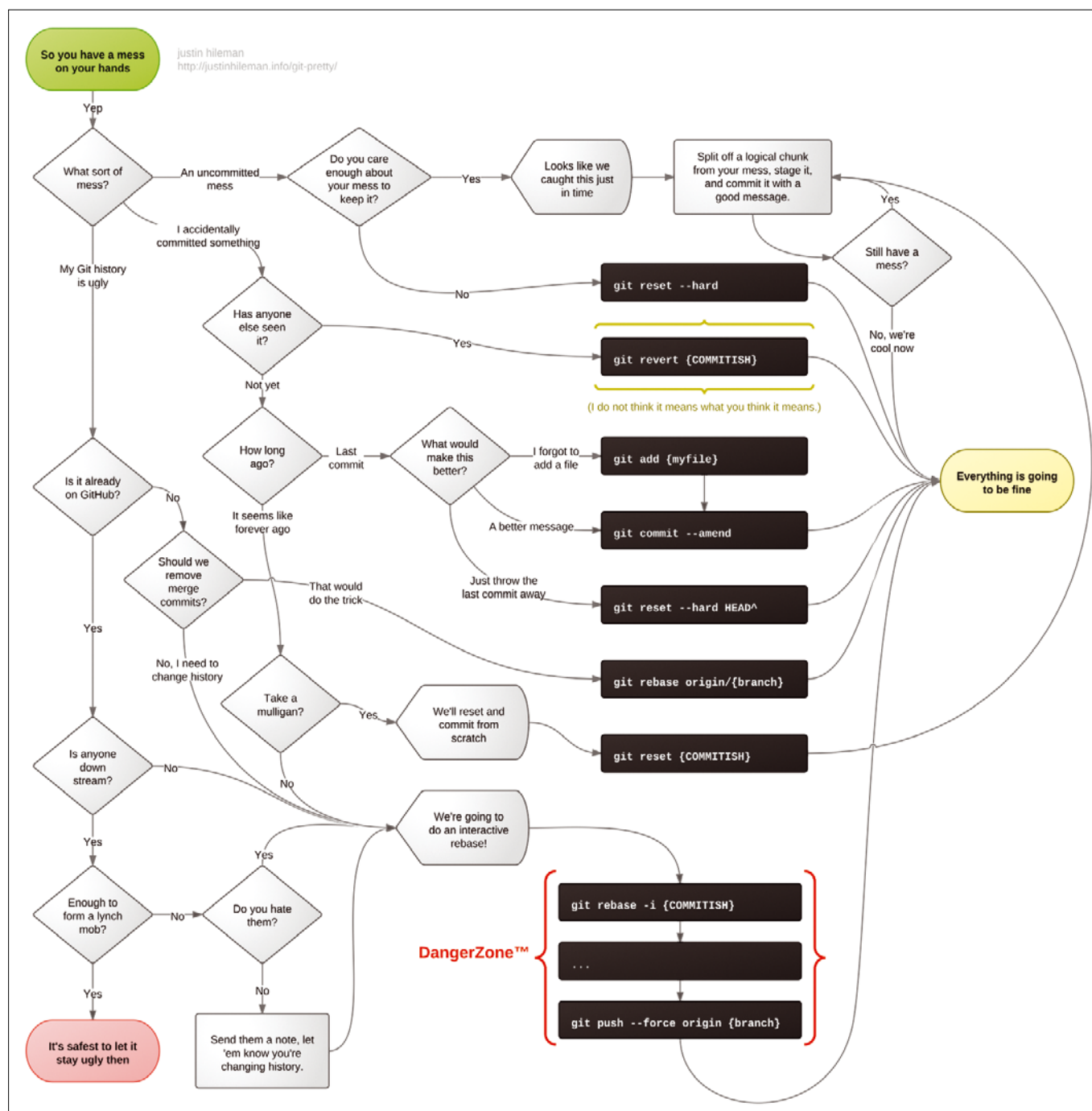


Figure 3: Git Pretty gives users a decision-making tree for cleaning up Git repositories.





## Listing 2: Vim Plugin for git blame gui

```
01 function LaunchGGBlame()
02     let cmd = printf("cd %s && git gui blame --line=%d %s &", expand("%:p:h::S"), line("."), expand("%:p::S"))
03     let output = system(cmd)
04     " if the background command fails, we won't know because of &
05     if v:shell_error != 0
06         let msg = printf("Error running '%s': %s", cmd, output)
07         echohl WarningMsg | echon msg | echohl None
08         return
09     endif
10 endfunction
11 command GGBlame call LaunchGGBlame()
```

If you want to use `~/.gitignore` globally, use the following command:

```
git config --global core.excludesfile ~/.gitignore
```

If you want to learn more about `~/.gitignore`, you will find further information online [9]. In addition, a tool from SUSE's senior engineer Adam Spiers might come in handy. Adam developed `git check-ignore` [10]. This command is built into Git and checks files and path names to discover whether or not they are on one of the local ignore lists. In this way, developers can avoid surprises when working with third-party repositories.

## Tip 5: Retroactive...

Thomas offers another tip, but one that developers will not want to apply to public commits. Git lets you retroactively add whole files to a commit. If you have staged files with `git add` and already checked them into the repository with `git commit`, but forgot some files, you can add them later with `git commit --amend`.

First, simply call `git add` again and specify the forgotten file as a parameter. Then `git commit --amend` comes into play; this command displays the commit message from the previous commit in the editor. If you want, you can change it now, but it's not absolutely necessary. By the way, `amend` also supports `rm` for later removal of files. More information is available in the Atlassian [11] wiki.

## Tip 6: Stashing with Stash

Every developer is familiar with this scenario: A developer is working on an important project when The Boss comes in with a new, even more important task. The new task suddenly has priority, but you will need to continue working on the old task at a later time. You will have to check in the code for the new task before you check in the code for the original task.

Git supports a strategy known as stashing for such situations: Changes that the developer has not yet checked in end up in stashes for processing at some later time.

`git status` first displays the list of changed files; `git stash` moves these changes to a stash. Now you can work on the new task. After the work is done, call `git stash pop`, and you can continue working at the point you reached before the interruption. Git Stash can be nested multiple times, so commands like this exist:

## Listing 3: Useful Aliases for Mercurial Users

```
01 [alias]
02     branches = !git branch --list
03     ci = commit
04     out = !git fetch && git log FETCH_HEAD..
05     outgoing = !git fetch && git log FETCH_HEAD..
06     in = !git fetch && git log ..FETCH_HEAD
07     incoming = !git fetch && git log ..FETCH_HEAD
08     tags = !git tag --list
```

## Listing 4: Cleaning up the Submodules

```
01 gitCleanSubmodules(){
02     git submodule sync
03     git submodule update -f --recursive --remote
04     git submodule foreach --recursive git clean -xdf
05     git submodule foreach --recursive git reset --hard
06     git submodule update -f --recursive --init
07 }
```

- `git stash list` – lists all stashes
- `git stash drop` – removes the last stash
- `git stash branch test` – generates a new branch based on the stash

Attention: Stash only works on added files by default. If you also want to have unmonitored files in your stash, you have to add `git add --all` and `git stash` explicitly. You'll find more information at the Git SCM site [12].

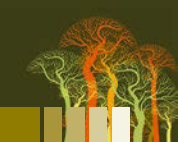
## Tip 7: In Memoriam Mercurial

Veteran developer Lenz Grimmer wrote: "Our team is involved in the development of the Ceph management solution openATTIC and its successor, the Ceph Dashboard. We migrated our openATTIC code repository from Mercurial to Git in April 2017 [13]. As former Mercurial users, we missed a few commands in Git, which we reverse engineered as aliases in `~/.gitconfig`." See Listing 3 for some of our favorites.

## Tip 8: Spring Cleaning

Stephan Müller integrated the function from Listing 4 into his `zshrc` to make life easier with Git and submodules. But be





careful, this function will clean up completely and irretrievably. After this command, all submodules are as clean as after a new clone process!

### Tip 9: git-vimdiff

Matt Oliver mentioned the benefits of `git-vimdiff` [14]. In addition to the normal `vimdiff`, which can also be connected to Git, `git-vimdiff` (Figure 4) offers each file in a separate tab. Vim users can conveniently scroll back and forth through them. “I use this so often now: Every time I go through a new patch set in Gerrit or GitHub, I just jump into the branch and call `git vimdiff HEAD~.`” You can then go through the diffs one by one and read the reviewer comments in parallel.

### Tip 10: Automatic Analysis

Adam Spiers has no qualms about advertising his own product, `git-deps` [15]. This tool is used to automatically analyze dependencies between commits in a Git repository. According to Spiers, `git-deps` has plenty of uses, including:

- porting between branches
- splitting a patch series into independent parts
- supporting cooperation in development teams
- automatic integration of fix-up commits

The tool also helps you clean up the private commit history before publishing.

### Tip 11: Fixups and Rebases

Alvaro Saurin stresses the importance of a correct understanding of `git commit -fixup=Commit`, which marks the commit as a fixup of a previous commit, and its counterpart `git rebase -autosquash`, which merges the original commit and the fixup into a new commit. For example, suppose a developer is at the following location:

```
Master -> Modification 2 -> 2
Modification 1 -> Current modification
```

In the next step, the developer could then initiate a fixup commit for Change 2. The command for this is `git commit --fixup=HEAD~1`. The results would look like this:

```
Master -> Change 2 -> Change 1 -> Fixup-Commit-for-Change-2
```

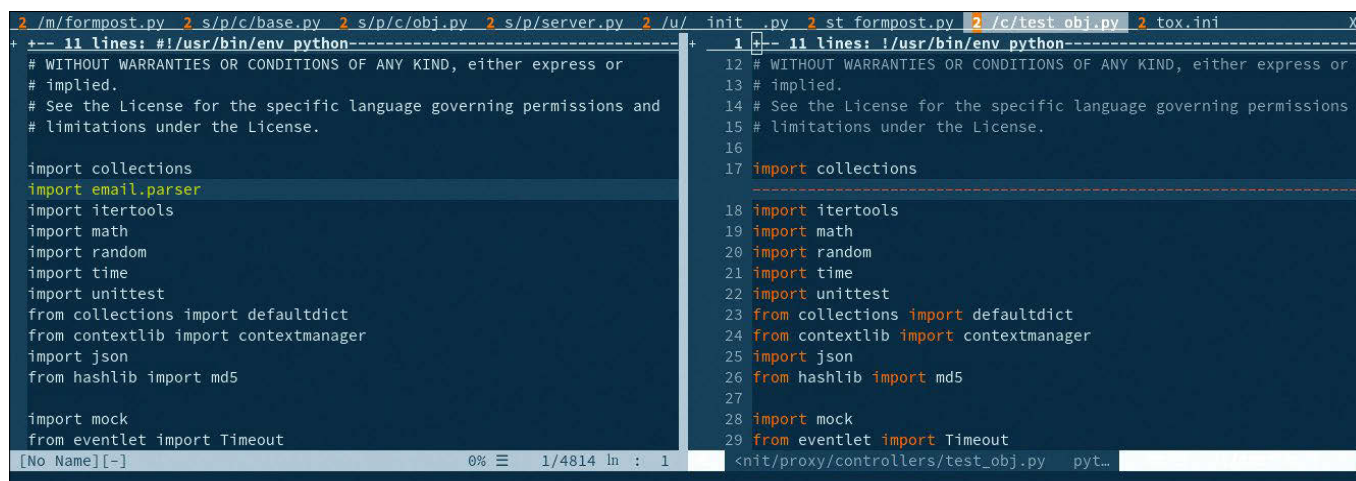


Figure 4: SUSE developer Matt Oliver calls `git-vimdiff` “one of the most productive git hacks I use regularly.”

A subsequent `git rebase -i --autosquash` then automatically merges the fixup with change 2 (shown here in two steps):

```
Master -> Change 2 -> Fixup-for-change-2 -> Change 1 Master -> 2
New change 2 -> Change 1
```

This sequence of steps could also be defined as a Git alias and executed in one step. Developer Bernhard Wiedemann shows how this works in Listing 5.

And this – as Benjamin Zeller notes – could be activated as the default via the Git configuration.

The alias, on the other hand, is not necessary for people who use Adam Spier’s `git-fixup` [16]. The developer is convinced that the approach described is one of the most fundamental that a developer needs to understand in Git.

Unsurprisingly, the sequence can also be embedded in Emacs and called with just three keys. `C` opens the commit menu; `F` starts the instant fixup; you can then choose what you want to fix. A “.” tells Emacs to create the fixup commit, automatically going through all unrecorded changes, interactive rebasing, and even leaving the stash at the end.

### Tip 12: In Color and Colorful!

A tip that adds color to Git also comes courtesy of Bernhard Wiedemann. According to his own statement, he found the following, not really pretty alias “somewhere on the Internet”:

```
lfi = log HEAD --graph --all --abbrev-commit --full-history 2
--color -pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset 2
%C%Cgreen(%cr) %C(bold blue)<an>%Creset' --date=relative
```

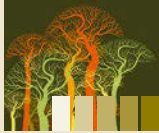
He quickly got used to the clarity of a colored representation of the tree diagram in the history.

### Fast Run

I’ll end with five more quick tips you can find online.

### Listing 5: Autosquash Alias

```
01 [alias]
02     pushf = push --force-with-lease
03     ri = rebase --interactive --autosquash
```



Web developer Thomas Hutterer uses a Git alias featuring `homegit` [17] to manage his configuration files (*dotfiles*). This little `GIT_WORK_TREE` trick makes it possible to manage files within your home directory and use them on multiple computers without any additional software. Thomas says “it helps you feel at home everywhere.”

Martin Wilck provided the links for a rebase without the loss of tags [18]. The kernel developer also provided a link to the git-related tool [19], which helps Git users track down developers and commits for a set of changes. The tool runs through the Git history and executes a `git blame` for each change.

Sergio Lindo added several recommendations. For instance, you can sign commits (via `gpgsign = true` in the `[commit]` section of the `.gitconfig` file).

The `git lola` alias [20] creates a colored graph for all branches of the repository at the command line. `git rup` is an alias for `git remote update --prune`. It downloads the last commits of all branches of all repositories in the remote ref branches but does not maintain them in the local branches. This can be done by the developer, for example with `git-sync-fork.sh` [21]. `diff-highlight` [22] highlights changes in diffs in color.

Storage guru Lars Marowsky-Brée recommends `vim-fugitive` [23]. The Vim extension makes typical Git commands like `git status`, `git blame`, and `git mv` usable for Vim users.

But the last word goes to Stephan “coolo” Kulow: It has to be fun! To ensure this, he recommends the Happy Git Commits [24]. ■■■

## Info

- [1] Git: <https://git-scm.com/>
- [2] Useful Git commands: <https://dzone.com/articles/useful-git-commands>
- [3] Git pro tips by Fedora developers: <https://www.linuxfoundation.org/blog/2015/04/7-pro-tips-for-using-git-from-fedora-developers/>
- [4] Cheat sheets for Git: <https://www.git-tower.com/blog/git-cheat-sheet/>, <https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf>
- [5] gitflow: <https://github.com/nvie/gitflow>
- [6] Git Pretty: <http://justinhileman.info/article/git-pretty/>
- [7] Undoing in Git: <http://www.git-scm.com/book/en/v2/Git-Basics-Undoing-Things>
- [8] Interactive adding in Git: <http://git-scm.com/book/en/v2/Git-Tools-Interactive-Staging>, <http://gitready.com/intermediate/2009/01/14/interactive-adding.html>
- [9] Ignoring in Git: <http://git-scm.com/docs/gitignore>
- [10] git-check-ignore: <https://git-scm.com/docs/git-check-ignore>
- [11] Rewriting the Git history: <https://www.atlassian.com/git/tutorials/rewriting-history>
- [12] Stashing in Git: <http://git-scm.com/book/en/v1/Git-Tools-Stashing>, <http://git-scm.com/docs/git-stash>
- [13] Migration from Mercurial to Git: <https://www.openattic.org/posts/openattic-code-repository-migrated-to-git/>
- [14] git-vimdiff: <https://github.com/frutiger/git-vimdiff>
- [15] git-deps: <https://github.com/aspiers/git-deps>
- [16] git-fixup: <https://github.com/aspiers/git-deps/blob/master/bin/git-fixup>
- [17] homegit: <https://github.com/thutterer/homegit>
- [18] Git rebase without losing the tags: <https://ownyourbits.com/2017/08/14/rebasing-in-git-without-losing-tags/>
- [19] git-related: <https://github.com/felipec/git-related>
- [20] git-lola: <http://blog.kfish.org/2010/04/git-lola.html>
- [21] git-sync-fork.sh: <https://github.com/software-for-life/git-helper/blob/development/src/git-sync-fork.sh>
- [22] Optimizing diffs: <https://github.com/git/git/tree/master/contrib/diff-highlight>
- [23] vim-fugitive: <https://github.com/tpope/vim-fugitive>
- [24] Happy Git Commits: <https://collectiveidea.com/blog/archives/2010/08/03/happy-git-commits>

Available Now

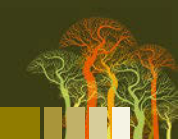
\* **2018 EDITION** \*

Linux Magazine Archive DVD

**ORDER NOW!**  
[shop.linuxnewmedia.com](http://shop.linuxnewmedia.com)







## Four graphic interfaces for Git

# Tree View

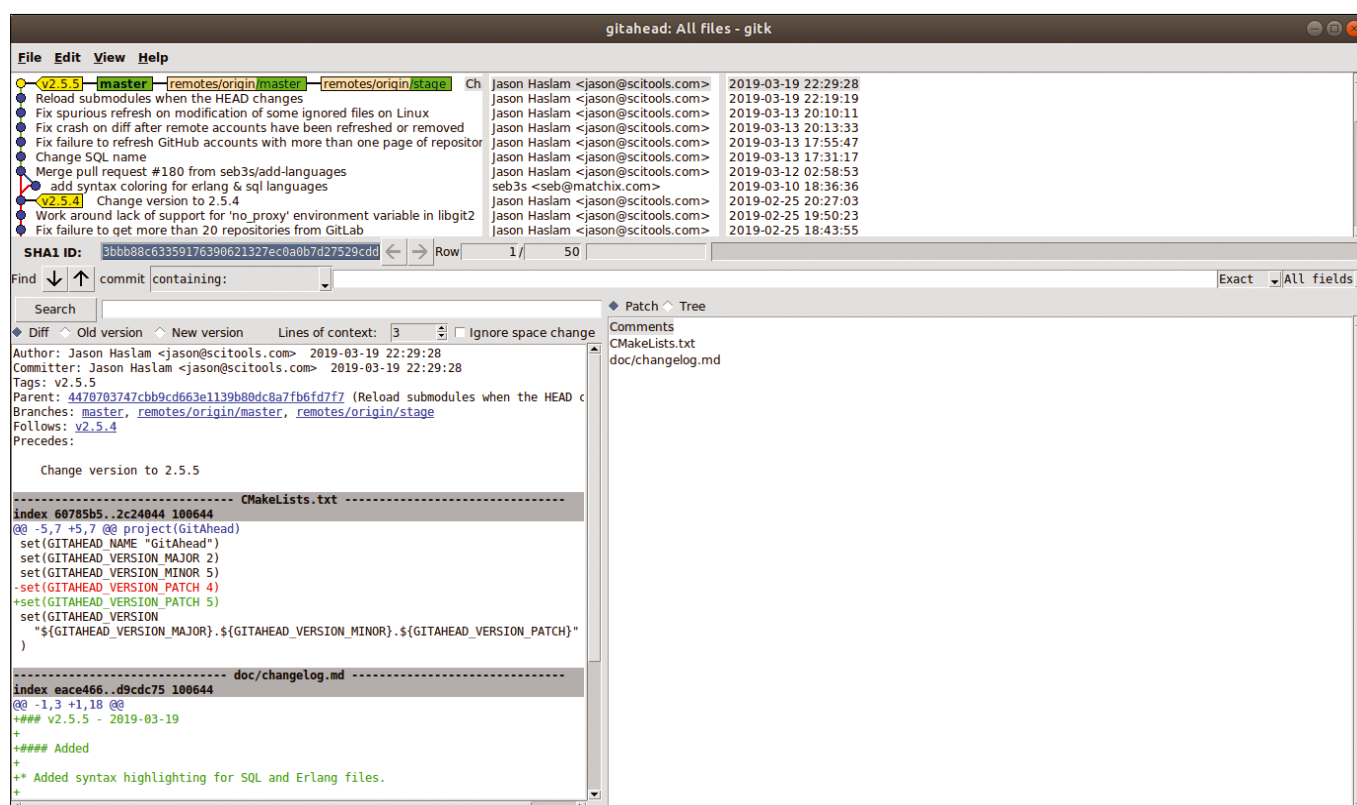
Complex Git projects sometimes require a better view of the dependencies and branches. Several tools offer GUI options for Git. We take a look at gitk, gitg, git-gui, and GitAhead.

By Kristian Kißling and Tim Schürmann

**G**raphical User interfaces (GUIs) for the Git version control system let users visualize branches and different version levels. Developers thus have a better overview of their projects' status. In addition, GUIs make Git easier to use, because programmers do not need to remember any of the cryptic Git commands.

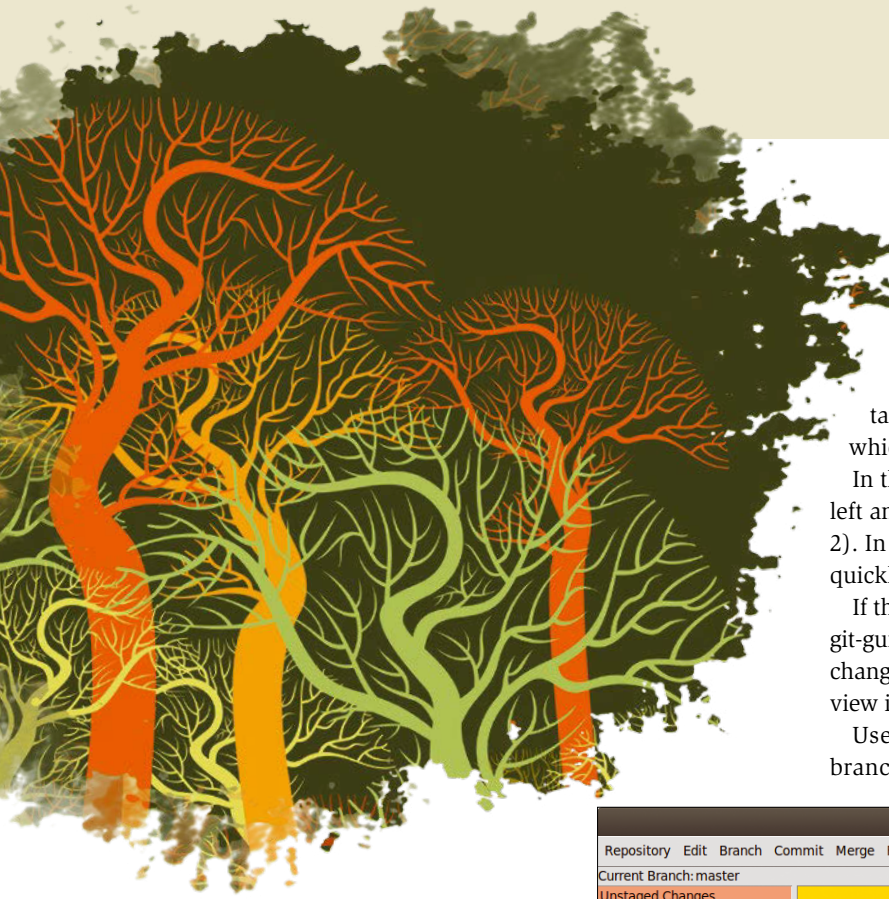
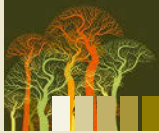
Thanks to the great popularity of Git, numerous GUIs are now vying for the attention of users, including some commercial offerings. Many users are not aware that Git already has two GUIs, gitk [1] and git-gui [2]. Although both are part of Git's official scope of supply, many distributions outsource them into separate packages. In Ubuntu 18.10, users have to install the git-gui or gitk packages. The code for the two GUIs has also been adopted by external projects. The Git developers always add the latest stable version to their branches.

The following comparison with the free competitors GitAhead [3] and gitg [4] is intended to reveal how the two GUIs perform compared to the rest of the field.



**Figure 1:** Gitk confronts users with information overkill, although you can only really use it to visualize branches and commits.





## gitk

Gitk [1] is based on the Tcl scripting language and the Tk toolkit. This dependence on Tcl/Tk is reflected by the old-fashioned appearance of the main window. The software was last updated two years ago. Gitk only visualizes the Git history (Figure 1); users can call it up directly in the project directory. The tool cannot manage and compare several projects. In the top-left corner of the main window, you will see the view of all branches, where merges and commits also appear.

A click on a commit shows the changelog in the lower-left corner together with the changes made. The changed files appear bottom-right, and the names of the persons responsible at the top-right.

Two search functions help you find information. One of the search functions digs through the open changelog and the changes; the other digs through the commits. Only the second search function evaluates regular expressions. Both highlight the search results in color when you enter a search term.

Users can create views to restrict the display to certain criteria. For example, you can create a view that displays only commits by Tim Schürmann. However, gitk does not open these views in separate windows; instead, users have to switch to them via a menu, which is inconvenient.

## git-gui

Git-gui, which comes with the Git package, is also based on Tcl and Tk. As with gitk, you call the tool directly in the project directory using the Git `git gui` command. In contrast to gitk, git-gui [2] simplifies the actual work with the Git commands. Graphical representation of the branches is left to the counterpart gitk utility, which has a menu item for this purpose.

In the main window, git-gui displays the changed files on the left and the changes in the selected file on the top-right (Figure 2). In the lower area, developers can add comments and quickly create a commit using the options provided.

If the Aspell dictionaries available under Linux are installed, git-gui even performs a spell check. After a developer has changed a source code file (Figure 3), they have to refresh the view in git-gui – the tool does not recognize changes on its own.

Users can create, rename, delete, or locally merge new branches via the menu. In addition, git-gui can fetch and

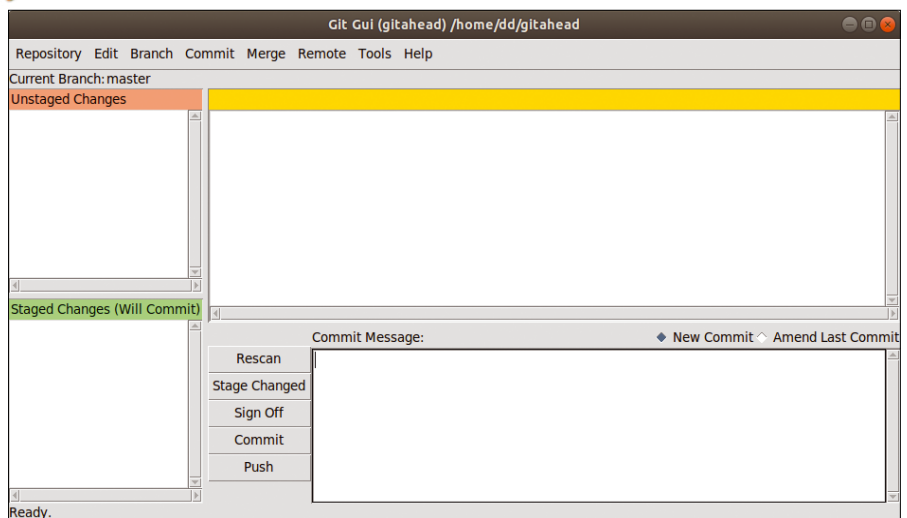


Figure 2: The git-gui main window.

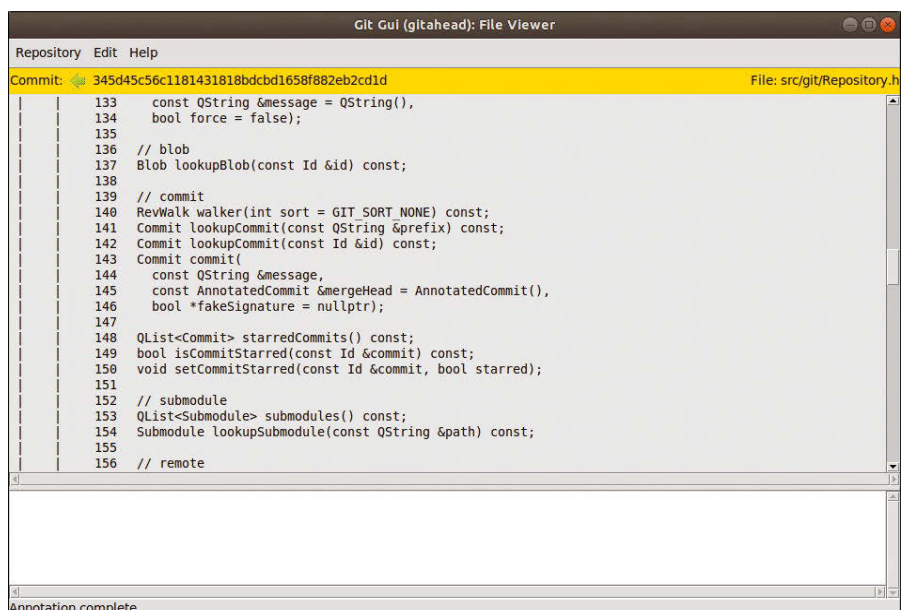


Figure 3: Git-gui can display individual files in this File Viewer, which also provides an interactive Blame view.

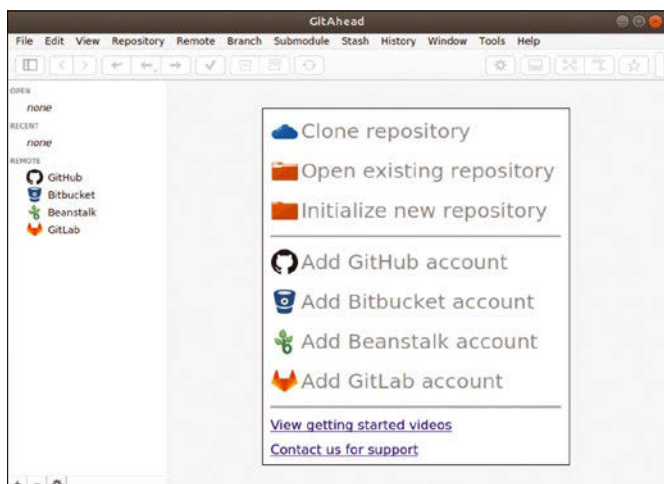


Figure 4: GitAhead can integrate four external services.

push from and to external repositories. Last but not least, the tool integrates external tools if required, although git-gui can only start one command-line command given by the user at a time.

## GitAhead

GitAhead [3] is not as well integrated into Git, and it is essentially the work of developer Jason Haslam. The software is available under the MIT license. We were able to install the current version 2.5.3 on Ubuntu 18.04 in our lab without any trouble. If you are interested in the product, you won't find it in your package manager's lists, but it is available on

GitHub [5]. The package with the .sh extension is a script, which you need to make executable by typing `chmod u+x GitAhead-2.5.3.sh` and then launch by typing `./GitAhead-2.5.3.sh`. The command unpacks the software contained in an archive into a subfolder. You then launch the software via the `gitahead` file.

GitAhead cooperates with a number of popular Git vendors, including GitHub, GitLab, Beanstalk, and Bitbucket (Figure 4). For our test, we added a source code archive on GitHub via Add GitHub account.

The documentation for the software is fairly sparse. If you have any questions, you are supposed to tag them as `[gitahead]` on Stack Overflow [6], which apparently nobody has done yet. Most Git users probably intuitively understand how the application works anyway, so they don't need a helping hand. The fact that people actually use GitAhead is shown by the open issues on GitHub.

In the left pane, the software presents the remote, open, and recently used repositories. Local changes can be moved to the remote repository using the *Remote | Push* menu item. If you want to create a new repository, click *File | Initialize New Repository* and then assign a local location for the repository.

*File | New File* lets you create new files in a simple editor; you save them with *File | Save* in the newly created repository. They are not checked in yet. A click on the new repository shows *Uncommitted Changes*. In the upper-right area, developers then create a commit message and select *Stage All* and *Commit* to check in all changes locally.

To the left of center, the branches and commits appear one on top of the other in two smaller windows. The upper window lists the branches (often the only branch is `master`); below

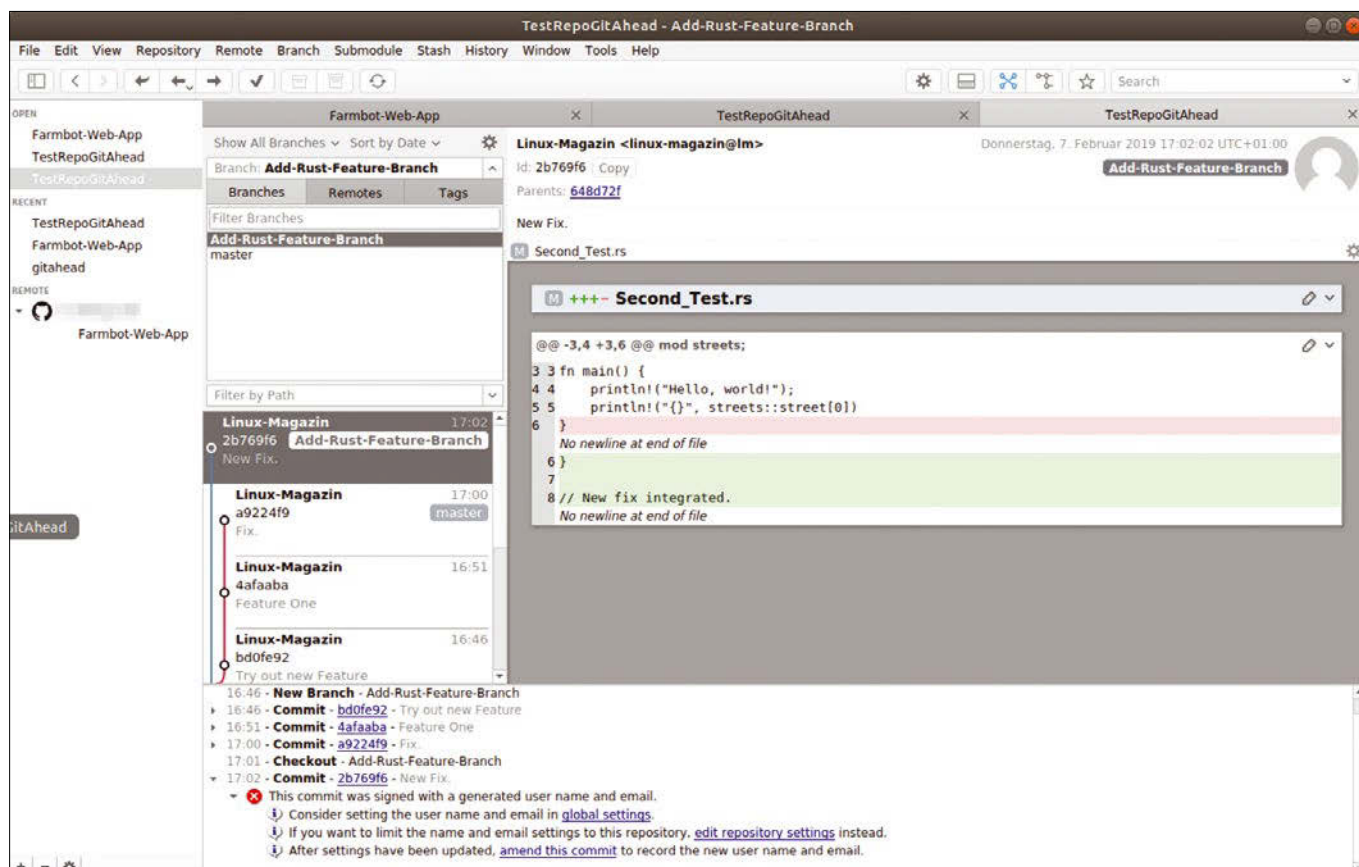


Figure 5: GitAhead traces the often confusing ramifications of the code with colored lines.



them you see the commits they contain. On the right side of the window, the commits appear in detail. A click on the pen icon next to an entry opens the entry in the built-in editor, where you can then modify it.

The interface also supports branches and merging. To work on a branch other than *master*, developers need to select a commit to branch and then select *Branch | New Branch*. If you then want to work in the new branch, select it in the *Branches* area and double-click on it. On the right you can now click on the pen, work on the code, and save your changes by pressing **Ctrl + S** or selecting *File | Save*. Check the code in again via *Stage All* and *Commit*.

Once all changes have been made to the branch, *Branch | Merge* puts the two branches back together again. If merge conflicts occur, GitAhead displays a diff in the details window and asks which changes the developer wants to apply – because third parties might have made changes to a file in the meantime.

The individual commits to the branches appear below the branches. As the number of branches increases, this is visualized by branching lines shown in different colors (Figure 5). The lines connect the commits and quickly show the developer which parts of the code are temporary spin-offs. You can see who is currently working on a branch in the top right-hand corner of each commit.

At the bottom of the window, a history of commits to a project also appears. There are also error messages if the *master* cannot be checked out due to unresolved conflicts or if Git Large File Storage (LFS) is not installed, but the developer wants to use it.

To push the local repository to GitHub, the developer first logs in, then creates a new repository via the browser, and copies the link, for example:

```
https://github.com/username/Repository-Name.git
```

Now, in GitAhead, first select the desired local branch and then *Repository | Configure Repository*. A small window opens. In the *Remotes* tab, there should not be a remote source directory for a local-only source directory. However, this can be added by pressing the plus sign.

First type a name, such as *origin/Repository-Name*. In the line below, you need the link to GitHub; the whole thing is sealed by a click on *Add Remote* and then *Ok*. Afterwards, move your local repository to GitHub via the *Remote | Push to* menu item. For the other Git providers, the procedure should work in the same way. Further commits can then be automatically moved to the remote server using the settings.

GitAhead has more capabilities that we have not yet talked about. *Tags* let you assign version numbers to software. In addition to a merge, a rebase is also possible, which, for example, transplants a feature branch to the top of the master branch. Code contributions can also be undone in the commit overview. Just right-click on the commit entry and select *Revert* from the context menu.

If time is of the essence, but you do not want to let your unpolished changes loose on the repository yet, you can use the *Stash* menu item. *Stash* lets you push your changes into a virtual icebox for the time being, and *Pop Stash* lets you

## 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?

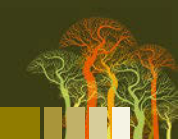
### DIGITAL & PRINT SUBSCRIPTIONS



### SPECIAL EDITIONS







continue to work on them. Last but not least, cherry picking is possible if you wish to select only certain commits from the history and transfer them to another branch.

*Tools | Options* makes it possible to adjust a few settings, such as character encoding, themes, fonts, and updates. You can also enter an external editor if it is located in `$PATH`. To call the editor, right-click on the bar above a commit in the detail view and choose *Edit in External Editor*.

If you usually work in a different editor, the files created with it are simply stored in the local Git repository. They then appear as unsubmitted changes in GitAhead.

## gitg

Gnome is dedicated to functional frugality, which makes taking a look at gitg [4] all the more exciting. Gitg is a Git GUI that visualizes Git directories and integrates seamlessly into the Gnome interface. The software can be installed via the package manager on Ubuntu 18.04.

Code repositories in the Gnome world are referred to as software repositories. When launched, Gitg first points out that it has not yet found any repositories; instead, it offers you a clickable link to search. If you follow the link, you will be taken to the local repositories that gitg displays in a simple overview.

In our lab, the somewhat confusing test repository, which we created with GitAhead, was one of the candidates presented. As with GitAhead, a click on the test repository showed a visualization of the commit history with colored lines (Figure 6).

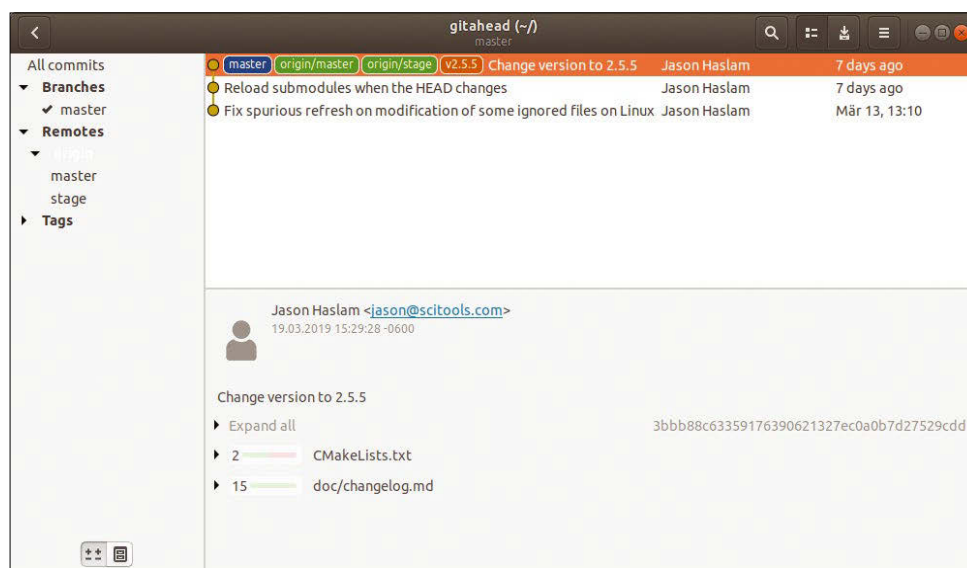
The upper-right area lists the different commits, and their contents appear in the lower window area. Gitg color-highlights changes in diff format. The leftmost column provides a complete overview of the branches in the selected repository.

If you right-click on a commit entry top-right on the screen, various options await you in a context menu. These options include *Create branch*, *Create tag*, *Create patch*, and *Cherry pick operation by branch\_name*. In fact, you can use gitg to create your own branches and merge them. For merges, you have to enter your data below *Details of the author* in the configuration menu; otherwise gitg protests.

If you choose the *Create tag* option, you can specify a version number for the commit that will appear next to this entry and to the left of the *Tags* entry. If the user chooses *Create patch*, gitg creates a file with a `.patch` extension, which you cannot edit, but at least you can save it in the source directory. You can also use gitg to supplement other local repositories and clone existing repositories.

## Conclusions

The GUIs provided by Git share the work. While gitk visualizes the dependencies between branches and provides search



**Figure 6:** Visually, gitg matches Gnome, but it is primarily used to visualize Git repositories.

functions to help you rummage through the commit history, git-gui transfers some of the typical Git commands to a graphical interface. However, both seem a bit antiquated due to the use of the Tcl/Tk toolkit.

Gitg is similar to gitk in that it visualizes the Git directories graphically, but it adorns the data in the window dressings of the Gnome desktop. However, gitg does not allow changes to be made to the files. Why the Gnome developers didn't decide to adapt functionally more extensive open source GUIs to Gnome remains their secret.

What is strange, however, is that the files can even be changed. In our lab, right-clicking on the filename at the bottom of the GUI was all it took. The *Open File* option calls it in an editor. However, these modifications did not appear in the directory browser even after saving in the editor and restarting gitg. Apparently, the software does not re-index the files in the imported Git directories after startup, although this makes them unusable for changes.

GitAhead made a good impression, accommodating the most important Git features in a graphical application. Developers (so far) only have to do without some of the more sophisticated functions. According to the bug tracker, GPG-signed commits are not yet possible. More exotic cases like `git bisect` are not yet supported by the software. It would also be desirable for the developers to integrate GitAhead into existing Linux distributions in a better way; a feature request for better integration already exists. ■■■

## Info

- [1] gitk: <https://git-scm.com/docs/gitk>
- [2] git-gui: <https://git-scm.com/docs/git-gui>
- [3] GitAhead: <https://gitahead.github.io/gitahead.com/>
- [4] gitg: <https://wiki.gnome.org/Apps/Gitg>
- [5] GitAhead download from GitHub: <https://github.com/gitahead/gitahead/releases>
- [6] GitAhead on Stack Overflow: <https://stackoverflow.com/questions/tagged/gitahead>

# Too Swamped to Surf?



## ADMIN

Network & Security

ADMIN offers additional news and technical articles you won't see in our print magazine.

Subscribe today to our free ADMIN Update newsletter and receive:

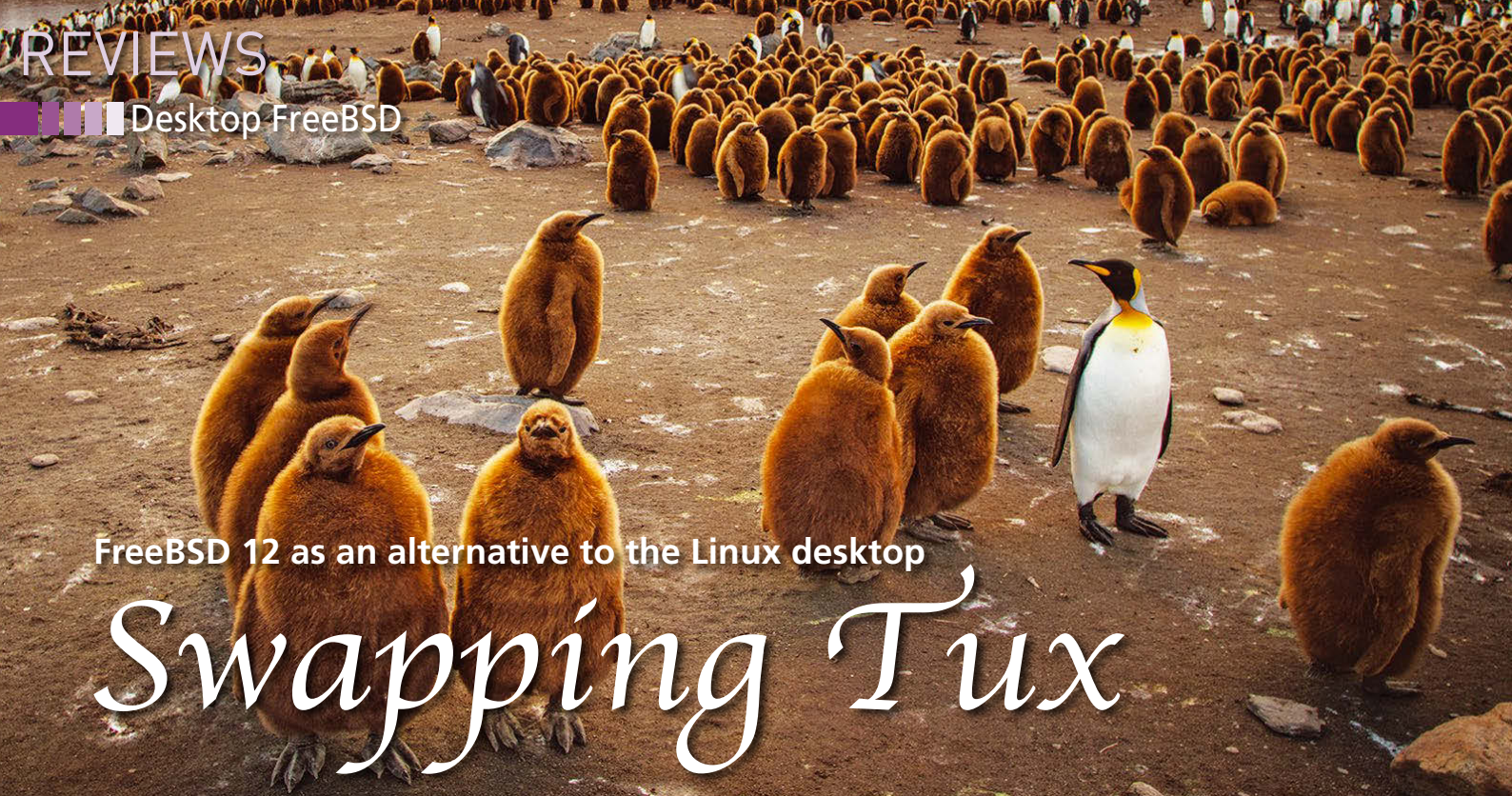
- Helpful updates on our best online features
- Timely discounts and special bonuses available only to newsletter readers
- Deep knowledge of the new IT

© Rachata Toyarsit, 123rf.com



<https://bit.ly/HPC-ADMIN-Update>





FreeBSD 12 as an alternative to the Linux desktop

# Swapping Tux

FreeBSD is a reliable and highly secure server operating system. We look at how FreeBSD fares as a desktop system. *By Erik Bärwaldt*

**F**reeBSD has been around since 1993 and enjoys an excellent reputation, especially in the server sector. The system is based on the Berkeley Software Distribution (BSD), a Unix-style operating system whose origins go back to 1977. Numerous BSD variants, such as TrueOS (the former PC-BSD), Dragonfly BSD, or GhostBSD, make the scene just as confusing as the Linux world.

With the exception of TrueOS and GhostBSD, BSD derivatives don't focus on the desktop but on servers, storage appliances, routers, and firewalls. However, mainline BSD variants like FreeBSD have extensive software repositories with plenty of desktop tools if you're ready to look for them. We decided to explore the possibility of setting up a desktop system on FreeBSD.

## Installation

FreeBSD is available as an ISO image for various computer architectures. Whether you are using a traditional Intel computer, a Raspberry Pi, a PowerPC system, or a Sparc workstation, you'll find a version of FreeBSD for your hardware at the project site [1].

### Listing 1: Burning the Image

```
$ dd if=FreeBSD-12.0-RELEASE-amd64-dvd1.iso of=/dev/sdX bs=1m conv=sync
```

After downloading the right image for your application, burn it onto an optical disc or transfer it to a memory stick. For example, on Linux, you can use the `dd` command (see Listing 1). Please be sure to replace the drive name `sdX` with the correct drive for your system.

Then boot your computer from the newly created removable disc. FreeBSD will soon provide you with a simple boot manager, where you can choose to re-boot or to run the system in multiuser or single-user mode. In addition, there is an option to switch to the prompt. If FreeBSD does not show the boot manager, you may be asked to check the list of compatible hardware, which usually means that a hardware component is not supported.

By default, FreeBSD starts in multiuser mode from the boot manager, and shortly afterwards, you will see an Ncurses dialog asking if you want to start a Live operating system or install right away. If you opt for Live mode, a login prompt immediately appears, and you can log on as `root` without password. You are taken back to the prompt again without a graphical user interface launching. Live mode is thus not suit-

able for beginners and users who prefer a graphical desktop.

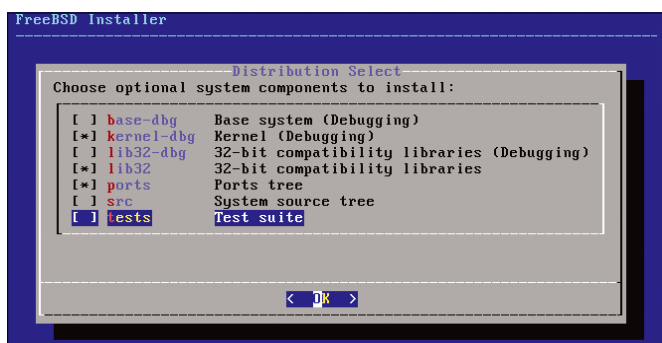
Static installation on a test partition is easier. Select the option for static installation. FreeBSD then calls the default `Bsdinstall` installer, which installs the system on your storage device in just a few steps. Since the BSD derivative uses a US keyboard layout by default, you may want to change the keyboard layout to suit your needs. Select one of the options and confirm by pressing *Select*. You then have the option of specifying a host name and importing optional system components of the BSD derivative via a selection window (Figure 1).

The default selection is usually for newcomers; you can thus move on to the next step, which involves partitioning the mass storage media, without making any adjustments. FreeBSD supports UFS and ZFS as native filesystems – both are firmly anchored in the Unix world and are primarily used in professional environments.

UFS2 is FreeBSD's default filesystem, while the sophisticated ZFS seems over the top for desktop systems. If the entire mass storage device is available for a complete reinstallation, you can tell the installer to perform automatic partitioning. In this case, the system creates a UFS2 root partition and a swap partition. If there are already other operating systems on the hard disk, the installer identifies them and supports manual partitioning.

Please note the completely different nomenclature of the drive names com-



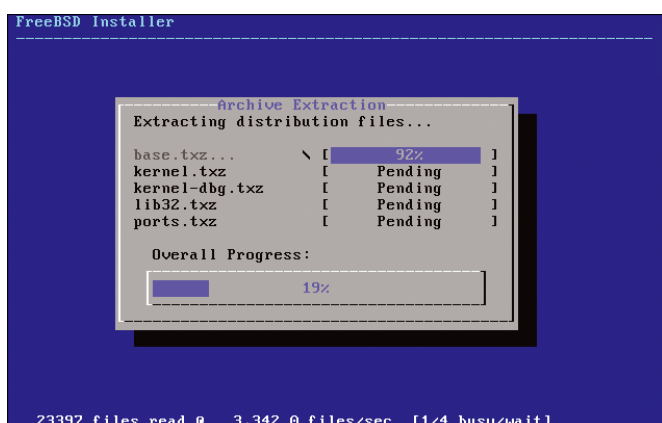


**Figure 1:** The FreeBSD installation takes place completely in text mode.

pared to Linux: The first hard disk or SSD under FreeBSD has a drive name of `ada0`; the other mass storage devices are counted in ascending order. Each partition is preceded by `p` and numbered in ascending order after the drive name. For example, the first partition on the first drive is `ada0p1`. When partitioning is complete, the installer copies the operating system to the mass storage device (Figure 2).

Enter a new password at the prompt for the `root` user. When you get to the configuration dialog, first enable network access and set the date and time. The routine will then branch to a dialog where you can select the system services that you want FreeBSD to launch automatically at boot time.

In another Ncurses window, you then compile a list of various security options that serve to harden the operating system against attacks. The configuration dialog then takes you through the process of creating new users and entering their authentication credentials. The installer asks for further optional information, such as the new users' membership in groups and the associated home directories (Figure 3).



**Figure 2:** When partitioning is complete, the copy process begins. A progress bar marks the progress.

In the *Final Configuration* window, you will find all previous configuration dialogs again. If you want to modify the configuration, activate the dialog with the arrow keys. When all changes are complete, you return to the *Final Con-*

*figuration* screen. Use the *Exit* option to reboot from the newly installed mass storage device.

## First Launch

The system first opens the same boot loader as on the installation disc and starts multiuser operation if you do not select any of the options offered. You'll soon reach a login prompt where you can log in as the `root` user. Now you can install a graphical user interface. The Ports Collection provides a variety of desktop environments for this purpose.

First you have to install the X11 server. At the prompt, type `pkg install xorg` (Listing 2, Line 1). Since no package management tool is installed as yet, the routine first asks if it should download and configure a package manager; you will want to say yes. FreeBSD then collects files for the missing package manager from the repository servers and installs them.

The routine then downloads the complete X11 system, including the matching fonts, and installs it on your mass storage device. Next, at the prompt, add the users previously created in the system to

the groups *video* and *wheel* using

the commands from Listing 2, Lines 3 and 4. Then open the Vi text editor at the prompt and edit the `/etc/rc.conf` file (Line 5).

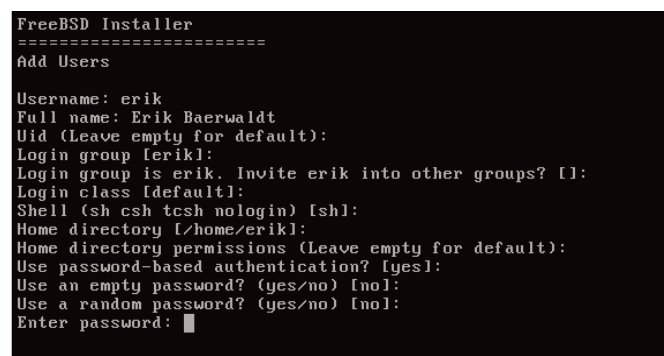
`/etc/rc.conf` is where you need to add the `dbus_enable="YES"` line (see the box entitled "Editors"). The `pkg install desktop_name` command lets you install the desired desktop environment on the system [2]. For example, to install XFCE, the command line is `pkg install xfce` (Line 7). At another prompt, the program lists the number of packages required for the desktop environment and their space requirements. The installer will then ask you if you really want to install these packages, which you can do by pressing [Y].

Now you have installed both the X11 server and a desktop environment. To ensure that they launch when the `startx` command is entered, create a suitable `.xinitrc` (Listing 2, line 8) at the prompt in the last step. Now you can load the desktop by entering `startx` together with the desktop environment.

If the X server does not start as expected but is closed with a note to the effect that a D-Bus UUID is missing, you need to enter the `dbus-uuidgen --ensure` command at the prompt. The X server and desktop environment will then launch correctly.

## Manual Work

FreeBSD is designed primarily for server systems, and it lacks many tools to set up the system on the desktop. Since the hardware detection and driver support for desktop systems also leave much to be desired [3], FreeBSD requires more manual work than most current Linux distributions if you want to use the operating system productively with a graphical user interface.



**Figure 3:** All configuration work, including creating new users, takes place in the terminal.



### Listing 2: Setting Up X11

```
# pkg install xorg
[...]  
# pw groupmod video -m user_name  
# pw groupmod wheel -m user_name  
# vi /etc/rc.conf  
[...]  
# pkg install xfce  
# echo ". /usr/local/etc/xdg/xfce4/xinitrc" > ~/.xinitrc
```

The graphical configuration tools of the working environments are available after the basic installation, but some of them only work in a limited way. In many cases, the system does not fully detect graphics cards and often offers only rudimentary display settings on the desktop [4]. For XFCE 4.12, only the SVGA and XGA resolutions were available in our lab (Figure 4).

In order to adapt the system to the graphics hardware and output devices, you either need to set the appropriate video modes via the `xrandr` program or modify various X server configuration files directly by hand to reflect the installed hardware. You can also perform these tasks in the terminal.

## Packages

The terminal is the linchpin in controlling and customizing FreeBSD, and it offers few surprises for experienced Linux users. FreeBSD's command set is very similar to Linux's, so you can use the common Linux commands in FreeBSD as well.

FreeBSD has features similar to Linux for installing additional software. Since the system installation including the desktop has virtually nothing in the way of additional programs, you can customize the installation to suit your own needs. The result is a lean and resource-saving system. However, even everyday users will spend a large amount of time adding applications, because there is no

## Editors

FreeBSD installs the Vi editor by default, but using this editor can be a little overwhelming for newcomers. As an alternative to Vi, you can also use the package manager to install a friendlier editor such as Nano. The package is named after the program, so the installation just requires you to type `pkg install nano`.

graphical front end to browse through the package sources.

The two common forms of software installation on FreeBSD

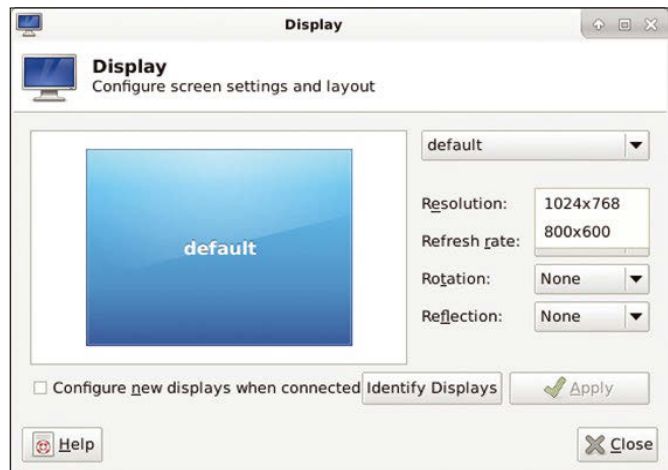
are direct integration using prepared binary packages and sourcing the available ports. You can import binary packages from the repositories into the system using the `pkg install package_name` command. Ports, on the other hand, consist of makefiles and fixups, so you have to compile them from the source with a "recipe".

To retrieve the Ports collection, use the `portsnap` command. With the appropriate parameters, `portsnap` lets you download compressed snapshots of the file archive from the Internet and store them in the `/usr/ports/` directory (Listing 3, first three lines). The programs can then be installed from the `ports` directories using the `make install clean` command sequence. If required, you can enter application-specific parameters for compiling after the `make` command.

The last two lines of Listing 3 build the Filezilla FTP client from the source code and install it on the FreeBSD system. The `make` script automatically loads the dependencies required by the application and builds them at the same time. Especially when building the first ports, the system has to install numerous tools.

In the meantime, prompts for special options continually pop up on screen – asking you, for example, whether you want to build the documentation at the same time. You will not typically need to adjust anything and can simply press `OK` to close the dialog. The whole process takes more than 20 minutes even on a faster computer.

The FreeBSD project provides a well maintained manual with detailed docu-



**Figure 4:** Due to missing hardware detection, FreeBSD severely limits the resolutions available for some graphics cards and monitors.

mentation on ports and the software installation process [5]. The project shows which ports are available under which names in the web-based Ports collection, which fortunately has a search function [6].

## Security

FreeBSD is considered an extremely secure operating system on server systems, and for good reason. The FreeBSD distribution provides multiple firewalls that all work at kernel level. The syntax of the packet filter firewall (Pf for short), which is ubiquitous in the BSD universe, is similar to the syntax of Ipfiler. The Netfilter/Iptables firewall used on Linux does not support FreeBSD. As with Linux, the packet filter firewall is controlled by easy-to-read rules.

FreeBSD also has jails [7] as an interesting security component. Jails are, as the name suggests, isolated instances that enhance the security of the operating system using a modified `chroot` environment. The virtual prison prevents individual processes from gaining access to resources outside the `chroot` environment.

## Updates

The FreeBSD community keeps its system up to date. There are no fixed patch

### Listing 3: portsnap Command

```
$ portsnap fetch  
$ portsnap extract  
$ portsnap update  
$ cd /usr/ports/ftp/filezilla  
$ make install clean
```

```

Terminal
File Edit View Terminal Tabs Help
root@Esprimo-Mobile-D9510:~ # freebsd-update
usage: freebsd-update [options] command ... [path]

Options:
  -b basedir  -- Operate on a system mounted at basedir
                (default: /)
  -d workdir  -- Store working files in workdir
                (default: /var/db/freebsd-update/)
  -f conffile -- Read configuration options from conffile
                (default: /etc/freebsd-update.conf)
  -F          -- Force a fetch operation to proceed
  -k KEY      -- Trust an RSA key with SHA256 hash of KEY
  -r release  -- Target for upgrade (e.g., 11.1-RELEASE)
  -s server   -- Server from which to fetch updates
                (default: update.FreeBSD.org)
  -t address  -- Mail output of cron command, if any, to address
                (default: root)
  --not-running-from-cron
                -- Run without a tty, for use by automated tools
  --currently-running release
                -- Update as if currently running this release

Commands:
  fetch      -- Fetch updates from server
  cron       -- Sleep rand(3600) seconds, fetch updates, and send an
                email if updates were found
  upgrade    -- Fetch upgrades to FreeBSD version specified via -r option
  install    -- Install downloaded updates or upgrades
  rollback   -- Uninstall most recently installed updates
  IDS        -- Compare the system against an index of "known good" files.

root@Esprimo-Mobile-D9510:~ #

```

**Figure 5:** The FreeBSD update routine has many options.

days for FreeBSD, as there are for other operating systems. The security and general system updates (which also include updates to new major and minor versions) are installed using the `freebsd-update` command (Figure 5) and corresponding options (Listing 4). If an update shows unwanted side effects, the last set of changes can be rolled back with `freebsd-update rollback`.

#### Listing 4: Installing an Update

```

$ freebsd-update fetch
$ freebsd-update install

```

You can customize the update using the `/etc/freebsd-update.conf` configuration file. The configuration file is where you will find, for example, the `Ignore-Paths` option, which lets you exclude certain directories from the update process. This setting prevents manual changes to the system from being overwritten by an update.

If you enter `freebsd-update cron` in `/etc/crontab`, you can automate the update process to execute once a day in the background. In this case, the process mails information on the updates and notes on system security to the `root` user.

You can either read the mails using the `spartan mail` command or install a text-based email client like `Mutt` or `Alpine` (Figure 6). Comprehensive documentation [8] is available for the functions of the update tool.

## Conclusions

FreeBSD is very stable and provides interesting security features, including the jails concept, but the system does not provide an easy or efficient operating system setup. In our lab, FreeBSD proved less suitable than the popular Linux variants as a desktop system.

The gappy hardware support and the lack of graphical tools for installing and customizing the system cause a high volume of manual work and time overhead before you have your FreeBSD desktop system to your liking. Beginners who are interested in BSD would be better off using a derivative such as `TrueOS` [9] or `GhostBSD` [10] if the goal is to set up a desktop system. Both of these BSD variants emphasize the desktop and focus on developing their own graphical tools. ■■■

## Info

- [1] Download: <https://www.freebsd.org/where.html>
- [2] "Installing a Desktop Environment on FreeBSD": <https://www.freebsdoundation.org/freebsd/how-to-guides/installing-a-desktop-environment-on-freebsd>
- [3] Hardware compatibility: [https://www.freebsd.org/doc/en\\_US.ISO8859-1/books/faq/hardware.html](https://www.freebsd.org/doc/en_US.ISO8859-1/books/faq/hardware.html)
- [4] Documentation for setting up a desktop: [https://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/x11-wm.html](https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/x11-wm.html)
- [5] "Installing applications: packages and ports": [https://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/ports.html](https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/ports.html)
- [6] Ports list: <https://www.freebsd.org/ports/>
- [7] Jails: [https://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/jails.html](https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/jails.html)
- [8] Freebsd update documentation: [https://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/updating-upgrading-freebsdupdate.html](https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/updating-upgrading-freebsdupdate.html)
- [9] TrueOS: <https://www.trueos.org>
- [10] GhostBSD: <https://ghostbsd.org>

```

Terminal
File Edit View Terminal Tabs Help
ALPINE 2.21.9999 MESSAGE TEXT Folder: INBOX Message 1 of 2 73%

Checking negative group permissions:
No /var/log/mount.today
Checking for uids of 0:
root 0
toor 0

Checking for passwordless accounts:
root::0:0:0:Charlie &:/root:/bin/csh

Checking login.conf permissions:
No /var/log/dmesg.today
freebsd login failures:
freebsd refused connections:

? Help      M MsgIndex  P PrevMsg   - PrevPage  D Delete    R Reply
0 OTHER CMDS > ViewAttach N NextMsg   SPC NextPage U Undelete  F Forward

```

**Figure 6:** The system mails important information about updates and status to the root account. You can read the information with text-based mail clients such as Alpine.



Using VolksPC OS to run  
Linux applications over Android

# The Best of Both Worlds



Linux desktop users can now use an estimated two million Android apps that were previously unavailable on Linux with VolksPC OS. *By Vasant Kanchan*

Open source enthusiasts have spent years waiting for the Linux desktop revolution. The Linux environment now supports hundreds of useful, stable, and secure desktop tools that are available for no cost, but Linux still has not displaced Windows or macOS in the race for desktop marketshare.

Android, however, which uses the Linux kernel, has actually become mainstream and is competing quite well. In fact, if you combine desktop, laptop, tablet, and mobile usage, Android has reached 38 percent of marketshare, narrowly overtaking Windows as the most popular operating system in the world.

Android runs on ARM chips that support low-power and low-cost computers. The Android option is therefore quite economical, but what about all that free software you've become accustomed to on your Linux system? Do you need to give up your Linux desktop just because you're running an Android tablet?

The answer is no. A handful of solutions provide options for running desktop Linux applications over Android. This article introduces one of those solutions: VolksPC OS [1].

VolksPC OS, developed by a company of the same name, is designed to let you

run a Debian Linux desktop on top of Android with the following features:

- Full support for running Android applications.
- Multi-windowed Linux desktop applications that work well with a keyboard and mouse.
- Instantaneous switching between Android and the Linux desktop.
- Quick launch of Android applications from the Linux desktop.
- Installing applications from Google Play and Debian repository.

Some Linux-over-Android solutions depend on the Android layer for hardware access and are therefore hardware neutral, running on most or all Android systems. VolksPC OS takes a different approach. The X client within VolksPC OS addresses the hardware drivers directly, which means VolksPC can operate much faster than many of the alternatives. On the other hand, direct hardware access means that the VolksPC developers must actively port it to the hardware systems on which it will run.

VolksPC sells its own ARM-based mini PC that runs Android and the Debian Buster Xfce desktop. You can also use VolksPC OS with the popular ODROID-C2 single-board computer (SBC). The company has also ported VolksPC OS to several other ARM single-board systems and Android TV boxes. In this article, I

refer to Debian as the distro used with VolksPC, but you can also use Fedora, Ubuntu, and other Linux alternatives.

Around 95 percent of the VolksPC software is open source, but the installer app and some graphics libraries included with the solution are proprietary. You can download the ODROID-C2 software from the VolksPC website for \$14.99 or purchase the Model-S905X mini PC (with VolksPC OS included) for \$99.

All modifications to open source software are available in source code form through the VolksPC website per the terms of the GPL.

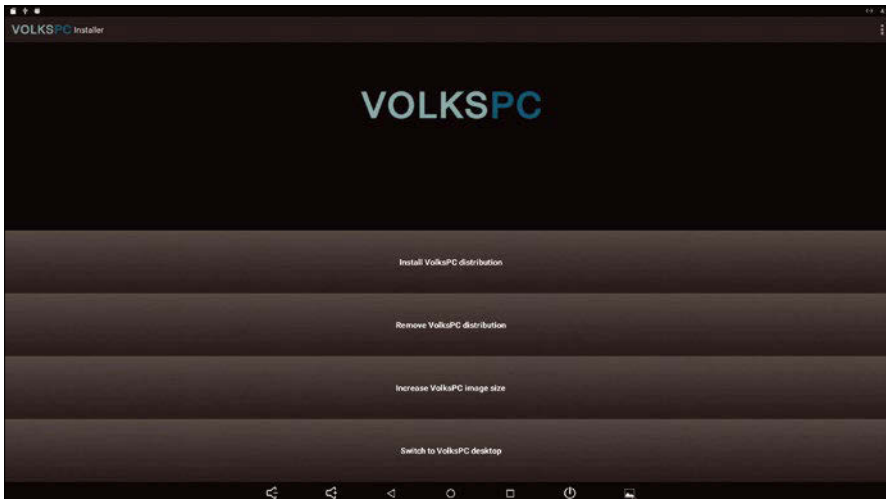
## Benefits of Running Android with Debian

Android offers many advantages, such as:

- A large number of available applications and games; none of which will be ported to Linux desktop.
- Very good support for touch interface.
- Hardware accelerated multimedia.
- A large number of ARM SoC's with Android implementation.
- Runs on an enhanced Linux kernel.
- Most phones already ship with Android.

However there are some limitations when using Android with large displays:

- Each application uses the whole screen.



**Figure 1: Android VolksPC Installer.**

- Cannot view multiple applications at the same time.
- No support for desktop-style word processing, email, etc. with keyboard and mouse input.

Linux desktop distributions such as Debian have very good support for legacy desktop-style applications, such as LibreOffice, the Firefox browser, and the Thunderbird email client. These applications also work very well with keyboard and mouse input.

Currently shipping phone hardware has enough resources to easily run the Debian desktop. In fact, VolksPC provides such a solution for the popular ODROID-C2 SBC, which is based on a quad-core Cortex A53 running at 1.5GHz. Indeed most phones that ship today are way more powerful. Another important hardware feature on the newer phones is the availability of USB-C port, which when connected to a hub can provide both HDMI output and USB keyboard and mouse input. In other words, your phone can be converted to a desktop or even a laptop (e.g., the Samsung DeX platform).

Consumers already use their cell phones as their primary computing device. The ability to run the Debian desktop on their phones would address those computing scenarios that work better with a large screen driven by a physical keyboard and mouse.

## Implementation and Performance

Android and the Debian desktop share the same kernel, so even if the userspace libraries are different, both applications can run simultaneously. The biggest dif-

ficulty in integrating the two systems is in the graphics technology. Android uses SurfaceFlinger and Debian uses X windows for drawing graphics.

VolksPC OS runs the Debian desktop using the Android Linux kernel and the Android init program. The Debian distribution is installed using VolksPC Installer (Figure 1) and appears to the rest of Android as a part of the VolksPC Installer files. The /data partition is where application data is stored in Android and is mounted read/write during system boot. The Debian root filesystem resides as a single 3GB file at /data/org.volkspc.installer/files/linuxfs.img and includes all the Debian installed applications and configuration. Android init scripts will mount this file and start Debian during system boot. VolksPC Installer allows you to manage the installation.

VolksPC Installer manages the Linux desktop with these options:

1. *Install VolkSPC distribution.* This will install the Debian distribution on Android. The installer first looks for the installation zip file, `volkspcimg.zip`, on either a microSD card, USB stick, or internal eMMC.
2. *Remove VolkSPC distribution.* This will remove the distribution from internal eMMC. You will need to log out of Debian before you can uninstall. Android factory reset will also remove Debian and Linux user data.
3. *Increase VolkSPC image size.* The root filesystem is distributed as a single file, `linuxfs.img`, emulating a disk and is of a fixed size. However, this image can get full if the user installs a lot of Debian applications. This option al-

lows a user to increase the VolksPC image size in 1GB increments.

The Linux desktop uses the X Window System (Figure 2) to implement graphics. This is a client-server approach where the X server controls the display and is responsible for drawing graphics on the screen. Applications, on the other hand, are X clients that exchange messages with the X server via the X windows protocol. Applications are typically written through high-level toolkits such as GTK, QT, Tcl/Tk, etc.

Figure 2 shows an application communicating with the X server in a Linux desktop environment. Most of the complexity of X windows is due to the client-server architecture. Design aspects that make X windows complex and slow are:

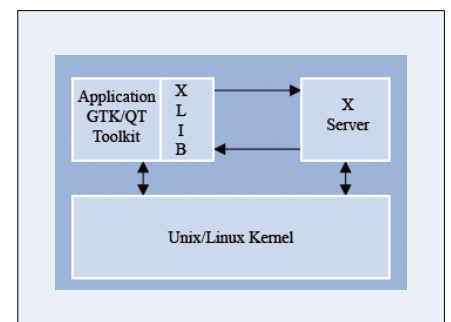
- The client and server have to format commands and responses (aka, the X11 protocol).
- Synchronous and round-trip requests are inefficient.
- Frequent context switching between the application and the X server degrades performance.
- Graphics rendering can only start after the X server starts running.
- The X server implementation is not multithreaded.

There are efforts within the open source community to move to a new display server technology, such as Wayland, which to date is still a work in progress.

Since in most use cases, the application needs to draw graphics on the client's machine, it makes sense to implement graphics processing as a driver. VolksPC implements core graphics functions as a kernel module. A modified userspace X11 library communicates directly with the kernel driver (Figure 3).

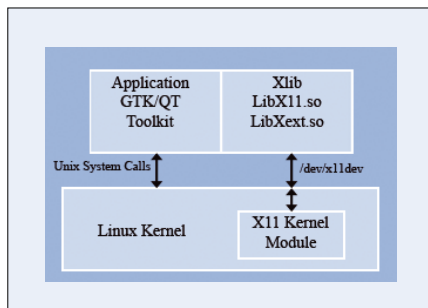
The advantages of this design are:

- Low latency and no penalty for round-trip requests



**Figure 2: The X Window System.**





**Figure 3: VolksPC X11 compatible graphics.**

- No buffering and formatting of requests and responses
- No context switch overhead
- Direct rendering of all graphics by the client
- No changes required for existing Xlib applications
- Two to 30 times faster than the standard X Window System.

With support from the kernel module, Debian applications can directly render to the frame buffer without going through Android's SurfaceFlinger. This is a simple and fast solution. Besides providing Xlib API support to Debian applications, the VolksPC kernel module provides the following additional features:

- Monitors keyboard shortcuts (LeftAlt + LeftMeta) and facilitates switching between Android and Debian display.
- State of Android graphics and Debian graphics is maintained.
- At any given time, only Debian or Android occupies the whole screen and switching is instantaneous.
- Applications are unaware of which desktop is currently being displayed on the screen.
- Provides a simple API for switching between Android and the Debian desktop.

Android application performance with VolksPC OS is identical to running standalone Android, whereas Debian GUI applications run faster than under X windows.

Besides application graphics rendering, there are many driver and configuration issues that need to be addressed to make a completely functional Debian desktop. These include:

- Support for a Bluetooth keyboard is handled by the kernel module.
- Audio for a Debian application is handled by the Linux ALSA driver.
- External drives mounted by Android are visible to the Debian desktop. This allows for easy sharing of files between Android and Debian.
- During startup, Android's time zone and language settings are used to configure Debian's time zone and keyboard.
- Android applications appear as icons on the Debian desktop and can be launched from Debian.

Hardkernel [2], the manufacturer of ODROID-C2, makes several ARM boards for which they provide both Android and Ubuntu distributions along with Linux kernel sources. VolksPC is ported to the ODROID-C2 board [3], which is based on an Amlogic S905, quad-core Cortex-A53 running at 1.5Ghz. This board is slightly faster than Raspberry Pi 3 B+ and more importantly comes with an Android Marshmallow port. The official ODROID-C2 Ubuntu 16.04.2 distribution is based on the MATE desktop 1.12.1 running a AARCH64 user space. The VolksPC distribution is based on Debian Jessie ARMHF. We also installed LXTask and GtkPerf to compare performance between VolksPC OS and Ubuntu [4] running on ODROID-C2 (see Table 1).

The memory usage is roughly the same except that with VolksPC OS I have also booted Android. The GtkPerf benchmark runs 176 percent faster on VolksPC OS. Also the client-server X11 runs on two cores when running the GtkPerf benchmark: one core running the GtkPerf client and another core running the X11 server. In the case of VolksPC OS, graphics are rendered directly on one core. It is impor-

tant to understand that X11 performs reasonably well when commands are streamed from an X11 client to an X11 server as is mostly the case with GtkPerf or x11perf benchmarks. Unfortunately, X11 architecture becomes an issue when you try to get information or events from the server. On the same ODROID-C2 system, the x11perf -prop command to get an X property from the X11 server runs 30 times faster with VolksPC OS.

So VolksPC OS provides a smoother graphics experience for the Linux desktop while also providing user access to an estimated two million Android applications.

The Debian Stretch version of VolksPC OS has already been released for ODROID-C2 and a preliminary port of Debian Buster is being tested internally.

## The Future of VolksPC OS

Android supports GPU and OpenGL ES applications, but Debian on VolksPC OS can only support 3D graphics if used with the Mesa open source OpenGL implementation. Unfortunately, most of Android's graphics implementation is proprietary. Still it is not a serious issue as most Debian applications only need 2D rendering, and as shown, VolksPC OS is much faster in this area. However, 3D rendering is important for games. VolksPC OS will support this if GPU vendors provide easy access to their graphics drivers and supporting libraries. Also Linux games require OpenGL, but most ARM SoCs only support OpenGL ES. Lack of 3D and video acceleration in Debian is not a show stopper, since VolksPC OS provides complete support for all the graphics and video acceleration on Android. Consequently, VolksPC offers an interim solution where Android applications are good for HD video playback and 3D graphics-rich games, while Debian applications are good for programming tools, scientific applications, word processing, and email. ■■■

**Table 1: VolksPC OS and Ubuntu Performance**

	Ubuntu 16.04.2 AARCH64, Mate 1.12.1	VolksPC OS Debian 8.8 ARMHF, Xfce 4.10
<b>Linux Kernel Version</b>	3.14.79	3.14.29
<b>LXTask</b>	After complete boot up shows 392MB used from the available 1717MB	After complete boot up shows 390MB used from the available 1717MB
<b>GtkPerf</b>	Took 13.89 seconds for 100 iterations	Took 7.89 seconds for 100 iterations

## Info

- [1] VolksPC OS: <https://www.volkspc.org>
- [2] Hardkernel: <http://www.hardkernel.com>
- [3] VolksPC OS running on ODROID-C2: <https://www.youtube.com/watch?v=tTW7OyTFkGQ>
- [4] VolksPC OS vs Ubuntu on ODROID-C2: <https://www.youtube.com/watch?v=5jmvYryz2hw>

Shop the Shop

[shop.linuxnewmedia.com](http://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.

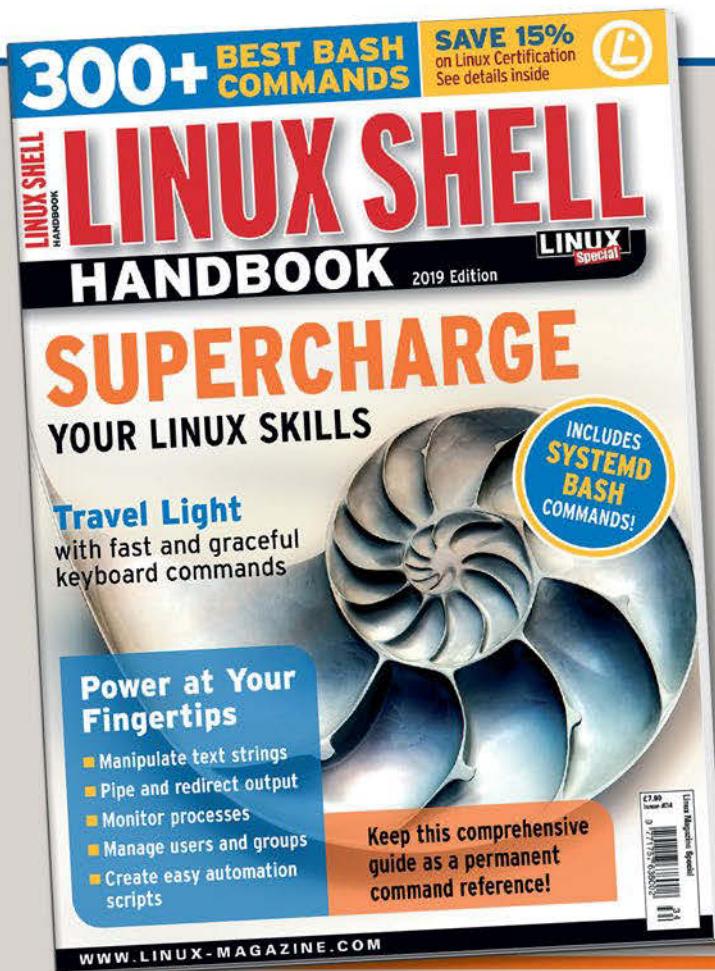
**2019  
Edition!**

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](http://shop.linuxnewmedia.com/specials)





## Using sudo options to enhance security

# SUDO SECURITY VOODOO

By taking the time to learn sudo's many options, you can make your system more secure.

By Bruce Byfield

**T**he `sudo` command [1] has been around since the 1980s, but it has gained popularity in recent years as the default tool for running commands as root in Ubuntu. However, there's far more to `sudo` than Ubuntu's policy. In fact, `sudo`'s man page is over 2,400 lines long, covering a staggering number of situations – some of which, like many powerful Linux commands, can get you in a lot of trouble if you are careless. `sudo` also offers options that can greatly enhance security, especially if you take the time to be creative.

Why would you want to enhance your security? The answer is that, from a security viewpoint, Ubuntu's use of `sudo` can be viewed as a problem (although opinions do differ). As you may know, when `sudo` is configured the way it is in Ubuntu, you can use the password for your everyday account to log in to `sudo` and run root commands. The trouble is that any password for an everyday account is exposed in a way that the root account is not, especially on the Internet. That means that if the everyday account is compromised, the intruder gains root access, too, if `sudo` is set up on the system. The traditional separate root password is more secure,

although less convenient. Fortunately, though, you can manage both convenience and security by taking the time to learn the details of `sudo`.

### Editing sudoers

`sudo` has a unique configuration system. You can configure the behavior of the `sudo` command using the `sudoers` file in the `/etc` directory (Figure 1). `sudoers` lists default behaviors and the privileges granted to individual users. As the top of the `sudoers` file warns, it should only be edited using the `visudo` command. `visudo` is designed to prevent you from editing `sudoers` in a way that would cripple or disable `sudo` by doing all editing in a temporary file and replacing the original file only when all editing is done. Should you make an error while editing `sudoers`, as you try to save, `visudo` will give you the option to reopen its temporary copy of `sudoers` to correct the errors (e) or discard your edits (X) – choices that you obviously should not ignore. Depending on the distribution, `visudo` may or may not display these choices, but they will be available whether displayed or not.

Starting `visudo` opens a command-line text editor. Usually, this editor is Vim,

but you can also set another editor. For instance, if you want `visudo` to open `nano` instead, you can change the environmental variable with the command:

```
export VISUAL=nano; visudo
```

The `sudoers` file itself is divided into three sections. The first section is the default behaviors. It lists one option per line. For example, if you want to use the `insults` option – a genuine option, which insults users who make mistakes trying to log in to `sudo` – the entry is:

```
Defaults insult
```

Default settings can be overridden by specific users' or groups' settings. However, above the specific settings is a section that defines aliases for hosts (host names, IP addresses, network numbers, or netgroups), for users (account names, UIDs, groups or netgroups), for users to run as (account names, UIDs, groups or netgroups), and for commands (usually with full path names). All aliases consist of uppercase characters or underscores.

These aliases exist to make the defining of specific settings less cumbersome.

For example, if you want user accounts `bab`, `plw`, and `vaf` to all have the same privileges, you could create the alias

```
User_Alias ADMIN = bab,plw,vaf
```

With this user alias defined, you can simply define privileges for `ADMIN`, instead of for `bab`, `plw`, and `vaf` separately. In the same way, you create an alias for a list of network terminals from which a `sudo` user can log in or a set of commands that a group can or cannot use. Although aliases can take time to set up, they make creating a new set of privileges or editing an old set much easier.

In the third section, individual privileges are defined one per line, using this structure:

```
[USER or ALIAS] [TERMINALS]=[USER  
RUN AS] [OPTIONS:] [PERMISSIONS]
```

Permissions are generally those that a user or alias can use, but adding an exclamation mark (!) in front of them turns the list into those that cannot be used.

For instance, the basic entry for the root user with all privileges is:

```
root ALL=(ALL) ALL
```

Individual terminals or commands can be entered instead of `ALL`. `USER RUN AS` and `OPTIONS` are optional, so that the line

```
bab ALL=(ROOT) passwd, chown,  
chgrp, chmod
```

is enough to give user `bab` the ability to change passwords and permissions from all terminals on the system, while running as root. More simply still, de-

fining an alias called `ATTRIBUTES` that included all four commands would reduce the line to:

```
bb ALL=(ROOT) PERMISSIONS
```

Remember, though, that specific permissions override those set as defaults.

## Defaults, Privileges, and Options

Defaults and privileges are defined using the options listed in the `sudoers` man page. The available options range from requiring no login whatsoever to specific settings for greater security.

Many security options affect how to log in to `sudo`. For example, `password_tries=NUMBER` sets how many times a user can try to log in before being denied. It is accompanied by `passwd_timeout=MINUTES`, which sets how long `sudo` runs before logging out a user – an especially useful feature when using root privileges, since basic security decrees that the root account should be used for as short a time as possible. With `passwd_timeout`, you no longer have to rely on your own memory to close root as soon as possible. Less drastically, `timestamp_timeout=MINUTES` sets the time before `sudo` prompts for another login.

Other options set which password `sudo` requires from you. The option `runaspw` requires the current account's password, just as in Ubuntu. However, you can do better than that simply by specifying `rootpw`, which requires the root password. Subtler still, `targetpw USER` can require another account's password, so that you set up a user with root privileges that is used only with `sudo`. With `targetpw`, an intruder will need to be able

to read the list of users in order to find the password for `sudo`.

Still another basic piece of security is `noexec`. `noexec` is designed to limit the running of applications from which other commands can be run. Without `noexec`, the running of one application could easily give intruders access to other applications in the system.

However, individual privileges are where ingenuity reigns. With a little planning, you could set up separate accounts with limited root privileges. For example, one account could only be permitted to run tools for installing package managers, such as Debian's `apt-get`, `apt`, and `dpkg`, while another would be limited to running commands for changing file attributes. With such an arrangement, Linux can be made to mimic other sophisticated Unix descendants. While each limited account can do specific functions, an intruder who gains access to one account via `sudo` would not have complete control over the system.

## Getting More from Sudo

`sudo` is a sophisticated command. However, as you can see, it is also a seriously under-used one. Even if you have no interest in such options as setting the command prompt, there are still a number of options that can make switching into `sudo` to temporarily gain root privileges safer – and all without sacrificing any convenience once everything is configured.

It is easy to think of `sudo` as a magical word that some distributions require at the front of administrative commands. And for many people, that may be enough. But `sudo` can also be much more. You might even investigate `sudo` plugins like `Privilege Manager for Sudo`, [2] which allows you to set policies for `sudo` graphically, or `sudo_pair`, which requires an admin to approve any use of `sudo`. But, one way or the other, if you are concerned about security, you owe yourself the time to learn what else `sudo` can do. ■■■

### Info

- [1] Sudo: <https://www.sudo.ws/>
- [2] Privilege Manager for Sudo: <https://www.oneidentity.com/products/privilege-manager-for-sudo/>
- [3] `sudo_pair`: [https://github.com/square/sudo\\_pair/tree/master/sudo\\_pair](https://github.com/square/sudo_pair/tree/master/sudo_pair)

```
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults    env_reset
Defaults    mail_badpass
Defaults    secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
#
# Host alias specification
#
# User alias specification
#
# Cmnd alias specification
#
# User privilege specification
root    ALL=(ALL:ALL) ALL
#
# Allow members of group sudo to execute any command
/etc/sudoers
```

**Figure 1:** The `sudo` configuration file, `sudoers`, lists the default behaviors and privileges for users.





The decade long wait for Bash 5

# Full House

It's a coincidence that the Linux kernel and Bash jumped to version 5.0 at about the same time. While Linus assigns the numbers as he sees fit, Bash changes its version when major adjustments are made. Here's what users can expect in Bash 5. *By Bernhard Bablok*

**M**y last article about a Bash version change is 10 years old [1]. Version 4 was in the starting blocks at that time, but it took some time for all distributions to switch to this version. Nobody puts their production system at risk without good reason.

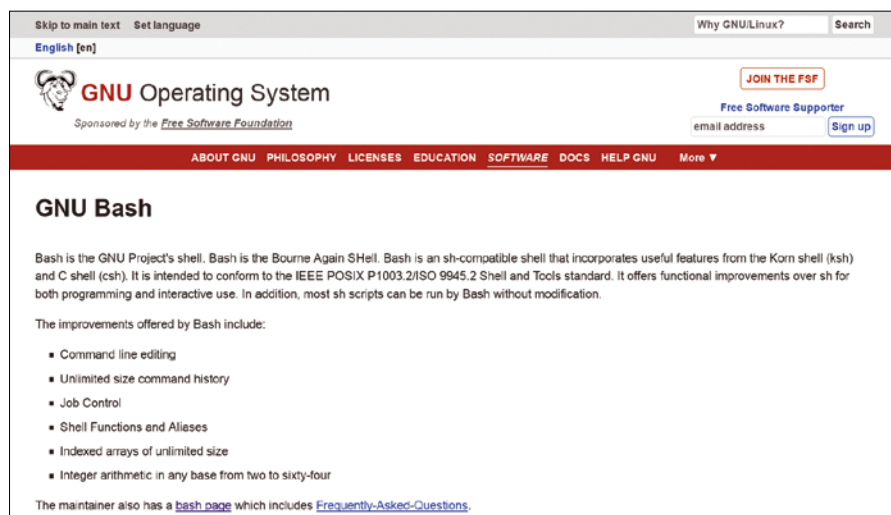
Nevertheless, the change was very attractive for developers of complex scripts, because – thanks to associative arrays – a completely new data structure was introduced. The advantages were more elegant, simpler programs that were also easier to maintain. Other important changes included the coproc command (which supports parallelization) and redirection operators.

## Treasure Hunt

For Bash 5, the first job is to look at files such as `NEWS` and `CHANGES` to separate the important changes from the unimportant ones. These files are linked from the Bash page of maintainer Chet Ramey [2], who – keeping to the true

spirit of Bash – almost completely does without graphical elements. The Bash homepage on [3] is in no way inferior (Figure 1). The `NEWS` file lists 44 changes to Bash and another 11 to the Readline library used since the last stable version. This is information overkill in its purest form.

On closer inspection, you feel more like a gold digger – the yield in terms of actual nuggets is extremely low. Even for me, and I have written several 10,000 lines of Bash code in the course of my life and exhausted the functionality of Bash in many areas, most of



**Figure 1:** The Bash homepage – without reference to the 30th anniversary.

**Table 1: Changes in Bash 5**

Change	Comment
New variable EPOCHSECONDS	Seconds since 01.01.1970
New variable EPOCHREALTIME	Seconds since 01.01.1970 (with microseconds)
New variable BASH_ARGV0	Program (\$0), can be redefined
New built-ins: rm, stat, fdflags, seq	Less forks for external commands
history -d with negative offsets	Delete history, oldest first
coproc Name now expands Name	Dynamic coprocesses possible
Bash is compilable with a static PATH	For secure shells
Associative and indexed arrays can also contain only spaces in the index	More a logic change than an exciting feature

the changes relate to fairly arcane border areas.

One example is associative arrays. In Bash 5, strings consisting only of blanks can also act as keys. This is certainly logical, but not a change that would make you want to migrate prematurely. The new EPOCHSECONDS and EPOCHREALTIME variables are more useful. The first now contains the Unix time (seconds since the beginning of the epoch); the second additionally has the microseconds.

For example, anyone who feeds measurement data into an RRD database every second benefits from this and saves the operating system an additional fork (Figure 2) by not having to call the external Date binary. A list of potentially interesting changes can be found in Table 1.

## Installation

Because of the very manageable number of epochal innovations, installing the new Bash is not really an attractive option. However, if you want to try out Bash 5.0 anyway, you can build the

shell yourself. Alternatively, openSUSE's build service offers several experimental and community builds. The problem with these builds, however, is the additional dependency on new *glibc* versions, so an installation is almost equivalent to a complete update of the whole system.

Users of the rolling release version Tumbleweed offered by openSUSE will find RPM packages for Bash 5.0 and Readline 8 that ensure a simple update path. Arch Linux now also provides a Bash 5.0 package; other well-known Linux distributions are likely to follow soon.

## Changes

Given the meager yield of genuine innovations, the question arises as to why the Bash developers changed the major version at all – at the end of the day, it still wasn't quite clear to me either. One potentially decisive hint results from the detailed study of the individual changes. There are also some incompatibilities to the previous version.

Unlike the Linux kernel, changing the version for compatibility reasons is a common procedure for many software projects. If backward compatibility is broken, the team increments the major number. This sometimes leads to the situation where, for years, several versions require active support – Python is a notorious example.

Bash 5.0 basically shows that the scripting language has reached a mature state in the meantime. Like with the last change from Bash 3 to Bash 4, the major distributions will update to the current Bash in upcoming versions and almost no one will notice this change. By then, the worst of the newly-introduced bugs should have been fixed. ■■■

## Info

- [1] "A Good Bash" by Bernhard Bablok and Nils Magnus, *Linux Magazine*, issue 105, August 2009  
<http://www.linuxpromagazine.com/Issues/2009/105/Bash-4>
- [2] Chet Ramey's Bash page: <https://tiswww.case.edu/php/chet/bash/bashtop.html>
- [3] Official Bash website:  
<https://www.gnu.org/software/bash/>

## Author

Bernhard Bablok works at Allianz Technology SE as an SAP HR developer. When he's not listening to music, cycling, or walking, he deals with topics related to Linux, programming, and small computers. He can be reached at [mail@bablobk.de](mailto:mail@bablobk.de).

```
[bablobk@sirius:~] > bash --version
GNU bash, version 5.0.0(1)-release (x86_64-suse-linux-gnu)
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
[bablobk@sirius:~] > echo -e "EPOCHSECONDS:      $EPOCHSECONDS\nEPOCHREALTIME:      $EPOCHREALTIME\n$(date '+%s.%N'): $(date '+%s.%N')"
```

```
EPOCHSECONDS:      1548422118
EPOCHREALTIME:      1548422118.382155
$(date '+%s.%N'): 1548422118.384211129
[bablobk@sirius:~] >
[bablobk@sirius:~] > bin/test-it
$0:      bin/test-it
$BASH_ARGV0: bin/test-it
$BASH_ARGV0: mein-programm
$0:      mein-programm
[bablobk@sirius:~] > cat bin/test-it
#!/bin/bash

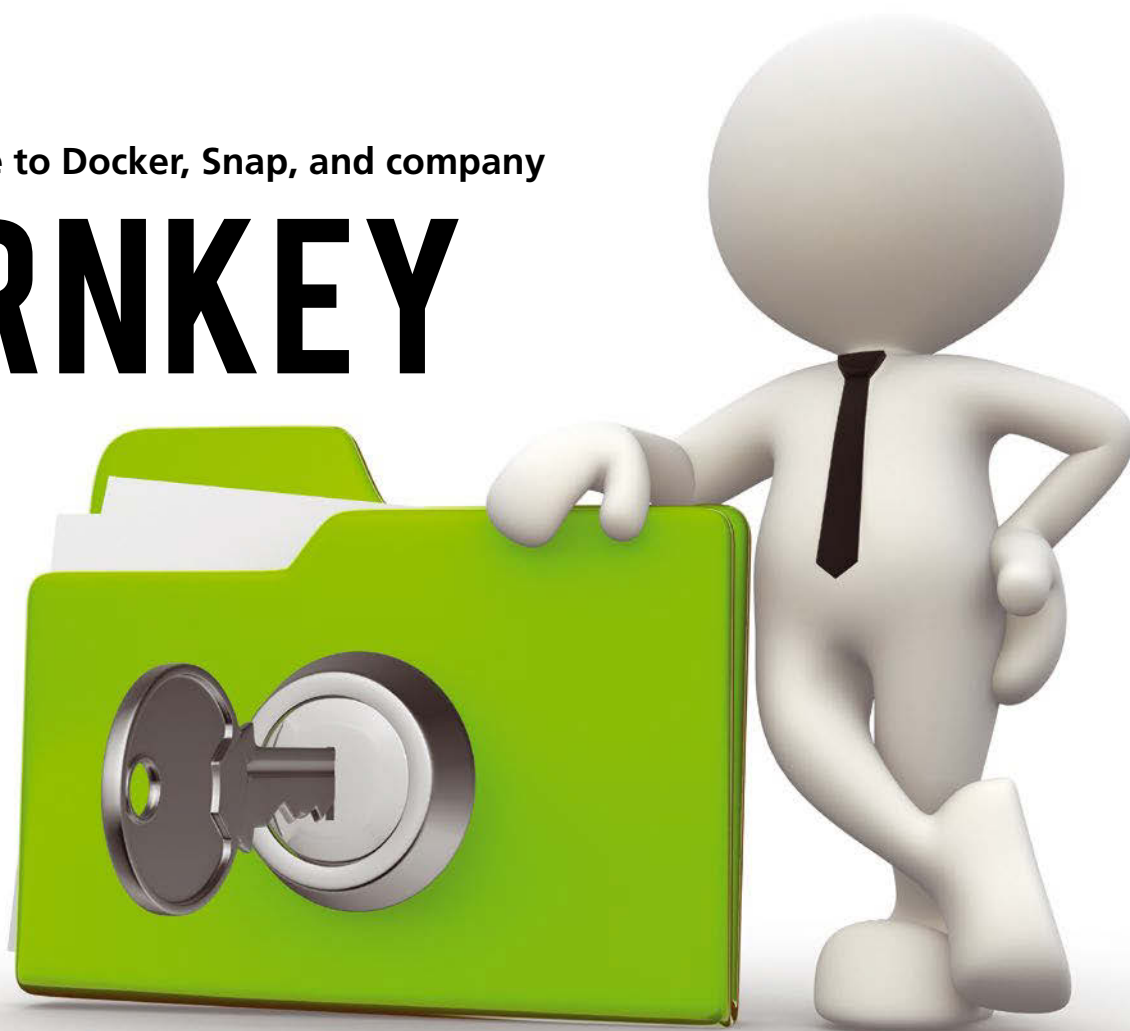
echo "\$0:      $0"
echo "\$BASH_ARGV0: $BASH_ARGV0"
BASH_ARGV0="mein-programm"
echo "\$BASH_ARGV0: $BASH_ARGV0"
echo "\$0:      $0"
[bablobk@sirius:~] >
```

**Figure 2: The new stable Bash version 5.0 by the Free Software Foundation in action.**



An alternative to Docker, Snap, and company

# TURNKEY



When setting up complex web-based services such as Drupal or Plone, there are many hurdles to overcome. Bitnami will make your job easier. *By Erik Bärwaldt*

**T**he task of installing servers – even on Linux – is often fraught with pitfalls; amateur admins face serious issues with finding all the dependencies and putting all the necessary pieces in place. Bitnami reduces the installation and configuration overhead associated with setting up a fully configured web server. The Bitnami project [1] provides complete packages, including the required infrastructure, and lets you install the whole stack all at once.

Bitnami is jointly developed and maintained by BitRock [2] and Bitnami, both from San Francisco, and it is available as free software under the Apache license. The individual stack components consist of the BitRock graphical installation program and the required server applications. Bitnami integrates the Linux-Apache-MySQL-PHP (LAMP) environment, which is often used with web ap-

plications, into the stack up front. Since the stacks run autonomously, like virtual machine (VM) appliances, they do not require clumsy and error-prone modifications to your Linux system.

In addition to providing a complete application stack, Bitnami also offers pre-configured VMs for some applications. Bitnami also provides some applications as cloud instances or Docker containers. The project offers numerous applications for free download, including WordPress, Joomla, MediaWiki, and Ruby.

See the catalog on the Bitnami website for a list of supported applications [3]. You'll find that Bitnami does not offer desktop apps but only complex server and infrastructure stacks.

## Setup

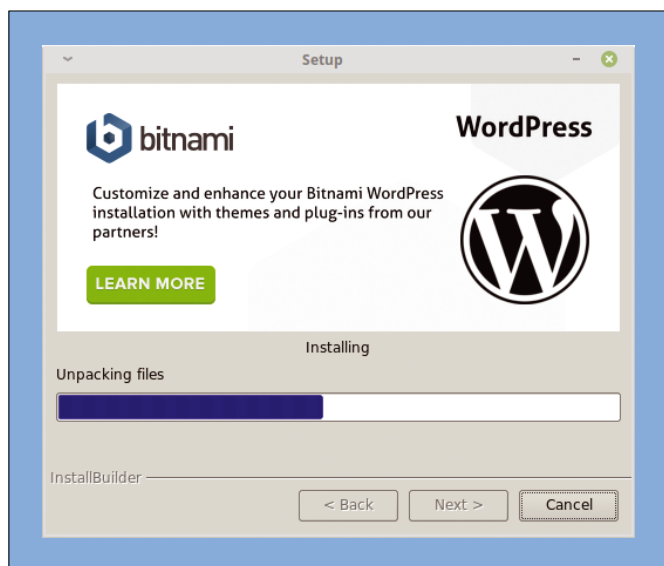
To install a Linux stack, select the desired application in the Bitnami Application Catalog and download it. You then

assign execution rights to the package and call it up with a mouse click. The installer, which now launches, first asks you about the language setting; you can change this in the selection field if needed. The actual installation dialog for the stack then appears.

In the second step, the installer lists the components it integrates into the stack. It makes sense to select all components; otherwise, the desired software may not work correctly.

After a click on *Next*, you are taken to the dialog for specifying the target path in which you want to install the stack. Bitnami creates a subfolder with the name of the logged-in user in `/opt/` by default. It creates a subdirectory there named after the application. You can change this path if so desired.

Then the dialog asks you for a username and password for the stack administrator account. Depending on the



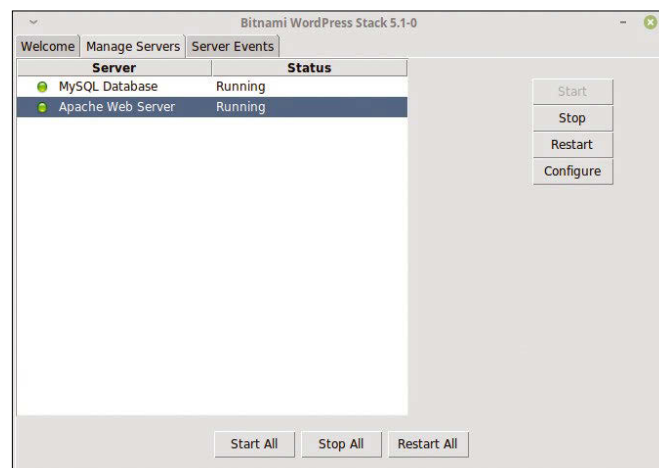
**Figure 1:** Bitnami installs the applications using a graphical front end.

stack, Bitnami offers additional optional input for the software it contains: For example, you can enter the name for the WordPress stack or forum software. In the next step, you specify whether you want to enable the email notification system – if so, you then need to enter the SMTP configuration for dispatching the mail. Bitnami will then install the stack including all the components on your hard disk (Figure 1).

Now a small welcome window with three tabs appears in the top left corner. In the active tab, *Welcome*, you will find five options in the bottom right corner. If Bitnami does not start the installed server application automatically, click on the *Go to Application* button to load the chosen program with the required basic applications.

If this does not work, one of the server applications that was configured as an additional component during the installation of the stack may not be running yet. The content of the *Manage Servers* tab gives you more details. This is typical of LAMP systems, where many server apps require an Apache web server and a MySQL database (Figure 2).

Use the buttons to the right of the server list to start, stop, or restart individual services. If you want to restart, terminate, or activate all services for the first time, use the buttons lined up at the bottom of the window. If necessary, you can also make fundamental changes to the configuration of a server by clicking on *Configure*. This includes,



**Figure 2:** Under *Manage Servers*, you can determine whether all the applications are working correctly.

for example, changing the port numbers, but also

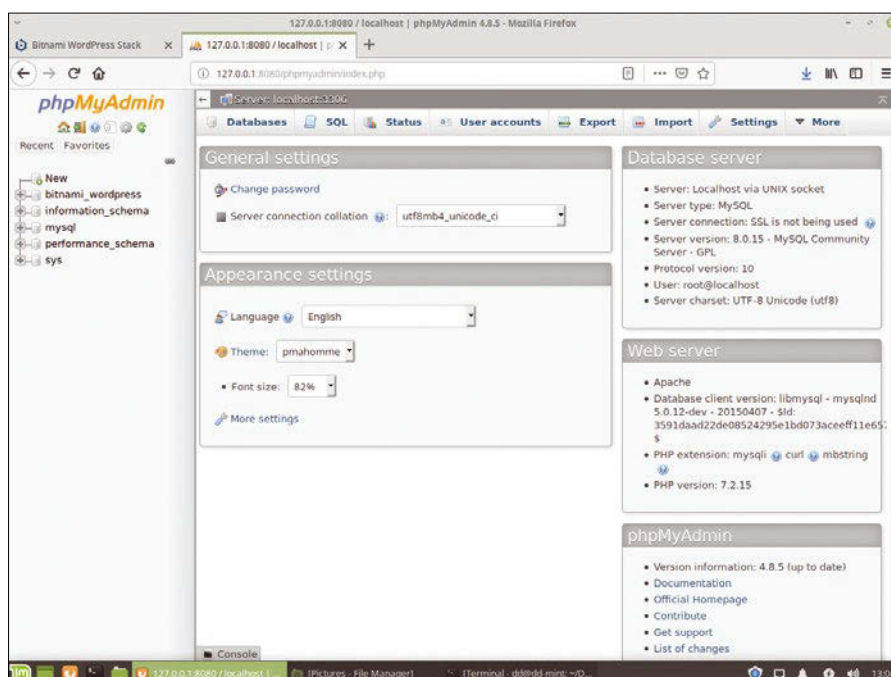
accessing the error and access logs. *Open Conf File* can also be used to open the configuration file of the service for editing.

## Administration

Manual administration work for the basic LAMP system database is usually done in a terminal. Bitnami integrates the graphical web front-end phpMyAdmin [4] into its stacks for convenient administration of MySQL databases. If you do not want this, you can deselect the option in the installation dialog.

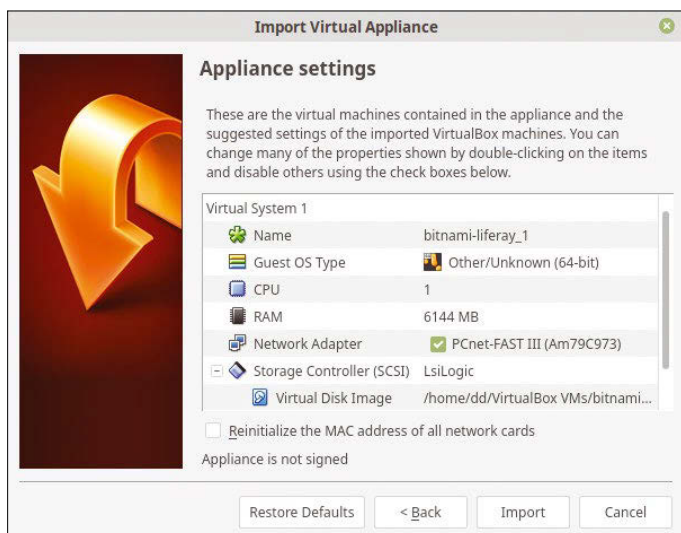
To start the interface, click the *Open phpMyAdmin* button in the Bitnami window. You then authenticate and set the language for the interface (if needed) when you call it up for the first time. The configuration interface then opens in the web browser (Figure 3). You can create, edit, or delete tables; create, manage, or delete databases; or display data records. PhpMyAdmin also comes with a user administration tool.

Extensive documentation can be found on the phpMyAdmin project website [5]. In addition, there is a manual in PDF format [6], which reflects the current development status of the software, online.



**Figure 3:** PhpMyAdmin simplifies the configuration of the MySQL database, especially for inexperienced users.





**Figure 4:** The VM apps deliver the project ready-to-use; you only have to load them.

## Manual

Since the individual Bitnami stacks do not create any start entries in the common user interfaces' menus, it makes sense to manually create a corresponding launcher, which you can use to conveniently switch a service on and off at the push of a button.

You start the software manually in the stack's directory with a click on `manager-linux-x64.run`; this opens the Bitnami window. You can make this application permanently available by creating a suitable menu item.

If you want to operate the stack without using the graphical front end, enter the `./ctlscript.sh <Parameter>` command in the stack directory at the prompt. As a parameter, you can pass in either `start`, `stop`, or `restart` to do precisely what it says on the box to all the services. The status parameter displays the current

will stop all of the stack's active services and uninstall the stack completely. The routine shows the progress with a progress indicator.

## VMs

Bitnami also provides numerous servers as preconfigured VMs in OVA and VMDK format that can be started as virtual appliances in VirtualBox, VMware, and KVM. VMs like this are particularly suitable for the simultaneous use of several services on a system that offers sufficient performance.

As a further advantage, each appliance runs in an isolated environment, which makes operations far more secure. Compared to the stacks, the VMs are far larger: File sizes over 1GB are not uncommon. This is due to the fact that the virtual appliances always come with a completely preconfigured, slimmed-

down operating system.

After downloading the appliance from the project page, integrate it into VirtualBox by selecting *File | Import appliance* in the menu. The import then takes place in a multiple-level dialog; the setup also shows you the preset resources.

status of the individual components.

## Uninstall

To uninstall a stack, you can use the `uninstall` routine in the respective directory. Clicking on it starts a graphical front end, which first asks you if you really want to remove the stack. As soon as you confirm this, Bitnami

If necessary, you can modify the assignment, just like for a VM created manually, at any time in the *Change* dialog (Figure 4).

Then start the appliance in VirtualBox like any manually installed system. The Bitnami machine first boots a Debian 9 system and then enables the preconfigured services. Then, the VM displays credentials for accessing the server services through a web browser, including the IP address, username and password, and the credentials for the console on the VM (Figure 5).

You can then log on to the server-based services with these credentials and work with them like on a dedicated server. You can make changes to the configuration of the machine or uninstall it using the VirtualBox dialogs.

## Conclusions

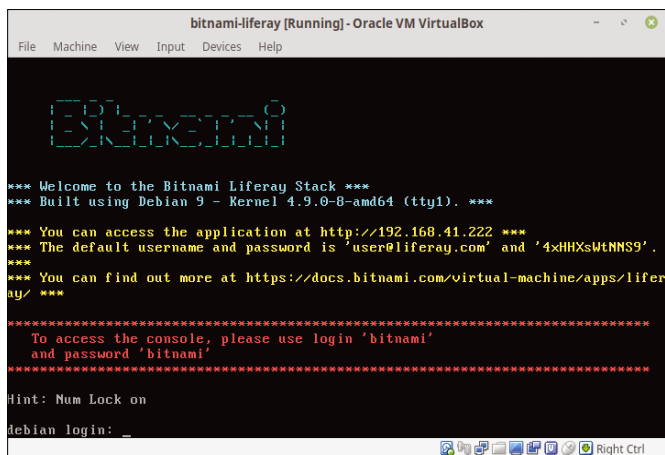
Thanks to Bitnami, complex server-based services can also be set up quickly and easily at home or in a small enterprise. Instead of reading extensive documentation, downloading countless packages, and installing and configuring them individually, you only need a single stack courtesy of the Bitnami project. The basis is either a Linux system or a virtual environment such as VirtualBox or VMware. Obsolete libraries or missing dependencies are a thing of the past. You can install the stacks with just a few mouse clicks – in our lab, they always worked reliably. And if anything is unclear, the detailed documentation will help you. ■■■

## Info

- [1] Bitnami: <https://bitnami.com>
- [2] BitRock: <https://BitRock.com/>
- [3] Bitnami Application Catalog: <https://bitnami.com/stacks>
- [4] phpMyAdmin: <https://www.phpmyadmin.net/>
- [5] phpMyAdmin documentation: <https://docs.phpmyadmin.net/en/latest/>
- [6] PDF manual: <https://media.readthedocs.org/pdf/phpmyadmin/latest/phpmyadmin.pdf>

## Author

**Erik Bärwaldt** is a self-employed IT-admin and technical author living in Scarborough (United Kingdom). He writes for several IT-magazines.



**Figure 5:** Bitnami also automatically configures the access credentials for the apps.

## The sys admin's daily grind: Log File Navigator

## Ghost Tracker

During a long trek through the verbose syslog, really important warnings and errors are scattered along the path. Sometimes a missing message can be the decisive event. Sys admin columnist Charly has now hired a tracker to help him search for clues: Log File Navigator.

By Charly Kühnast

Searching in logfiles is the sys admin's bread and butter. Finding a specific piece of information often requires long cascades of `grep` commands. What makes this even more difficult is if a log message that I expect every five minutes is delayed. Of course, this is a warning signal, but I can't use `grep` to figure this out. What can draw my attention to the fact that warning messages are piling up? These difficulties prompted me to onboard Log File Navigator (`Inav`, [1]).

If you launch `Inav` without any options, it opens `/var/log/syslog` (Figure 1). Using:

```
Inav /var/log/syslog*
```

makes more sense, because it then includes older syslog files – whether compressed or not. `Inav` bears the name “Navigator,” because it makes it easy to walk through the logfiles in small steps or giant leaps. For example, `Shift + D` beams you back 24 hours into the past, and pressing `D` without `Shift` takes you back to the present. `Shift + 1` lets you jump back to 10 minutes after the last full hour, while `Shift + 2` jumps back to 20 minutes after the last full hour, and so on. `Shift + G` always takes you to the end of the log.

Searching is easy, too. You simply type / followed by a search term. Besides strings, `Inav` also accepts regular expressions, which makes complex and fuzzy searches possible. `N` and `Shift + N` let you jump between the hits. A search function using SQL syntax is currently still experimental.

`W` and `Shift + W` jump to the next/previous warning, while `E` and `Shift + E` jump to errors. Great stuff: `S` and `Shift + S` navigate to events that are out of sync – such as delayed events.

`Inav` keeps statistics in the background. The History view (Figure 2) proves to be

practical. It displays a graph showing the number of messages received and the proportion of warnings and errors. In the screenshot, the entries are totaled in 10-minute blocks. `Z` and `Shift + Z` let you zoom in and out of the time periods.

Once you have familiarized yourself with the keyboard shortcuts, working with `Inav` will be as easy as pie for you. I only mentioned what are the most important shortcuts for me here; the complete list is available under “Hotkey Reference” on [2]. If I could wish for something in a future version, it would be more color schemes. I like to work with dark screens, but some color-highlighted areas in the log are not easy to read. ■■■

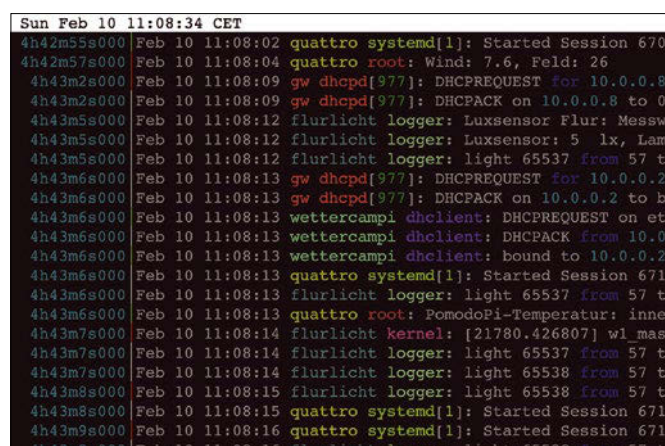


Figure 1: `Inav` not only displays logs, but also waits for keystrokes.

## Info

- [1] `Inav`: <https://Inav.org>
- [2] `Inav` documentation: <https://Inav.readthedocs.io/en/latest/>

## 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.



Figure 2: History view: `Inav` records the messages as totals – of 10 minutes duration in this case.





Custom file monitoring

# A Tale of tails

When it comes to file monitoring, **tail**'s replacements, **colortail** and **MultiTail**, offer more sophisticated control over how your information is displayed. *By Bruce Byfield*

**P**agers are a basic necessity for administering a system. That necessity is especially strong in Linux, where configuration settings are stored in text files. The best-known pagers, of course, are **cat**, **less**, and more, all of which present the entire contents of a file. However, if you want to monitor a file over time, such as the logs in **/var/log** or **/tmp**, the required tool is one that displays the lines at the end of the file, where new information is appended. The original tool for this purpose is **tail**, but, these days, it is increasingly being replaced by **colortail**

## 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>

or **MultiTail**. All these commands share the same basic functionality, but the replacements offer more control over how information is displayed (Figure 1).

## tail and colortail

The original command, of course, is **tail** [1] (not to be confused with Tails Linux,

which is used for secure browsing). By default, **tail** displays the last 10 lines of a file that it is monitoring. However, you can use

```
--lines=NUMBER (-n NUMBER)
```

to change the number of lines to display. Alternately, you can use

```
bb@nanday:/var/log$ ls
alternatives.log      dswmlog.log.4.gz    kern.log.3.gz        mssmlog.log.4.gz
alternatives.log.1    debug               kern.log.4.gz        mssmlog.log.5.gz
alternatives.log.10.gz debug.1             lastlog              mssmlog.log.6.gz
alternatives.log.11.gz debug.2.gz          lightdm              mssmlog.log.7.gz
alternatives.log.12.gz debug.3.gz          lpr.log              prey.log
alternatives.log.2.gz debug.4.gz          lpr.log.1            sddm.log
alternatives.log.3.gz dmesg               lpr.log.2.gz         speech-dispatcher
alternatives.log.4.gz dpkg.log             lpr.log.3.gz         syslog
alternatives.log.5.gz dpkg.log.1          lpr.log.4.gz         syslog.1
alternatives.log.6.gz dpkg.log.10.gz      lynis.log             syslog.2.gz
alternatives.log.7.gz dpkg.log.11.gz      lynis-report.dat     syslog.3.gz
alternatives.log.8.gz dpkg.log.12.gz      mail.info             syslog.4.gz
alternatives.log.9.gz dpkg.log.13.gz      mail.info.1          syslog.5.gz
apache2               dpkg.log.14.gz      mail.log              syslog.6.gz
apt                   dpkg.log.15.gz      mail.log.1            syslog.7.gz
aptitude              dpkg.log.16.gz      messages              trim.log
aptitude.1.gz         dpkg.log.17.gz      messages.1            unattended-upgrades
auth.log               dpkg.log.18.gz      messages.2.gz         user.log
auth.log.1            dpkg.log.19.gz      messages.3.gz         user.log.1
auth.log.2.gz          dpkg.log.20.gz      messages.4.gz         user.log.2.gz
auth.log.3.gz          dpkg.log.21.gz      minidlna.log          user.log.3.gz
auth.log.4.gz          exim4               minidlna.log.1        user.log.4.gz
```

**Figure 1:** The **tail** family is especially useful for monitoring the system logs found in **/var/log**.

Lead Image photo by Andre Mouton on Unsplash



```

root@nanday:~# colortail /var/log/syslog
==> /var/log/syslog <==
Mar 28 12:57:09 nanday gfsd[1982]: mkdir failed on directory /var/cache/samba: Permission denied.
Mar 28 12:57:09 nanday gfsd[1982]: mkdir failed on directory /var/cache/samba: Permission denied.
Mar 28 12:58:00 nanday gfsd[1982]: dbus_mount_reply: Error from org.gtk.vfs.Mountable::mount(): Failed to retrieve share list from server: Connection refused
Mar 28 12:58:00 nanday gfsd[1982]: Couldn't create directory monitor on smb:/x-gnome-default-workgroup/. Error: The specified location is not mounted
Mar 28 13:00:01 nanday CMD[3431]: (read) CMD (/usr/lib/prey/prey.sh >/var/log/prey.log)
Mar 28 13:00:01 nanday CMD[3431]: (www-data) CMD ( if test -x /usr/share/drupal7/scripts/cron.sh ; then /usr/share/drupal7/scripts/cron.sh ; fi)
Mar 28 13:01:04 nanday systemd[1]: Started Run anacron jobs.
Mar 28 13:01:04 nanday anacron[5401]: Anacron 2.3 started on 2019-03-20
Mar 28 13:01:04 nanday anacron[5401]: Normal exit (0 jobs run)
Mar 28 13:01:04 nanday systemd[1]: anacron.timer: Adding 2min 49.728976s random time.

```

Figure 2: colortail color codes by column in the output.

### Listing 1: colortail Color Code Map

```

1 = magenta
2 = cyan
3 = green
4 = yellow
5 = brightblue
6 = brightred

```

```
--bytes=NUMBER (-c NUMBER)
```

to set the number of kilobytes to display. However, because you are dealing with text files, fine-tuning can be difficult. In addition, rather than counting from the end of the file, you can set where to start the display with `c +KILOBYTES`.

Other options for `tail` are few. With `--max-unchanged-stats=REITERATIONS`, you can stop following a file if it remains unchanged after the defined number of updates. You can also use

```
--follow=NAME / DESCRIPTOR (-f)
```

to save output to a file.

This is a very limited set of options, which is probably why the latest release of Debian does not even bother to include the original command. Instead, it includes `colortail` (Figure 2). Like the original `tail`, `colortail` [2] allows users to set the number of lines to display, although it dispenses with setting the display in kilobytes. It also includes the `-f` option, which immediately displays any change in the hardware that a logfile is monitoring.

However, `colortail`'s main advantage is that it color codes each column in the display with the option:

```
--config=CONFIG-FILE (-k CONFIG-FILE)
```

tion for the date, followed by the day, and then the time. In this case,

```

Mar  8 8:41
[-1--][--2--]

```

this map would color the month magenta, and the day and time cyan.

## MultiTail

Of all the `tail` commands, `MultiTail` [3] is by far the most full-featured. Like `tail`, it lets you view multiple files, but, unlike `tail`, it uses ncurses to create sub-windows in the terminal window. In addition, it monitors wildcards intelligently, using the most recently modified match by default, which helps to monitor a directory of files. The command can also update the status line to show the arrival of new information, and, like `colortail`, can be color-coded. The command can either display specific files, or, with the use of regular expressions, files in a specific directory (Figure 3).

```

00.2 Recommended version: 0.101.1
Feb 26 12:04:50 nanday freshclam 673 : Tue Feb 26 12:04:50 2019 -> DON'T PANIC! Read
https://www.clamav.net/documents/upgrading-clamav
Feb 26 12:04:50 nanday freshclam 673 : Tue Feb 26 12:04:50 2019 -> main.cvd is up to
date (version: 58, sigs: 4566249, f-level: 60, builder: sigmgr)
Feb 26 12:04:50 nanday freshclam 673 : Tue Feb 26 12:04:50 2019 -> daily.cld is up to
date (version: 25372, sigs: 2261684, f-level: 63, builder: raynman)
Feb 26 12:04:50 nanday freshclam 673 : Tue Feb 26 12:04:50 2019 -> bytecode.cld is up
to date (version: 328, sigs: 94, f-level: 63, builder: neo)
Feb 26 12:09:24 nanday systemd 1 : Starting Clean php session files...
Feb 26 12:09:24 nanday systemd 1 : Started Clean php session files.
00] /var/log/daemon.log 80KB - 2019/02/26 12:25:5
2019-02-25 16:50:30 status unpacked multitail:amd64 6.4.2-1+b1
2019-02-25 16:50:30 status unpacked multitail:amd64 6.4.2-1+b1
2019-02-25 16:50:30 startup packages configure
2019-02-25 16:50:30 configure multitail:amd64 6.4.2-1+b1 <none>
2019-02-25 16:50:30 status unpacked multitail:amd64 6.4.2-1+b1
2019-02-25 16:50:30 status unpacked multitail:amd64 6.4.2-1+b1
2019-02-25 16:50:30 status half-configured multitail:amd64 6.4.2-1+b1
2019-02-25 16:50:30 status installed multitail:amd64 6.4.2-1+b1
2019-02-25 16:50:30 trigproc man-db:amd64 2.7.6.1-2 <none>
2019-02-25 16:50:30 status half-configured man-db:amd64 2.7.6.1-2
2019-02-25 16:50:31 status installed man-db:amd64 2.7.6.1-2
01] /var/log/dpkg.log 23KB - 2019/02/26 12:25:5

```

Figure 3: MultiTail is the most fully-featured member of the `tail` family of commands.

```

Use cursor UP/DOWN to scroll, ctrl+g to exit
MultiTail lets you create one or more windows in
one terminal. In each window one can monitor one
or more logfiles and(!)/or the output of external
programs: when using multiple inputs, they get
automatically merged.

In the main menu one can do add/delete windows,
move them, swap them, set regular expressions, set
color(-schemes) etc.

One can press the following keys:
q      quit the program
F1     this help
/      search in all windows
shift + / highlight in all windows

```

Figure 4: Press F1 for a list of MultiTail keyboard shortcuts.



**Table 1: Selected Keyboard Shortcuts for MultiTail**

Shortcut	Description
q	Quit
/	Search in all windows
Shift + /	Search in all windows and highlight results
b	Scroll back
B	Scroll back in all windows when merged into a single window
e	Enter a regular expression
l	Toggle case sensitivity in a search
a	Add a new file in a new window
d	Delete a file and its window from a display
c	Set or change colors
C	Edit RGB definition for a color
b	Scroll back in window buffer
v	Toggle vertical arrangement of windows
0-9	Add bookmark to window
R	Reset a window
y	Set line wrap
o	Clear a window
O	Clear all windows
g	Take screenshot
l	List keybindings
j	Set window sizes
z	Hide window
U	Unhide all hidden windows
P	Toggle pause in window

Since MultiTail runs from the command line, it uses keyboard shortcuts to navigate the display. These shortcuts can select the active window, scroll and search, and change the display. The controls are not those used by most programs – for example, the arrow keys have no effect whatsoever within a file, although they do function in pop-up windows. However, a list of the available shortcuts is available when you press F1 (Table 1). Other pop-up windows, such as lists of files and windows, display when you need to make a choice (Figure 4).

Other behaviors in MultiTail are set using command options. The default number of lines displayed depends on the window's size, but can be set precisely with `-n <number of lines>`. Similarly, rather than have the same line repeated, you can set MultiTail to print the

number of times that a message is repeated using `--no repeat`, or indicate the lack of new messages with an "x" by adding `--mark-interval x`. Alternatively, using `--closeidle <number of seconds>`, you can set a window to close if no information is given in the time specified. Should you want to stop and restart the display of a file, you can use `-r <number of seconds>` to set the interval before restarting. If you do restart, `-R <number of seconds>` will show the difference between the current reading and the previous one.

Still other commands modify the display once it is running. For example, `-q <number of seconds> "<path to files>"` allows new files to be added to a running display in separate windows, while `-Q <number of seconds> "<path to files>"` displays them in a single window. These options are especially well-suited for watching a directory of files and for making full use of regular expressions. However, note that the path must be placed in quotation marks, so that the shell does not try to parse it.

Output from MultiTail can also be directed by options. The option `-a FILE` saves the output. The file can be further defined by `-5`, which prepends the file name with the window's sub-number. You can also send output directly to a command using `-g COMMAND` for further editing or, perhaps, to take advantage of advanced search and replace tools like those in an advanced text editor like Bluefish or Kate.

Throughout MultiTail, standard regular expressions can be used. However, regular expressions are especially useful for scanning a directory of files for specific content. When a regular expression follows `-e`, you can search for a specific string in the files listed in MultiTail. With `-ex`, you can search for a command mentioned in a file and then execute it. More simply, you can use `-ec` to generate a list of matches on the regular expression. Such commands can make analyzing the output of files much easier.

## Customizing the Display

Many users may be content to use MultiTail's default display parameters.

```
# Format of this file:
#
# include:configfile
#       Also pars 'configfile'.
#
# defaultcscheme:<name of colorscheme>
#       Selects the default color scheme to use. If this one is set, you
#       no longer need -cS/-CS.
#
# colorscheme:<name of colorscheme>
#       This name can be given for the commandline-parameter -cS. That
#       way, one can select what colorscheme to use for the next
#       logfile.
#
# cs_re:<color>:<regular expression>
#       This defines a regular expression to find a particular string.
#
#       color: [fg],[bg],[attribute[/otherattribute]][/other colorpair+attribute
#       2, etc.
#       e.g.: red,,bold|red would give bold red for line 1 and just red for line
#       Possible colors: red, green, yellow, blue, magenta, cyan and white.
#
# /etc/multitail.conf
```

**Figure 5: MultiTail's configuration file is the easiest place to customize columns, colors, and other aspects of the display.**

However, you can choose to customize the display as part of the command. Using `-s NUMBER`, you can set the number of default columns. More specifically, you can follow `-sw` with a comma-separated list that specifies the number of pixels in each column – for instance:

```
-sw 20, 30, 10, 10
```

The number of vertical rows can be set in the same way with the option `-sn`. More generally, MultiTail's window height can be set with `-wh NUMBER`; although if you specify a height greater than your screen, the height will be automatically adjusted.

Specific parts of the window can also be customized. Some users might want to add `-ts` to give each line a time stamp, configuring the format in `/etc/multitail.conf` (Figure 5). Other users might add `-D` so as not to display the status line, or `-du` to position the status line at the top of the window.

Numerous options for coloring different aspects of MultiTail and the contents of

files can be set at the command line. However, since many color options depend on color schemes defined in `/etc/multitail.conf`, you might prefer to edit that file directly instead. If nothing else, the comments in the file give detailed instructions about how to write a color scheme and the available colors and fields. Very likely, the only time you might want to override `multitail.conf` from the command line is when you want to suppress the use of color altogether, either in a single file (`-c`) or in a list of files (`-C`).

## The Evolution of a Command

From a simple beginning, the `tail` family of commands has evolved steadily towards greater sophistication. The ability to customize colors alone can make `colortail` more efficient than the original `tail` command. MultiTail takes that evolution several steps further, allowing admins not only complete control over how files are displayed, but also enhanced monitoring of multiple files.

So which should you use? The answer is both a matter of taste and

complexity. For simple monitoring, especially at home, `colortail` should be more than enough to meet your needs. Although it lacks most of the features of MultiTail, often `colortail` can get the job done.

By contrast, MultiTail is likely to cope best with networks or the need to monitor several files at a time. However, its sophistication comes at the price of greater complexity. The fact that its options and keyboard commands can be eccentric only makes it more difficult to learn.

Personally, I install both on my machines. That way, I can use `colortail` for simple file paging and MultiTail for more advanced work. In other words, I let the task decide. ■■■

## Info

- [1] `tail`: <https://linux.die.net/man/1/tail>
- [2] `colortail`: <http://manpages.ubuntu.com/manpages/bionic/man1/colortail.1.html>
- [3] MultiTail: <https://linux.die.net/man/1/multitail>

# FIND THE RIGHT TOOL!

## Check out our new Linux Administration corner!

<http://www.linux-magazine.com/administration>

SPONSORED BY



Linux  
Professional  
Institute







Remote access to Wayland desktops under Fedora 29

# Bird's Eye View

In Fedora 29, you can enable a VNC server on Wayland with a few mouse clicks, thus enabling remote desktop access. *By Tim Schürmann*

**O**n Linux, the X Window System (short X11) still draws the graphical user interface on the screen in most cases. Thanks to its integrated network functions, it can even transport program windows from remote computers onto the screen if required. This facilitates remote maintenance and simplifies thin client setup. While the computationally intensive application runs on a powerful computer, the user looks at the output on their local PC, which requires very little in terms of resources.

In Wayland, the newly-developed X11 successor, these neat network functions are missing in the plain vanilla version. Although Wayland is leaner and significantly more secure than X11, it lacks some proven functions. If you want to share your desktop on Wayland, you need separate remote desktop software.

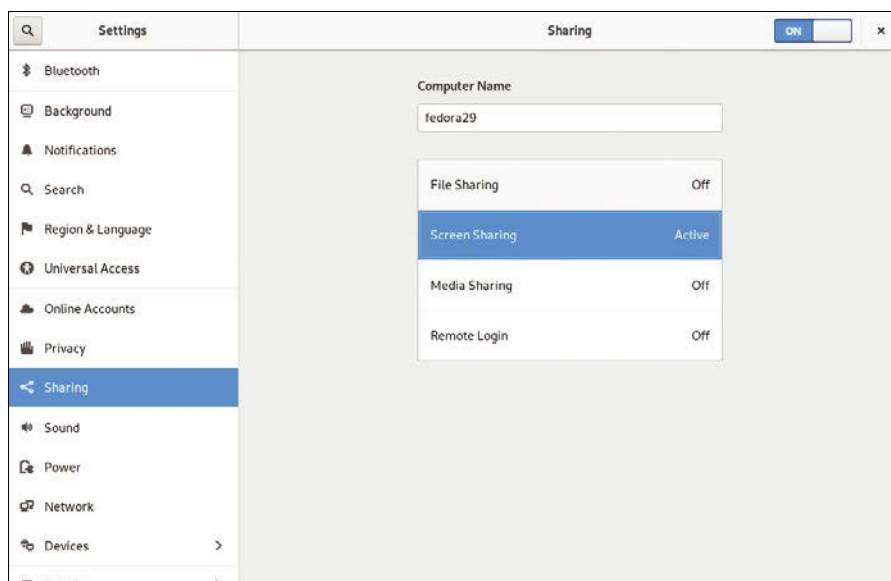
## Pimped Up

The lack of remote access in Wayland was a thorn in the Fedora team's side. While they used proven components and protocols to restore these missing net-

work capabilities, they had to adapt or slightly modify existing software to meet their goal [1].

The idea was for communication to use the established Virtual Network Computing (VNC) protocol, with encrypted data

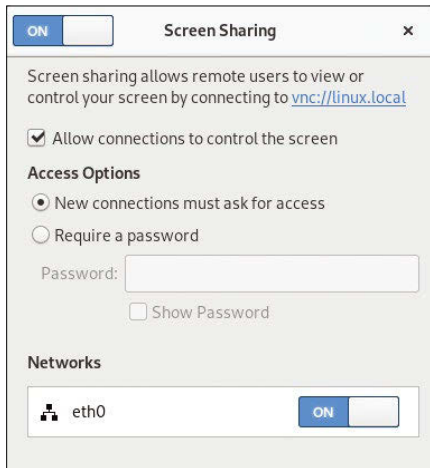
on the wire and access to the remote desktop only possible for legitimate users. VNC relies on the client-server principle: A server program sends the content to be transferred to a client program, which in turn displays it on the screen.



**Figure 1:** The Gnome Remote Desktop is hidden under *Screen Sharing*. With the settings shown here, clients can later access the desktop on the computer named *Fedora 29*.

Lead Image © M Tomczak, fotolia.com





**Figure 2:** With the settings shown here, you have to explicitly confirm every client's connection request.

The LibVNCServer library provides help in creating a VNC server. The Fedora developers first extended this with the functions required for encryption. Using LibVNCServer, the Gnome Remote Desktop implements a VNC server. The developers added the possibility to authenticate users with a password.

They used the relatively young PipeWire for data transfer and data format negotiation [2]. PipeWire's long-term goal is to transport audio and video data to the required locations as quickly as possible.

For the remote desktop function under Fedora, the developers had to ad-

just the systemd configuration accordingly. The init system can now enable PipeWire via a socket. Finally, the Fedora developers added the matching options in Gnome's System Preferences that allow users to enable or disable Gnome Remote Desktop.

## Turn It On

The result of all this work can be found for the first time in Fedora 29 Workstation. The distribution automatically installs the *gnome-remote-desktop* package and all other components necessary for sharing the desktop. If you are using Fedora 29 with the Wayland session enabled by default, it is not difficult to display the desktop on other computers. In fact, it takes just a few clicks.

To do this, under Settings, select *Sharing*, and toggle the *ON* button in the titlebar, as shown in Figure 1. Then click *Screen Sharing* and enable this with the corresponding *ON* button in the dialog's titlebar (Figure 2).

Now you have a VNC server running on your system, which allows other computers to display your desktop on their local system. If you want to let the users work actively with the desktop environment, check the *Allow connections to control the screen* box.

You can expect some of your desktop's would-be users to be up to no good. To prevent this, the software requires you to explicitly grant each request with a

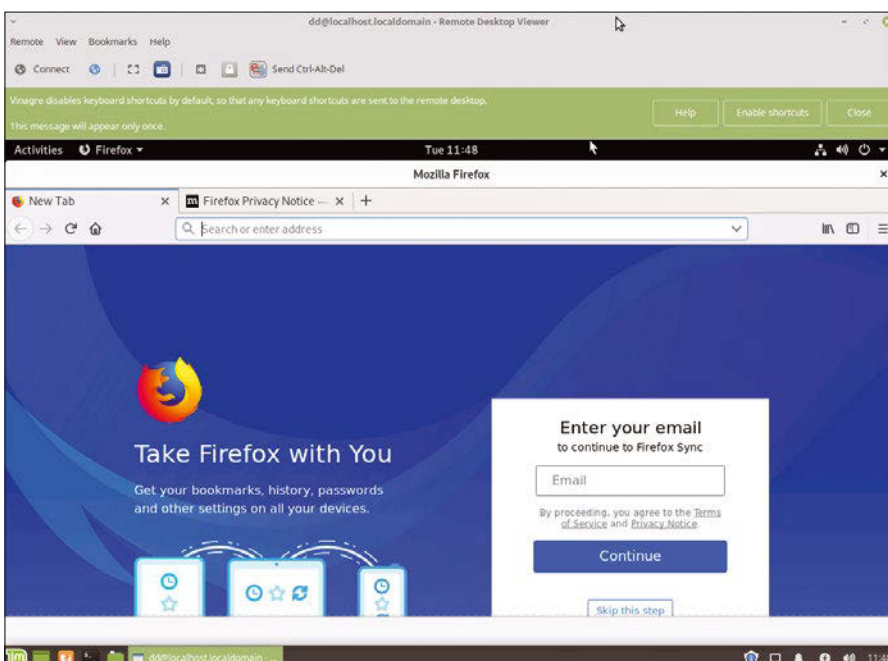
mouse click. Alternatively, you can give other people password-based access. To do this, enable the *Require a password* option and enter a strong password in the input field. Anyone with the password will be able to view the desktop environment from their computer in the future.

## Clientele

Next, you need to set up a VNC client for the computers that you want to access your desktop. You can do this on Fedora via *Software* by installing either Remmina or the Remote Desktop Viewer (aka Vinagre).

Start the client program and connect it to the VNC server – and thus to the machine on which you enabled sharing. In the case of Vinagre, click *Connect*, set *Protocol* to VNC, and type the VNC server's IP address or hostname in the *Computer* field. After you click *Connect*, Vinagre establishes contact with the remote computer (Figure 3).

If you select Remmina as the remote desktop client, set the combobox at the top left of the input field to VNC. Then enter the VNC server's IP address or hostname and press Enter (Figure 4).



**Figure 3:** Connecting to a remote computer with Vinagre.

## 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.

ADMIN Magazine



Linux Magazine



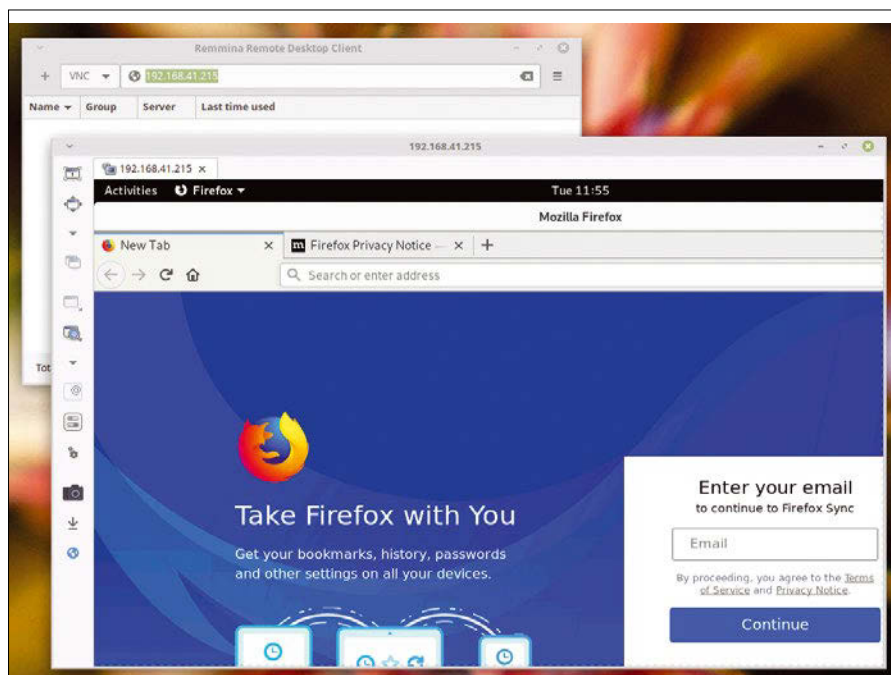
Visit our apps page for more information  
[shop.linuxnewmedia.com/us/apps](http://shop.linuxnewmedia.com/us/apps)



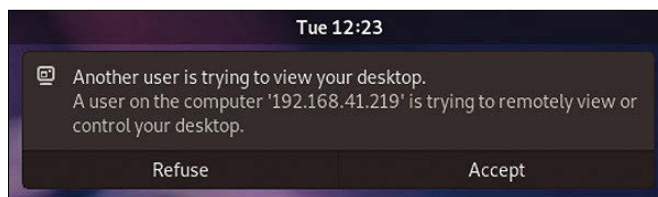
# GOT CLUSTER?

Tune in to the HPC Update newsletter for news, views, and real-world technical articles on high-performance computing.

<https://bit.ly/HPC-ADMIN-Update>



**Figure 4:** Enter the IP address in the Remmina Remote Desktop Client dialog box.



**Figure 5:** By pressing *Accept*, a waiting client is permitted to view the desktop.

If you previously set a password, you must tell it to the client. Otherwise, Fedora will ask you on the VNC server if you want to share the desktop with the corresponding client (Figure 5). You can

confirm by pressing *Accept*.

In any case, an orange icon in the upper right corner (Figure 6) indicates that at least one other person can see the desk-

top. The client also presents the VNC server's desktop live.

If you disconnect from the client program, the VNC server may report an error (*Oops! ...*). You can ignore this error: In tests, the release then continued to function without any problems.

## Conclusions

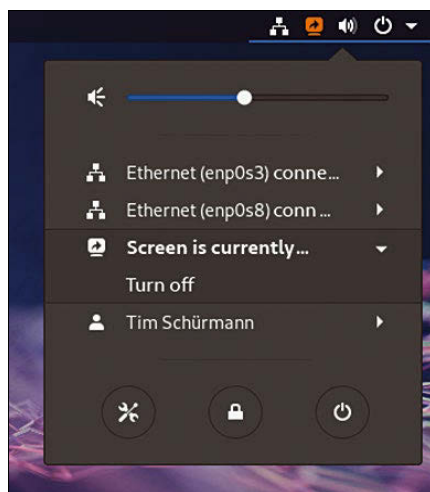
Contrary to some web news, Fedora developers have simply provided their distribution with an easy-to-enable VNC server. Wayland itself still does not offer any functions for transparent access via the network. The corresponding functions are limited to the Gnome desktop for now, but, at the end of the day, remote access is possible with any VNC client, even Windows or Mac OS X. ■■■

## Info

[1] Wayland remote desktop:

<https://fedoraproject.org/wiki/Changes/WaylandRemoteDesktop>

[2] PipeWire: <https://pipewire.org>



**Figure 6:** You can disconnect from the client program at any time via the System menu, which will terminate the client's access to your desktop.





**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](http://shop.linuxnewmedia.com/subs)

**GET IT NOW!**

**FAST DELIVERY WITH OUR PDF EDITION**



Screen scraping  
with Colly in Go

# Data Scraper

The Colly scraper helps developers who work with the Go programming language to collect data off the web. Mike Schilli illustrates the capabilities of this powerful tool with a few practical examples. *By Mike Schilli*

As long as there are websites to view for the masses of browser customers on the web, there will also be individuals on the consumer side who want the data in a different format and write scraper scripts to automatically extract the data to fit their needs.

Many sites do not like the idea of users scraping their data. Check the website's terms of service for more information, and be aware of the copyright laws for your jurisdiction. In general, as long as the scrapers do not republish or commercially exploit the data, or bombard the website too overtly with their requests, nobody is likely to get too upset about it.

Different languages offer different tools for this. Perl aficionados will probably appreciate the qualities of `WWW::Mechanize` as a scraping tool, while Python fans might prefer the `selenium` package [1]. In Go, there are several

**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](mailto:mschilli@perlmeister.com) he will gladly answer any questions.



projects dedicated to scraping that attempt to woo developers.

One of the newer ones is Colly (possibly from “collect”). As usual in Go, it can be easily compiled and installed directly from its GitHub repository like this:

```
go get github.com/gocolly/colly
```

After installation, a program like Listing 1 [2], for example, can access the website of the CNN news channel, dig through the links hidden in its HTML, and display them for testing purposes.

**Goodbye,  
Dependency Hell**

Customary in Go, `go build linkfind.go` creates a binary named `linkfind`, which weighs in at a hefty 14MB, but already contains all the dependent libraries and runs standalone on similar architectures without further ado. What a concept! Go shifting today's typical “Dependency Hell” from run time to compile time, and thus recruiting developers to do the heavy lifting instead of the end user, is probably one of the greatest ideas of recent times.

Listing 1 uses `NewCollector()` to initialize a new `colly` structure in Line 9; its `Visit()` function later connects to the CNN website's URL and kicks off the `OnHTML()` callback as soon as the page's HTML has arrived. As its first argument, the “`a[href]`” selector calls the subsequent

`func()` code only for links in the format `<AHREF=...>`. The Colly library ensures that each call receives a pointer to a structure of the `colly.HTML` type, containing all relevant data of the matching HTML structure, from which line 13 extracts the link URL as a string via `e.Attr("href")`.

Moving on, how difficult would it be to determine which links branch to ex-

**Listing 1: linkfind.go**

```
01 package main
02
03 import (
04     "fmt"
05     "github.com/gocolly/colly"
06 )
07
08 func main() {
09     c := colly.NewCollector()
10
11     c.OnHTML("a[href]",
12         func(e *colly.HTML) {
13             fmt.Println(e.Attr("href"))
14         })
15
16     c.Visit("https://cnn.com")
17 }
```

Lead Image © Hannu Viitanen, 123RF.com

## Listing 2: linkstats.go

```

01 package main
02
03 import (
04     "fmt"
05     "github.com/gocolly/colly"
06     "net/url"
07     "os"
08 )
09
10 type Stats struct {
11     external int
12     internal int
13 }
14
15 func main() {
16     c := colly.NewCollector()
17     baseURL := os.Args[1]
18
19     stats := Stats{}
20
21     c.OnHTML("a[href]",
22         func(e *colly.HTML_Element) {
23             link := e.Attr("href")
24             if linkIsExternal(link, baseURL) {
25                 stats.external++
26             } else {
27                 stats.internal++
28             }
29         })
30
31     c.Visit(baseURL)
32
33     fmt.Printf("%s has %d internal "+
34         "and %d external links.\n", baseURL,
35         stats.internal, stats.external)
36 }
37
38 func linkIsExternal(link string,
39     base string) bool {
40     u, err := url.Parse(link)
41     if err != nil {
42         panic(err)
43     }
44     ubase, _ := url.Parse(base)
45
46     if u.Scheme == "" ||
47        ubase.Host == u.Host {
48         return false
49     }
50     return true
51 }

```

ternal websites and which reference the site internally? Listing 2 is based on the same basic structure, but also defines a counter structure, `Stats`, and no longer hardwires the URL to be examined in the code, but accepts it as a parameter on the command line.

To distinguish external links from internal ones, the `linkIsExternal()` function uses `Parse()` from the `net/url` package to split the link URL and the original base URL passed into the function into their component parts. It then checks via `Scheme()` if the link is missing the typical `http(s)://` protocol or if the host is identical in both URLs – in both cases, the link points to the original page, so it is internal.

## Structured by Default

Line 19 initializes an instance of the `Stats` structure defined previously in Line 10, and – as usual in Go – all members are assigned default values; in the case of the two integers, each starts out at 0. This means that lines 25 or 27 only need to increment the integer value by 1 for each link examined; at the end of the program, line 33 can then output the

number of internal and external links. For the *Linux Magazine* home page, this results in:

```

$ ./linkstats 2
https://www.linux-magazine.com
https://www.linux-magazine.com has 2
64 internal and 12 external links.

```

It's a pretty complex website! But collecting link stats does not exhaust Colly's

usefulness. Colly's documentation [3] is still somewhat sparse, but the examples published there might give creative minds some ideas for future projects.

## Let's Go Surfing

For example, I frequently visit the website *surfline.com*, which shows the wave height at selected beaches around the world. Since I love surfing (at a casual level, mind you), I've always wanted a command-line tool that quickly checks the site for my local beach (Ocean Beach in San Francisco) and discovers whether there are any monster waves preventing me from paddling out, because – as a hobby surfer – anything more than eight feet has me shaking in my wetsuit. Figure 1 shows the relevant information as it's displayed on the web page; Figure 2 illustrates where the data is hidden in the page's HTML according

to Chrome's DevTools. It is now the scraper's task to shimmy through the tags and trickle out the numerical values indicating today's surf size.

Squinting at the HTML in Figure 2, the wave height is indicated in a `span` tag of the `quiver-surf-height` class. However, this tag occurs several times in the document, because the page also displays the conditions at other neighboring surf spots. The trick now is to

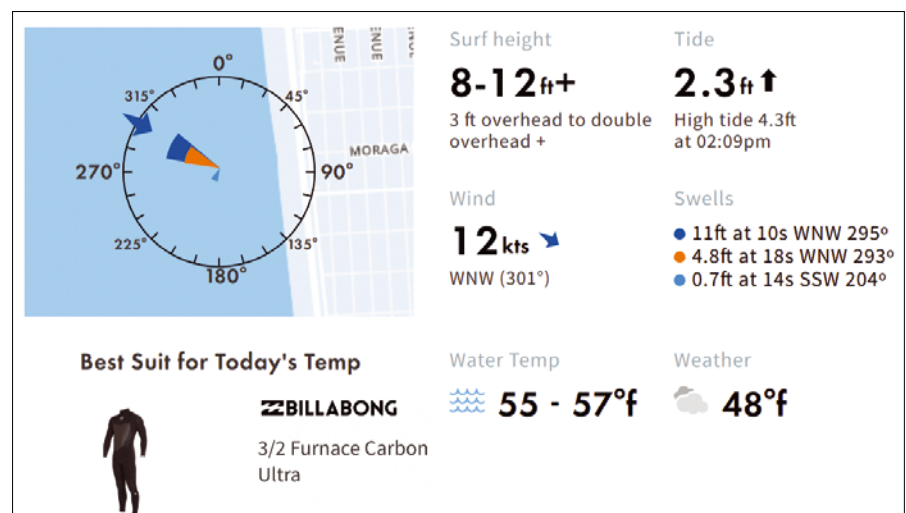


Figure 1: Current surf conditions on *surfline.com*.



find a path from the document root to the data that is unique and therefore only matches the main spot's data. As Figure 2 shows, this path is winding through an element of the `sl-spot-forecast-summary` class.

Programming this is an easy job; you just pass the two class names to the

`OnHTML()` function in line 12 of Listing 3 as space-separated strings in the first argument. The query processor digs down on this path into the document to find just one, and thus the correct, wave height for the current choice of spot.

### goquery: jQuery for Go

But oops, if you look closer, the `span` element contains not one but three lines. The first one, "8-12", shows the wave height I was searching for, and the second one contains `<sup>FT</sup>` to add feet as the unit of length next to it in the browser. The third line contains a "+",

which indicates that the waves could be a bit higher than indicated.

How can the query processor separate these three lines? I didn't find anything helpful on this topic in the Colly documentation, but luckily Colly uses the goquery language internally, which is very similar to jQuery. Starting with a structure of `HTMLElement` type found in Colly, you can quickly extract to the corresponding goquery structure via its `DOM` attribute.

The corresponding goquery documentation [4] states that the `Contents()` function splits the text from the discovered

element into its three components, and a subsequent call to `Slice(0,1)` then extracts the first bit.

As usual in Go, the index numbers for a slice always refer to the first (inclusive) and the last elements (exclusive). The following `Each()` in line 15 grabs the one and only result and calls the provided callback function with the selection at hand. The callback then uses `s.Text()` to extract the wave height in feet as a string, and we're in business.

### High Waves

The following call to the compiled binary

Listing 3: `surfline.go`

```
01 package main
02
03 import (
04     "fmt"
05     "github.com/gocolly/colly"
06     "github.com/PuerkitoBio/goquery"
07 )
08
09 func main() {
10     c := colly.NewCollector()
11
12     c.OnHTML("sl-spot-forecast-summary " +
13             ".quiver-surf-height",
14         func(e *colly.HTMLElement) {
15             e.DOM.Contents().Slice(0,1).Each(
16                 func(_ int, s *goquery.Selection) {
17                     fmt.Printf("%s\n", s.Text())
18                 })
19         })
20
21     c.Visit("https://www.surfline.com/" +
22           "surf-report/ocean-beach-overview/" +
23           "5842041f4e65fad6a77087f8")
24 }
```

```
▶<div class="quiver-content-container">...</div>
▼<div class="sl-spot-current-conditions-section sl-spot-current-conditions-section--with-cam sl-spot-current-conditions-section--with-report sl-spot-current-conditions-section--free">
  ▶<div class="quiver-content-container">...</div>
  ▼<div class="quiver-content-container">
    ▼<div class="sl-spot-module sl-spot-module--with-cam">
      ▼<div class="sl-spot-report">
        <div class="sl-colored-condition-bar sl-colored-condition-bar--poor">P00R</div>
        ▼<div class="sl-spot-report__content-wrapper">
          ▼<div class="sl-spot-report__condition-values sl-spot-report__condition-values--with-report">
            ▼<div class="sl-spot-forecast-summary">
              ▼<div class="sl-spot-forecast-summary__wrapper">
                ▼<div class="sl-spot-forecast-summary__stat">
                  ▼<div class="sl-spot-forecast-summary__stat-container sl-spot-forecast-summary__stat-container--surf-height">
                    <div class="sl-spot-forecast-summary__stat-title">
                      Surf height</div>
                    ▼<span class="quiver-surf-height">
                      "8-12"
                      <sup>FT</sup>
                      "+"
                    </span>
                    ▶<div class="sl-wave-summary">...</div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

Figure 2: The wave height is hidden somewhere in the HTML, inside a `quiver-surf-height` class tag.

```
$ ./surfline
8-12
```

thus reveals that the waves are 8 to 12 feet tall (or 2.4 to 3.6 meters if you prefer). That is a little beyond my capabilities, forcing me to leave my surfboard at home today for safety reasons and ensuring timely production of future columns. But tomorrow is another day! ■■■

### Info

- [1] "Programming Snapshot: Protectli" by Mike Schilli, *Linux Magazine*, issue 208, March 2018, pp. 46-49, [http://www.linux-magazine.com/Issues/2018/208/Servile-Guardian/\(language\)/eng-US](http://www.linux-magazine.com/Issues/2018/208/Servile-Guardian/(language)/eng-US)
- [2] Listings for this article: <ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/208/>
- [3] Colly documentation: <http://go-colly.org/docs/>
- [4] goquery documentation: <https://godoc.org/github.com/PuerkitoBio/goquery>

---

# 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](http://shop.linuxnewmedia.com)





# MakerSpace

The EOMA68 Laptop

## The Road to Production

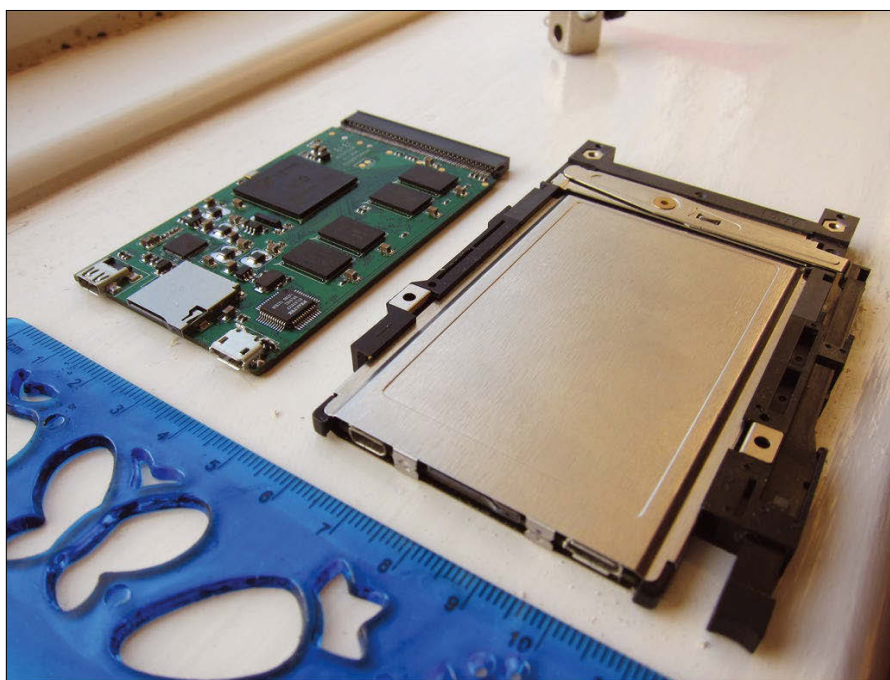
Despite challenges, hardship, and delays, the EOMA68 laptop project is set to test its first PCBs. Through this learning curve, Leighton, the project's developer, has laid the groundwork for other open source hardware pioneers. *By Bruce Byfield*

**I**n 2016, I wrote an article about Luke Leighton's [1] crowdfunding campaign to build a modular, recyclable computer (Figure 1). Three years, and dozens of updates later, the project is about to test its first printed circuit boards (PCBs), and production appears just around the corner

(Figure 2). Behind this milestone is a complicated story of changing specifications, the challenges of production in China and Taiwan, personal hardship, and delays; all of which illustrates the challenges that new manufacturers face when bringing open hardware to release.



**Figure 1:** Luke Leighton has been working towards a module computer for seven years.



**Figure 2:** Three years after a successful crowdfunding campaign, the project is testing its PCBs.

Lead Image © lightwise, 123RF.com

Leighton's EOMA68 design [2] has several distinguishing features. Designed to be a completely free laptop, it is also designed to be environmentally responsible, with a bamboo frame and a case that can be repaired with a 3D printer or even wooden parts (Figure 3). Technically, its greatest innovation is the storing of computer cards housed in recycled PCMCIA cards that can be easily swapped in and out of the machine (Figure 4). The building of EOMA68 laptops is accompanied by the establishment of an open EOMA68 standard that can be used by anyone. Parts of the project are already certified as free hardware by the Free Software Foundation.

The project's idealism has attracted loyal supporters, many of whom have volunteered their services to make both the hardware and the standard a reality. Leighton specifically singles out Chris Waid of ThinkPenguin, who sponsored the design of the 15.6-inch laptop housing. Another long-term volunteer is Richard Wilbur, who spent eight months working with high-speed differential pairs necessary for HDMI support.

Leighton himself has spent seven years on the project, first developing the standard and then trying to manufacture the laptop on a shoestring budget, often at personal expense. In the course of his efforts, Leighton has seen his already modest income halved as well as his family's eviction from their home. The experience might have been eased by seeking investors, but Leighton declares that the EOMA68 standard "cannot be allowed to be compromised by profit maximizing. By logical implication, I am absolutely prevented from approaching or accepting all and any share-allocation funding, including standard venture capital investor funding." He is, however, applying for grants from various nonprofit foundations.

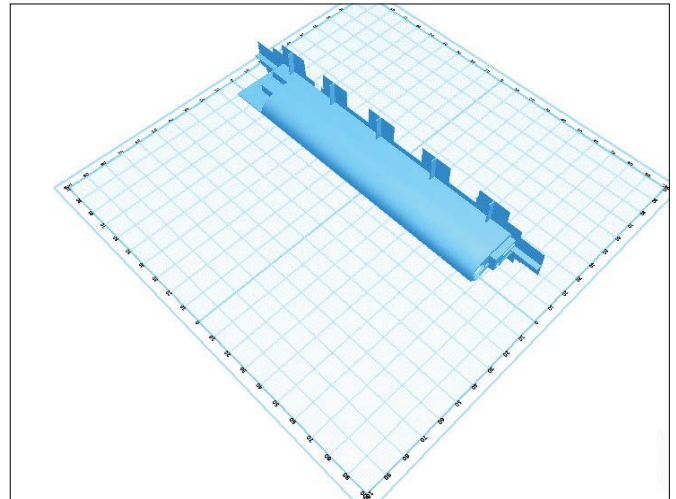
### Shifting Specifications and Components

Seven years is a long time in computing specifications. It can mean several generations in accepted standards. The EOMA68 standard has seen at least two major revisions: the replacement of SATA with a second USB port, and the removal of gigabyte ethernet with optional USB

support. According to Leighton, each of these changes cost around \$10,000, and delayed progress by six months. "However," he adds, "without them, EOMA68 would not be a viable long-term standard, so it was costly but absolutely necessary." The alternative would be a standard and product that was obsolete before either was released.

A similar problem has slowed manufacturing. The project requires very specific components, such as PCMCIA connectors and mid-mount USB-OTG and Micro HDMI Type D. "We're currently on the *third* mid-mount USB-OTG connector, and the *fourth* Micro HDMI Type D connector," Leighton says. "Each time the chosen component went to end-of-life, we had to do a complete redesign of the PCB." Each change required a thorough review of components, a process that, because of the project's limited budget, could take "3-4 months including component ordering, PCB manufacturing, assembling, [and] then testing. It's amazing how much time is spent waiting for other people."

To further complicate matters, this updating can often not be done by outsiders. "It is the most frustrating and irritating thing in the world to find no contact details for a supposedly open project, and no source or design repositories. [And] on finally making contact, the designer says, 'Oh, I will release the design files when I have finished them. I do not want anyone to criticize or perhaps steal my work.' Such a lack of trust terminates any possibility for others to help you to avoid serious basic design flaws."



**Figure 3:** The EOMA68 laptop features a modular case that can be repaired with 3D printing.



**Figure 4:** Computer cards make swapping operating systems easy.

Instead, such work has to be done in-house. For instance, after finding that outsourcing CAD and PCB layout was often time-consuming and unsatisfactory in itself, Leighton found it easier to learn this work himself and design the laptop's case as he learned.

### Manufacturing in Asia

Like many first-time open hardware manufacturers, Leighton chose to manufacture in China or Taiwan. However, as others have found before him, manufacturing in Asia can be challenging for those who live in Europe or North America.

To start with, parts that are sold in North America may not be available in Asia. Sometimes, the parts have reached their end-of-life in Asia. At other times, components have been sold to larger manufacturers. The only way to get the parts that would be common in North America would be to order them from an American source, which means a six to eight week delay, special paperwork, and





**Figure 5:** An early prototype of the laptop: Note the do-it-yourself housing for the computer card.

additional expenses, including 40 per cent import duty.

“It’s just absolutely critical to make sure that the components you use are the ones that the factory can actually get,” Leighton observes. Except, he adds, “even just finding out what’s available is a near-impossible task” – even if you are actually on site. [Sourcing materials] is truly a bazaar, and you are expected to have word-of-mouth contacts. The bottom line is that if you want to design a product that is to be manufactured in volume, make absolutely sure that you scheduled at least 6-12 months for component sourcing. No, that’s not a joke or a misprint.”

Leighton goes on to advise: “Take the time to establish good relationships with Chinese sourcing agents – respect and pay them adequately. If you happen to have friends there, cherish them. Alternatively, if you can move to Taiwan or Hong Kong, do so, as the cost savings will be immense over time. If not, you have to take into consideration the fact that each PCB and every component sent to you will have a 3-6 weeks delay for arrival and will incur international courier-level costs, plus import duty. Suddenly, the cost of moving seems like a sensible financial decision.”

### With the Goal in Sight

Leighton admits that some early backers have become discouraged by the delays. However, he adds that “there’s nothing I can do about that. If they want to receive a failed product that is non-upgradeable and has no long-term future, I could deliver that to them, but I won’t. We have one shot at getting this right. A few years’ delay is worth it to create a stable decades-long stan-

dard that can be relied on.”

Still, with release in sight, Leighton is already looking ahead to what comes next (Figure 5). First priority is a complete free reference design. Although as trademark holder, Leighton cannot compete with li-

censees of the design, he hopes to establish an EOMA68 certified program to maintain standards. He hopes to establish a foundation to sub-license the EOMA68 certification mark to ensure that everyone will “properly conform to the safety and interoperability aspects.” There will be no charge or licensing fee, but certification will be mandatory before any product can be certified as compliant. “This is just how it has to be,” Leighton says. “It’s primarily down to end-user safety.”

In addition, he plans to work on a free, RISC-V processor. It “will be fully libre right to the bedrock: CPU, GPU, VPU, everything. Even the hardware design source code is libre and is already being developed.” Ultimately, he hopes to see an EOMA68 prototype and a design that can be used generally in the development of open hardware devices of every form factor.

Meanwhile, Leighton gives this advice to would-be open hardware developers:

- “Do the research. Find similar projects and study them; I learned from projects such as *Openmoko* and *OpenPandora*. Hardly anyone aged 25 has even heard of those, despite them being incredibly important and containing extremely valuable lessons.
- “Have a clear goal as to the scope and scale of what you want to achieve. If you only intend to make 50 units, the approach is radically different from wanting to do 1,000 or 10,000 or 100,000 or a million units. Each level of ambition requires a totally different strategy. 50 units, you can easily buy components off of Digi-Key and can use a European or USA-based PCB manufacturer. 5,000 units, you’d best find a Chinese factory.

- “Work out a strategy in advance, which allows you to fund the ongoing development for several years.
- “Don’t just set up a blog. Set up a mailing list, because a mailing list lets people talk amongst themselves, whereas a blog restricts them to talking only to you. People who go to the trouble of talking to you and helping you are your front-leaders and your indirect word-of-mouth marketers. Respect, appreciate, and empower them.
- “Write a blog anyway. Demonstrate to people that you have something rational and useful to say, and they are much more likely to offer assistance and advice. If they learned something from you because you presented it clearly, it should come as no surprise that they want to help you, and you should accept that help and make it easy to find you.
- “Be prepared to write talks and do presentations at conferences. Accept and appreciate that, at the end of the talk, the people who ask questions and also those who come up afterwards to thank and to talk to you would like an opportunity to communicate with you. Listen attentively to what they have to say, and with good grace.
- “Do not make the mistake of ceding control of the business to others. A long-term project such as designing hardware and properly seeing it through to product is something that requires huge sustained year-on-year effort. Plan accordingly and retain 100% control of the business whilst at the same time respecting the value and worth of contributors.”

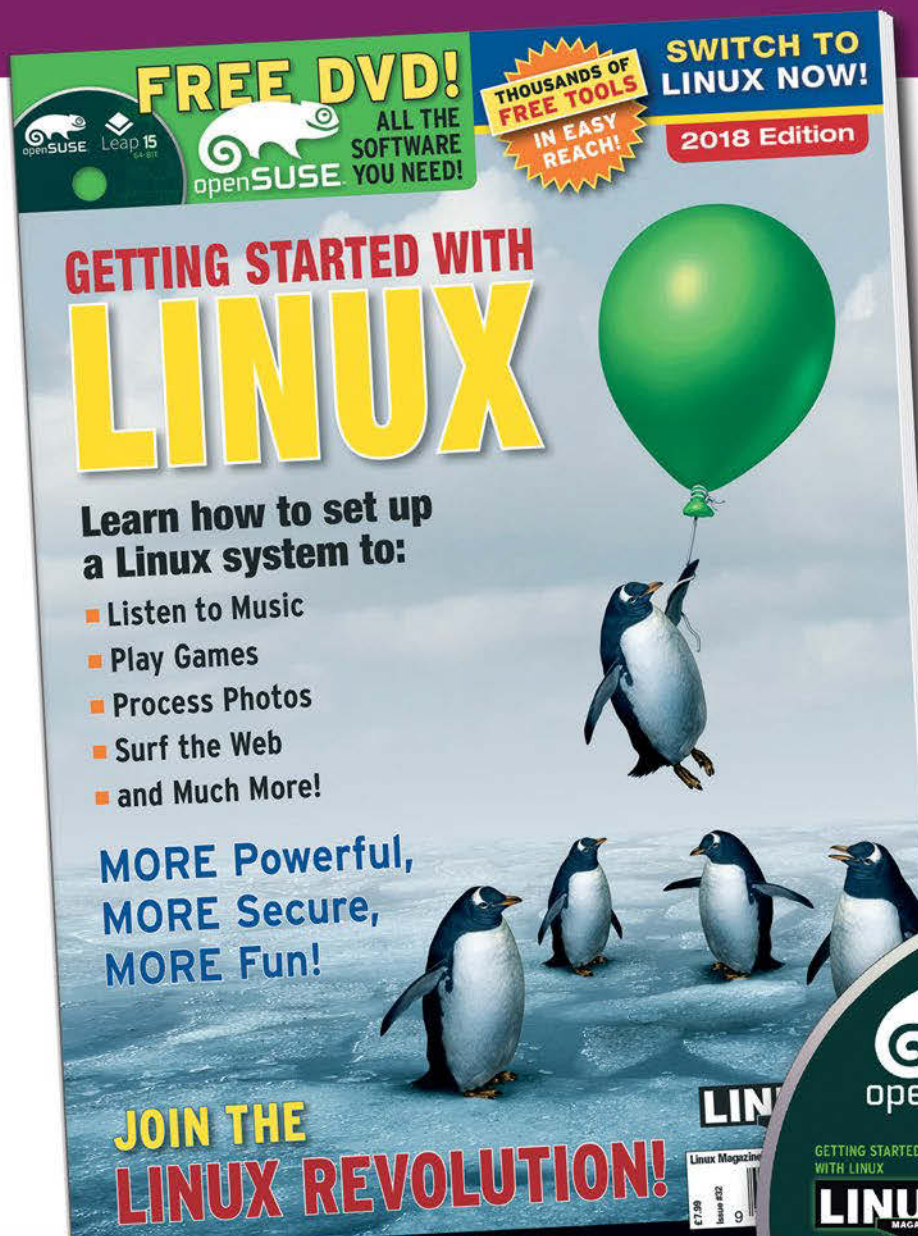
How successful the EOMA68 laptop will be remains to be seen. However, one thing is clear: By being open about the challenges of producing open hardware, Leighton and other first-time entrepreneurs like him are already creating a pool of knowledge that will increase the chance for others to succeed. ■■■

### Info

- [1] “A Free Laptop Project” by Bruce Byfield, *Linux Magazine* online, July 2016, <http://www.linux-magazine.com/Online/Features/A-Free-Laptop-Project>
- [2] EOMA68 campaign: <https://www.crowdsupply.com/eoma68>

# 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](http://shop.linuxnewmedia.com/specials)





# MakerSpace

## Cloud storage for your IoT projects On Fire

IoT projects on the Google Firebase platform promise future expandability and features. *By Pete Metcalfe*

**F**or many Internet of Things (IoT) projects, a message-queuing system like Message Queue Telemetry Transport (MQTT) is all you need to connect sensors, devices, and graphic interfaces. If, however, you require a database with sorting, queuing, and multimedia support, some great cloud storage platforms are available, and one that is definitely worth taking a look at is Google Firebase.

Like any IoT solution, Google Firebase can take various programmatic and sensor data inputs and has a variety of client applications to view the data (Figure 1). However, Google Firebase also offers other features, such as

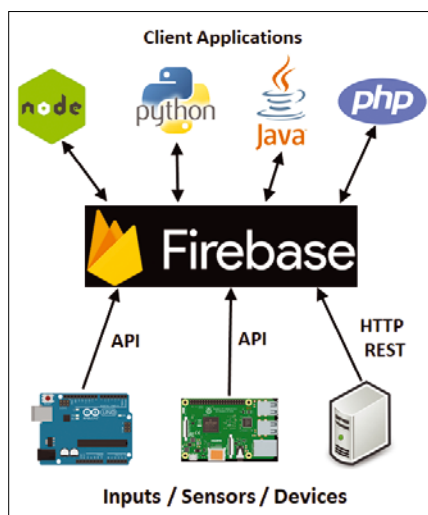
file storage, machine learning, messaging, and server-side functions. In this article, I will:

- set up a sample Firebase IoT project,
- use Python to simulate I/O data,
- create a web dashboard with Node-RED to view and write data,
- create an Android app with App Inventor to view and write data, and
- look at a more complex data monitoring example in Python.

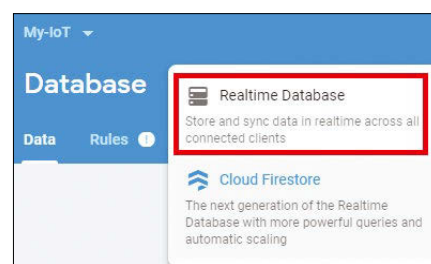
### Getting Started

Google Firebase [1] might not have the huge variety of options that Amazon Web Services (AWS) has, but I found as an IoT engineer that Google Firebase had all the features I need to get up and running quickly, and I don't need a credit card for my free activation.

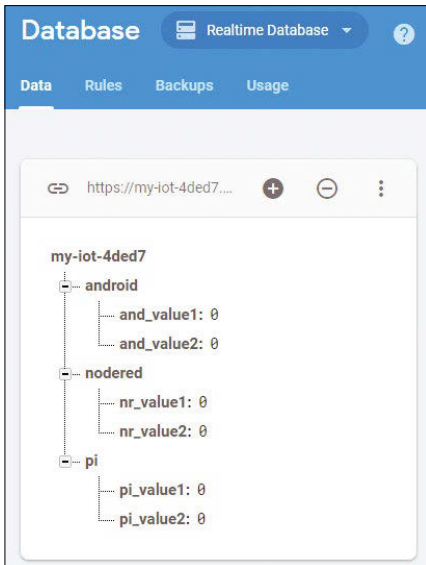
To begin, log in with your Google account [1] and select *Get Started*. After I created a sample project called *My-IoT*, Firebase gave it a unique identifier name (e.g., *my-iot-4ded7*).



**Figure 1:** Firebase IoT overview.



**Figure 2:** Creating a Firebase Realtime Database.



**Figure 3:** Create/view a database structure.

Firebase data can be stored in either a Firebase Realtime Database or a Cloud Firestore. The Realtime Database is an unstructured NoSQL format that has been around for about four or five years, so a lot of third-party components exist. The Cloud Firestore stores the data in structured collections and is fairly new,

so the number of APIs isn't as extensive as you can find for the Realtime Database. For my test IoT project, I use the Realtime Database (Figure 2), which you can create from the *Database | Data* menu option.

The database structure can be built directly from the Firebase web console or imported from a JSON file. For this database, I defined three groupings, one each for Android, Node-RED, and Raspberry Pi I/O values. From the Firebase console, you can set, change, and view database values (Figure 3). Security is configured under *Database | Rules*.

To keep things simple for testing, I set read and write security to true for public access, but I will have to remember to change this when I put the project into production (Figure 4). The project's remote access settings are shown in the Authentication section from the *Web setup* button (Figure 5).

At this point, I have an empty database, I've opened up the security,

and I have the credentials for remote access, so now it's time for some programming.

## Python

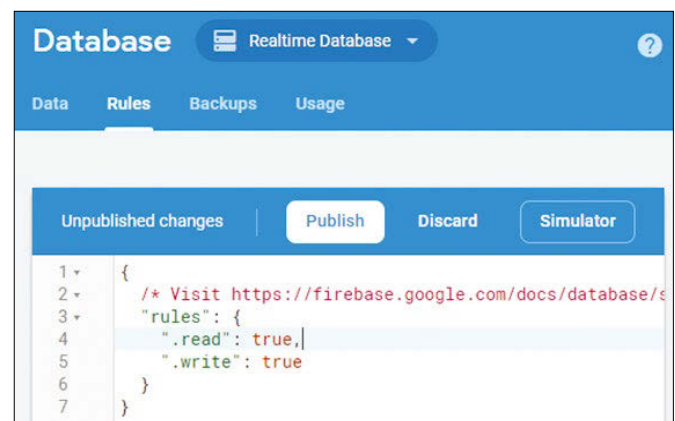
A number of Python libraries access Firebase. I like the *pyrebase* library because it has some added functionality, such as queries, sorting, file downloads, and streaming support. The *pyrebase* library only supports Python 3, and it needs an upgrade of the Google authorization library:

```
sudo pip3 install pyrebase
sudo pip3 install --upgrade google-auth-oauthlib
```

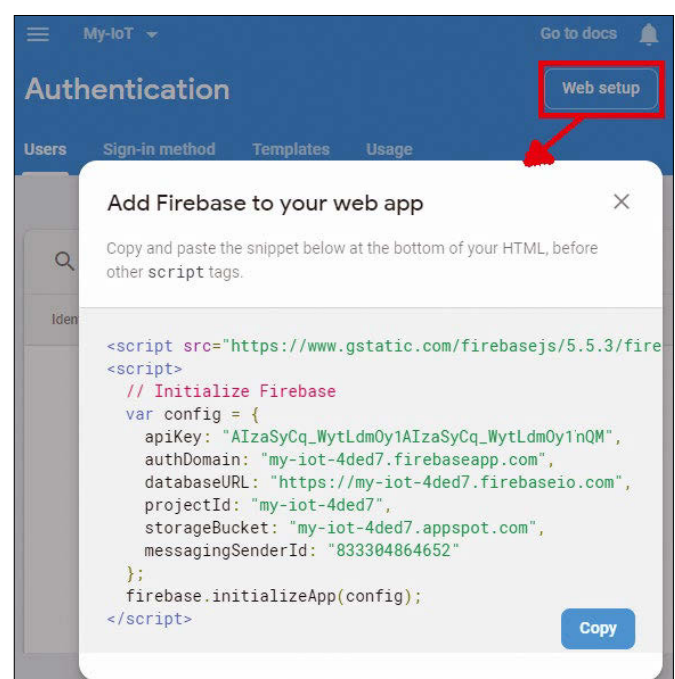
Listing 1 is a Python example I ran on a Raspberry Pi that sets two values (pi/pi\_value1 and pi/pi\_value2) in

### Listing 1: firebase1.py

```
01 import pyrebase, random
02
03 # define the Firebase as per your credentials
04 config = {
05     "apiKey": "AIzaSyCq_WytLdmOy1AIzaSyCq_WytLdmOy1",
06     "authDomain": "my-iot-4ded7.firebaseio.com",
07     "databaseURL": "https://my-iot-4ded7.firebaseio.com",
08     "storageBucket": "my-iot-4ded7.appspot.com"
09 }
10
11 firebase = pyrebase.initialize_app(config)
12 db = firebase.database()
13
14 # set 2 values with random numbers
15 db.child("pi").child("pi_value1").set(random.randint(0,100))
16 db.child("pi").child("pi_value2").set(random.randint(0,100))
17
18 # readback a single value
19 thevalue = db.child("pi").child("pi_value1").get().val()
20 print ("Pi Value 1: ", thevalue)
21
22 # get all android values
23 all_points = db.child("android").get()
24 for apoint in all_points.each():
25     print(apoint.key(), apoint.val())
```

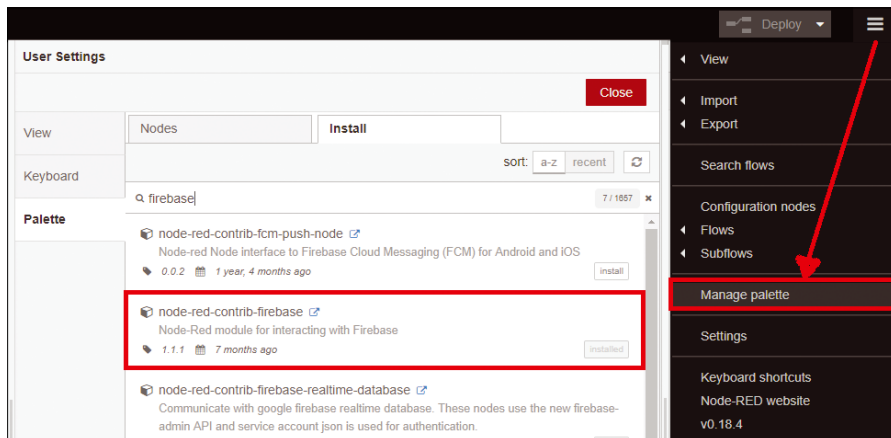


**Figure 4:** Firebase database security.



**Figure 5:** Firebase authentication information.





**Figure 6:** Installing Firebase on Node-RED.

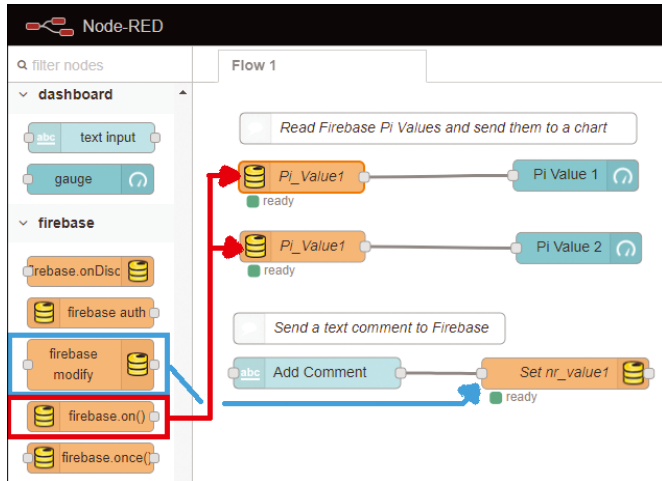
lines 15 and 16 and reads a single point (lines 19 and 20) and a group of points (lines 23 to 25). Remember to change the configuration settings to match your project's security credentials. From the Firebase web console, you can check the results from the test program.

## Node-RED

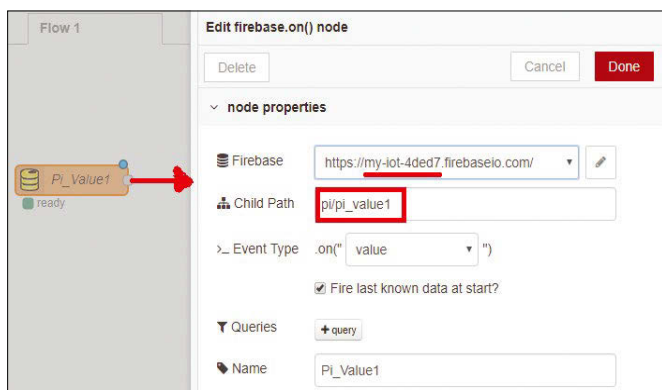
Node-RED [2] is a visual programming environment that allows you to create applications by dragging and dropping

nodes onto the screen and logic flows by connecting the nodes. Node-RED has been preinstalled on Raspbian Jesse since the November 2015 release and can be installed on Windows, Linux, and macOS, as well. You can find instructions online [3] to install and run Node-RED on your specific system.

To install the Firebase components, select the *Manage palette* option from the right side of the menubar, then



**Figure 7:** Node-RED logic.



**Figure 8:** Firebase input configuration.

text comment back to the Firebase database. The complete Node-RED logic (Figure 7) only needs six nodes.

To read the Raspberry Pi values, I use two `firebase.on()` nodes, and to set the Firebase database and data point information (Figure 8), I double-click these nodes. The Firebase field has an edit (pencil) button that allows you to enter your project credentials. To write or set a value in Firebase, you configure a `firebase modify` node.

The output from the `firebase.on()` node is connected to two dashboard gauge nodes. Double-clicking on the gauge node allows you to edit its labels and ranges and create a dashboard layout. A text input node lets you enter text from a web dashboard that then can be passed into Firebase.

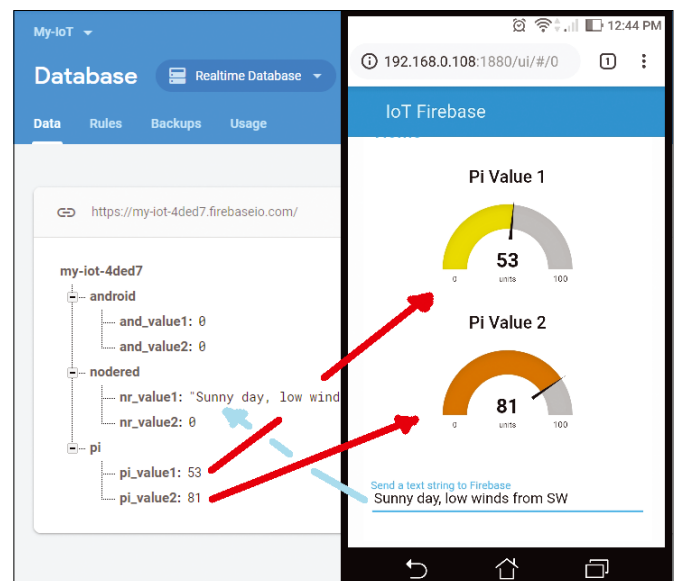
After the logic is complete, hit the *Deploy* button on the right side of the menubar to run the logic. The Node-RED dashboard user interface is accessed at `http:// <ipaddress> :1880/ui` (e.g., `http://192.168.1.108:1880/ui`) (Figure 9).

## App Inventor

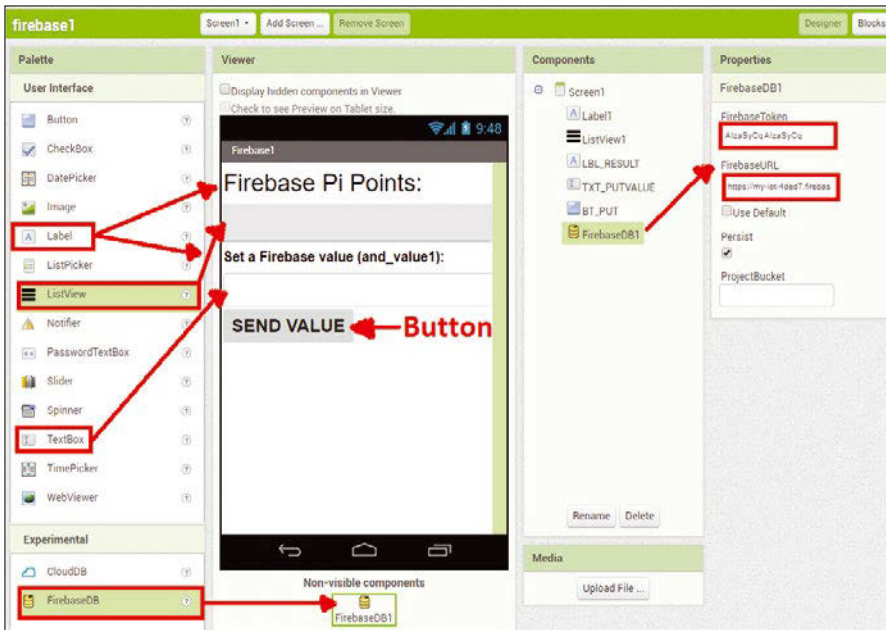
App Inventor [4] [5] is a web-based Android app creation tool with a graphical programming environment. App Inventor has two main screens: the Designer screen, on which you lay out the Android app, and the Blocks screen, where you build the logic. On the right side of the top menubar, the *Designer* and *Blocks* buttons allow you to toggle between the two screens.

search for *firebase*, and install the *node-red-contrib-firebase* module (Figure 6).

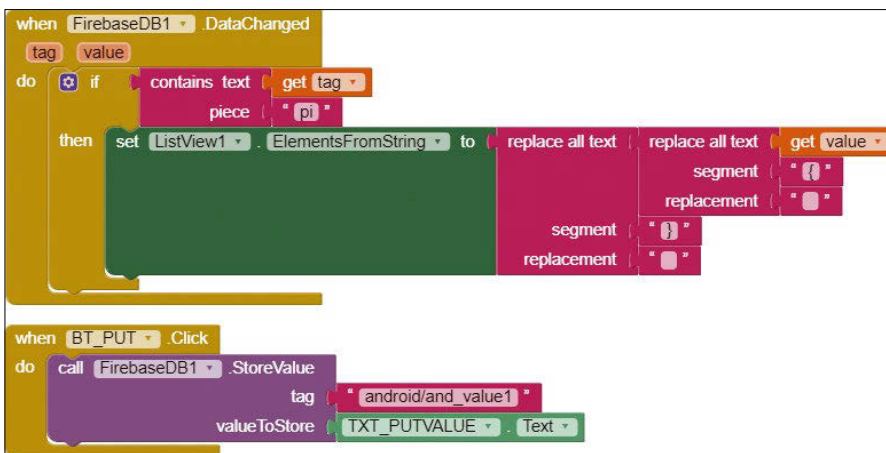
For my Node-RED example, I use web gauges to show two Python simulated test points and send a



**Figure 9:** Node-RED web dashboard.



**Figure 10:** App Inventor screen layout.



**Figure 11:** App Inventor logic to read/write to Firebase.

On the Designer screen, an app layout is created by dragging a component from the Palette window onto the Viewer. The App Inventor Android app I create here will read Raspberry Pi values from the Firebase IoT database and write a value back.

The visual layout of this application uses a *Button*, two *Labels*, a *ListView*, and a *TextBox* component. A non-visual *FirebaseDB* component connects to the Firebase Realtime Database (Figure 10). After a component is added to the application, you can rename or delete that component from the Components window. In the Properties window, you can edit the features of a component. For example, for the *FirebaseDB* component, I need to configure the Token and URL to match my project credentials.

Once the layout design is complete, you add logic by clicking on the *Blocks* button in the top menubar. To build the logic, you select an object in the Blocks window and click on the specific block you want to use.

App Inventor is pretty amazing when it comes to doing some very quick prototyping. The entire app in this example only uses two main blocks (Figure 11).

The *when FirebaseDB1.DataChanged* block is executed whenever new data arrives into the Firebase database. The *DataChanged* block returns tag and value variables. The tag variable is the top-level item name, (i.e., *pi* in this example). The value will be a string of the items and their values, for example:

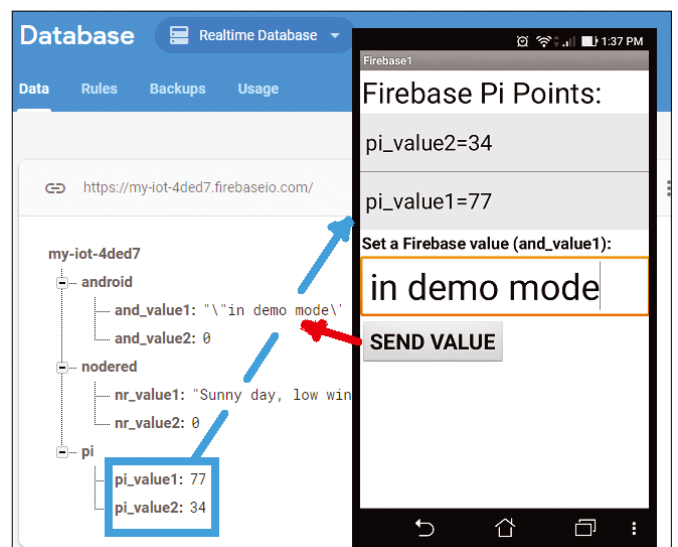
```
{pi_value2 = 34, pi_value1=77}
```

A *replace all text* block removes the leading and trailing *}* characters and then passes the string to the *ListView* component. Note this same code will pass two or 2,000 tags in the *pi* tag section. The *when BT\_PUT.Click* block writes the text entered on the screen into the *android/and\_value1* item in the Firebase database.

After the screen layout and logic are complete, the *Build* menu item compiles the app, which can then be made available as an APK downloadable file or as a QR code link. Figure 12 shows the final Android app communicating with the Firebase Realtime Database.

## Python Data Monitoring Example

The example IoT Firebase Realtime Database is quite simple. Data monitoring or Supervisory Control and Data Acquisition (SCADA) projects usually require more information than just a value. A more realistic sensor database would include fields such as a tag name, description, status, time, and units. By adding some indexing in the



**Figure 12:** Android App Inventor app talking to Firebase.



Firebase security rules (Figure 13), you can create queries and sort the data, such as *Points in alarm* or *Point values between 2:00 and 2:15*.

In Listing 2, the statement

```
tag_sum = db.child("pi").  
    order_by_child("status").  
    equal_to("ALARM").get()
```

(line 14) shows only records that have the `ALARM` status. Some other filter options include `.limit_to_first(n)`,

`.start_at(time1).end_at(time2).get()`, and `.order_by_value()`.

A good next step would be to pass important filtered information to Firebase's messaging feature.

## Summary

Without a lot of configuration, you can configure a Google Firebase project and have Python, Node-RED, and App Inventor apps read and write values. The level of programming complexity is on

par with an MQTT implementation; however, Google Firebase offers a lot more future functionality that you wouldn't have with MQTT, such as file storage, machine learning, messaging, and server-side functions. ■■■

## Author

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



**Figure 13:** Indexing on Firebase to allow queries.

## Info

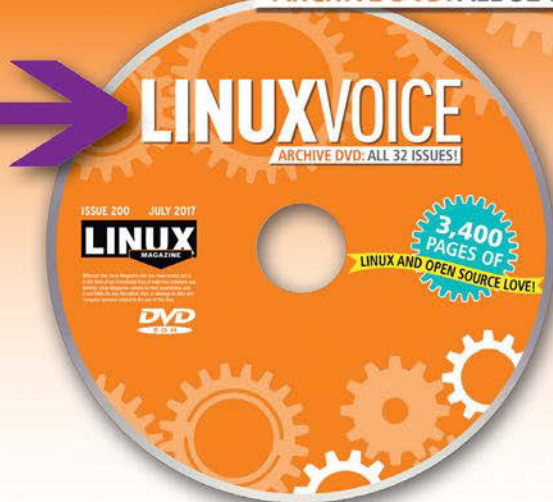
- [1] Google Firebase: <https://firebase.google.com>
- [2] Node-RED: <https://nodered.org>
- [3] Installing Node-RED:  
<https://nodered.org/docs/getting-started/installation>
- [4] App Inventor project: <http://appinventor.mit.edu>
- [5] App Inventor access: <http://ai2.appinventor.mit.edu/>

## Listing 2: firebase2.py

```
01 import pyrebase, random
02
03 # define the Firebase as per your settings
04 config = {
05     "apiKey": "AIzaSyCq_AIzaSyCq_Wyt",
06     "authDomain": "my-iot-7ded7.firebaseio.com",
07     "databaseURL": "https://my-iot-7ded7.firebaseio.com",
08     "storageBucket": "my-iot-7ded7.appspot.com"
09 }
10
11 firebase = pyrebase.initialize_app(config)
12 db = firebase.database()
13
14 tag_sum = db.child("pi").order_by_child("status").equal_
15     to("ALARM").get()
16
17 # print alarm points
18 for tag in tag_sum.each():
19     info = tag.val()
20     print (tag.key(), " - ", info["description"], ":",
21         info["value"], info["units"], info["status"])
```

# THE COMPLETE LINUXVOICE

ARCHIVE DVD: ALL 32 ISSUES!



**3,400  
PAGES OF  
LINUX AND OPEN SOURCE LOVE!**

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](http://shop.linuxnewmedia.com)

**Today's computer magazines** talk about big ideas and new age concepts, like containers, cloud computing, and software-defined infrastructure, but we at Linux Voice know that one of the reasons you own your computer in the first place is to take care of everyday tasks more efficiently and without the clutter.

Linux is home to dozens of useful tools for taking notes and managing to-do lists. This month we feature Joplin, a powerful open source note-taking app that organizes your notes in a searchable form and even supports synchronization with several popular cloud platforms. We also investigate the Unforeseen Incidents point-and-click mystery game, and our tutorial series continues with a look at Bash math functions and more on designing 3D objects with OpenSCAD.



Image © Olexandr Moroz, 123RF.com

# LINUXVOICE

## Doghouse – Freedom Zero 69

*Jon "maddog" Hall*

The GPL's "freedom zero" can be applied to more than just open source software.

## Game Review 70

*Tim Schürmann*

In Unforeseen Incidents, a deadly virus and a spooky government quarantine are the prelude to an exciting point-and-click adventure for adults.

## Joplin 74

*Bernhard Bablok*

If you need an open source alternative to the Evernote note-taking tool, why not switch to Joplin?

## FOSSPicks 78

*Graham Morrison*

This month Graham looks at Eureka, Bookworm, Kdenlive 19.04, KStars 3.1, digiKam 6, CRRCsim, and more!

## Tutorials – Shell Math 84

*Marco Fioretti*

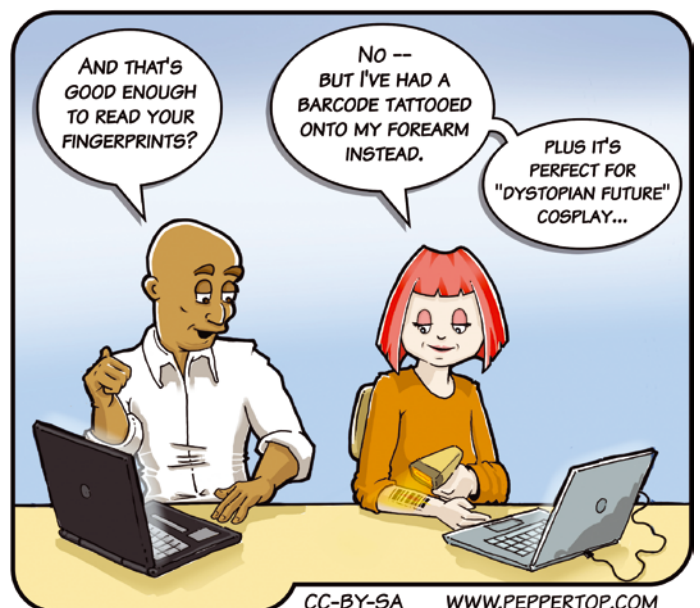
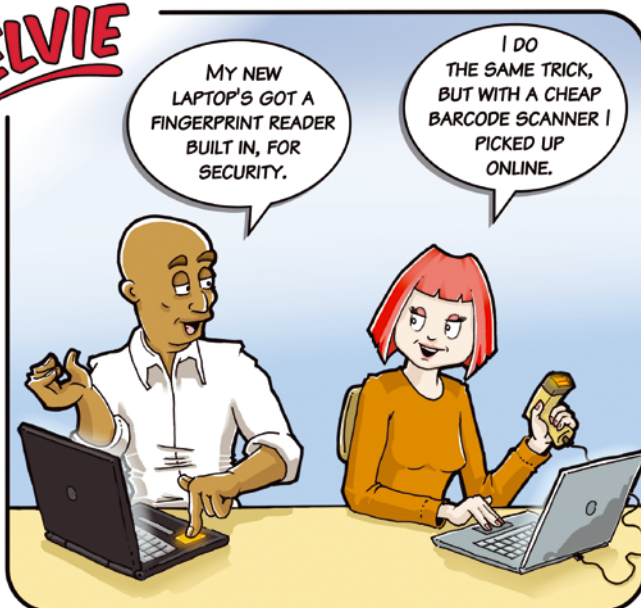
Although Bash is not the most advanced environment for doing and visualizing math, its power will surprise you. Learn how to calculate and display your results with shell scripts.

## Tutorials – OpenSCAD 90

*Paul Brown*

OpenSCAD lets you use simple scripts to build 3D images that you can send to your 3D printer.

**ELVIE**



CC-BY-SA WWW.PEPPERTOP.COM



# REAL SOLUTIONS *for* REAL NETWORKS

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

Learn the latest  
techniques for better:

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

on Windows, Linux,  
and popular varieties  
of Unix.

**GET IT  
FAST**  
with a digital  
subscription!

**6 issues per year!**

**ORDER NOW**

[shop.linuxnewmedia.com](http://shop.linuxnewmedia.com)

# MADDOG'S DOGHOUSE



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®.

The GPL's “freedom zero” can be applied to more than just open-source software. BY JON “MADDOG” HALL

## Cooperation and freedom for all

**R**ecently, a discussion came up on one of the mailing lists for a GNU/Linux distribution, on which I feel it is necessary to comment. Because this discussion has a place in world politics today, I am bringing my input to this column.

I started working for Digital Equipment Corporation (DEC) in 1983. At that time, I had traveled only domestically in the USA, never internationally.

Then DEC realized that I was fairly good at explaining complex technical issues in terms that both technical employees and managers could understand.

About 1986, I went to Tel Aviv to talk about DEC's Unix products for a DEC Users' Society (DECUS) meeting. DEC had an office on the shore of the Mediterranean Sea. In fact, the back door of the office opened up onto the beach.

Before I went there, many friends asked if I was afraid of the bombings that were going on at that time. I told them that I was not. If other people could live there, I could too.

I had several good friends working at the DEC office in Tel Aviv, and one day at lunch we went out the back door to sit in the sun and watch the beach volleyball game that was being played there.

I asked one of my friends (a citizen of Israel) what the conflict was between the Israelis and the Palestinians. He wrinkled his nose and pointed at the volleyball game.

“See that guy? He is a Jew. The next one? He is an Arab – now two more Jews and three more Arabs. They are all getting along, because they all want the same thing. A good life for themselves, a better life for their children, and to be left alone by the government.”

At the center of the conflict are real people on both sides who need real places to live, real abilities to live and travel back and forth without harassment, and real water and land rights.

As I continued to travel the world I found that almost all people only want a good life for themselves, a better life for their children, and to be left alone by the government.

Twenty-five years ago, I became involved with the Linux Kernel Project. When I started with the project, it was mostly a “techie” project and had little commercial value. However, I

could see that sooner or later it would have commercial value, and people could start making money with a GNU/Linux distribution.

Initially, there were developers that said they did not want people making money with software that they developed “for free.” Others said that they did not want banks using their software, because they did not like banks. Other people said they did not want the military using their software, because they did not like the military, or governments using their software, because they did not like the government. And so it went.

If you applied all their wishes, you would find that none of the software could be used by everyone.

If you did not allow others to make money, then they would fight the spread of free software. If you allow them to make money, they would help to spread it.

Finally, someone pointed to the GNU Public License (GPL) and “freedom zero”: The freedom to run the program as you wish, for any purpose.

Now, I pull the two points together, because a large free software development group has chosen to have their main conference in Haifa, Israel. Some of the developers argue that it should not be held there because of the conflict between Israelis and Palestinians.

I think that the conference should be held there for several reasons:

- 1 People who live in Israel are involved with many things, and are not defined solely by the ongoing conflicts.
- 2 Developers of this group live in Israel just as they do in many other countries around the world.
- 3 The committee for choosing the host country evaluated all of the submissions fairly and chose Israel.
- 4 Having a conference there creates an avenue for putting forth the hand of understanding.

Some of the developers say that if the conference is held there, they will not go. It is certainly their right.

On the other hand, I hope that the developers that do go, from all over the world, act as shining beacons for what free software really stands for, cooperation and freedom for everyone. ■■■



Unforeseen Incidents combines suspense with easy puzzles

# Point-and-Click Mystery

In *Unforeseen Incidents*, a deadly virus and a spooky government quarantine are the prelude to an exciting point-and-click adventure for adults.

BY TIM SCHÜRMANN

The phone rings as the protagonist, Harper Pendrell, reluctantly gets up from a mattress on his hobby room floor. It's Professor MacBride on the line (Figure 1), and once again, he is having problems with his laptop, which is urgently needed to evaluate research results. Harper grabs his universal multitool, which resembles a large Swiss army knife, and sets off for MacBride's lab – unaware that an adventure is about to begin.

## Epidemic with a Question Mark

In the lab, a quick cable repair solves the problem, and Harper can now return to his mundane life – if it weren't for a copiously bleeding woman whom he finds on his way home. She has all the symptoms of a highly contagious and quickly fatal virus currently afflicting the residents of Harper's small town.

The woman asks Harper to take an important document about the virus's origin to a journalist in a hotel on the town's outskirts. Even before Harper fully understands the situation, a panicked resident alerts the RHC, the health organization responsible for both investigating the disease on behalf of the government and providing appropriate quarantine measures.

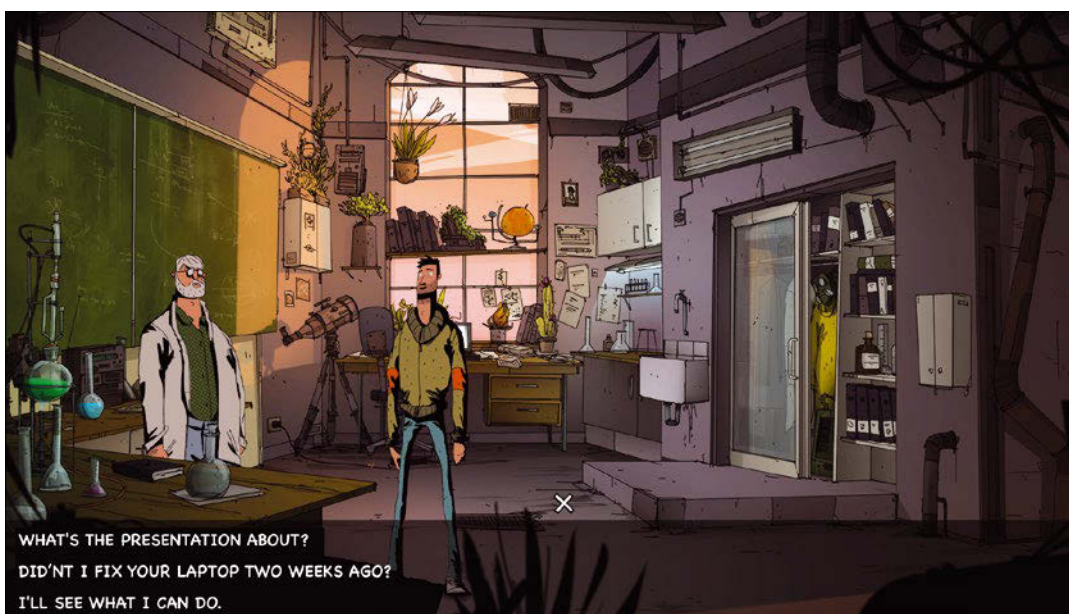
Although the government claims to have the virus under control, it is slowly spreading throughout the country. Infected persons who have been quarantined never reappear. Moreover, the RHC's response is surprisingly quick: Only a few seconds after the panicked resident signals an alert, men in yellow protective suits pick up the sick woman (Figure 2).

With the documents in hand, Harper has no choice but to search for the journalist. He receives help from Professor MacBride, who is interested

in the new virus and is searching for an antidote. From here, Harper embarks on a dangerous journey meeting many strange characters and using his multitool along the way.

## Classic Clicks

*Unforeseen Incidents* [1] is a point-and-click adventure in which you control the fortunes of Harper Pendrell. A mouse click sends the hero to the matching point in the landscape. After clicking on an object, Harper either makes a comment (Figure 3) or picks up the



**Figure 1:** In conversations such as this one with Professor MacBride, you not only learn interesting facts about the characters, but also get tips for solving the puzzles.

object, which the game stores along with all the other collected loot in a bar at the top of the screen (Figure 4).

You can use drag and drop to combine different objects. Talking with the town's residents follows the proven multiple-choice principle. In this way, numerous puzzles must be solved to move the story line forward. If you don't want to search the entire scene, press the space bar and the game will display all the items and people with whom interactions are possible.

While many of today's point-and-click adventures use cartoon or pixelated diagrams, *Unforeseen Incidents* offers a totally unique look. The hand-painted graphics (part graphic novel, part scribbled color sketch) underlines the somewhat gloomy background story creating an atmospheric mood.

In contrast, the animation is extremely reduced and awkward: Harper looks like he's walking on wooden legs. In addition, the game does not show some important actions onscreen. For instance, in a key scene with a large tractor, the game simply turns the screen black and outputs a sound effect.

## Details

*Unforeseen Incidents* is the first release by Backwoods Entertainment, a small game developer studio from Bochum, Germany. You can purchase the game on Gog.com [2], Steam [3], and Humble [4] for \$19.99. If you buy the game from Humble, however, you only receive one key to unlock the game on Steam. Gog.com offers a version without copy protection. On Gog.com, you can also purchase a digital art book and the game soundtrack for \$2.99 each. Steam offers these extras bundled together with the



**Figure 2:** Even before Harper can finish questioning the dying woman, two RHC employees take her away.

game in a single package for \$22.85. All sites offer the game in both a 32-bit and a 64-bit version.

To play *Unforeseen Incidents* on Linux, at a minimum, you will need Ubuntu 16.04 with SteamOS+, an Intel Core 2 Duo from 2GHz, and a graphics card with 1GB memory, 4GB RAM, and 6GB hard disk space. For Gnome users, see the "Gnome Bug" box.

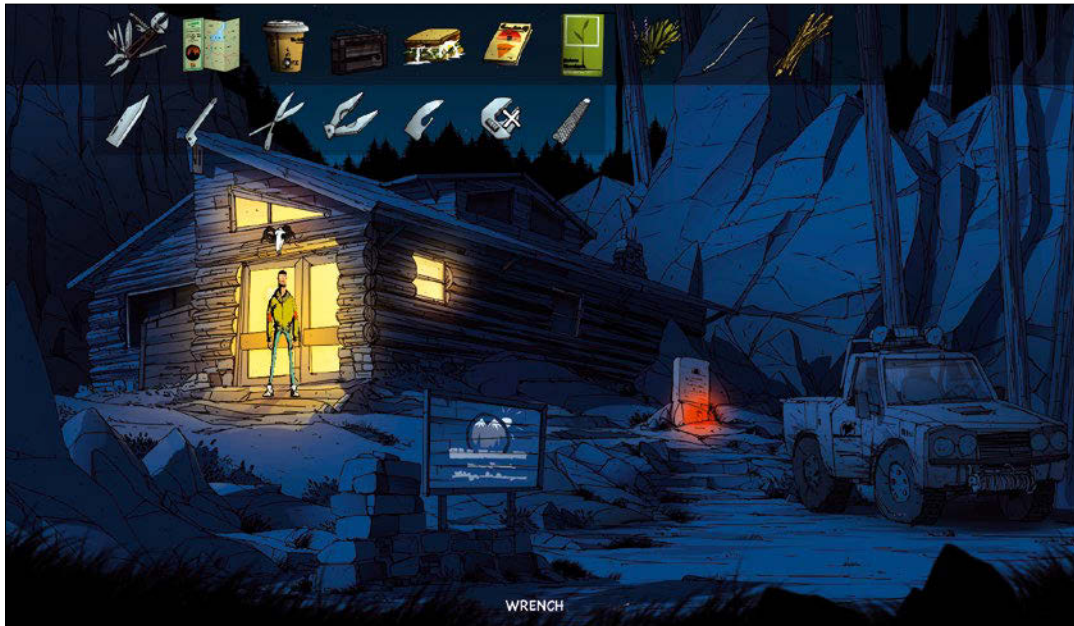
## Conclusion

*Unforeseen Incidents* is a classic point-and-click adventure with an excitingly staged story. For experienced adventurers, however, the puzzles are not massively challenging. The puzzles themselves, however, have a logical structure and are meaningfully embedded into the action (Figure 5). The bizarre and adorable characters and the



**Figure 3:** Harper tends to be slightly sarcastic. If you click on an object repeatedly, he even gives you different comments.





**Figure 4:** Objects belonging together are grouped in the inventory at the top of the screen. Clicking on the multitool opens all the tools, for example.

protagonist's sarcastic humor loosen up what is otherwise a serious and somewhat mysterious plot. Despite the jerky animation, *Unforeseen Incidents* is one of the best new adventures and more than worth its \$19.99 price tag. **■■■**

## Gnome Bug

If you start Unforeseen Incidents under Gnome, you will get a message that the game is not responding. Just wait a moment: After a short break, Unforeseen Incidents invites you to play despite the message.

## Info

- [1] Unforeseen Incidents:  
<http://www.backwoods-entertainment.com/unforeseenincidents.html>
- [2] Gog.com:  
[https://www.gog.com/game/unforeseen\\_incidents](https://www.gog.com/game/unforeseen_incidents)
- [3] Steam:  
[https://store.steampowered.com/app/501790/Unforeseen\\_Incidents/](https://store.steampowered.com/app/501790/Unforeseen_Incidents/)
- [4] Humble:  
<https://www.humblebundle.com/store/unforeseen-incidents>



**Figure 5:** Every now and then you need to solve puzzles. In this example, rotate the squares with the mouse so that all folders light up red.





# 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](http://shop.linuxnewmedia.com/digisub)





# Open source note taking with Joplin

## Listed and Distributed

If you are looking for an open source alternative to Evernote, why not switch to Joplin? **BY BERNHARD BABLOK**

**L**ong before the digital age, it was commonplace to quickly jot something down, perhaps on a yellow sticky note. Today, there are countless electronic note-taking apps for every operating system. Some even reproduce the yellow stickies visually, while others offer additional functions such as to-do lists or reminders.

Well-integrated programs of this kind exist for the popular Linux desktops. However, exchanging data between different systems is often problematic. For example, the Xfce Notes app does not save changes immediately. If you synchronize the file with the notes at the wrong time, errors are inevitable. Furthermore, mobile operating systems are usually ignored.

Everyone has a different understanding of what makes a perfect notes app, but some basic features, like ease of use, are considered important by most. Ideally, the program supports the ability to attach images and other media. In addition, it should be easy to share the notes across devices with different operating systems, with automatic synchronization and conflict resolution added into the bargain. Last, but definitely not least, privacy must be guaranteed.

The note-taking king of the hill, Evernote, meets many of the requirements mentioned above. However, Evernote requires you to register with the vendor, and then only the basic functions are freely accessible. In addition, Evernote is not open source, and it lacks a native Linux client.

Joplin, a free note and to-do list app, offers an open source alternative to Evernote.

### Architecture

Joplin is implemented in JavaScript. All data except images and other attached resources are stored in a SQLite database. The program's guiding principle is "local first" – the application works completely independently of a cloud; all notes are available on the computer at all times. In terms of content, the application organizes notes and to-do lists in nesting notepads.

For desktop systems, Joplin provides Web Clipper, an add-on for Chrome/Chromium or Firefox. Once installed in the browser, Web Clipper sends a web page's content to Joplin at the touch of a button – a real killer feature, because Web Clipper saves the page's text, not just a link or a screenshot. Since the notes within the software are accessible for full text searches, this makes a decisive difference.

Finally, the Joplin architecture provides synchronization. Joplin supports a whole range of possibilities, from simple network drives to WebDAV servers to OneDrive or Dropbox. Since the data can be stored on external servers, the software offers the possibility to encrypt the contents before synchronization.

### Installation

Besides the source code, binaries for various platforms are available on the Joplin website [1]. The Android version is available directly from Google Play. Under Linux, you can alternatively download and run a script, which has the advantage of updating an existing Joplin installation. Since Joplin's developers are very active, there are sometimes only a few days between new versions.

If you use AppImage, you will find the `Joplin.AppImage` file located in the `.joplin/` directory below your home directory. To launch automati-

#### Listing 1: Automatic Startup

```
[Desktop Entry]
Name=Joplin for Linux
Comment=Joplin Notes App
Exec=$HOME/.joplin/Joplin.AppImage
Icon=false
Terminal=false
Type=Application
StartupNotify=false
X-GNOME-autostart-enabled=true
```

cally at startup, simply create an entry in the Auto-start folder (Listing 1). For Xfce, you need to place this file in the `$HOME/.config/autostart/` directory. The installation program also stores a desktop file in `.local/share/applications/joplin.desktop`, which creates a menu entry.

After the first launch, you will see an empty window with a note on how to create your first notebook (Figure 1). Before you do this, however, you will want to configure various settings using the entries in the Tools menu.

## Configuration

The Tools menu contains three options: *Webclipper Options*, *Encryption Options* and *General Settings*. If you want to use Web Clipper, enable it via the corresponding option. In this dialog, you will find links to the browser extensions for Firefox and Chrome.

*Encryption Options* contains only an activation button and a link to the Joplin documentation.

*General Settings*, in contrast, offers various options, most of which are self-explanatory (Figure 2). If you are working on a high-resolution display, it may be worth your while to increase the zoom level. You can also define the type of synchronization in this dialog if required.

All in all, you can set up the software for local operation within a very short time. Under Android, the configuration dialog is even more compact – here it is especially important to fill out the parameters for synchronization.

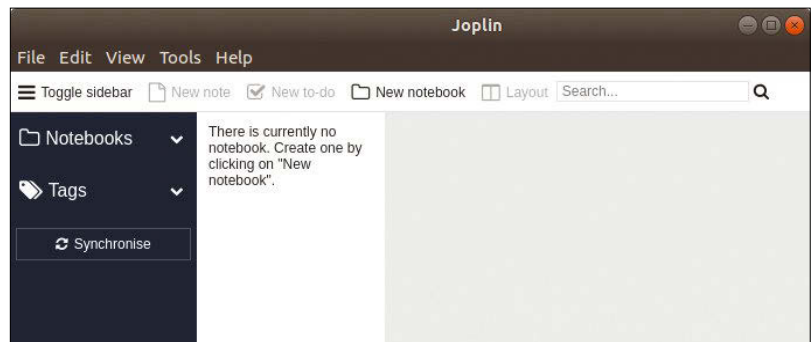
## Creating Notes

You create notes either via Web Clipper or manually. Joplin uses Markdown [2], a simple markup language, to define the text's structure. The built-in editor can handle three layouts: Either you see only the markdown text, only the interpreted version – or both (Figure 3).

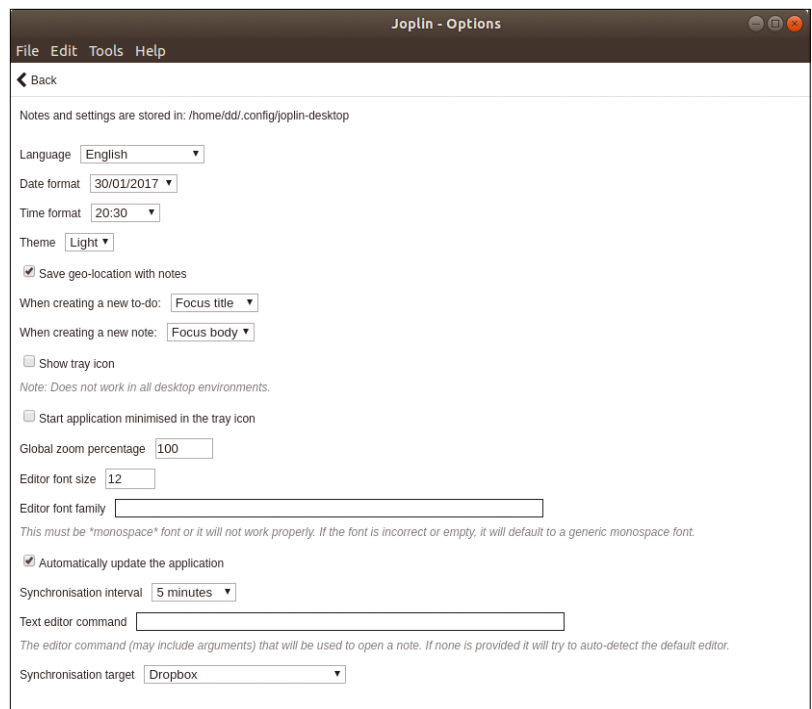
To create a new note manually, click on *New note* or *New to-do*. You can use the context menu to switch between the two at any time. Attachments can be added using the paperclip button, and notes can be categorized using tags. In addition, you can set an alarm; however, this requires an installed and enabled Notify daemon on Linux.

## Secure Sharing

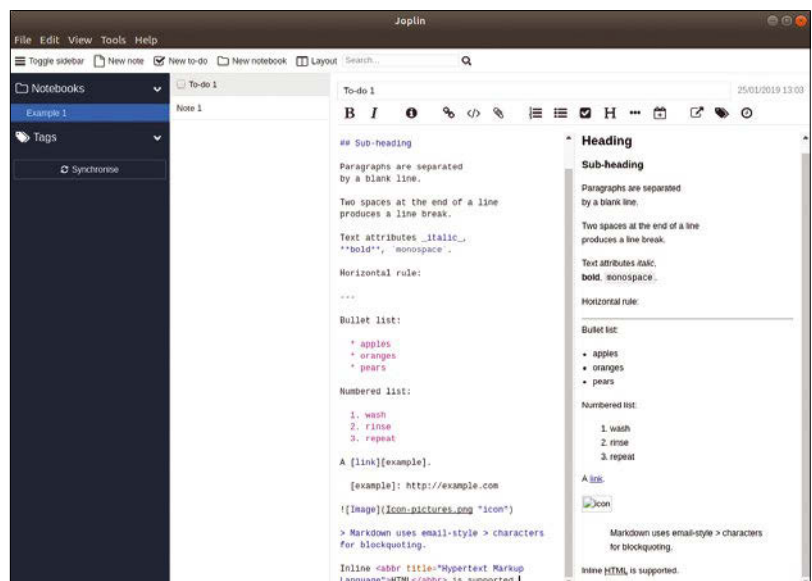
If you are using a small network attached storage (NAS), it is a good idea to choose a network drive for note storage. Because mobile devices don't have easy access to a network drive, you should use another option when you include such clients. You can then either use a cloud service in combination with encryption, or you can set up a WebDAV server yourself (see the "WebDAV Server Setup" box).



**Figure 1:** After the first start, Joplin offers to create a notepad directly, but it is worth configuring some basic settings first.



**Figure 2:** In *General Settings* you can adjust synchronization intervals, and change font preferences, among other things.



**Figure 3:** The editor shows both the markdown text and the interpreted version in split layout mode. This can make input easier.



**WebDAV Server Setup**

If you run a small home server, you can set up a WebDAV server on it with little effort. All common web servers offer a module for this. For Apache, first install the actual web server with the commands from Listing 2, then copy the contents of Listing 3 to the file `/etc/apache2/conf-available/joplin.conf`, and enable the configuration with the commands from Listing 4. The configuration shown here grants access to everyone. If you do not want this, set up a username and password. The whole thing will even run on a small-board computer like the Raspberry Pi Zero without any trouble. Apache is considered to be a resource hog, but that will not necessarily play a role in a home setup with low access figures.

**Listing 2: Installing a WebDAV Server**

```
$ sudo apt-get update
$ sudo apt-get -y install apache2
$ sudo a2enmod dav
$ sudo a2enmod dav_fs
```

**Listing 3: /etc/apache2/conf-available/joplin.conf**

```
Alias /joplin "/var/www/joplin/"
<Directory "/var/www/joplin/">
    DAV on
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

**Listing 4: Enabling a WebDAV Server**

```
$ sudo a2enconf joplin
$ sudo apache2ctl -k graceful
```

**Using a Service**

Email providers like GMX offer an even simpler alternative to WebDAV. With your GMX email account, you are also given access to the MediaCenter, which provides 2GB of storage space. Via the service's website, you then log in with your mail address and password and create a folder in the MediaCenter (e.g., `joplin/`). Then enter the WebDAV URL (in our example `https://webdav.mc.gmx.net/joplin/`) into the corresponding fields of the Joplin configuration (see Figure 2) and add the mail address and password as the username.

Click on *Check synchronization settings* to check if everything is good to go. Then press *OK* to complete the configuration. In the current version, data synchronization can only be switched on or off globally; the software does not support restricting this to individual notebooks.

When this issue went to press, the Linux desktop version had a bug that caused synchronization to run extremely slowly. This does not affect normal notes, but only those with attached resources such as images or other files – in particular all pages stored using Web Clipper.

Given Joplin's fast update cycles, the chances are pretty good that the problem will be fixed by the time this issue is published. Apart from patience, the only thing that helps is to switch on the synchronization in time to avoid a backlog of unsynchronized notes.

**Encryption**

If you only sync your devices locally, you can do without encryption, especially since it can be enabled retroactively. Depending on the data volume, however, the first encryption can take a long time. For this rea-

son, it makes sense to use the encryption function right from the start regardless of which devices you plan to sync or whether you sync at all.

When setting up encryption, Joplin generates a password-protected master key. The password must be entered on each device that you include in the synchronization setup. Keep the password in a safe place, because if it's lost, you won't be able to access your notes.

If you are retroactively enabling encryption, the Joplin developers recommend doing this on the device with the highest computing power. Joplin then encrypts the content, and all devices involved are updated by synchronization. Only enable encryption on the other devices after they have received the master key via synchronization. Otherwise you will have several master keys with several passwords – not a technical problem, but a major organizational headache.

**Conclusions**

Joplin is unlikely to win a beauty contest. Additionally, if you are already pushing Evernote's limits, you will probably not be totally satisfied with its free competitor. However, for most users, Joplin covers all your daily needs and, as a massive benefit, offers you full control over your own data. The program can even import Evernote data, so there is nothing standing in the way of a change. ■■■

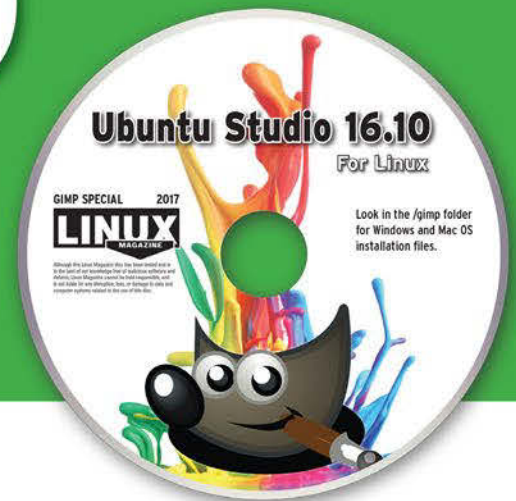
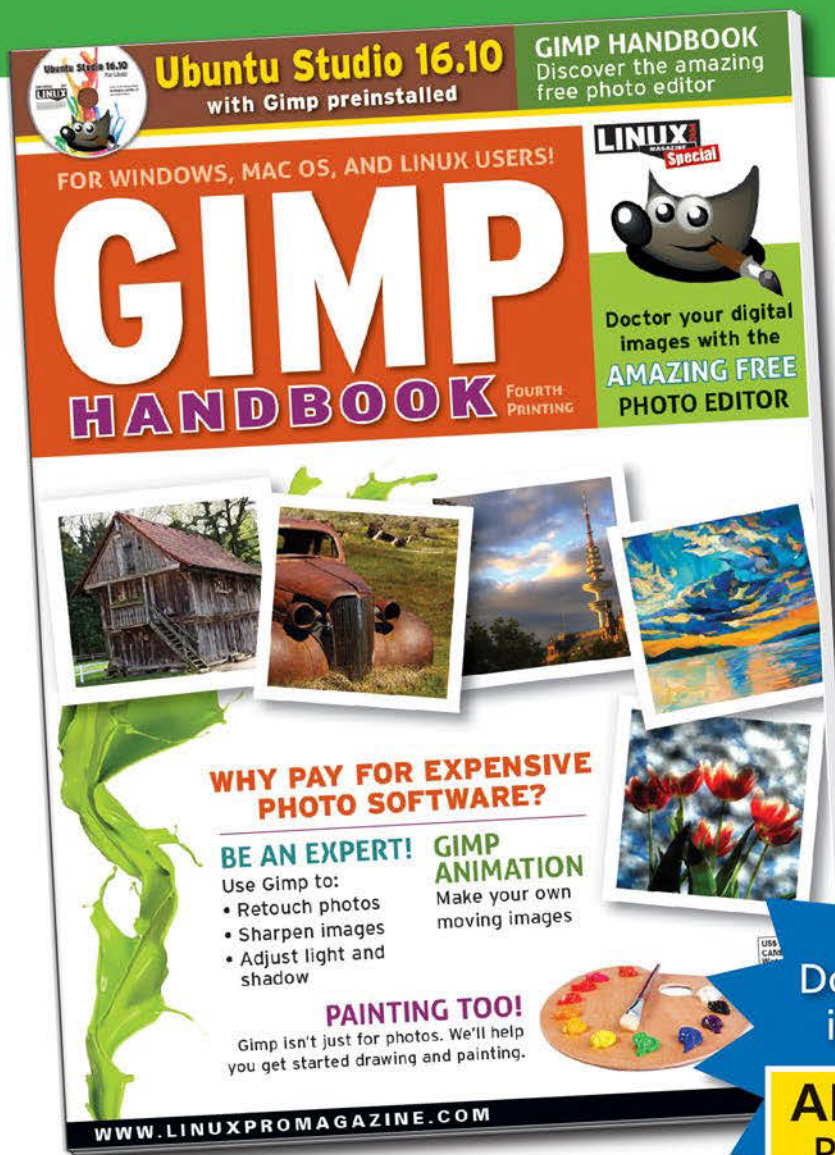
**Info**

- [1] Joplin: <https://joplin.cozic.net/>
- [2] Markdown language: <https://guides.github.com/features/mastering-markdown/>

Shop the Shop

[shop.linuxnewmedia.com](http://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](http://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



If there's one thing writing FOSSPicks has taught Graham, it's that there are as many ways to install and build a package as there are to cook an egg. **BY GRAHAM MORRISON**

## Video Editor

# Kdenlive 19.04

**K**denlive is one of those applications that catches you by surprise. One minute it seems little more than a quick Qt-built GUI wrapped around some command-line tools to concatenate video files, and the next minute it's spending months on hiatus being rewritten and refactored into something that can genuinely start to compete with Final Cut Pro on macOS. This is what's happening with Kden-

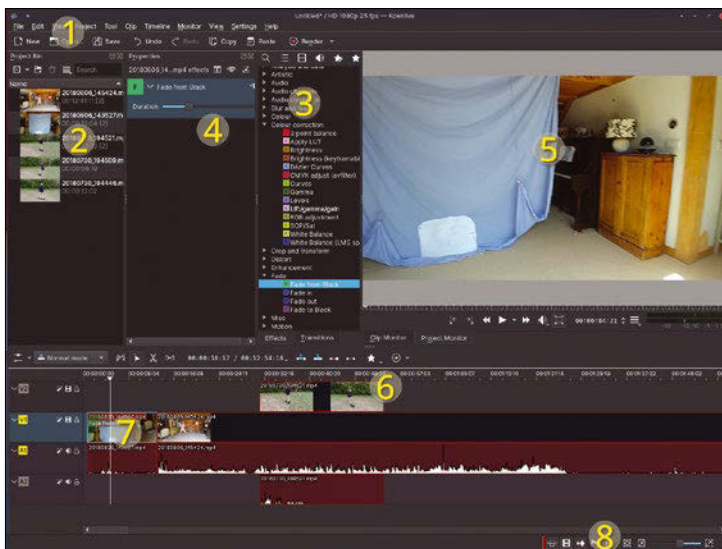
live; you'll find the all new version in the KDE Applications 19.04 release. To be fair, it already was the best open source video editor available, barring perhaps Blender if you needed absolute power and had the patience to master its idiosyncratic user interface.

Many who have perhaps not used Kdenlive for a while won't realize that it now includes some rather advanced features. One, for example, allows you to

use low quality copies of a clip as proxies for edits you want to make. This saves CPU and storage resources and is perfect for our 8K future. There's also a brilliant Title Editor toolbar that enables you to create 2D text frames without having to resort to an external package. This important function is always overlooked in open source video editors, as they often focus on performance and clip editing. However, adding titles is equally important. You only have to look at the most popular YouTube videos to see how words, spacing, shadows, gradients, and images are spliced into video segments to create a professional and snappy video. Dropping an alpha-blended Gimp text render doesn't really cut it, unless you're creating a video about Gimp. Kdenlive has done all this for some time, which has perhaps been our only criticism: It was difficult to see where new developments were taking the project.

All this has changed with 19.04, because it's where users will finally get their hands (or mice) on the results of a significant refactor and redesign, especially to the time line. For everyone that remembers KDE's refactor between the KDE 3 and KDE 4 releases, this likely inspires feelings of intense trepidation – it took years for KDE to regain the same functionality and trust it had in the KDE 3 era as features, applications, and paradigms were wantonly dropped in the name of progress and reinventing the wheel. There is an element of this in the refactoring of Kdenlive. Stability is still an issue, and some menus no longer offer the same options they did previously. As an example, there are currently no curves in the keyframe transitions. However, these elements are hugely outweighed by new features, such as the default audio split, easier monitor viewing, and a lovely user interface that allows almost every element to be dragged and rescaled, including tabs. With this update, Kdenlive remains not just an excellent open source video editor, but one that's excellent regardless of whether it's open or proprietary or how the project is developed.

**Project Website**  
<https://kdenlive.org>



1. Import/export: There's a huge number of ways to import and export your work, including integration with **dvdauthor**. 2. Clip view: Create a tray of clips to work on and process them, such as with image stabilization. 3. Effects: These and transitions can be dragged into the time line and moved without affecting their properties. 4. Effect properties: Drag and drop effects, change parameters, and see a preview in real time. 5. Clip and project monitors: Drag either monitor out of the main view into its own screen. 6. Time line: Video and audio is now separated by default. 7. Control: Resize, drag clip length, split, and remove, all from the time line. 8. Flags: Enable automatic transitions, comments, and audio thumbnail.

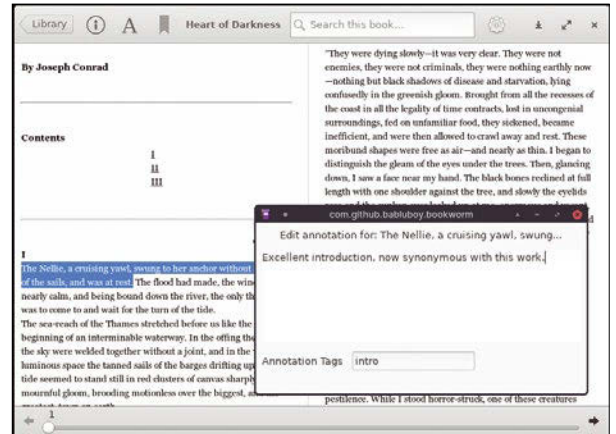
## Book management and reader

# Bookworm

**D**espite the digital revolution supposedly revolutionizing print and the initial huge success of e-readers like Amazon's Kindle, there hasn't been a huge blossoming in desktop or computer-based readers. There are a handful of minimal text readers, and of course there's Calibre. However, the latter is more like an unwieldy electronic book development environment than a reader for quiet contemplation. There's nothing like the range of applications you find for music playback, for instance, and you're often better off browsing a book's HTML directly on Project Gutenberg's website. But this doesn't help if you like to actually purchase your books, and this is where Bookworm can help. Bookworm feels a lot like an old iTunes

version, only geared to books rather than music, and without an online store.

Just like a music player, you can ask Bookworm to import your collection by scanning a directory or by importing each file manually. The directory scanning can be found in the Preferences pane, but you'll need to restart the application before Bookworm will attempt to update your library. At least you can add more than one folder if you have books in multiple locations. You can also enable dark mode from the preferences pane, as well as two-page reading; these preferences are useful if you're reading at night or on a wide screen. Another great feature is the ability to create different reading profiles by setting different colors for background, text,



Manage and view your ebook collection saved in EPUB, PDF, CBR, and MOBI formats.

and highlights. You can then easily switch between profiles. The library view uses either a thumbnail or a list, with the thumbnail images showing the cover and a small progress bar when you mouse over the image. However, reading is where you'll spend the most time, and the exceptional clarity and font rendering is coupled with bookmarks, search, annotations, and word definition, all available with a click of the *i* icon.

**Project Website**

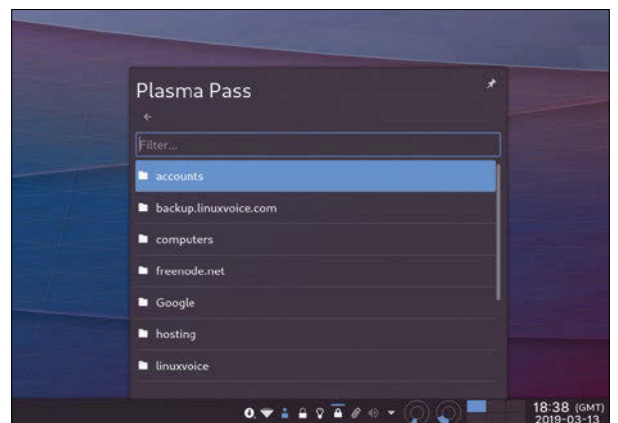
<https://babluboy.github.io/bookworm/>

## Password widget

# Plasma Pass

**T**here are many tools for managing your passwords. As a general rule, the simplest are often the most effective, because they're the ones you're most likely to use and trust. But simplicity doesn't always equal security, especially when it comes to online services that push and pull passwords back and forth to your browser. This is why we're big fans of `pass`, the simple command-line tool that effectively GPG encrypts your passwords into your regular filesystem, making them easily accessible from ordinary Unix tools. There are even tools that will automatically fill fields in your browser. We're big fans of QtPass GUI too, albeit with the caveat that one release broke its random password generator.

If you use the KDE desktop, there's now an even more convenient GUI, Plasma Pass. It's a Plasma widget that takes up very little space and is always available. As it's early days for development, installation is currently a little problematic. We needed to manually build and install from the source code. Hopefully, by the time you read this, you'll be able to download the widget directly from KDE's *Get New Widgets* requester or via a package for Neon. With the package installed and added to your desktop or panel, Plasma Pass couldn't be easier to use. It mimics the filesystem layout of your passwords, and you just need to click through this to select the entry you wish to decode to the clipboard. You never see the



Add quick and easy pass integration with the KDE desktop, thanks to Plasma Pass.

password itself, unless you paste it into a text field; your system's password manager can be used to decrypt the key. The widget will show a countdown for the password's lifespan on the clipboard before it takes care of removing it for you. It's basic, but exactly what you need.

**Project Website**

<https://cgit.kde.org/plasma-pass>



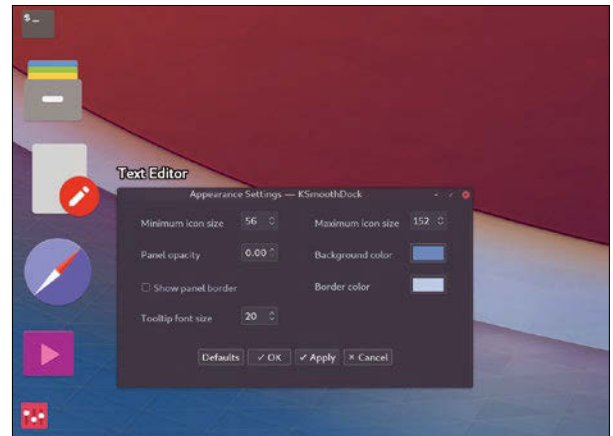
## System launcher

# KSmoothDock

For something that most users don't think about, there's plenty of choice when it comes to the Linux desktop launch panel. While the overall principle hasn't changed since the early days of Windows, its presentation has changed dramatically. This seems mostly thanks to Apple, who for many years did without a dock completely and then introduced a simplified icon-only launcher in Mac OS X. The same launcher then appeared on its iPhone and iPad, cementing the idea of what kind of tools work best when it comes to managing running apps. There are now several Linux launchers that mimic the style of Apple's dock, but none quite get to the same level of integration and refine-

ment as the one you find in macOS. However, KSmoothDock gets very close.

KDE is rather well equipped to handle new docks, because you can often use more than one panel at a time and work with as many as you need until you find a configuration you like. You can even do this with the panels that come by default. The first options KSmoothDock presents to you are where you'd like your new panel to exist and which functions you'd like it to perform, including launch menu, pager, task manager, launchers, and clock. When it appears, it looks and feels very similar to the equivalent macOS dock, complete with fast and efficient zooming, running icons in the lower panel, and



KSmoothDock is actually a stand-alone application, rather than a Plasma widget, which means it can be easily added to your current configuration.

configurable opacity. There are even options to control the small and large icon sizes, background color, and whether there's a panel border – none of which you can do on macOS without a hack. It looks and feels fantastic. If you've been after that macOS look and responsiveness, this is perhaps the best we've come across on the Linux desktop.

## Project Website

<https://store.kde.org/p/1081169/>

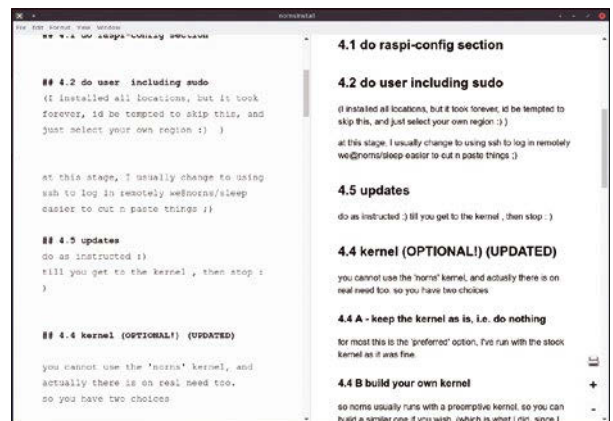
## Text editor

# PanWriter

If you've ever needed to convert one raw text document to another, you'll know just how arduous the process can be. There are so many different formats (and formats within formats), and they conspire with the subtleties of language to make the task almost impossible. Even programmers who can easily manage binary file conversion struggle with text formats. However, there is one text-processing tool to rule them all, and that's Pandoc. Running from the command line, `pandoc` will let you convert one text input stream to an output stream. It knows about the different versions of HTML, Markdown, DOCX, and LaTeX, and it can take much of the pain away. This is why Pan-

Writer is potentially such an interesting text editor, because the "pan" element in its name refers to Pandoc.

PanWriter uses Pandoc to import and export from its native Markdown. This ensures that you can get as close to a format's canonical version when exporting and work with as many different formats as possible. As with similar Markdown editors, you can split the view vertically to show your writer's rendered output on the right of the main window. This is useful if you know your words are going to be presented in this way as it gives you a great impression of the amount of space your words will use. But what's also unique in PanWriter is that this preview



Thanks to the Electron platform, we imagine adding an in-line CSS preview to a Markdown editor is relatively easy.

is rendered via CSS, which you can then edit in-file by adding the CSS to your Markdown document. The results of your tinkering are shown immediately in the preview, making this ideal for writers working with static sites that take Markdown as an input.

## Project Website

<https://github.com/mb21/panwriter>

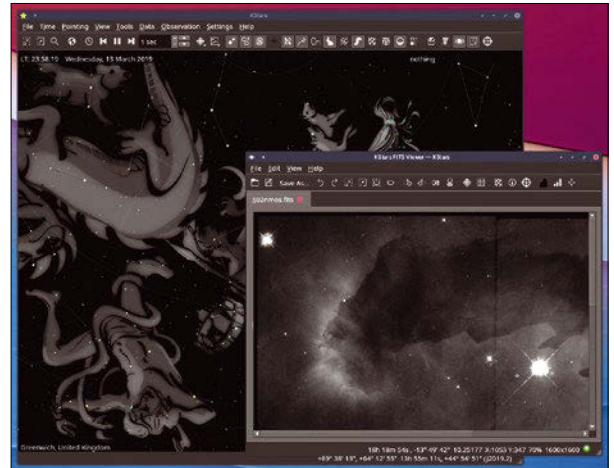
## Planetary

# KStars 3.1

**T**he KStars application has long been a stalwart of the KDE educational packages. And like the stars themselves, changes to KStars take an epoch to become visible to the human eye, which means we must have switched epochs. In the last few months, there have been two huge updates to this venerable planetarium. The first came at the end of 2018 with the release of KStars 3.0, with version 3.1 coming a few months later. The major update for these releases is the planet viewer, which borrows its code from the XPlanet solar system viewer. This adds a new settings page to the main KStars application, and you visit this first to trigger the download of more detailed XPlanet

image maps. With that done, you can now pan and zoom around the surface of planets, like Mars, which really helps to break through the 2D barrier where KStars sometimes traps you, unlike Celestia, which is a pure 3D planetarium.

Another new feature that also helps you break through the false ceiling is the FITS viewer GUI. FITS is an astronomical image format, where astronomical doesn't just mean "from space" but also size and resolution. They can be generated or downloaded from locations such as the European Space Agency. With a decently optimized viewer that offers histograms, exporting a FITS image can be akin to going outside and looking through a powerful



Explore the universe and images from Hubble with a desktop planetarium.

computer-controlled telescope. If you do go outside, the latest release adds all kinds of features for previewing images on DSLR cameras and controlling remote telescopes. And vitally, lots of this has been documented, so you don't have to go guessing or searching through the menus and settings pages. You can now just read the excellently updated manual.

**Project Website**

<https://edu.kde.org/kstars/>

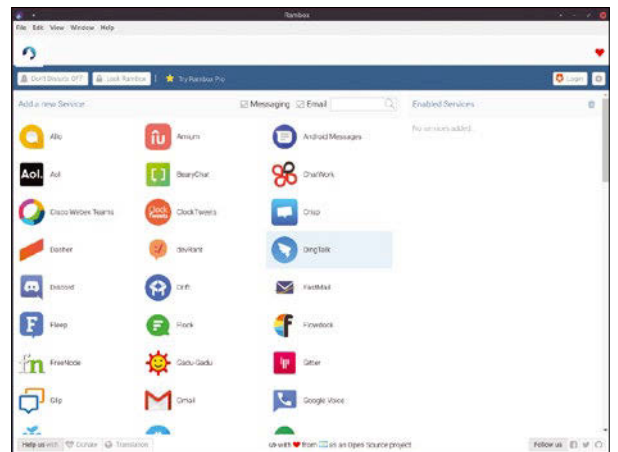
## Coms aggregation

# Rambox

**R**ambox is an interesting project. At one end of the scale, it's a paid-for hosted service that aims to help your productivity, but at the other end, it's an open source client that offers the same functionality running locally with your data stored locally as well. This is all thanks to the magic of `npm` and Electron. Normally, Electron seems like a cumbersome approach to creating a desktop application, but it's appropriate in the case of Rambox, because the application is itself a way of aggregating various online portals and messaging services. It unifies login with a single password, as well as themes and notifications, along with when to terminate them. When you first start Rambox, you're presented with a

huge list of these services. There's 103 in total, and the list includes the usual suspects such as Skype, Slack, Gmail, Twitter, and LinkedIn, as well as more esoteric services like ICQ, Steam, and Riot.

You need to enter your account details for at least a few of these for Rambox's functionality to make sense. Each service you enter is then added to your top panel, which will also include notifications on unread messages. You can also add your own custom services, which like everything else, needs to be either messaging (like email) or chat. With that done, you can then aggregate your various services into the same view, and it works quite well. It's certainly less distracting than having multiple win-



With Rambox you can aggregate all your messaging and chat applications into a single window.

dows open and skipping between them all without getting any real work done. You can limit yourself to a few minutes checking updates before closing the main window and getting back to being productive. This is where Rambox works best, especially if your work makes you use several similar services, because you can limit the amount of distraction and close just a single window when you reach EOD.

**Project Website**

<https://github.com/ramboxapp/community-edition>



Photo manager

# digiKam 6

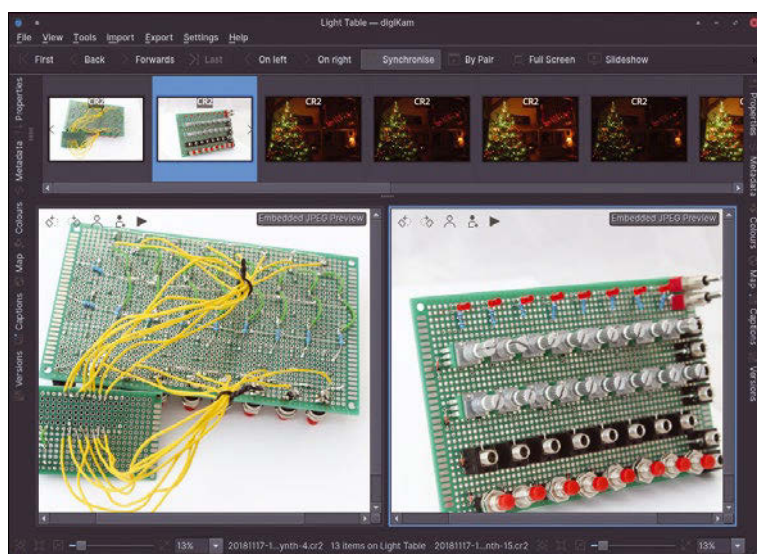
We're looking at three major releases from long-lived KDE applications in this issue: Kdenlive and KStars (see earlier), and now digiKam 6. While we've never previously mentioned Kdenlive, we did feature digiKam almost two years ago with the release of digiKam 5, and it's fantastic to report on its strong and continued development. And digiKam 6 is a fantastic release, albeit one that focuses on managing your collection rather than editing photos. Top of its new features is the updated ability to list video files alongside your photos. This may seem a little divergent for a photo management tool, but it makes a lot of sense.

Many of us take both photos and videos from the same device and keep them in the same folders. It's even what you expect when you browse your media on a phone. The digiKam developers have been playing with this idea for some time, but they've finally settled on a back-end solution that uses FFmpeg to generate the thumbnails. You wouldn't want to use digiKam to view or edit them (and you can't), but it's handy

seeing your content all in one place, and it follows a similar model to that used by many other photo management applications.

Improving photo management further, you can now drag and drop thumbnails within a collection to create your own ad hoc order. This work stemmed from a 2004 bug report, when other photo management applications, including Kalbum, already offered this feature. It's particularly useful if you want to create a presentation or create an album for someone else. Thanks to Yingjie Liu and the Google Summer of Code project, you can now move around thumbnails using manual mode with the latest release.

To get more photos into your collection, there's also lots of new support for RAW image formats, as this is the format used

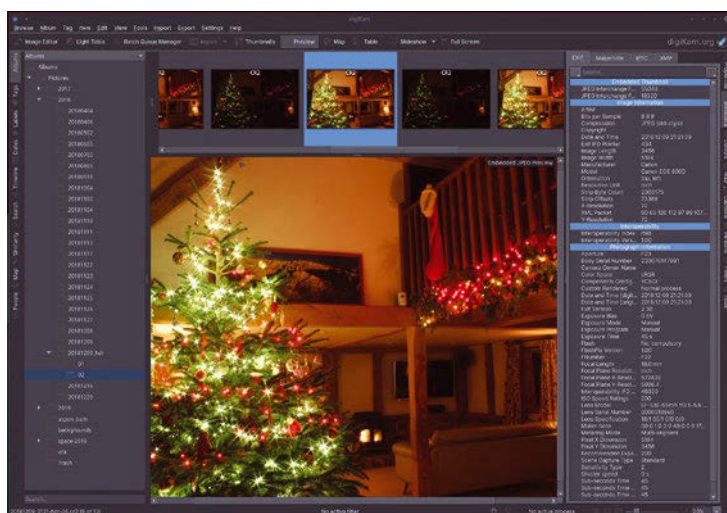


The light table is perfect for side-by-side comparisons of photos.

by most professionals and ambitious amateurs. We certainly found it picked up our new Canon CR2 images without issue. New iPhone, Samsung, Huawei, and Sony phones, and even PARROT and DJI drones, are supported. In addition, digiKam generated preview thumbnails for our huge collection quicker than darktable. Sharing also has become easier thanks to integrated OAuth and support for extra online services, including Pinterest, OneDrive, and Box. There was a time where digiKam was trying to be both an excellent photo and album manager and a brilliant and capable photo editor. With the rise of edit-specific applications like RawTherapee and darktable, it's no longer quite so important that one application does it all. We think the digiKam developers understand this. DigiKam excels at letting you manage huge collections, including integrated Marble for geolocation, excellent similarity processing to catch duplicates, and even face recognition. With this new release, these management functions are better than ever. As Linux users, we've never had free photo management and editing software so good, and it's at a point where just a couple of free and open source applications can rival proprietary offerings. If that isn't enough, you can now even recommend digiKam to your Windows using friends and family, thanks to its new and quickly developing Windows version.

**Project Website**

<https://www.digikam.org>



Embedded jpegs can be used to generate previews of high resolution RAW images.

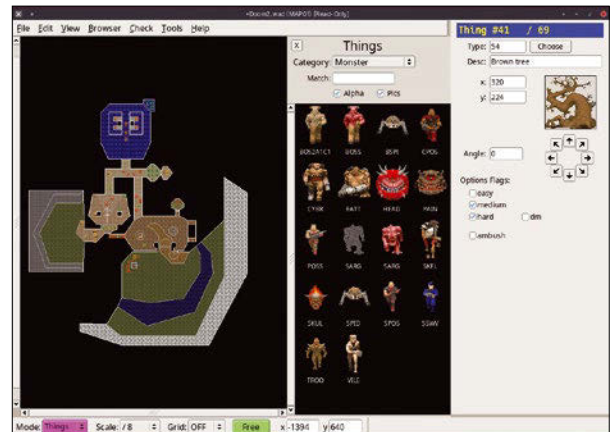
## Doom editor

## Eureka

Only after a significant amount of time has passed can you really begin to assess whether something is an art form. Video games seem to be passing that threshold, with titles like Doom still demanding our attention even a quarter of a century after their original releases. This trailblazing first-person shooter is still being played, modified, reverse-engineered, and ported to modern systems, which this excellent level editor proves. When it launches, Eureka starts off by asking for the location of your local Doom installation's WAD file. This file will be familiar to anyone still tinkering with Doom, because it contains all the level details and defines which version of Doom

you're playing. Third-party engines can often load multiple releases, as well as user-created WAD files, and this is exactly what you can create using the Eureka editor.

With your WAD file loaded, you see a top-down view of its entire layout. You can now select a vertex used to build the model and adjust its location. You can then switch between vertices, sectors, linedefs, and things with the drop-down menu on the bottom left. The parameter view on the right changes depending on what type of element you've selected. In *Things* mode, for instance, you can select individual monsters and adjust their difficulty, whether they're going to ambush the player, and their



Many game designers got into the business by starting off with simple level design. Eureka and Doom could be your chance!

location, as well as their type. Select type to see all the possibilities for your objects from a huge palette of images. There are similar browsers for textures and "flat" elements used within the game. However, the best feature appears when you select *Toggle 3D View* from the View menu. This reveals a full first-person rendering of your level, and you can move about this and select various elements just as you can with the flat editor.

## Project Website

<http://eureka-editor.sourceforge.net/>

## Model simulator

## CRRCSim

The brilliant flight simulator, FlightGear, gets a lot of positive publicity, because it's a hugely ambitious open source project that works. It's complex, comprehensive, and deeply immersive, allowing anyone with the bandwidth plus the storage to hold its huge amounts of terrain data to fly many different aircraft types across almost anywhere in the world. However, FlightGear isn't the only flight simulator available. We also have CRRCSim. CRRCSim isn't quite as ambitious as FlightGear, but that's literally because of its scale – it simulates flying model aircraft in a field (or over a beach) rather than real aircraft across the American Midwest.

This lack of scale also means that CRRCSim has a real and very practical use, because one of the hardest elements of learning to fly model aircraft is getting an intuition of where left, right, up, and down might be while you're standing in the middle of a field looking at your aircraft flying towards you. It's cheaper to crash two dozen virtual models than it is to crash the real thing, and CRRCSim really helps you get your bearings. This means that even though it has not been updated since drones took most of this thinking out of the process, it's still a valuable tool. You can select your environment, use a USB controller, choose between different aircraft, create your own models and fly them, and even fly with your mouse



Ideal for drone pilots in training everywhere.

and mouse wheel if you have nothing else at hand. It also is great fun, as you can quickly restart with a new piece of kit and test it out flying into the headwind. There's even a plan view of the wind, so you can take your experimentation further and see how you're able to handle the changing environment.

## Project Website

<https://sourceforge.net/projects/crrcsim>



# Prepare calculations and chart results with Bash Math, Shell Style

While Bash is not the most advanced environment for doing and visualizing math, its power will surprise you. Learn how to calculate and display your results with shell scripts.

BY MARCO FIORETTI

In this series' previous installments on Bash scripting [1, 2, 3, 4], I have explained how to use data structures, flow control commands, and testing operators. Now, I will show you how to handle number-related tasks with shell scripts.

Before getting started, I want to discuss the strengths and weaknesses of doing math inside shell scripts.

Bash cannot do floating-point math or plotting. As for speed, fast calculations of any kind have never been a selling point of Unix shells. When it comes to performance, there have been many, much better solutions since before C and Fortran. Today, with the right libraries, even other scripting languages are better choices if raw execution speed is your main requirement.

The same applies to interactivity and user interfaces, especially when you want to quickly visualize how changing some input data or formula impacts a math function or changes the solution to some problem. If this is what you need, a LibreOffice spreadsheet or tools like Genius [5] or Graph [6], for actually studying math, may be much more effective solutions.

The real strengths of doing calculations in a shell script are "human efficiency" and "glue logic." While a shell script will most likely run slower than equivalent code in other languages, writing that code as a Bash script will very likely be faster. Run-time performance only really matters with code that you must run many times, possibly every day.

For an average desktop user, however, most problems that require writing code will be one-off issues, different enough from each other to require different code each time. In all those cases, getting working code quickly saves much more human time than writing the fastest possible code. Shell scripts are made to order for this kind of need, even when number-crunching is required.

Efficiency is even greater when you consider what I call the shell script's glue logic. Shell scripts are born as all-purpose tools that can easily

launch, and connect with pipes, any other command-line program. Consider any problem spread across different areas, such as file management and actual numeric calculations of any kind: If you can quickly implement the complete solution as *one* script, what you gain in implementation time and self-documentation greatly outweighs what you may lose in raw performance.

## Variables

One of shell scripts' advantages is that you do not need to explicitly declare the variable type. At the same time, the Bash interpreter has native support only for integer arithmetic. Therefore, the first best practice to deal with numeric variables is to explicitly declare their nature with the `-i` (integer) switch:

```
declare -i MYCOUNTER=10
```

This will reduce the possibility of unnoticed errors, because the script will abort if you try to assign to `$MYCOUNTER` any value that is not an integer number. For the same reason, you should really declare as read-only all the numeric constants you want to use:

```
declare -r TIMEOUTVALUE=100
```

## Operators

Integer arithmetic's basic operations are supported with the same operators you would use in calculations done by hand, with the same relative precedences: first exponentiation (\*\*), then multiplication (\*) and division (/), sum and subtraction, and finally remainder or modulo (%):

```
X=8
X=$(( $X / 2 ))
Y=3
Z=$(( X ** Y ))
Q=$(( Z % 5 ))
```

### 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>).



In the example above, *Z* would be equal to 64, and *Q* to four, which is exactly the remainder of 64 divided by five. The modulo operator is often used to determine whether a number is odd or even. Other essential operators are those for automatic increment or decrement, which have the highest precedence of all:

```
X=4
echo $(( X++ ))
echo $X
```

The first `echo` command would print 4, and the second would print 5. This happens because the value of `$X` is incremented by one unit, but only *after* passing its previous value to `echo`. The automatic decrement operator (`$X--`) works in the same way.

I already introduced the Bash syntax for arithmetic expressions in this series' first installment "Shell Variables", but here is a quick recap. The examples above use the most flexible syntax, the double parentheses. Inside the double parentheses, you can execute any mathematical expression with or without spaces and use all variants of all operators. If you prefer (or simply want to copy code from older scripts), however, you can also evaluate arithmetic expressions with the `let` and `expr` keywords. Both of these commands:

```
let "X /= 2"
X=`expr $X / 2`
```

divide by two the current value of *X*.

## Floating-point Math with `bc`

The Bash interpreter has no built-in commands to handle floating-point calculations or complex mathematical functions like those for trigonometry. In practice, this has never been a problem, thanks to the command-line, script-ready calculator (and language) known as `bc`.

`bc` supports arbitrary precision numbers, for which you can separately define both "length" and "scale." The `bc` man page [7] defines length as "the total number of significant decimal digits in a number." The scale, instead, is the total number of decimal digits after the decimal point.

When you tell it to load its math library with the `-l` option, `bc` can use functions like `c(x)`, `l(x)` and `sqr(x)` to calculate cosine, natural logarithm and the square root of *x* respectively. `bc` can be used interactively at the prompt, but what makes it really powerful is the capability to load code from files or standard input. In the first case, `bc` executes code from all the files passed to it in the order they are listed one line at a time. In the second case, you can pipe to `bc` one or more lines of code generated on the fly, according to

the current values of some variables, as in this small example:

```
#> Y=55
#> X=`echo "scale=8; c($Y/39)" | bc -l`
#> echo $X
.15985120
```

A common, quick-and-dirty usage of `bc` in Bash scripts is to round up decimal numbers to the lower, or higher integer, as shown here:

```
X=4.5
echo "($X + 0.5)/1" | bc
5
echo "($X + 0.49)/1" | bc
4
X=4.51
echo "($X + 0.49)/1" | bc
5
```

These examples also highlight that you should be careful with the rounding value. `bc`'s internal programming language supports internal variables and arrays and comments in the same syntax as the C language:

```
/* any text between these characters */
is a comment */
```

It also supports control flow structures and print statements. Due to space constraints, I can't cover the language in more detail here. However, it is simple and similar enough to Bash that if you have already mastered the previous installments of this series, it will not be a problem for you (also see the detailed `bc` man page [7]).

## Playing with Time

Time-related calculations do not involve math functions or handling decimal numbers. However, they may still seem a daunting task for Bash unless you know and creatively use one or two quick and dirty tricks. In many cases, this can be handled by calculating the difference between moments in time. To do this, you use the `date` command options `-d` and `%s`. As an example, Listing 1 calculates the duration, in days, of the Space Shuttle program, from its first flight (April 12, 1981) to its last (July 8, 2011).

### Listing 1: Calculating a Time Duration

```
01 FIRSTFLIGHT=`date -d "1981-04-12 10:05" +%s`
02 LASTFLIGHT=`date -d "2011-07-08 10:05" +%s`
03 DURATION=$(( LASTFLIGHT - FIRSTFLIGHT ))
04 DURATION=$(( DURATION/(3600*24) ))
05 echo "Flights of the Space Shuttle Program went on for $DURATION days"
```



The `-d` switch in lines 1 and 2 tells `date` to print out the date that follows, instead of the current one. The `+` sign defines the format to use, and its `%s` option returns that date, expressed as the number of seconds since January 1, 1970. Now, all that is left to do is calculate the difference between `$LASTFLIGHT` and `$FIRSTFLIGHT`, which for the reason I just explained are both integer numbers, and convert it from seconds to days (line 4). In case you can't wait to solve this yourself, the answer is 11,044 days.

It is easy to see how the trick in Listing 1 can be generalized to calculate any difference between two moments in time, including future ones. As easy as it is, however, this trick has one weakness that may be irrelevant for some and cause problems for others if ignored.

The double parentheses construct may not round correctly the result when the subtraction result is *not* an integer multiple of 24. This is exactly what happens when the time interval considered includes the day when daylight savings time ends or begins (i.e., a day with 23 or 25 hours). The solution to this problem is simpler than it may seem: It consists of replacing the double parentheses in the example above with the `bc` command. For a practical example with a very detailed explanation, see reference [8].

### Beyond Vanilla Math

Before moving on, I want to introduce, for completeness, three niche types of calculations, each of which could fill its own tutorial. When you encounter these types of calculations, you might incorrectly assume that they cannot be handled with shell scripts.

The first is bit-level arithmetic, which means acting directly on the *bits* that compose each number (or any string really). It has two applications:

- Doing arithmetic basically in the same, very low-level way it happens in machine language, mostly for didactic purposes.
- Using numbers as very compact status registers, where each bit indicates, for example, if one element of an array is in one of two states. See the examples online [9] for more information:

```
15 >> 3 = 1 # '1111' >> 3 = '0001'
15 & 3 = 3 # '1111' & '0011' = '0011'
```

While it might seem cryptic, the first operation uses the right shift operator (`>>`), which shifts to the right all of its left term's bits of a number of positions equal to the right term, and replaces the empty spaces with zeroes. Since the bit-level, binary representation of 15 is 1111 ( $8 + 4 + 2 + 1$ ), shifting those four bits three times to the right yields 0001, which is exactly the number one in bi-

nary format. The second operation is a bitwise AND: It performs the Boolean AND operation on each pair of bits of the two numbers you pass to it. Since AND returns 1 only if both terms are equal to one, the result has non-null bits only in the two rightmost positions. This is more visible if you put the binary numbers, and the result, one below the other:

```
'1111' = 15
'0011' = 3
  ^^
'0011' = still 3!
```

Another niche type that some bold Bash users dare to enter from time to time is geographic calculations. This branch of mathematics answers questions like “what is the distance between two points on Earth whose latitude and longitude are known?” And “what is the bearing (i.e., the direction to follow, with respect to geographic North) to go from one to the other?”

Short answer: It is possible to solve these problems with a Bash script (with external help from some simple programs). If you want to know how, see [10], [11], and [12]. Be warned: These methods, while adequate for simple applications, lack the same precision customary of GPS navigators!

The final niche I am only going to mention here is advanced statistical calculations. The right open source command-line tool for this job is R [13], which can be used as a standalone tool or called from inside shell scripts.

### Calling the Right Tools

Much of what Bash can do in the “math” realm is outside its direct support for arithmetic expressions. Several Bash built-in commands, as well as some little command-line programs, have lots of number-related applications. A little known example of built-in commands, which I myself only discovered after years of successfully using Bash for fun and profit, is `factor`. Unsurprisingly, this command prints the prime factors of each integer number it receives from the command line or from standard input:

```
#> factor 35 63
35: 5 7
63: 3 3 7
```

Another very handy command is `seq`, which prints sequences of numbers in a given range and in constant steps. Writing `seq 80` in a script would generate all the integer numbers from one to 80. Writing `seq 37 2 80`, instead, would return a sequence like 37, 39, 41, and so on all the way up to 79: The first parameter is the starting value, and

the second the increment to use. All parameters are interpreted as floating-point values, and you can use all the formats supported by the Bash `printf` command for the output:

```
for COUNTER in `seq -f '%5.2f' 65.3 1.5 71.7`
```

This would make `$COUNTER` cycle over the values 65.30, 66.80, 68.30, 69.80, and 71.30. Piping the output of `seq` to `sort -R` (for **R**andom), would return the same numbers, but in random order.

Two other programs you need to know to efficiently collect and filter numbers for further processing are `grep` and `cut`. Knowing about these programs is a mandatory requirement for calculations in Bash scripts, whenever the numbers to process are surrounded by larger amounts of other numbers or by raw text in general.

Typical examples are listings of file sizes and timestamps when scanning hard drives, server logs, database backup files, or even traditional spreadsheets in Comma Separated Values (CSV) plain text formats. In all these cases, you can imagine the data as one (potentially endless) table in which only certain cells (that is, certain intersections of rows and columns) contain useful numbers. You can then use `grep` and `cut` to slice all and only those cells, using `grep` to cut rows, and `cut` to extract columns (or vice-versa, of course). Another more useful tool that can be used for the same types of jobs is the `awk` utility. Let's look at a practical example.

## Database Backup Example

MySQL or MariaDB relational databases are very common inside Linux servers or desktops. Their content can be saved in plain text files, whose content can then be used for any calculation necessary. Of course, the first question that comes to mind is "why not perform those calculations *inside* the database? Aren't database engines optimized to do just that?"

The answer to this question is yes, relational databases are much faster than Bash, but they can only process their own content! If you need to do anything more than that with your files (from comparing the numbers inside the database with those inside some other file to using the same numbers to decide what to do next), you must do it outside the database. Bash is an ideal environment for such tasks. There are two other advantages of working on plain text backups instead of the actual tables in the live database. One is that you can compare at any moment backups made at different times. Another is that, starting from the same database backup file, you can generate with Bash many partial dumps to combine their content in Bash in ways that would not be possible with any direct data-

base queries. One application of this approach in small business would be finding all customers with unpaid balances over some threshold inside the database, in order to automatically send them reminders by email, each with the corresponding invoice attached. In the same way, a teacher could generate custom exercises for each student using research or financial databases, depending on previous results.

To produce output that is immediately usable for this kind of work, use:

```
mysqldump -u DBUSER --password=PASSWORD \
--skip-extended-insert > database-backup.sql
```

This is the simplest approach, because the `--skip-extended-insert` option dumps each database record on a separate line. If the database contained only one table called `purchaseorders`, composed of four columns that corresponded to the unique identifier, date, description, and amount of each purchase, the file `database-backup.sql` would contain *lots* of lines with this format:

```
INSERT INTO `purchaseorders` VALUES (
44, '2004-10-20', 'stationery', 1855.0000);
```

Knowing that the database has the above format, you may save in a separate file all the records that are about stationery purchased between 2004 and 2005 with this command:

```
egrep "'2004\|-|2005\|'" \
stationery-expenses-2004-2005.sql
```

Then extract the amounts only, as follows:

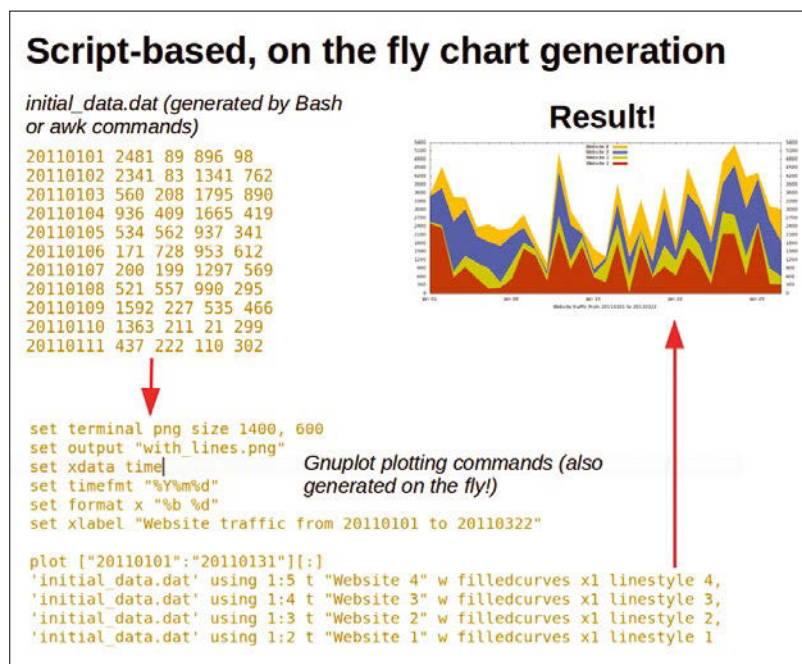
```
cut '-d, ' \
-f2,4 stationery-expenses-2004-2005.sql > \
expense-amounts.txt
```

Here, I have used the `cut` command, setting the column delimiter to the comma (`'-d, '`) and selecting only the second and fourth column (`-f2,4`) to save only dates and amounts inside the file `expense-amounts.txt`. Now, for example, if I wanted to calculate the *average* import of those stationery expenses, I would feed the second column of `expense-amount` to the `awk` program:

```
cat expense-amounts.txt | awk '-F, ' \
'{sum+=$2} END { print "Average = ",sum/NR}'
```

The `-F` switch tells `awk` to use commas as column separators instead of spaces, which are its default. The commands inside braces mean "read the file one line at a time, adding to the `sum`





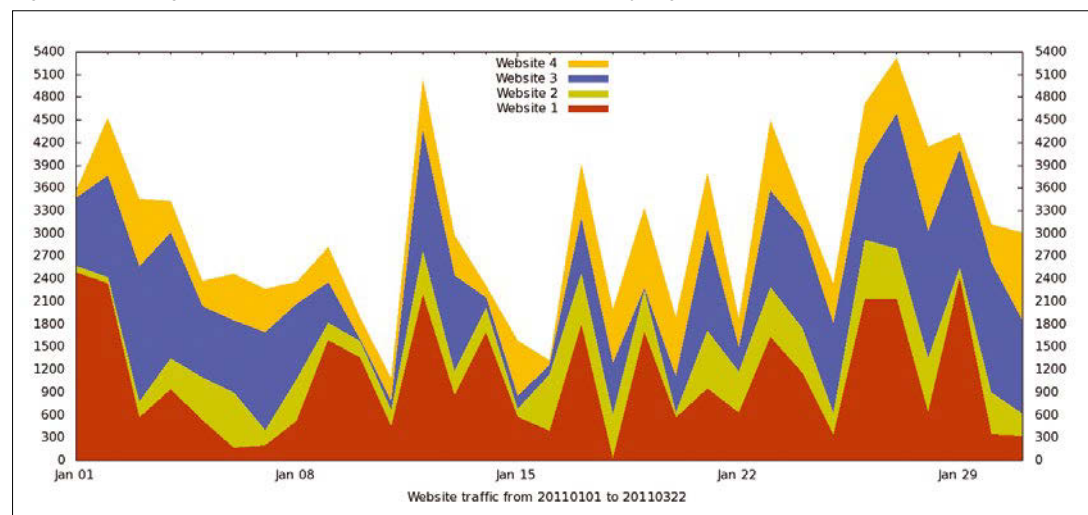
**Figure 1:** One plain text file of data and one plain text file of commands, both created on the fly with Bash code: Gnuplot could not be simpler to use or more flexible.

variable the value in the second column; when the file is finished, print the string **Average**, followed by the value of **sum** divided by the number of records (**NR**).

While the above mix of **grep**, **cut**, and **awk** is not the most efficient, I wanted to show you all three tools working together to give you a good idea of their potential for massive number crunching in shell scripts. Combining these programs, you can extract, reformat, and preprocess, with very little effort on your part, lots of numeric data of any nature. The **awk** trick above works the same way, for example, when you want to calculate the average size of all the files in the current directory, including all its subfolders. Try it for yourself:

```
find . -type f -exec ls -l {} \; | 2
awk '{sum+=$5} END { print "Average = ",sum/NR}'
```

**Figure 2:** Gnuplot graphs are still adequate for most situations and easy to generate.



## Charts

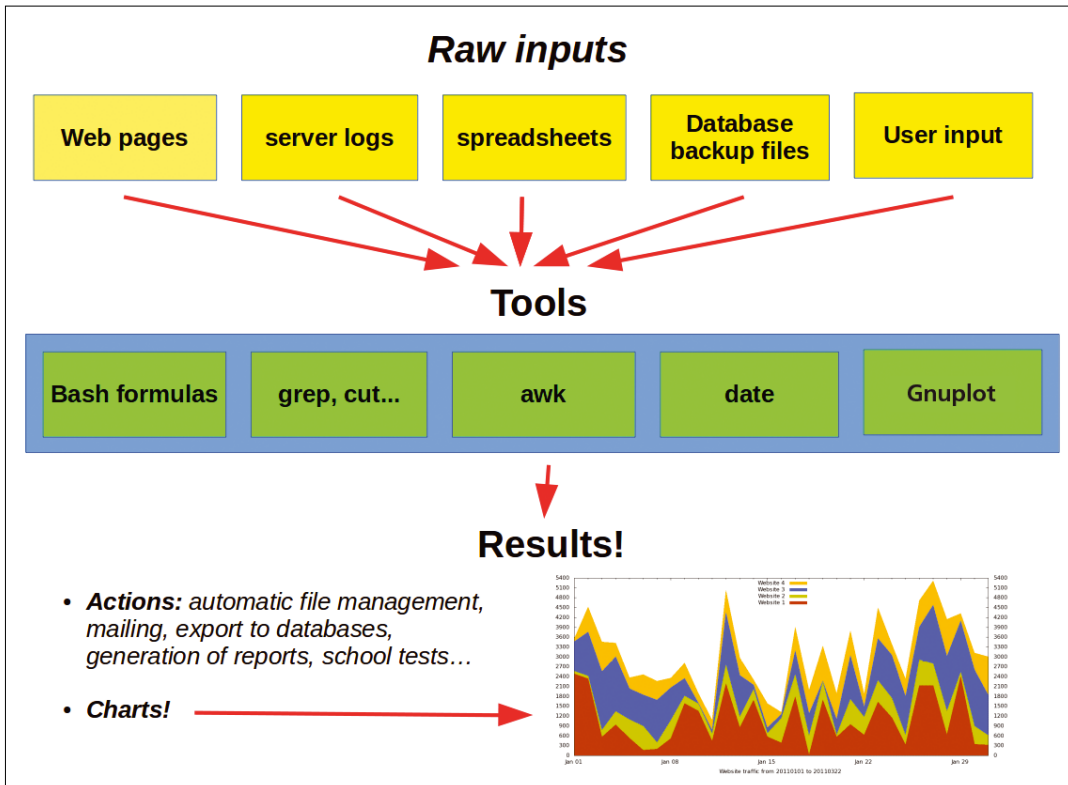
While I am not sure that a picture is always worth a thousand words, a chart is surely worth much more than a huge, unreadable table full of numbers. The tool I recommend to handle such tasks from shell scripts today is the same one I used in the 90s: the venerable Gnuplot [14]. The graphs it creates are not the fanciest, but Gnuplot is very well documented and available on almost any operating system, including Linux distributions optimized for tiny servers. Besides, using Gnuplot is really simple (see Figure 1):

- 1 Save the numbers to plot in a plain text data file.
- 2 Save the plotting commands for Gnuplot in an instruction file.
- 3 Tell Gnuplot to execute the instruction file on the data file.

Creating the data file on the fly consists of running **awk** commands similar to the one used in the database example or applying the techniques I presented in this series' previous installments. Concerning plotting commands, unless you have really uncommon needs, 90 percent of the time a quick online search will return commands ready to copy and paste into your script with little or no changes. The result will be spartan, but it will deliver clear charts like Figure 2 (to see how I generated this, see [15]) that your script can then embed in web pages, send by email, or use in any other way desired.

## The Big Picture

If all you needed to know about "math" in Bash were its arithmetic operators, a tiny cheatsheet would suffice. Since this is not the case, this installment is different from the previous ones: This time, presenting one, relatively complex script that solves *one* specific real-world problem would have been too reductive. I wanted to introduce, instead, with the minimum amount of working code, approaches and tools suitable for any kind of num-



**Figure 3:** Bash makes it easy to collect raw data from any conceivable source and pass the data to the right tools to make your computer do all your required numeric processing.

ber-related processing and show how they fit together (see Figure 3).

There are two take-away lessons from this installment. First, it is easy to insert single formulas in Bash scripts, using `bc` if necessary. Second, very often a Bash script is the most ef-

ficient way to run or prepare calculations on large quantities of numbers of any nature, from any source (including the Internet). The real power of Gnuplot, `awk`, `bc`, and so on is in how you combine them in the same shell script. Give it a try! ■■■

## Info

- [1] "Tutorials – Custom Shell Scripts" by Marco Fioretti, *Linux Magazine*, issue 219, February 2019, pp. 84-88
- [2] "Tutorials – Complex Containers" by Marco Fioretti, *Linux Magazine*, issue 220, March 2019, pp. 84-89
- [3] "Tutorials – Shell Flow Control" by Marco Fioretti, *Linux Magazine*, issue 221, April 2019, pp. 86-91
- [4] "Tutorials – Shell Test Conditions and Exit Codes" by Marco Fioretti, *Linux Magazine*, issue 222, May 2019, pp. 84-88
- [5] Genius: [www.jirka.org/genius.html](http://www.jirka.org/genius.html)
- [6] Graph: [www.padowan.dk](http://www.padowan.dk)
- [7] `bc` man page: <https://linux.die.net/man/1/bc>
- [8] Handling time differences: <https://stackoverflow.com/questions/4946785/how-to-find-the-difference-in-days-between-two-dates>
- [9] "Bash Scripting – Arithmetic, Logical, Relational and Bitwise Operators": [www.yourownlinux.com/2016/12/bash-arithmetic-logical-relational-bitwise-operators.html](http://www.yourownlinux.com/2016/12/bash-arithmetic-logical-relational-bitwise-operators.html)
- [10] "Geographical Distance Between Long and Lat in Bash": [www.unix.com/shell-programming-and-scripting/250356-geographical-distance-between-long-lat-bash.html](http://www.unix.com/shell-programming-and-scripting/250356-geographical-distance-between-long-lat-bash.html)
- [11] "Calculate Distance and Azimuth": [www.unix.com/shell-programming-and-scripting/129989-calculate-distance-azimuth.html](http://www.unix.com/shell-programming-and-scripting/129989-calculate-distance-azimuth.html)
- [12] "Work the Shell – Calculating the Distance between Two Latitude/Longitude Points" by Dave Taylor, *Linux Journal*, December 1, 2009: [www.linuxjournal.com/magazine/work-shell-calculating-distance-between-two-latitude-longitude-points](http://www.linuxjournal.com/magazine/work-shell-calculating-distance-between-two-latitude-longitude-points)
- [13] R: [www.r-project.org/](http://www.r-project.org/)
- [14] Gnuplot: [www.gnuplot.info](http://www.gnuplot.info)
- [15] "How to Create Stacked Area Graphs with Gnuplot": <http://freeware.zona-m.net/how-to-create-stacked-area-graphs-with-gnuplot/>



# Using OpenSCAD to build custom 3D pieces

## Build Your Own Body

OpenSCAD lets you use simple scripts to build 3D bodies from primitive shapes that you can then send to your 3D printer. It also lets you create custom shapes for pieces and objects. In this article, we look at two ways to do just that.

BY PAUL BROWN

In last month's installment [1], you saw how to use primitive 3D bodies (cubes, spheres, cylinders, etc.) to build complex 3D objects. Apart from combining basic primitives, there are more ways of making bodies in OpenSCAD [2]. One is to extrude them from 2D shapes; another is to build them from a bunch of vertices and faces you define in arrays.

Let's see how both of these work by building an "arm" that will let you attach a sports camera to a printer's hotbed. This arm could look something like what you can see in Figure 1. With the arm attached to the bed, you will be able to monitor

your print with the action always in the center of the frame.

You will also be able to make cool, if slightly dizzying, time-lapse photographs of the proceedings.

### Anatomy of an Arm

By studying my own printer (a Creality 3D Ender 3), I concluded that the best place to affix the arm would be to the hotbed's undercarriage. It doesn't get hot down there, so there is no danger of the plastic getting soft or melting. In addition, by affixing the arm to a corner and threading the screw used for leveling the bed through it, the arm has extra grip and doesn't fall off.

To clear the hotbed and allow the camera to focus properly, the arm would have to be relatively long, at least 10cm. It would also have to bend upwards at some point, so the camera had a good view of what was happening on the bed; otherwise, it would be too low.

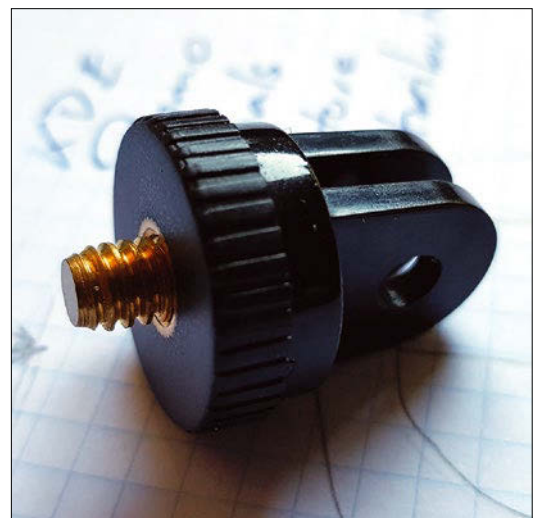
Most sports camera accessories have a pretty standard way of connecting one with another. They usually end in two or three semicircular



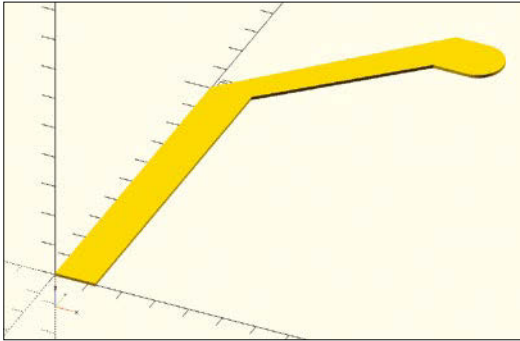
**Figure 1:** An arm for holding a sports camera helps you film your prints.



**Figure 2:** A typical array of sports camera accessories.



**Figure 3:** An accessory that allows you to connect nearly any camera to the arm.



**Figure 4:** A 2D projection of what your 3D arm will look like.

prongs (see Figure 2). You slot a piece with two prongs into a piece with three prongs, and, thanks to the fact that the prongs are curved, you can adjust the angle to your liking. Once adjusted, you use a screw-like bit (Figure 2, bottom right) to tighten the pieces to make sure they don't move.

This is how you can connect the arm to the camera, too. You can also use the piece that screws into the bottom of the camera (Figure 3) to attach several other cameras to the arm, since the screw is the standard size used for tripods.

To summarize, our arm has a forearm that is 10cm long, plus another section that bends up at a 45-degree angle. At the end, it has a semicircular section. In profile, it would look like Figure 4.

## 2D Polygons

Figure 4 shows an OpenSCAD polygon. You create polygons by using the (unsurprisingly) inbuilt `polygon()` function. In its simplest form, you pass it a list of vertices that are in the order in which you want to draw the polygon itself:

```
polygon([[0,0], [0,10], [5,15], [10,10], 2
[10,0]]);
```

The renderer then draws the polygon, in this case, a house-like shape. OpenSCAD will draw from `[0,0]` to `[0,10]`, then from `[0,10]` to `[5,15]`, and so on and will finish by closing the polygon, joining `[10,0]` with the starting point at `[0,0]`.

You can also use the `paths()` parameter for more complex polygons.

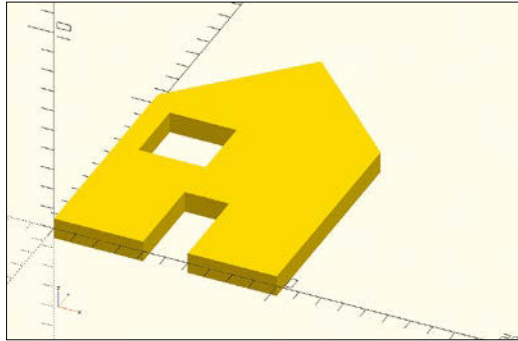
The line

```
polygon(points=[[5,15], [0,10], [10,10], 2
[0,0], [10,0]], paths=[[3, 1, 0, 2, 4]]);
```

also draws a house-like polygon, but the points are not in the given order. It is the `paths()` parameter that tells OpenSCAD in which order to draw the shape.

The usefulness of `paths()` becomes clear when you need to cut holes in your 2D object.

Consider, for example, Listing 1. You have each object (`facade`, `window`, and `door`) as an array of points (lines 1, 4, and 7). Then you have a path for each ob-



**Figure 5:** Using `polygon()`, `points()`, and `paths()`, you can make custom 2D shapes.

ject (lines 2, 5, and 8). Concatenate the array of points (line 10) and put the paths into an array of arrays (line 11). Pass everything to `polygon()` (line 13), and, hey presto, you have yourself a cute little house with a window and a door (Figure 5).

Getting back to the arm and the shape shown in Figure 4, the module shown in Listing 2 would do the job of drawing it nicely.

First you draw all the straight-sided parts of the shape (line 2) and then add a circle at the business end (lines 3 and 4). To get the exact diameter you need, you are going to have to dust off your trusty caliper (Figure 6). A regular ruler is not going to cut it for the precise measurements you need when making pieces that need to fit in with others in the physical world.

Once you have your polygon, the next step is to turn the 2D shape into an actual 3D object. You can do that with the `linear_extrude()` function.

### Listing 1: house.scad

```
01 facade=[[0,0], [0,10], [5,15], [10,10], [10,0]];
02 facade_path=[0, 1, 2, 3, 4];
03
04 window=[[1,6], [1,9], [4,9], [4,6]];
05 window_path=[5, 6, 7, 8];
06
07 door=[[4,0], [4,4], [6,4], [6,0]];
08 door_path= [9, 10, 11, 12];
09
10 house= concat (facade, window, door);
11 house_paths= [facade_path, window_path, door_path];
12
13 polygon(points=house, paths=house_paths);
```

### Listing 2: Arm base() Module

```
01 module base() {
02   polygon([[0,0], [0,100], [50,150], [60,150], [60,135], [50,135],
03     [12,100], [12,0]]);
04   translate([60, 142.5, 0]){
05     circle(d=15, $fn=50, center=true);
06   }
```





**Figure 6:** A caliper is an essential tool when designing pieces that have to fit in with other physical objects.

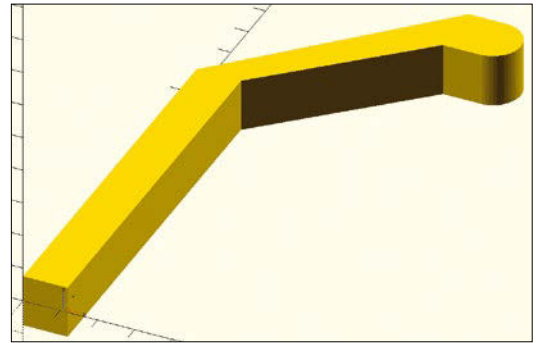
Here you are again going to need your caliper to measure the object's height. In the case of our arm, placed on its side, it is 15 units (millimeters) high. For an accurate measurement, it is a good idea to **center** the object, too. This will make it easier to place the shapes you want to subtract from it later.

Listing 3 shows the script for generating the basic body for your arm (Figure 7).

The final step is to create all the bits and pieces that you have to take away from the arm (Listing 4). You need a slot at one end, so you can fit the arm to your hotbed's undercarriage (lines 2 to 4), and a circular hole through that, so you can thread the leveling screw and spring through it (lines 6 to 10). You will also need two slots at the other end to convert the circular part into three prongs (lines 12 to 18). Again, you will need a hole through all of that, so you can insert the tightening screw and secure it on the other side with a bolt (lines 20 to 23).

Taking away `slots()` from `body()` with the inbuilt `difference()` function, you get a usable piece like the one shown in Figure 8.

For the record, I did print this piece and it worked. However, it was very weak: I could easily snap it without much effort. For a tiny sports camera, it would probably be fine. How-

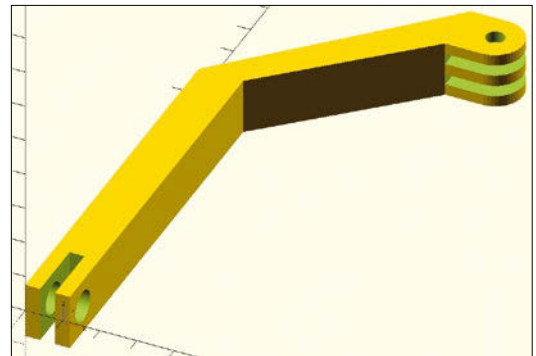


**Figure 7:** A 3D body extruded from a 2D shape.

ever, for anything heavier, it would probably break mid-print and drop the camera – not ideal.

### Custom Polyhedron

I figured I'd need some reinforcement and looked up online how others solved this problem. One



**Figure 8:** An arm you can print.

#### Listing 3: body.scad

```
01 module body() {
02   linear_extrude(height=15, center = true) {
03     base ();
04   }
05 }
06
07 module base() {
08   polygon([[0,0], [0,100], [50,150], [60,150], [60,135],
09     [50,135], [12,100], [12,0]]);
10   translate([60, 142.5, 0]){
11     circle(d=15, $fn=50, center=true);
12   }
13 }
```

#### Listing 4: slots.scad

```
01 module slots() {
02   translate([4, -5, -10]) {
03     cube([4,22,20]);
04   }
05
06   translate ([6,7.5,0]) {
07     rotate([0,90,0]) {
08       cylinder(20, d=8.5, center=true, $fn=50);
09     }
10   }
11
12   translate([60,142.5,3]) {
13     cube([20,20,3], center=true);
14   }
15
16   translate([60,142.5,-3]) {
17     cube([20,20,3], center=true);
18   }
19
20   translate([60,142.5,0]){
21     cylinder(20, d=5, center=true, $fn=50);
22   }
23 }
```

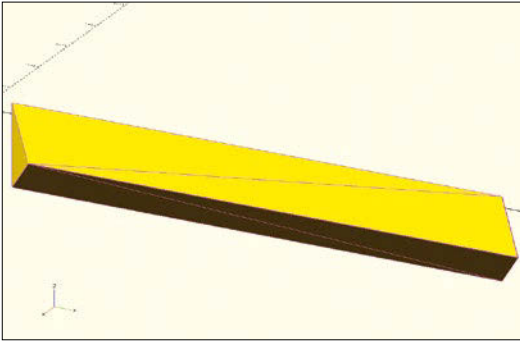


Figure 9: A triangular beam for extra strength.

way was to run a triangular beam along the arm and have it connect to the upward curving bit. Apparently, a triangular-shaped beam gives the arm extra resistance.

Each end of the beam would also have to be slanted at a 45-degree angle on one end, so as not to impede the leveling screw and spring, and on the other to meet the bend. The piece looks like Figure 9.

As this is not a cube, sphere, cylinder, or cone, you can build this piece either by painstakingly rotating and subtracting cubes from one another or by constructing your own polyhedron from scratch.

The second way is easier, and the idea is similar to what you did with the polygon: You define all the polyhedron's vertices (but this time in 3D space instead of on a flat surface), and then you define how they go together in faces.

Listing 5 shows a polyhedron that will draw the beam. Notice that the beam is already drawn in position so that, when rendered along with the rest of the piece, it will appear affixed to the body of the arm in the correct place (Figure 10).

There is one thing you have to take into account when describing your object to the renderer: You can start describing each face with any vertex you

#### Listing 5: reinforcement.scad

```
01 module reinforcement() {
02   polyhedron(
03     points= [
04       [12,11.75, -7.5],
05       [18,17.75,0],
06       [12,11.75,7.5],
07       [12,100,-7.5],
08       [18,106,0],
09       [12,100,7.5]
10     ],
11
12     faces= [
13       [0,2,1],
14       [0,1,4,3],
15       [1,2,5,4],
16       [3,4,5],
17       [0,3,5,2]
18     ]
19   );
20 }
```

want, but you then have to move clockwise after that. And “clockwise” here means “clockwise as if you were facing the object.” So, while you would run through the vertices clockwise on a face at the front or on the top of an object, you would have to run through the vertices counterclockwise for faces on the back or bottom of the object.

Say you defined a rectangular face in which the bottom left vertex was 0, the bottom right was 1, the top left was 2, and the top right was 3. If it were on the front of your object, you would tell the renderer to draw it with:

```
[0, 2, 3, 1]
```

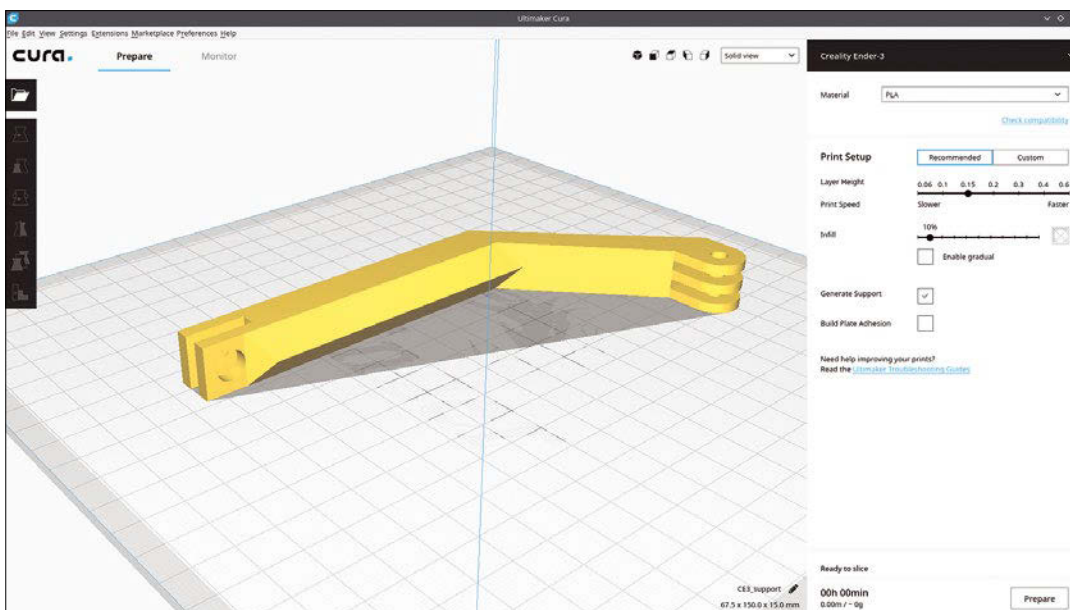


Figure 10: The new arm with its reinforcement in the Cura slicer ready to be printed.



However, if it were on the back of your object, you'd say:

```
[0, 1, 3, 2]
```

This may take some mental gymnastics to get right if you are spatially challenged like me. It does help if you draft your body on paper. Either way, if you make a mistake, you will get an error that says *WARNING: Object may not be a valid 2-manifold and may need repair!*, even if the piece seems to show up correctly in the preview window or even in the slicer.

### Final Piece

Listing 6 shows how you would bring all the bits and pieces together. You import all the modules from their files (lines 1 to 3), then subtract the slots and holes from the body (lines 5 to 8), and finally add the reinforcement beam (line 10).

Press F6 to build the piece and then use the *STL* button at the end of the editor's toolbar to export the piece to STL.

You can then use a slicer like Cura [3] (Figure 10 again) to make the object ready for your printer. Cura is available in the repositories of most Linux distributions and will open your STL file. Cura also comes with configurations for many of the most popular printers. Choose yours from the list, and it will allow you to place, rotate, and scale your object, as well as let you set the resolution and infill, and so on.

Just in case, the piece you have seen in this article does require supports, as you can see in Figure 11.

### Listing 6: support.scad

```
01 use <body.scad>;
02 use <reinforcement.scad>;
03 use <slots.scad>;
04
05 difference () {
06     body();
07     slots();
08 }
09
10 reinforcement();
```

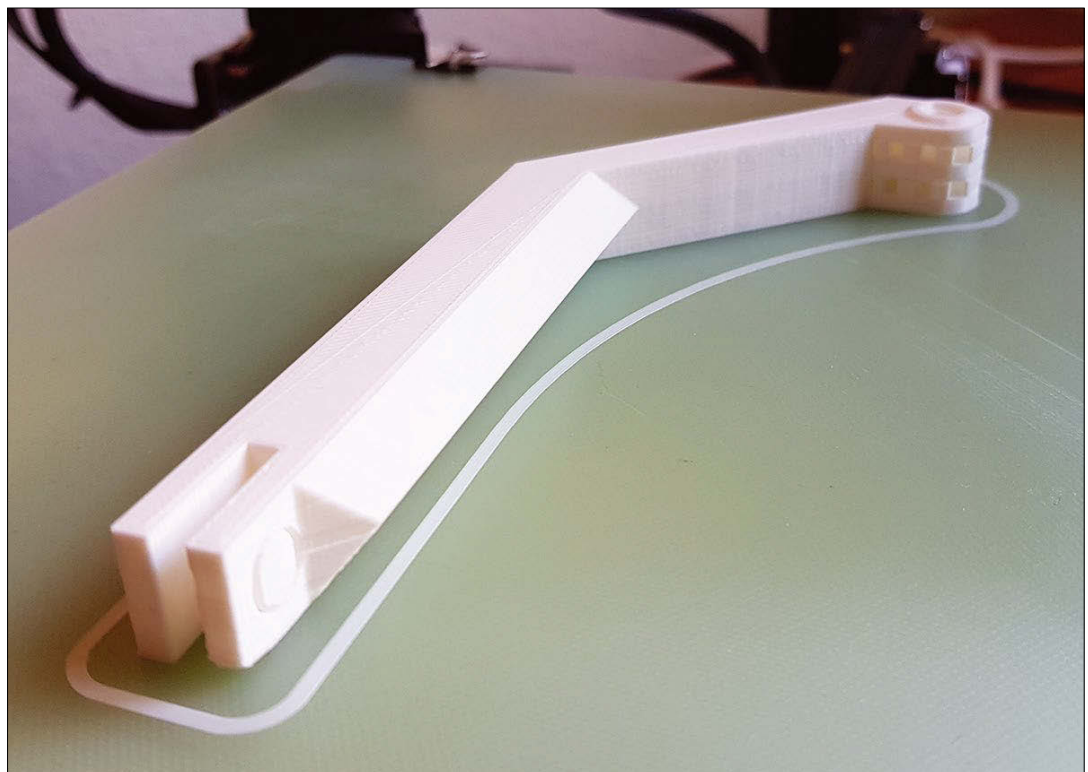
### Conclusion

Although this example is very specific, you should be able to use the object for your own printer even if it is a different model with some minor tweaks.

The more interesting point is that, with OpenSCAD, you can create pretty complex pieces that then translate into real world, useful objects. The potential for these kinds of things is huge, and OpenSCAD is a good way to reach it. ■■■

### Info

- [1] "Designing Your Own Stuff with OpenSCAD" by Paul Brown, *Linux Magazine*, issue 222, May 2019, pp. 90-94, <http://www.linux-magazine.com/Issues/2019/222/Designing-your-own-stuff-with-OpenSCAD>
- [2] OpenSCAD: <http://www.openscad.org/>
- [3] Cura: <https://ultimaker.com/en/products/ultimaker-cura-software>

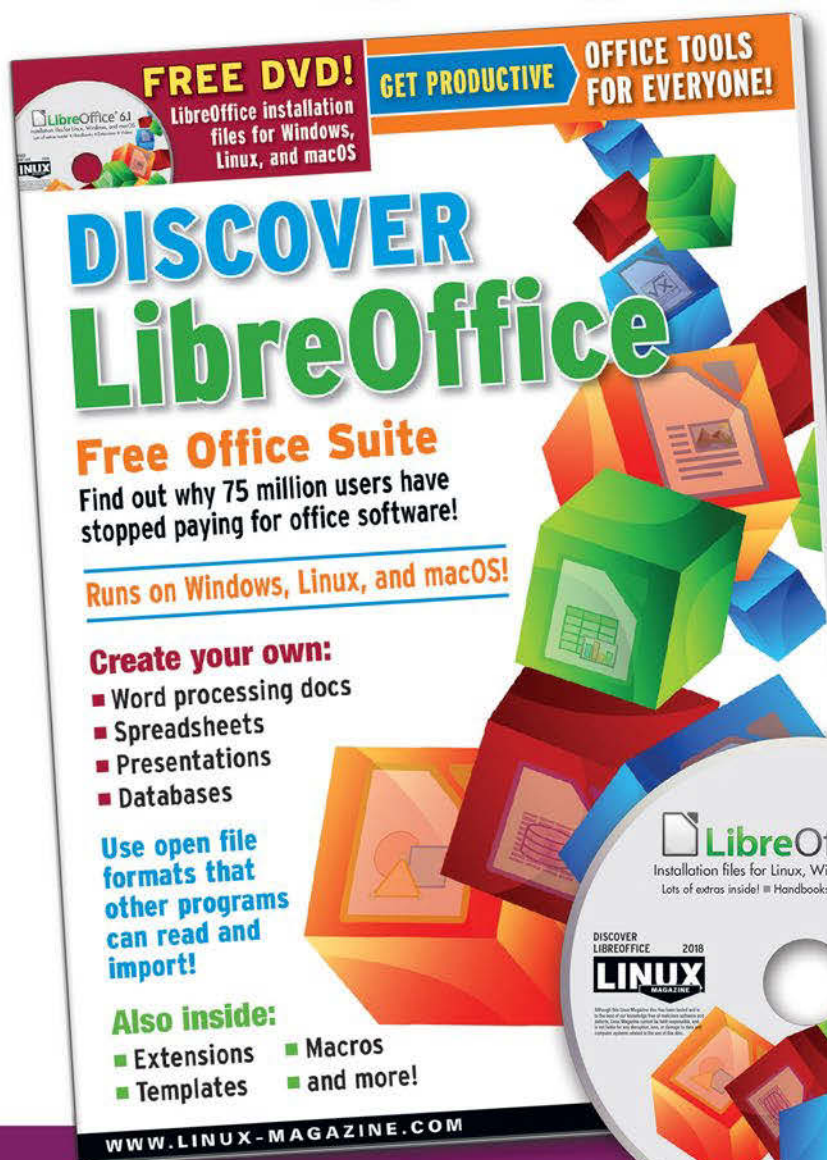


**Figure 11:** The final printed piece, still on the bed and with supports in place that need to be removed.

Shop the Shop

[shop.linuxnewmedia.com](http://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](http://shop.linuxnewmedia.com/specials)

For Windows, macOS, and Linux users!



# 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](mailto:events@linux-magazine.com).

## KubeCon + CloudNativeCon Europe 2019

**Date:** May 20-23, 2019

**Location:** Barcelona, Spain

**Website:** <https://events.linuxfoundation.org/events/kubecon-cloudnativecon-europe-2019/>

Join Kubernetes, Prometheus, Envoy, CoreDNS, OpenTracing, Fluentd, gRPC, containerd, rkt, CNI, Jaeger, Notary, TUF, Vitess, NATS, Linkerd, Helm, Harbor, and etcd at the Cloud Native Computing Foundation's flagship conference.

## 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.

## USENIX ATC '19

**Date:** July 10-12, 2019

**Location:** Renton, Washington

**Website:** <https://www.usenix.org/conference/atc19>

The 2019 USENIX Annual Technical Conference will bring together leading systems researchers for cutting-edge systems research and the opportunity to gain insight into a wealth of must-know topics.

## Events

Cloud Migration Summit	May 14	London, United Kingdom	<a href="https://www.cloudmigrationsummit.com/events/781-strategic-cloud-migration-summit">https://www.cloudmigrationsummit.com/events/781-strategic-cloud-migration-summit</a>
Open Source Data Center Conference (OSDC)	May 14-15	Berlin, Germany	<a href="https://osdc.de/">https://osdc.de/</a>
JAX DevOps Conference 2019	May 14-17	London, United Kingdom	<a href="https://devops.jaxlondon.com/">https://devops.jaxlondon.com/</a>
OSCamp Ansible	May 16	Berlin, Germany	<a href="https://opensourcecamp.de/">https://opensourcecamp.de/</a>
Linux Presentation Day 2019	May 18	Cities across Europe	<a href="http://www.l-p-d.org/">http://www.l-p-d.org/</a>
Open Source Conference Albania (OSCAL '19)	May 18-19	Tirana, Albania	<a href="https://oscal.openlabs.cc/">https://oscal.openlabs.cc/</a>
Cephcon Barcelona	May 19-20	Barcelona, Spain	<a href="https://ceph.com/cephcon/barcelona-2019/">https://ceph.com/cephcon/barcelona-2019/</a>
Kubecon + Cloud Native Con Europe 2019	May 20-23	Barcelona, Spain	<a href="https://events.linuxfoundation.org/events/kubecon-cloudnativecon-europe-2019/">https://events.linuxfoundation.org/events/kubecon-cloudnativecon-europe-2019/</a>
openSUSE Conference 2019	May 24-26	Nuremberg, Germany	<a href="https://events.opensuse.org/conferences/oSC19">https://events.opensuse.org/conferences/oSC19</a>
Secure Linux Administration Conference (SLAC)	May 27-29	Berlin, Germany	<a href="https://www.heinlein-support.de/secure-linux-administration-conference">https://www.heinlein-support.de/secure-linux-administration-conference</a>
ISC High Performance 2019	June 16-20	Frankfurt, Germany	<a href="https://www.isc-hpc.com/">https://www.isc-hpc.com/</a>
OpenExpo Europe 2019	June 20	Madrid, Spain	<a href="https://openexpo-europe.com/">https://openexpo-europe.com/</a>
KubeCon + CloudNativeCon China 2019	June 24-26	Shanghai, China	<a href="https://www.lfasiallc.com/events/kubecon-cloud-nativecon-china-2019/">https://www.lfasiallc.com/events/kubecon-cloud-nativecon-china-2019/</a>
HotStorage '19	July 8-9	Renton, Washington	<a href="https://www.usenix.org/conference/hotstorage19">https://www.usenix.org/conference/hotstorage19</a>
USENIX ATC '19	July 10-12	Renton, Washington	<a href="https://www.usenix.org/conference/atc19">https://www.usenix.org/conference/atc19</a>
Open Source Summit Japan	July 17-19	Tokyo, Japan	<a href="https://events.linuxfoundation.org/events/open-source-summit-japan-2019/">https://events.linuxfoundation.org/events/open-source-summit-japan-2019/</a>
Debconf 2019	July 21-28	Curitiba, Brazil	<a href="https://wiki.debian.org/DebConf/19">https://wiki.debian.org/DebConf/19</a>

# 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](mailto: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](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.

## Contact Info

### Editor in Chief

Joe Casad, [jcasad@linux-magazine.com](mailto:jcasad@linux-magazine.com)

### Copy Editor

Amy Pettie

### 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

© artqu, 123RF.com

### Advertising

Brian Osborn, [bosborn@linuxnewmedia.com](mailto:bosborn@linuxnewmedia.com)  
phone +49 89 3090 5128

### Marketing Communications

Gwen Clark, [gclark@linuxnewmedia.com](mailto: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](mailto:cs@linuxpromagazine.com)  
Phone: 1-866-247-2802  
(Toll Free from the US and Canada)

For all other countries:  
Email: [subs@linux-magazine.com](mailto:subs@linux-magazine.com)

[www.linuxpromagazine.com](http://www.linuxpromagazine.com) – North America

[www.linux-magazine.com](http://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.

## Authors

Bernhard Bablok	40, 74
Erik Bärwaldt	30, 42
Swapnil Bhartiya	8
Paul Brown	90
Zack Brown	11
Bruce Byfield	38, 46, 58
Mark Crutch	67
Markus Feilner	18
Marco Fioretti	84
Jon "maddog" Hall	69
Vasant Kanchan	34
Kristian Kißling	24
Charly Kühnast	45
Vincent Mealing	67
Pete Metcalfe	62
Graham Morrison	78
Brian Osborn	14
Mike Schilli	54
Tim Schürmann	24, 50, 70



Issue 224 / July 2019

# Energy Consumption

Looking for ways to reduce your carbon footprint?  
Next month we explore some expert techniques for  
evaluating the energy efficiency of Linux software.

## Approximate

UK / Europe	Jun 01
USA / Canada	Jun 28
Australia	Jul 29

## On Sale Date

## 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](http://www.linux-magazine.com/newsletter)

# ISC HIGH PERFORMANCE 2019

FUELING  
INNOVATION

- 450 SPEAKERS
- 160 EXHIBITORS
- 3500 ATTENDEES

Come find out how high performance computing is fueling innovation in science, engineering and commerce.



**Early bird rates until May 8**



Platinum Sponsors:



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



# SUPERMICRO

## Better. Faster. Greener.

Expect Better Data Center Performance, TCO & Impact on the Environment



35% Faster | Up to 50% TCO Reduction | Reduce E-Waste  
Featuring 2<sup>nd</sup> Generation Intel® Xeon® Scalable Processors



Learn More at [www.supermicro.com/X11](http://www.supermicro.com/X11)

© Supermicro and Supermicro logo are trademarks of Super Micro Computer, Inc. in the U.S. and/or other countries.

