



THE BEST LINUX APP STORES

Delivering hot open source software direct to your desktop!



Run full-fat Linux on the Valve Steam Deck

LINUX FORMAT

The #1 open source mag

BULLET-PROOF UBUNTU 22.04

Enter the Linux multiverse with a rock-solid PC install today!

- > Five-year long-term support
- > Improved network shares
- > Latest hardware updates
- > All-new Wayland display



PLUS: HOW TO

- Quickly clean up your cluttered hard drive
- Run your own email server the easy way
- Emulate the classic Z80 Amstrad PCW

FULL HOUSE!

Start coding your own card games

RUST TOOLS

Enhance your skills with filesystem access

TERMINAL TIPS

Manage your ebooks from the terminal



**THE
BRAIN
TUMOUR
CHARITY**

A CURE CAN'T WAIT

**BRAIN TUMOURS
MOVE FAST.**

WITH YOUR HELP,

WE CAN TOO!

We're working to create a future where brain tumours are curable.
We urgently need your help to accelerate research.

Text DEFEAT5 to 70507 to donate £5, please help us to find a cure.

thebraintumourcharity.org

© The Brain Tumour Charity 2020. Registered Charity in England and Wales
(1150054) and Scotland (SC045081)

 Registered with
**FUNDRAISING
REGULATOR**

**LINUX
FORMAT**

» MEET THE TEAM

With this issue's focus on the latest LTS release of Ubuntu, will you be upgrading to version 22.04 Jammy Jellyfish? If not, what are you installing instead?

**Jonni Bidwell**

I'll be keeping my Jammy Jellyfish installed. It's not yet stung me. But will it become my new daily driver? I'll reserve judgement till the next Pop!_OS LTS comes out, I suppose. Plus there will surely be a plethora of other great Jellyfish-spawn distros with which to burden my bootloader.

**Nick Peers**

I'll be installing Ubuntu 22.04 on all my machines, but not all at once. My desktop will be upgraded first, and I'll hold off updating my server until some of the early bugs get ironed out. The thought of being unable to access my

NickFlix media server sends shivers down my spine...

**Les Pounder**

I'll upgrade my laptop to Kubunto 22.04. When I first started with Linux I enjoyed KDE, then I spent some time with Gnome, Openbox and Regolith. However, for 22.04 I'll be moving back to KDE because it's been fantastic both

on my desktop and on a Lenovo ThinkCentre.

**Michael Reed**

Interesting that you should ask, as I fancy a change at the moment, but I'm undecided. I've got years of experience invested into Ubuntu, but I like what I've seen of Mint with Cinnamon as the desktop. Another route would be to start with Debian and then customise from there.

**Mayank Sharma**

I wouldn't think of letting LTS releases anywhere near my desktops. I used regular Ubuntu releases for the longest time, but I've been a Fedora man for the past few years.

Fedora does a bang-up job of running a stable desktop, and has a wonderful update policy that suits me.

Jammy upgrades



It's LTS time again! If you're down with the Linux lingo then you'll already know what this means, but to the rest of the world it's a Long Term Support release and for Ubuntu this comes around every two years, with 20.04 being the last such release.

Why is any of this important? An LTS freezes development and provides a stable and secure install that Canonical – the company behind Ubuntu – guarantees will

receive five years of security and kernel updates, with potentially a further three years (and perhaps as long as five) of Extended Support Maintenance, although you need to register for this.

It offers peace of mind for those wanting a quiet time of it or more likely admins who prefer stable, secure and predictable systems to deal with. While non-LTS releases tend to be used to test up-and-coming technology, the LTS releases are when that tech is ready for the prime time. With 22.04 LTS we're finally seeing the new graphical stack Wayland implemented with crazy essentials such as remote desktop and screen grabbing working out of the box. But I won't spoil all of the surprises here – that's Jonni's job, and he takes you through everything on page 34.

We're also having fun this issue because we had a Valve Steam Deck to play with, but rather than use it for its intended purposes we ran Desktop Linux instead. What else were we going to do, play games?! There's also a fun bag of system tools, VM management, Wine tips and classic retro emulation, so as always enjoy!

Neil

Neil Mohr Editor
neil.mohr@futurenet.com



Subscribe & save!

On digital and print
– see p18

Contents



REVIEWS

Seagate EXOS 20TB HDD 20

Mark Pickavance is stunned by the storage but not the endurance of this drive. The cost is another matter altogether, mind...



HostGator 21

A basic website hosting option for small and medium businesses that Shashank Sharma thinks should be on your short list.

Linux Mint Debian Edition 5 22

Whenever there's a new LMDE release, Mayank Sharma can't help but think of the old adage: "It's the thought that counts."

Fedora 36 23

Mayank Sharma calls the latest Fedora release just as boring as its predecessors. But is that necessarily a bad thing?

Ubuntu 22.04 LTS 24

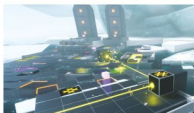
If there's one thing Mayank Sharma likes even less than Ubuntu, it's Ubuntu LTS releases, which are stable but not notable.

A Musical Story 26

Fun, folk and funk are three things that Management hate with a passion, so Tom Sykes has turned down his speakers.

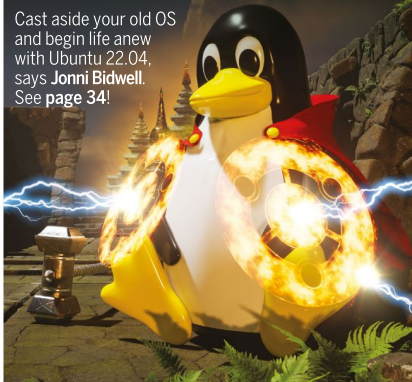
The Last Cube 27

Management isn't keen on sentient anything – "it's bad for productivity" – so Neil Mohr keeps the thinking to a minimum, which isn't helping here at all...



BULLET-PROOF UBUNTU 22.04

Cast aside your old OS and begin life anew with Ubuntu 22.04, says Jonni Bidwell. See page 34!



ROUNDUP



Open source app stores 28

Many Linux flavours and open source OSes have their own software stores. Alexander Tolstoy wonders if it's worth your time browsing their wares.

TOP OF THE FOSS



Get packing! 48

Are you keen to give back to the FOSS community, but don't know where to start? Mike McCallister shows you one way to do a good deed.

Pi USER

Raspberry Pi news 43

Introduced by *Linux Format*'s own **Matthew Holder**. We take a look at the latest moves to manage the Pi supply, a new arm is in town but this one isn't a processor, and Arduino takes on the Pi Compute.

Armbian 22.02 Jammy XFCE 44

Les Pounder assesses a distro that supports 64 different Linux single board computers, and now he has to buy them all.

Arducam 16MP camera module 45

Quick-off-the-draw **Les Pounder** can't shoot shots faster than this camera, which is now available to everyone.

Python-based reaction game 46

Les Pounder goes back to the early days of the Raspberry Pi to look at a board that made a big difference to his career.



CODING ACADEMY

File system tools in Rust 90

Mihalis Tsoukalos explains how to manipulate and examine files and directories in Rust, enabling you to write filesystem tools.

Updating old blackjack code 94

Updating old projects can be fun and educational. **Andrew Smith** ensures that your cards are dealt correctly at any resolution.



REGULARS AT A GLANCE

News 6

How the open source community is reacting to the Ukraine war, a new RISC-V computer, real Apple M1 support, a rolling Ubuntu release and Fedora is to be 32-bit no more.

Kernel watch 10

Answers 12

Backing up and syncing Keepass, trying to backup PhotoRec files, controlling screen mirroring to a TV, sorting out-of-tree module support, plus a look at the tmpfs system.

Mailserver 16

Demands to do more router coverage, demands that we should be nicer to Bohdi, demands to be nicer to new users and demands to appreciate Android!

Subscriptions 18

Get your monthly Linux dose and save cash!

Back issues 62

Get hold of previous *Linux Format* editions.

Overseas subscriptions 63

Get *Linux Format* shipped around the globe.

HotPicks 83

Alexander Tolstoy doesn't have to worry about silly old Russian jokes any longer, things aren't really funny any more. But we still have a fine open source selection including: *FireDM*, *Mplayer*, *Rnote*, *Mailspring*, *Airsane*, *Fbcat*, *Topgrade*, *Superdub*, *Winterapples*, *Casper-fs* and *Zplora*.

Next month 98

TUTORIALS

TERMINAL: Reading ebooks 52

Shashank Sharma shows how the terminal can serve as the pathway to the myriad worlds of book-based storytelling.

CZKAWKA: Clean your drives 54

Nick Peers takes a deep dive into this fast-evolving and brilliant tool for finding and removing redundant files from your PC.

EMULATION: Amstrad PCW 58

Les Pounder goes back to school, a time when his form room was full of Z80 computers and noisy dot matrix printers.



ASSISTANT: Home automation 64

Explore NFC tags, energy monitoring and the Wireguard VPN to expand your smart home with **Matthew Holder**.

WINE: Run Windows software 68

Michael Reed teaches you everything you need to know about using Microsoft Windows compatibility system *WINE* with some handy tips..

LXF SERVER: Email 72

David Rutland does the impossible and sets up a VPS-based email server and a webmail front-end, then writes about it.

MULTIPASS: Easy virtual machines 76

Stuart Burns helps you to master Multipass, the virtualisation platform that runs on any desktop system.

IN-DEPTH



All hands on the Deck! 78

Jonni Bidwell pries a Steam Deck from PC Gamer's cold anthropomorphised hands and gets his game on.

Newsdesk

THIS ISSUE: FOSS community on Russian war » VisionFive V1 SBC
» Linux on Apple's M1 » Rolling Rhino » Fedora sidelines 32-bit apps

CURRENT AFFAIRS

FOSS community stands up to Russian atrocities

Russia's invasion of Ukraine has sent shockwaves around the world, and it's keenly felt in the open source community, too.

Since last issue, more of the open source community has spoken out to condemn Russia's invasion of Ukraine and show support for Ukrainians. This includes Scarf (<https://about.scarf.sh>), an open-source gateway that helps distribute software and offers analytics for companies.

In a recent blog post (<https://bit.ly/lxf289scarf>), Scarf's co-founder and CEO Avi Press announced that Scarf will block package downloads from the Russian Government. Avi had discovered that "Scarf has fulfilled ongoing software download requests from at least 17 distinct sources that have been confirmed to originate from the Russian Government," and that the "notion of Russian government cyber attack operations leveraging software downloaded through Scarf's platform is unacceptable." Because of this, Scarf will block all "package and container downloads originating from Russian government sources." Avi also urged other open-source projects, companies and individuals to do the same where they can.

While many businesses have stopped their proprietary and closed-source products and services from being used in Russia or by the Russian government, it's a trickier situation when it comes to open-source products. By their very nature these are made available to everyone, anywhere – one aspect are licences such as the GPL that permit military use.

GitHub and GitLab, have resisted calls to limit access for Russians. In a debate on GitHub's forums (<https://bit.ly/lxf289githubforum>), a

staff member stated that "GitHub's vision is to be the home for all developers, no matter where they reside," and it would instead comply with "stringent new export controls aimed at severely restricting Russia's access to technologies and other items needed to sustain its aggressive military capabilities" instead.

Red Hat, however, has announced (<https://red.ht/3u9MaR7>) that it's ceased sales and services in Russia and Belarus, and has discontinued partner relationships with organisations based in those two countries. To its credit, the company has also gone further, by organising buses that have "safely transported several dozen of our Ukrainian associates' family members across the border to Poland," and the company promises to "continue to help those who remain in the country in any way possible."

Canonical followed suit termination all support, professional services, and channel partnerships with Russian enterprises.

An unethical response from the authors of node-ipc, was releasing malware that erases your hard drive if you have a Russian or Belorussian IP address, but has apparently hit aid workers in the region. This war has certainly prompted difficult decisions by the open-source community, and we hope it ends swiftly. Slava Ukraini!



Many open-source institutions are having to decide how to react to Russia's invasion of Ukraine.

OPEN SOURCE SANCTIONS

"Red Hat has announced that it's ceased sales and services in Russia and Belarus."

HARDWARE

VisionFive V1 RISC-V SBC on sale

After the cancellation of the BeagleV StarFive, a new SBC based on a RISC-V processor is now on sale.

For anyone who was upset when the BeagleV StarFive RISC-V SBC (single board computer) was cancelled a few years ago, there's some good news. StarFive has essentially resurrected it, with a collaboration with Radxa repurposing the JH7100 dual-core 64-bit RISC-V processor for a brand new SBC: the VisionFive V1.

It features similar specs as the previous SBC, although the VisionFive V1 increases the RAM from 4GB to 8GB, and also supports USB PD and Quick Charge with its USB-C power port. There's some interesting specifications for this SBC, and a starter kit went on sale for \$179 (around £140) from <https://bit.ly/lxf289visionfive>, although at the time of writing the starter kit has sold out, and you can't buy the board on its own either, because that's also sold out, too.

However, the Advanced Kit is still available for \$199 (around £150), which comes with a 32GB SD card with Fedora preinstalled (which also comes with the Starter Kit), as well as a power supply and cable, 40-pin header extension and an acrylic case. This SBC can also run other Linux distros, and is compatible with the U-Boot and GRUB2 bootloaders. As Electronics-Lab points out (at <https://bit.ly/lxf289e-lab>) while the specs (which can be seen on the product

page linked above), seem good, there are some odd choices as well. The bump in RAM is welcome, but the LPDDR4 memory is divided into two 4GB units clocked at 2,800MHz, which will offer poorer performance than if it were 8GB of fully integrated RAM.

The release announcement (at the end of last year (which you can read at <https://bit.ly/lxf289visionfiveannounce>) certainly promises big things, pointing out that the Neural Network Engine, NVLDA Engine and DSP of the device makes it a "powerful piece of hardware for human-machine interface, smart home tech, surveillance, NAS, and even multimedia applications." It goes on to highlight the board's "H.264/H.265 video decoder supporting up to 4Kp60 and dual-stream decoding up to 4Kp30."



The VisionFive V1 is a single board PC based on a RISC-V processor.

HARDWARE

Linux comes to the Apple M1

The first Asahi Linux Alpha release is very promising.

Apple's M1 processor is one of the most interesting products the company has made in recent times, and the Asahi Linux project (<https://asahilinux.org>) brings a customised remix of Arch Linux ARM with the Plasma desktop to M1-powered Macs and MacBooks.

As the release notes reveal (<https://bit.ly/lxf289asahi>), Mac and MacBooks that use the M1 chip, or the more powerful M1 Pro and M1 Max chips are supported. However there's no mention of the brand-new M1 Ultra chip (which is essentially two M1 Max chips connected via a low-latency connection), and the announcement says that the new Mac Studio, which is currently the only product to use the M1 Ultra, is not currently supported.

The ability to run Linux on these new Mac devices is exciting because the ARM-based M1 chips offer excellent performance and battery life. However, it's still early days, and while there's a growing number of features that now work, including Wi-Fi, screen and built-in keyboards, there's also a huge amount that doesn't work, including DisplayPort (when do they ever work?—ED) connections, Bluetooth, GPU acceleration and the webcam. There are also plenty of bugs, so this is best left for testers at the moment.

The wait for a more stable release could prove frustrating, however. A post on the r/linux subreddit (<https://bit.ly/lxf289reddit>) shows how compiling code is around twice as fast on an M1 Mac running Asahi Linux compared to the same Mac running macOS.

OPINION

FILESYSTEM MONITORING



Gabriel Krisman Bertazio is a senior software engineer at Collabora.

While filesystems developers do their best to avoid corruption, it's impossible to completely protect a system from accidental issues. Whether they're caused by random bit flips, disk crashes or software bugs, users don't enjoy losing their data for no reason. This is why filesystem developers put a huge effort in not only testing their code, but also in developing recovery tools. In fact, all persistent filesystems deployed in production are accompanied by some support infrastructure.

When an error happens, administrators and recovery daemons must be notified ASAP so they can begin emergency recovery procedures, like recover from backups, rebuild RAID's, replace disks or run *fsck*. When one needs to watch over a large quantity of machines, like in a cloud provider with hundreds of machines, a reliable monitoring tool is essential.

This is why we worked on a new mechanism based on Fanotify for closely monitoring volumes and trigger warnings in real-time when an error occurred. The feature, merged in kernel 5.16, won't prevent failures from happening, but will reduce their impact by ensuring any listener receives the message. ”

OPINION

TOO
CLEVER

Keith Edmunds

is MD of Tiger Computing Ltd, which provides support for businesses using Linux.

“We were once asked to look at a Linux firewall that was in place at a Premier League football club. The firewall wouldn't boot, so no one had any internet access, and that's why they called us.

As is often the way, the firewall has been put in place by “somebody who has left.” That somebody had set up the firewall to PXE boot – that is, to download its bootable image from somewhere else, which in this case was a file server.

My first thought was, “Why on earth would you set up a firewall to PXE boot?” And hot on the heels of this question was the answer: because it's clever.

Never mind that the firewall now depended upon another, unrelated, server to be able to boot. Look how clever I am! No disk needed in the firewall! (Although it had one anyway: it just didn't use it.) Oh yes, I thought, this a clever solution... right up to the point when it stopped working. Technology is complex enough without adding to the equation.

People who work with Linux tend to love tech. That's fine, but let's make sure that we keep the solutions we build as simple as possible. That's what's really clever. 

DISTROS

Rolling Rhino announced



Rolling Rhino is an intriguing remix of Ubuntu that turns the distro into a rolling-release.

A new community remix of Ubuntu turns the popular distro into a rolling-release project.

A new community remix of Ubuntu has been announced (<https://bit.ly/lxf289rollingrhino>). Unlike some more straightforward remixes, Rolling Rhino essentially turns Ubuntu into a rolling release distro such as Arch Linux and OpenSUSE Tumbleweed, using a script to install Ubuntu's daily images from the devel branch.

It's certainly an interesting new way of getting Ubuntu, and the brave souls trying it out will be receiving cutting-edge features and bug fixes.

However, caution is advised because Ubuntu isn't designed as a rolling-release, and Rolling Rhino relies on testing versions of the distro, which can come out at random times, and can also introduce bugs and issues.

Support may also be hard to get hold of due to the niche nature of this release, so we recommend that this is only tried out by anyone who's confident about what they're doing, and has methods to roll back to a working release if something does go wrong.

SOFTWARE

Fedora drops 32-bit app support

Others are likely to follow the mainstream distro's actions.

While support of 32-bit hardware has been abandoned by many major distros, none have stopped supporting 32-bit applications, until now. While Fedora stopped supporting 32-bit PCs in 2019, developers are now being asked to stop building i686 versions of packages that don't have any dependencies – also known as “leaf packages”.

As a wiki entry (<https://bit.ly/lxf289fedora>) explains, as long as the packages aren't depended on by other i686 packages, developers are asked not to put in “significant investment of time or resources” to support i686 architecture, and this will no longer be considered a breaking change. This could cause issues for proprietary software such as Steam, which still features 32-bit games. We'll be

keeping an eye on the community response to this.



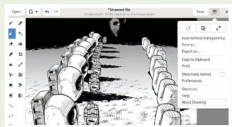
Fedora is paving the way to drop support for 32-bit applications, and other distros are sure to follow.

SOFTWARE

Drawing 1.0.0 released

The open-source MS Paint alternative hits a milestone.

Drawing is a simple-to-use art tool that's considered to be a great open-source alternative to Microsoft's iconic MS Paint, and it's now hit a major milestone with version 1.0.0 released. In a Reddit post celebrating the release (<https://bit.ly/lxf289drawing>), rendering performance has been improved (although as the post explains, editing large images using integrated graphics of some CPUs is “still kinda bad but, huh, less than before.”) New keyboard shortcuts have been added, and the Scale tool has also been improved, as has zooming. Download the latest version at <https://bit.ly/lxf289drawingdownload>.



Drawing 1.0.0 is a great tool for simple and easy image creation and editing.

Distro watch

What's down the side of the free software sofa?

Q4OS 4.8

This Debian-based distro has a new version out, and is now based on Debian 11.3. This includes the latest stable kernel, along with security and bug fixes. The installation process has also been upgraded to offer a more polished experience when you're setting up the distro. As the release announcement states (see <https://bit.ly/lxf289q4os>), language and localisation support has been improved in the API library for various tools as well.



Q4OS 4.8 is available to download with a much-improved installation process.

DEEPIIN 20.5

Another Debian-based distro has a new release out, with Deepin 20.5 bringing in a number of new features that were suggested by user feedback. Perhaps the most notable addition is face recognition, which enables you to securely log into Deepin by glancing at your webcam. The kernel has also been updated to version 5.15.24, and numerous vulnerabilities have been fixed. To find out more about this new release, see the release announcement at <https://bit.ly/lxf289deepin>.



Biometric logins are all the rage, and now deepin 20.5 makes it possible for you to sign in using your face.

FINNIX 124

To celebrate its 22nd anniversary, this compact Live CD distro has a new version out, which includes several fixes, alongside new packages and features. You can find nearby Wi-Fi hotspots with the *wifi-connect* helper utility, and RISC-V support has been added (unofficially, however, because AMD64 is the only officially supported architecture). It's well worth checking out the entire release announcement at <https://bit.ly/lxf289finnix> for more information about this release.



Finnix is now in its 22nd year, having first been released in March 2000.

PARROT 5.0

This Debian-based distro with a focus on security has a major new update out, which the team claims makes "the system extremely stable and flexible." It now follows a Long Term Support release model, yet with backporting ensures that its tools are all kept up to date. A new Architect edition has also been introduced, and the aim of this version is to offer a wide range of customisability. Head over to the project's download page at <https://bit.ly/lxf289parrot> to get hold of the latest version.



Parrot is a security-based distro that offers a range of privacy and testing tools.

OPINION

EVENTS ARE BACK, BABY!



Matt Yonkovit is the head of open source strategy at Percona

By the time you read this, I'll have finished my first in-person conference since the pandemic. Themed around the PostgreSQL open source database and taking place in San Jose, the Postgres Conference brought together people who wanted to know the latest happenings around data and how to manage it.

So what did we learn from this event? Well, there was a range of attitudes. For some, this was their first event out in two years, and they were apprehensive. For others, this was the latest chance to get out and meet up with people, chat through problems, and learn more. Wherever people were in this range, the event was a great opportunity to share in the world of open source and the community.

At the same time, we couldn't leave behind all the good things that we've picked up from two years of remote events. During the conference we ran live streams and online events to share the knowledge and bring a flavour of the event to those that couldn't attend in person.

We'll be using what we learnt in our next open source conference in Austin, Texas, in May. If you're interested in open source databases, maybe we'll see you there.

OPINION

A.OUT WITH THE OLD



Jon Masters has been involved with Linux for more than 22 years.

The end may finally be nearing for the venerable “a.out” file format. With the removal of support from the last two architectures that still implemented it, it’s just a matter of time (perhaps next release). There aren’t many users of “a.out” left out there and readers would be forgiven for wondering what this is. Several decades ago, before there was ELF (Executable and Linkable Format), there was “a.out” and it was the original way Linux handled binaries.

The year was 1995. I was 13 years old and had just been accepted into a local university part-time while still in secondary school. At the same time, the great Linux migration from “a.out” to ELF was underway. And it was messy. It was also necessary. “a.out” statically linked everything (including “shared” libraries) at a specific virtual memory address, requiring every library on a machine to be precompiled into its own location, and the maintenance of central registries of known locations so as to not break applications. It was flimsy.

There was a better way (ELF), but it required an incompatible “flag day” break with the past. The early Linux distros (Debian, Slackware, Red Hat Linux, SuSE, etc.) migrated as part of an upgrade from one GNU C library (libc4) to another (libc5). Everything needed to be upgraded at once – almost akin to swapping out your entire disk/filesystem – otherwise everything would break at once. It was a very painful experience for those who lived it and we still remember it today.

Fortunately, “a.out” is more of an historical relic, soon to be museum material.

KERNEL WATCH

Jon Masters summarises the latest happenings in the Linux kernel, because someone has to...

Linus Torvalds announced the release of Linux 5.17, saying “we had an extra week at the end of this release cycle, and I’m happy to report that it was very calm indeed.” The latest kernel includes many new features, such as support for “Compile Once, Run Everywhere” or “CO-RE” BPF programs that enable developers to ship BPF (Berkeley Packet Filter, a kind of JITable small program that’s loaded directly into the kernel on behalf of a user) portably without recompiling for each target machine’s kernel.

Another new feature is support for a new AMD P-State driver, replacing the legacy ACPI P-State driver with an ACPI CPPC-enabled one instead. The upshot of this is that those with recent Zen CPUs should see improved power and performance management, with the kernel able to more precisely communicate to the platform the desired level of responsiveness to particular workloads. In turn, the underlying Zen platform is more able to communicate its own available capabilities to the operating system, similarly to the intel_pstate driver.

Linux 5.18 merge window

With the release of 5.17 came the opening of the “merge window” for new features that will

land in the upcoming 5.18 release. At the time of writing, the merge window is closed and Linux has posted several release candidates. These include a number of typical “big ticket” items, as well as smaller internal changes.

The kernel random-number generator has been overhauled by RNG (Random Number Generator) maintainer Jason Donenfeld. The first of these is that there is now no difference between `/dev/random` and `/dev/urandom`. The second is a new mechanism to handle the case of duplicated (at runtime) virtual machines, enabling entropy to be injected in such a way that two identical virtual machines nonetheless generate different random numbers.

This might happen, for example, if the duplication were used as a means to quickly start up VMs. Each were then personalised independent of the other and expected to run different workloads, anticipating unique randomness for the generation and use of private keys, and for other security purposes.

If things go according to plan, we should be summarising the tail end of the 5.18 development cycle and the preparation for 5.19 in the next issue. Which does make us wonder, just how far away is 6.0. While it might only be symbolic, we’re likely to see a Linux 6.0 release in a few months, perhaps before Linux turns 31.

» ONGOING DEVELOPMENT

Catalin Marinas posted a patch updating some Arm contacts since Grant Likely has taken a new role as CTO of Linaro. This author has personally known Grant for many years and is very excited for his new opportunity.

Anup Patel posted some patches to clean up IPI (Inter-Processor-Interrupt) support for RISC-V machines. IPIs are commonly used for code running on one processor core (“hart” in RISC-V) to signal another that it needs to perform an action. An example of this is tearing down an address space that’s been used by multiple other cores which may still contain cached data in their local TLB (Translation Lookaside Buffer – a kind of cache structure). Traditionally, RISC-V had a less-performant mechanism for IPIs, but a new interrupt controller architecture (AIA

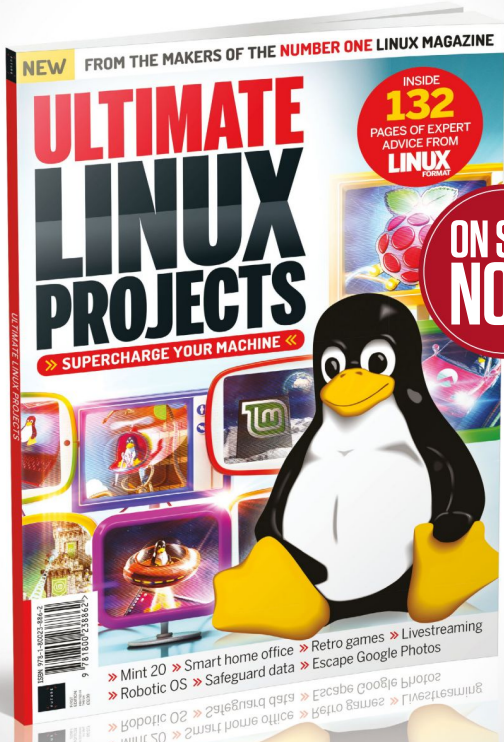
with IMSIC) aims to improve this performance significantly.

Byungchul Park posted version 3 of a patch series implementing DEPT (which stands for Dependency Tracker), a “tool for detecting deadlock possibilities by tracking wait/event rather than lock... acquisition order to try to cover all synchronisation mechanisms.” A little feedback came from one intrepid tester, and some updates were posted, but nothing further just yet. Still, this seems interesting as an extra means to find potential deadlocks.

Finally this month, an SVG version of Tux, the Linux logo has been added to the kernel in a response to the request not to add binary artefacts as part of new documentation (SVG is a descriptive text format for images). [DW](#)

EXPLORE THE POSSIBILITIES THAT LINUX HAS TO OFFER

From open-source software to coding masterclasses and Raspberry Pi projects, get the most from your machine with this exciting array of expert tutorials, guides and advice from the minds behind Linux Format magazine.



FUTURE



Ordering is easy. Go online at:

magazinesdirect.com

Or get it from selected supermarkets & newsagents

Answers

Got a burning question about open source or the kernel? Whatever your level, email it to lxformat@futurenet.com



Neil Bothwick

tweaks your troublesome Tux to make it tick smoothly

Q Secrecy without conflict

I use KeePass on my phone, laptop and desktop. I also use *Syncthing* to sync the database among the three devices.

Syncthing creates “conflict backup files” of the KeePass database when data is updated on more than one device. I’m looking for a solution, either a way for *Syncthing* to not generate these files or an alternative method of keeping the same password database on all devices. Is there a tool that will enable me to verify and combine both KeePass database files (since it’s encrypted)?

Jordan Welch

A We went through the same ourselves and tried several solutions. The problem with *Syncthing* is that it is, by its nature, non-interactive. When faced with a conflict it can’t ask what to do. It either makes a decision (possibly picking the wrong file), or saves both sets of data.

Our first approach was to store the KeePass database on *Dropbox*, so each application was working with the same file. It doesn’t avoid the problem of two programs writing different versions of the same file, but with KeePass, that only means you may occasionally miss saving

some data. Because KeePass is read a lot more often than written, this was a minor inconvenience. Even though the database file is encrypted, we weren’t comfortable leaving it on a remote server, so we switched to using *NextCloud* (<https://nextcloud.com>) to give basically the same effect as using *Dropbox*, but self-hosted.

However, we then found another password storage solution: *Bitwarden* (<https://bitwarden.com>). *Bitwarden* is like *LastPass*, but open source. There are browser plugins for *Firefox* and *Chrome* as well as desktop and mobile clients. The database is hosted on *Bitwarden*’s servers by default, but it doesn’t have to be. *Bitwarden* is open source, you can run the server yourself.

Unlike *NextCloud*, where you have to have a web server to host it, *Bitwarden* is available as a Docker image that you can run locally. You just need to open the relevant port on your router if you want to give yourself access from outside. The database contents are protected by a passphrase, but the database itself isn’t accessible from outside. As a bonus, you can import your KeePass database directly, making the transition simple.

If you’re going to host it yourself, the official *Bitwarden* docker image is a

heavyweight with a lot of dependencies, but there’s a lighter version called *Vaultwarden* (<https://hub.docker.com/r/vaultwarden/server>), which is more suitable for personal usage. Instructions for setting this up are on the web site.

It’s possible to merge the files – sort of. Load each one into *KeePass* and export it as a CSV file. This has each entry on a single line. Then you can combine those files and use *sort* together with its option to remove duplicate lines to extract only unique entries. You may get duplicates where the password for a site has changed, but that’s easy to clean up. If you exported the files as *keepassNN.csv*, you can perform the cleanup in one pass with:

```
$ cat keepass?.csv | sort -u ->keepassnew.csv
```

Then you can import the new CSV and save it out as a database. The CSV files contain all of your passwords in plain text, so it’s best to save them to a filesystem on RAM like */tmp*, or a USB stick that you can zero afterwards. Either way, delete them as soon as you’re done.

Q Do you copy?

I’ve got about 30 folders each containing a few hundred files. I want to copy all of the files in all of the folders into a new folder. Essentially, I’ve used *PhotoRec* on an old 3GB IDE drive I found in the garage and I want to put all the files into one folder. I’m not bothered about the names of them because *PhotoRec* has given them all obscure names already.

I’ve been trying to use *rsync*, but I’m not sure it can do it. Internet searches keep resulting in programming scripts and the obvious:

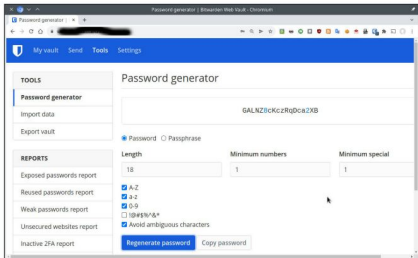
```
cp my_file_name.txt my_file2_name.txt
```

Can it be carried out with a somewhat simple command?

Kyle Marsh

A The exact method you use depends on whether there are duplicate names among the files. If this isn’t the case, you can do what you want with find:

```
$ find sourcedir -type f -exec cp {}
```



Bitwarden has desktop and mobile versions, along with browser extensions, as well as this web interface.

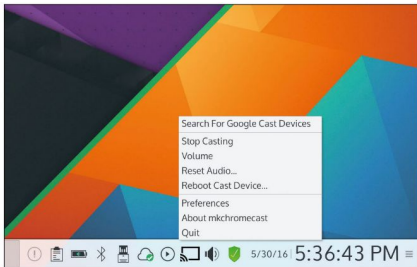
`destdir;`

This finds all files of type file – remember in Linux everything is a file, but you don't want to copy directories, symlinks or any other file-like objects – and passes them to the command specified with `-exec`. The curly braces are replaced with the name of the file and the semi-colon marks the end of the `-exec` command. It is escaped – putting a slash before the command – because the semi-colon would otherwise be interpreted by the shell. So this command finds every file in `sourcedir`, no matter where it is in the directory structure of that directory, and copies it to `destdir`. The problem occurs if there are files with the same name in different directories. Adding `-i` to the `cp` command will cause it to stop and ask before overwriting files.

To keep all the files you would need to make sure each one has a unique name. One way to do this is to use the full path of the file, replacing all directory separators with a different character. We can do this with a bash loop that reads all the files from the find command and acts on them:

```
find sourcedir -type f | while read F; do
mv "$F" "destdir/${F//\/|/}"
done
```

The first line runs `find` as before and sends the list of files to a pipe, where the script reads each file into a variable and acts on it until there are no more files. The `mv` command looks a bit messy, but it takes advantage of the Bash shell's built-in text replacement options. If you have a variable `X`, then `$X` gives the contents of `X`, `${X}/y/z` gives the contents of `X` with the first `y` replaced by `z`. Add another slash like this – `${X//y/z}` – and all occurrences of `y` are replaced. The way that we're using it here looks messy because the character



! Mkchromecast can mirror your desktop to any TV with a Chromecast dongle fitted or built in.

we want to replace is the directory separator, which is also a slash. However, we don't want the shell to treat it as part of the substitution structure, so we prefix it with a backslash. We now have the variable name, followed by two slashes to say replace all occurrences of the next character, which is also a slash, then another slash to specify the replacement character.

There are GUI tools available for moving and renaming files, such as `KRename`. However, because they need you to select files to work on, they generally handle one folder at a time.

Q Mirror, mirror on TV
How can I mirror my screen to a TV in Debian? I can do that easily in Windows, but in Debian I think it's missing the necessary software to do so.

I have seen some Ubuntu videos where they go to the Settings window and in its window's bar, at the right top corner, there is a SHARE button. With that they can share the screen, but it seems to be only a PC-to-PC VNC connection only, and not the Wi-Fi direct connection I'm looking for here. Is there any easy to use GUI software to do that?

Chris Lawson

A There are a number of ways to do this, depending on the desktop and software you have installed. If you're running a recent enough version of GNOME (at least GNOME Shell 3.34) you can install the `Cast To TV` extension. It's a somewhat fiddly process, but it does allow mirroring video and audio from the full GNOME desktop. See the instructions at www.linuxuprising.com/2020/04/how-to-cast-your-gnome-shell-desktop-to.html.

A simpler option is to use the `Chrome` browser. This used to make use of a `Cast` extension, but now it's built-in. Press the "hamburger" menu button and there will be a `Cast` option. When selected, it will give a list of available devices. If you select one of these now, it will cast the current tab, with video and sound, to the selected device. If you want to cast the entire desktop, press the `Sources` button at the bottom of the popup to pick desktop instead of the default of the current tab. This is a simple solution that only has one significant drawback: casting the full desktop doesn't mirror sound to the TV.

There is a third option – a program called `Mkchromecast`, which is available from <https://github.com/muammar/mkchromecast>. There is a package for Debian that's included in the official repositories. You only need to run

» A QUICK REFERENCE TO... TMPFS

Linux has more filesystems than you can shake a stick at. There are the traditional disk-based filesystems such as `ext2/3/4` and `XFS`. Then we have the all-in-one filesystems and volume managers like `ZFS` and `btrfs`, not to mention the virtual filesystems using `/proc`, `/sys` and `/dev`.

So here's one more: `tmpfs`. This is a filesystem that exists only in memory. When the computer is switched off, its contents are gone – forever. This makes it useful for data that you don't want hanging around. It's most commonly used to mount the `/tmp` directory because that's supposed to be wiped when you reboot anyway. Because the

files are stored in memory only, `tmpfs` is fast – much faster than a spinning disk and also faster than an SSD. You create a `tmpfs` from the command line with `sudo mount tmpfs /mountpoint -t tmpfs` or from `/etc/fstab` with a line like `tmpfs /tmp tmpfs size=50% 0 0`

The size option sets the maximum size of the filesystem, either in bytes or as a percentage of total RAM: 50 per cent is the default. This doesn't mean that mounting a `tmpfs` will eat up half of your memory. This is the maximum size; it only uses as much space as it needs to store files. When you unmount the filesystem, all memory is returned.

`$ sudo apt install mkchromecast`

or install it from your preferred graphical package manager. Once installed and running, *mkchromecast* adds a tray icon that you can use to search for and select suitable devices for mirroring your desktop.

Q Meddlesome modules

I have two modules that are causing me problems. They are filling up my syslog with errors. It's my fault because I installed them to keep time on an Odroid C2 [a 64-bit quad-core single board computer] in between reboots, but the battery piece was ripped off and now I have no use for the modules.

The two modules are `aml_i2c` and `rtc_pcf8563`. I know that I can temporarily remove them with `modprobe -r` but they come back after a reboot. How can I remove them for good?

Billy Howell

A If these are "out of tree" modules then those are not part of the kernel package. This means that you can uninstall them in the same way they were installed – with your distro's package manager. If you're using a distro based on Debian or Ubuntu then the easiest way to find out which package installed a file is to search for it at <http://packages.debian.org>.

However, the module names you give correspond to the ones that are installed with the kernel. While it's possible to remove the module files and rebuild the module database, this is likely to cause more problems than it fixes, and they will only reappear when your kernel is updated.

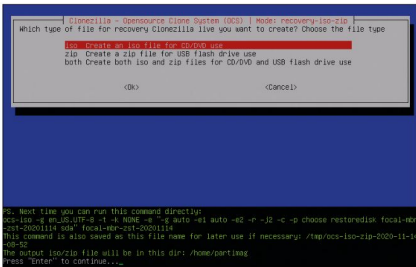
Instead, it's simplest to leave the modules where they are and tell the system to not load them at boot time, or at any later time. To do this create a file in `/etc/modprobe.d`. The name must end in `.conf` and should make it clear what it's for, so something like `/etc/modprobe.d/blacklist.conf`. Add a line to blacklist each module, like so:

```
blacklist aml_i2c
blacklist rtc_pcf8563
```

This will prevent the modules from being loaded, but something on your system was asking for them to be loaded in the first place. It could be worthwhile finding out what this was. Searching the output of `dmesg` after boot should give some clues.

Q Instant reinstallation

Do you know any tool to create an ISO image of your system as is and then be installable, with your users, configuration files and everything else? I want to be able to make backups and in case of loss of data or configurations,



In addition to making full disk backups, Clonezilla can create a restore disc tailored to your backup.

and install the latest version from an ISO image.

Alex Dyer

A Having spent many years mastering custom ISO images for the Linux Format DVDs, we can say that what you want to do is possible, but not easy. For one thing, you can't safely create an install ISO from a running system. Files will change during the process, so you will have to boot from either a live distro or another distro installed on your hard disk. Second, the remastering process is time-consuming, often taking several hours for a run. This was when creating ISOs of 2-4GB for the DVDs. Once you start including data from your home directory, you could end up with a much larger image, possibly exceeding the ISO specification and difficult to write to a suitable medium for booting.

Because you'd have to boot from a live distro to create the image, you may as well use one specifically designed for creating total system backups, such as Clonezilla (<https://clonezilla.org>). This is available as a live image that can be written to a CD or USB stick and boots to a menu that provides options for backing up and restoring hard disks. In contrast to using the likes of `dd` to image a disk, Clonezilla images only the parts of the disk in use, so it produces smaller images. It can write to a local medium, such as a USB drive, or to network storage.

When backing up a complete drive, it saves the partition table and bootloader as well as filesystem data, so you can easily restore your system back to the state in which it was at the time of the backup. Clonezilla's ace up its sleeve is that far as your needs are concerned is after creating your backup image, you can then use

Clonezilla to create a single bootable image file that contains the backup. Put this on a USB drive and in the event of disaster, you can boot and restore from the same drive.

There's a full tutorial on this on the Clonezilla web site. The process may seem a little involved, but after you've done it once, it should be straightforward to do it again for the following backups.

One problem with full disk backups like this is that they mean not having your system available while backing up, which discourages frequent backups. We would recommend using a traditional backup program. There are plenty to choose from to run scheduled backups, preferably daily, in between your full imaging sessions. **EW**

GET HELP NOW!

We'd love to try and answer any questions you send to lfanswers@futurenet.com, no matter what the level. We've all been stuck before, so don't be shy. However, we're only human (although many suspect Jonni is a robot), so it's important that you include as much information as you can. If something works on one distro but not another, then tell us. If you get an error message, please tell us the exact message and precisely what you did to invoke it.

If you have, or suspect, a hardware problem, let us know about the hardware. Consider installing *hardinfo* or *lshw*. These programs list the hardware on your machine, so send us their output. If you're unwilling, or unable, to install these, run the following commands in a root terminal and send us the `system.txt` file too.

```
uname -a >> system.txt
lspci >> system.txt
lspci -vv >> system.txt
```

FREE! BACKPACK WORTH £90 FOR NEW SUBSCRIBERS

NO.1 FOR DIGITAL ARTISTS

ImagineFX

IMPROVE YOUR

CHARACTER DESIGN

Learn with our expert advice from veteran Blizzard artist **Dave Greco**



MASTER YOUR PORTRAITS

Create striking art with help from pro artists



IN-DEPTH

PAINT A HIGH FANTASY SCENE WITH OILS

HOW TO AVOID AND MANAGE BURNOUT
ANATOMY: GUIDE TO BONE STRUCTURE



WORKSHOP

COLOURISING TECHNIQUES

How to create the woodblock look in Photoshop



AMIR ZAND TALKS ABOUT LIFE AS AN ARTIST, WITH WORKFLOW TIPS & TRICKS

ON SALE NOW! PRINT AND DIGITAL EDITIONS AVAILABLE AT

magazinesdirect.com

Digital editions are also available on iOS, Android, and more



Mailservers

WRITE TO US

Do you have a burning Linux-related issue that you want to discuss? Write to us at Linux.Format.Future.Publishing.Quay.House.The.Ambury.Bath.BAI11UA@gmail.com or email bfletters@futurenet.com.

Route of the problem

Further to the Newsdesk article about Linux malware in LXF287, which finished with "Implementing router-level security to help protect any device... is also worth investing in", I wonder whether an article about the main things to check/configure would be worthwhile? Many of us tend to lose interest, or at least get distracted by other things, after the relief of getting a new router working, and perhaps don't pay enough attention to security aspects.

Steve Holmes

Neil says...

It can't be a good sign when I think "we wrote what?" This is a good point though, as there are novel networking configurations that can help compartmentalise networks and bolster security. I was hoping to get someone to look at OpenWrt (<https://openwrt.org>) that could perhaps be the basis of a network feature, as otherwise every other router differs so much in configuration and features.

Helping to unlock the potential of your routers.

Bodhi is great

Thanks for reviewing Bodhi again in your most recent issue. We're always humbled and excited when we get



Bodhi is indeed great and you should give the distro a whirl. We'd put it on the cover disc but ooooooh.

a mention in your magazine. However, a few of our community members and our small team feel you have missed a few quite important things that make Bodhi what it is. The tiny footprint and lightweight philosophy, as well as the quick launcher (Ctrl+Esc) are just a few of the main ones.

We hope that when you next look at desktop environments or compare distros that you will also take into account that we are pretty unique in that we use EFL and not KDE or GDM and have used this to develop Moksha DE. In doing this we are able to ensure superb performance on older hardware without compromising on delivering a modern

Another bug bear is that reviewers tend to get sidetracked with the default theme, which is green. This can be changed!

Garth Williams.

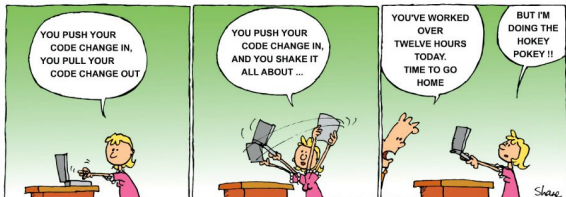
Neil says...

Thanks for the feedback. It's always good to have specific pointers on what to look into with distros. While I understand people will be defensive of their projects and when we come to review and compare anything we have to be objective, I would say a number of the things you point out we didn't actually do, in the LXF287 Roundup that I think you're referring to.

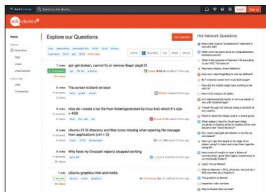
I don't think we even mentioned that Moshka was green (we did that for Linux Mint), plus we specifically highlighted a benefit of Bodhi is that it is indeed lightweight while offering plenty of visual bling. In a Roundup we're always going to end up skipping over specifics of what makes a project great, which I can imagine is frustrating, but we're limited on space so lots

Internet		System
IPv4 Internet	IPv6 Internet	Uptime: 16d 9h 58m 46s
Connected: ON	Connected: ON	Load Time: 2022-01-02 02:39:22
Connected since: 16d 9h 34m 34s	Connected since: -	Kernel Version: 5.4.80
Protocol: DHCP client	Connected since: -	Model: FriendlyElec NanoPi R4S
IPv4: [redacted]	Protocol: -	Architecture: ARMv6 Processor rev 4
GatewayV4: [redacted]	IPv6 profile: -	Firmware Version: AO Build@20211214
DHCP: 111110.01	GatewayV6: -	

Helpdex



* For full terms and conditions see: www.futureplc.com/terms-conditions



Canonical has fostered a fantastic Q&A forum, which is one reason why Ubuntu and similar are so popular.

of details never get a mention, but then that's why we'll still do individual reviews, too.

One-track minds

I keep noticing that many established Linux users, when dealing with new users, seem to demand that the new person has to change their system setup to fix any problem they might be complaining about. It's maddening. It's like a "I'm right, you're wrong" mentality – a bit like the old Vim vs Emacs flame wars of years gone by.

Say a new user pops up running Ubuntu and they're running a Nvidia graphics card. If they post online for help, then while there might be some useful replies asking for more details, the majority of posts will be "I don't have this problem on Arch", "Nvidia is proprietary" or "the Gnome desktop sucks, use KDE". And don't get me started on people going RTFM.

John Blaze

Neil says...

"The quickest way to get the right answer online is to post the wrong answer," goes Cunningham's Law. Okay, it's not quite what you're saying but the natural response for people online seems to be to tell you you're doing it wrong. Possibly more to the point here it's easier just to post "my way works" than to explore what the person's actual problems are and how to fix them. It's partly why well-managed Q&A forums are good for problem solving. I think the real solution here is to use those like <https://askubuntu.com>.

» LETTER OF THE MONTH

Don't dismiss Android

As a long-time reader of *Linux Format*, I've often noted the inconsistency with which your writers describe the relationship of Android to Linux. Sometimes you describe it as a type of Linux, sometimes as an alternative to Linux. The latter isn't accurate, as Android is based on the Linux kernel. Android belongs in the distro branch diagram on page 43 of your tribute to 30 years of Linux (LXF280), but it's not there.

The last page of that article (page 45) is the best example I've seen of the inconsistency. Its text includes both "distros, such as Android" and in the image caption "Android (not a distro)".

It's both accurate and politically valuable to the Linux world for you to point out at every opportunity that Android is a variant of Linux, as evidence of its impact.

Gregory Miller, Chicago

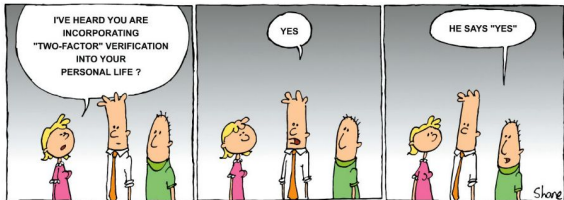
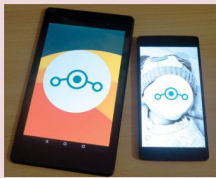
Neil says...

You're right that we're being dismissive and inconsistent, I think we were trying to be funny when we wrote "(not a distro)" in the caption. Android certainly is not an alternative and I don't think we have said as such. As for the distro element, technically Android is a distro: it's a specialised distribution usually for a specific model of phone. Although desktop Android x86 fits far more closely to what you'd normally think of as a distro. It was left out of the diagram (with plenty of others) for space, as it's not a "desktop" distro as such.

We do have a love/hate relationship with Android due to Google's permissive nature, but projects such as Lineage OS (based on Android) show just how valuable open source and the use of the Linux kernel are in this area. Just as people love to proclaim how they've extended the life of their laptops or desktop PCs, Lineage OS helps to do the same with many mobile devices that would otherwise be left hopelessly out of date software-wise.

Let us know if you'd like to see more Android coverage. We often discuss it, but our general sentiment is people would prefer to see desktop and laptop tutorials.

Jonni still has an "ancient" original Motorola Moto G Falcon from 2013 that's running Lineage OS.



shane_collinge@yahoo.com

SUBSCRIBE Save money today!

SUBSCRIBE

Sign up today and get your

AUSTRIAN AUDIO

Hi-X15 Over-Ear Headphones

YOUR GIFT!

WORTH £89

Don't miss out, subscribe now!



Product features

- » Low-impedance headphones, ideal for low-power outputs like mobile devices
- » Soft memory foam earpads to keep you comfortable during long listening sessions
- » Foldable all-metal hinges for easy storage – uncommon at this price range
- » Detachable 1.4m cable
- » Designed and engineered in Austria



WHAT HI-FI?

AWARDS 2021

BEST WIRED ON-EAR
HEADPHONES UNDER £100

Austrian Audio HI-X15

SUBSCRIBE NOW!

www.magazinesdirect.com/lin/a33x

Call **0330 333 1113** and quote **A33X**



» **PLUS:** Exclusive access¹ to the *Linux Format* subs area!

1,000s of DRM-free PDF back issues and articles! Get **instant access** back to issue 66 (May 2005) with tutorials, interviews, features and reviews. At linuxformat.com

DON'T MISS!
Includes 5 years of *Linux User & Developer* issues

NOT FROM THE UK?
Turn to page 63 for more great subscriber deals!

» **CHOOSE YOUR PACKAGE!**

ANNUAL PRINT EDITION

Only £67.50

13 issues of *Linux Format* in print by Direct Debit

ANNUAL PRINT + DIGITAL EDITION

Only £92.50

13 issues of *Linux Format* in print and digital by Direct Debit

ANNUAL DIGITAL EDITION ONLY

Only £54.99

13 issues of *Linux Format* in digital by Direct Debit

Terms and conditions: Offer closes 31 May, 2022. Offer open to new UK subscribers only. Pricing is guaranteed for the first 12 months and we will notify you in advance of any price changes. Please allow up to six weeks for delivery of your first subscription issue (up to eight weeks overseas). Your gift will be delivered separately within 60 days after your first payment has cleared. Gifts only available to subscribers on the UK mainland. Gift not available with a digital subscription. The full subscription rate is for 12 months (13 issues) and includes postage and packaging. If the magazine ordered changes frequency per annum, we will honour the number of issues paid for, not the term of the subscription. For full terms and conditions, visit www.magazinesdirect.com/terms. For enquiries please call +44 (0) 330 333 1113. Lines are open Monday to Friday, 9am to 5pm UK time, or email help@magazinesdirect.com. Calls to 0330 numbers will be charged at no more than a national landline call, and may be included in your phone provider's call bundle.

Seagate EXOS 20TB

Mark Pickavance is stunned by the storage but not the endurance of this drive. The cost is another matter altogether, mind...

SPECS

Size: 20TB

Interface: SATA or SAS

RPM: 7200

Max rate: 285MB/s

Power: 5.4w idle, 9.4w max

Design: Helium sealed

Cache: 256MB

MTBF: 2.5M hours

Warranty: Five years

The Seagate EXOS 20TB is targeted at those with cloud data centres, massive scale-out data centre applications, bulk storage and enterprise NAS systems. Whereas the IronWolf Pro was meant for commercial and Enterprise NAS to support the needs of Creative Pro and Medium-to-Large businesses. They look practically identical from the outside, so is the EXOS 20TB just wearing a different hat from its IronWolf Pro brother?

The Seagate EXOS 20TB sells for an eye-watering £480. We found the 18TB model for £295. That's a 70 per cent price increase for a 2TB, or 11 per cent increase in capacity. To the casual observer, this is the same 670g block of metal that houses a CMR (conventional magnetic recording) 7200RPM hard drive with 10 2TB platters and 20 heads, functioning within a factory-sealed helium atmosphere.

All the differences are internal, and based on the lack of weight difference. What differences exist are most likely firmware based rather than physical. Seagate has tuned the EXOS for data centre use, where high-capacity RAID arrays are used for big data applications, distributed file systems and disaster recovery platforms.

The IronWolf Pro 20TB offered an MTBF (mean time between failure) is 1.2 million hours, and the yearly workload is 300TB. The EXOS 20TB exceeds those levels with a 2.5-million-hour MTBF rating and 550TB annual workload. That converts into a warranty that lasts for five years with a TBW of 2,750TB, compared with the 1,500TBW of the IronWolf Pro.

Another difference is that the EXOS 20TB comes in both SATA and SAS connections, whereas the IronWolf Pro is exclusively SATA. For those that use SAS to connect arrays to multiple servers for fail-over functionality, this might be another reason for going with the EXOS, because it doubles the potential pathways from one to two on each drive.

We found it difficult to separate the IronWolf Pro and EXOS 20TB in our benchmarking. Both could read and write at close to 285MB/s in most tests. That's a 10 per cent improvement which can be linked to the extra platter and heads that this drive has over the 18TB model. But unlike the IronWolf Pro, the 20TB design shows no improvements in operational power demands over the 18TB models, and the SAS models uses an extra 0.4W over the SATA versions at idle.

Endurance

We had concerns about IronWolf Pro 20TB and its 300TB per year workload. These are addressed to a point by the 550TB per year workload offered by the EXOS 20TB. Doing the same calculations that we did for



20TB is an awful lot of .config files.

unbranded NAND SSD, though it's much better than the 75TB of workload transfer that the IronWolf Pro offers.

A problem we noted when covering the IronWolf is that for the sanity of those maintaining the data centre, regular array inspections are carried out to make sure the integrity of the data stored on them is good. The complete reading of the drive for an integrity test once a week would use up 1,040TB per year, nearly twice the yearly limit, and that's without any operational use.

For the customer, the choice is between the biggest drives available, allowing the largest possible arrays, or spreading the workload between less-expensive drives with potentially increased levels of redundancy. The phrase, 'between a rock and a hard place' seems most appropriate for this dilemma. **LF**

VERDICT

DEVELOPER: Seagate

WEB: www.seagate.com

PRICE: £480 SATA (£470 SAS)

FEATURES	9/10	EASE OF USE	9/10
PERFORMANCE	8/10	VALUE	6/10

Crazy amounts of storage and better than the IronWolf Pro 20TB due to an increased workload.

» Rating 8/10

HostGator

A basic website hosing option for small and medium businesses that **Shashank Sharma** thinks should be on your short list.

IN BRIEF

Thanks to its economical pricing and offerings, HostGator should appeal to most beginners, and even professionals and SMBs. The website builder, offered as part of the shared hosting plans, makes it easy to deploy a basic website in almost no time at all.

Based in the US, HostGator offers a range of solutions including shared hosting, VPS hosting, dedicated hosting, cloud hosting or even reseller hosting. While the dedicated hosting plans give the choice of Windows and Linux as the underlying operating systems, all shared hosting and VPS plans are Linux based. HostGator defaults to CentOS 7 for all its Linux installations, but you can request CentOS 6 if that's more your speed.

The shared hosting plans start at \$2.95 per month for a three-year subscription and might seem tempting because of the unmetered disk space and bandwidth. However, the hosting provider will ask you to scale down usage of the resources if your needs exceed 25 per cent of the total available resources.

Unlike many other hosting providers that allow backups out of the box, with HostGator you must separately purchase the CodeGuardBasic service for \$2 per month. The service provides daily automatic backups, 1GB storage space and three restorations per month, and you can also restore the entire site, or any file, to a previous state with a single click.

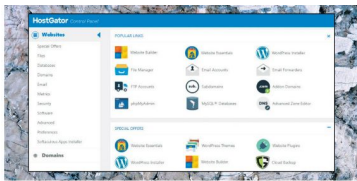
Although the plans include a website builder as well, it's rather limited, and you'd have to purchase a separate plan for it if you want to assemble a complete website spanning multiple pages.

Pleasant surprise

All shared hosting plans include some of the most used features, including a free email account. This is quite surprising as most hosting providers insist that you shell out extra for a professional email address that matches your chosen domain name.

HostGator also widely promotes its WordPress hosting plans. The cheapest Start plan priced at \$5.95 per month for a three-year subscription, with each renewal at \$9.95 per month. The Start plan enables you to deploy a single website, and limits the number of monthly visitors to 100,000. Also included is a free domain name, professional support and SSL certificate.

Users looking for more resources on their server must opt for either VPS or dedicated hosting. HostGator offers three VPS plans, starting at \$19.95 per month (renews at \$79.95 per month). It's also not possible to customise the plans to your needs, so you must opt for the next closest plan. HostGator's Baby plan includes a free domain for a year, unmetered disk space, free site migration, free email account, SSH access, and one-click installation of dozens of applications using *Softaculous* package manager and more. All hosting



↑ cPanel gives you everything you need to control and manage your site.

plans include a 45-day money-back guarantee, and you can also utilise \$150 credit for GoogleAds.

The HostGator Dashboard provides quick access to frequently used features such as cPanel, set up email accounts, install WordPress and more. The panel on the left features additional categories such as Hosting, Marketplace, Email & Office, Domains and Billing.

Unsure how to do something? Begin typing in the search box and the site displays matching articles. There's a lot of content that's sensibly organised, but if you need extra help, support is available 24/7/365 via live chat and telephone (toll-free in the US.)

We used the service [Uptime.com](#) to check the availability and response time of our HostGator site for a week. Our site was never down during the monitoring period. That's what we would expect after only a week of testing, but it was still good to see. Response times averaged 378ms, which is at the slower end of the basic shared hosting market (most providers average between 200 and 400ms.) There's better news in HostGator's 'worst case' time. In more than 2,000 tests, the slowest response logged was only 556ms, considerably better than many hosts (the current average for our last 30 reviews is 891ms). **1/5**

VERDICT

DEVELOPER: HostGator
WEB: www.hostgator.com
PRICE: From \$2.75

FEATURES	8/10	EASE OF USE	8/10
PERFORMANCE	8/10	VALUE	7/10

Although the pricing of the shared hosting plans make them a great starter option for SMBs and professionals, you can opt for VPS or dedicated plans as your website grows.

➤ **Rating 8/10**

Linux Mint DE 5

Whenever there's a new LMDE release, **Mayank Sharma** can't help but think of the old adage: "It's the thought that counts."

IN BRIEF

LMDE stands for Linux Mint Debian Edition, and as its name suggests, sources its packages from Debian. Unlike the main Linux Mint release, which is based on Ubuntu, in the project's own words, the distro exists to ensure Linux Mint continues to deliver its goodies, even if Ubuntu were to disappear.

SPECS

CPU: Any x86 processor
Mem: 4GB recommended
HDD: 100GB recommended
Build: 32- and 64-bit

LMDE 5, codenamed Elsie, is the latest edition of Linux Mint's Debian-based distro, and is based on Debian 11. LMDE exists because Linux Mint needed a fallback option in case Ubuntu made such significant changes to the distro that it no longer became feasible to use as a base.

For this reason, LMDE is also one of Mint's development targets, and helps guarantee that its homebrewed software work outside of the Ubuntu sphere.

Given its objectives, LMDE strives to be as similar as possible to Linux Mint. One of the reasons for Mint's popularity is its Cinnamon desktop environment, which sports familiar-looking desktop furniture, unlike the now-defunct Unity desktop, and even the new-fangled Gnome 3 desktop that's the default on Ubuntu.

The similarities don't end with the Cinnamon desktop. In fact, with LMDE 5, Linux Mint has tried to replicate the desktop experience of the latest edition of Linux Mint 20.3, with some noticeable differences.

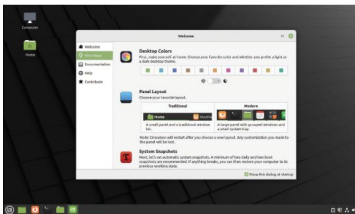
The first is the mechanism for transferring the ISO to a USB drive. According to LMDE's release notes, the distro's ISO uses a different structure than the ones used by other distros, such as Linux Mint, which means that it isn't compatible with multiboot tools such as Yumi, and has to be manually *dd'ed* onto a USB.

Another difference is the installer. LMDE doesn't use the installer it does on Linux Mint, but it's just as easy to navigate. The only real difference is the partitioning step. While both the Linux Mint and LMDE installers support automatic partitioning, if you need to manually partition your disk, LMDE fires up *Gparted* instead of handling this within the installer, like Linux Mint.

Proof of concept

The other differences aren't as innocuous. For starters, LMDE hardly has any documentation of its own. The project relies on documentation from the main distro, which is fine for the most part since the distros are so alike. However, for tasks such as upgrading drivers, the instructions don't work in LMDE because the distro doesn't include the driver manager.

Furthermore, LMDE is only offered with the Cinnamon desktop, unlike Mint that comes in several flavours. Then there's the fact that LMDE 5 comes about seven months after the release of Debian 11. Even then the developers haven't yet published the steps for users of LMDE 4 (LMDE doesn't have point releases) to upgrade to the latest release.



LMDE 5 is well stocked, and fully usable right-out-of-the-box, thanks to a host of desktop software you won't find on a stock Debian installation.

These quirks, taken along with LMDE's lack of original distro-specific documentation, and the project's own *raison d'être*, leads us to believe that perhaps the Linux Mint project itself looks at LMDE as more of a showcase, rather than something you'd want to run on a production machine.

The one key difference between LMDE and Mint is that the former is available for 32-bit machines as well. This leads some to suggest that LMDE is a good distro for older machines. We have our doubts though, because LMDE chooses to ship only with the Cinnamon desktop. Although Cinnamon isn't as resource hungry as Gnome 3 or KDE 4, it also isn't as lightweight as Mate or LXDE, that both offer a lot more bang for the buck on older, underpowered machines.

There's no denying that LMDE is an important distro for the Linux Mint project, but from the looks of it, LMDE isn't worked on with the same urgency and priority as the main edition. We have no issues with that, but it does prevent us from recommending the distro for any real use case beyond satisfying one's curiosity. **BT**

VERDICT

DEVELOPER: Linux Mint
WEB: <https://linuxmint.com>
LICENCE: Various

FEATURES	7/10	EASE OF USE	8/10
PERFORMANCE	8/10	DOCUMENTATION	5/10

Unless you're a Linux Mint developer, we'd suggest you stick to the project's main release, which is where all the action is.

» **Rating 7/10**

Fedora 36

Mayank Sharma calls the latest Fedora release just as boring as its predecessors. But is that a bad thing?

IN BRIEF

One of the top RPM-based desktop distros, Fedora also toils hard to ensure it delivers the best Gnome desktop. Supported by Red Hat, which employs several Fedora core developers, Fedora serves as a playground for bleeding-edge features to mature before they make their way into Red Hat's enterprise offerings.

SPECS

Minimum CPU: 2GHz
Memory: 2GB
HDD: 20GB
Builds: x86-64, AArch64

Generally speaking, it's difficult to become excited at Fedora releases. And that's a good thing. Unlike its often-fidgety peers, the Fedora developers quietly keep cranking out releases. Usually there's hardly any noticeable difference, and most of it can be attributed to the Gnome desktop environment. The majority of changes in Fedora usually happen behind the scenes, and manifest themselves in the nooks and crannies of the userland in very subtle, but important ways.

And the developers don't intend to change this winning formula with the Fedora 36 release. The most obvious evolutionary change in the distro is the inclusion of the Gnome 42 desktop environment, which is also part of the Ubuntu 22.04 release (see page 34). However, unlike Ubuntu's tweaked rendition, Fedora 36 ships with a more or less pristine Gnome 42 release.

The highlight of Gnome's latest version is a system-wide dark theme, with wallpapers for both dark and light themes, and tweaks to the folder icon theme to bring their appearance in line with the desktop.

The new desktop also brings an improved cache of programs, many of which have been ported to GTK4. The move to GTK4, thanks to all the behind-the-scenes work, will add a spring to the step of these applications, along with subtle changes to make them appear more modern. The most noticeable change is the new interactive screenshot tool that can now also record screencasts of the screen or a part of it (in the WebM format), in addition to taking static screenshots. The release also brings a replacement to the Gedit new text editor. The new one's simply called *Text Editor*, and in addition to the improved UI, the GTK4-powered tool also includes useful new features, such as auto-save.

Deep extra cover

A Fedora release is made up of several distros. Besides the Fedora Workstation release that's designed for desktop users, there's also Fedora Server, and Fedora IoT for their namesake environments. These will soon be joined by Fedora CoreOS, for cloud computing, and Fedora Silverblue, which delivers an immutable desktop ideal for containerised environments. While Workstation defaults to Gnome, the project has a few official spins for desktop environments. A notable change in one of these, the LXQt spin, is the inclusion of the 1.0 release of the lightweight desktop.

Looking under the covers, one of the most relevant changes for desktop users, especially the ones with



Fedora 36 is the latest Linux distribution to offer users a dark mode interface that can be switched on easily with a single click.

Nvidia graphics, is that the release will use the Wayland server even on installations that use the Nvidia driver.

Elsewhere, the rpm-ostree Fedora 36 variants, such as Fedora Silverblue, will now have the `/var` directory on a separate subvolume to help users maintain snapshots of dynamic data separate from the system snapshots. In the same vein, the RPM database has moved from `/usr` to the `/var` directory, which again will simplify snapshots and rollbacks, especially for admins. This is being touted as one of those changes that come into being in Fedora, and will eventually get replicated in Red Hat Enterprise Linux, after it gets a shakedown.

All things considered, Fedora 36 is like any Fedora release: pleasingly rock-solid and stable. In fact, we've been tinkering with the code-complete Beta release, which came out after missing a couple of deadlines (another testament to Fedora's insistence on stability over anything else). We didn't encounter any bugs during our testing on real and physical hardware, and are confident the Beta bears a very strong resemblance to the final release, which at time of writing is currently scheduled for release before the end of April. **TOP**

VERDICT

DEVELOPER: The Fedora Project
WEB: www.fedoraproject.org
LICENCE: Various

FEATURES	8/10	EASE OF USE	9/10
PERFORMANCE	8/10	DOCUMENTATION	9/10

As good as any release for new users to get into Fedora, and a must upgrade release for existing users.

» **Rating 8/10**

Ubuntu 22.04 LTS

If there's one thing **Mayank Sharma** likes even less than Ubuntu, it's Ubuntu LTS releases, which are notably stable but not notable generally.

IN BRIEF

One of the two main Gnome-bearing distros, arguably Ubuntu helped put Linux on the radar of all kinds of desktop users. Ubuntu, like its nearest rival Fedora, is supported by a for-profit company. The developers also use Ubuntu to spin distros for a variety of use-cases, from the desktop, to the data centre, and everything in between.

SPECS

CPU: 2GHz
Memory: 2 GB
HDD: 25 GB
Build: x86-64, Arm, RISC-V

This being an LTS release, Ubuntu treads cautiously and continues to use Pulseaudio as default sound server rather than PipeWire.

Unlike regular releases, long term support (LTS) releases like such as 22.04 code-named Jammy Jellyfish aren't designed to show off new features. Instead, LTS releases focus on stability over anything else.

Ubuntu will support Jammy Jellyfish for the next five years. Even after the expiry of that period enterprise customers can pay for the Extended Support Maintenance (ESM) contract to ensure their 22.04 installation is supported for another three to five years.

Because of its focus on stability, the developers are reluctant to bundle the latest bleeding-edge versions of the core components that do the heavy lifting. However, it's unfair to compare LTS releases with regular releases that Ubuntu churns out every six months.

So while Ubuntu 21.10 users won't notice any stark differences in the 22.04 release, Jammy Jellyfish will be a breath of fresh air for users on the previous LTS release, Ubuntu 20.04 Focal Fossa.

Is there a Gnome at home?

The first noticeable change in the distro is the placement of the Home and Install icons in the bottom-right corner of the screen, from their traditional position on the top left corner. Yet the difference worth noting is the inclusion of Gnome 42, which is the latest edition of the popular desktop environment. In fact, Fedora, which prides itself for bundling the latest Gnome desktop, also includes Gnome 42 in its latest release, Fedora 36 (see page 23).

Unlike Fedora however, Ubuntu ships with a customised version of the desktop environment with a handful of extensions and other tweaks to make it conform to their user experience expectations. Furthermore, while Gnome 42 brings in some radical new



The eagle-eyed would probably notice that after over a decade, Ubuntu 22.04 ships with a revamped logo with a contemporary lift to the Circle of Friends promise.

changes and applications, which are visible in Fedora 36, a majority of the default Gnome tools in Ubuntu 22.04 are from Gnome 41. Thanks to the LTS nature of the release, Ubuntu has erred on the side of caution and avoided including any Gnome program that have been ported to the GTK4 library.

For instance, Jammy Jellyfish continues to roll with the Gedit text editor and the old Terminal tool, which have been replaced by new variants in Gnome 42. The one new Gnome tool that Ubuntu 22.04 does include is the new Screenshot tool, which can also record screencasts.

Similarly, Ubuntu's Yaru theme (now sporting orange instead of purple) and its icon set have been ported to work with the latest Gnome desktop. In addition to the new lick of paint, customisers will appreciate that they can now select different accent colours to tweak the appearance of various elements on the desktop, including folders and the notification area.

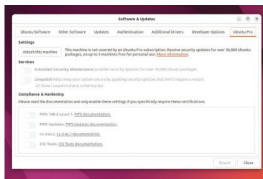
If you like customising your desktop, you'll love the new Personalise section in the Settings panel, which houses various settings to help you adjust important aspects of the desktop such as the dock. For instance, you can use the Personalise section to switch between light and dark versions of the Yaru themes, change the layout and behaviour of the dock, and lots more.

The one major graphics improvement that isn't noticeable is the distro defaulting to running Gnome on Wayland, even when the installation is using the proprietary Nvidia driver. This has been the default behaviour for Intel and AMD Radeon graphics hardware since Ubuntu 21.04, and will now work on machines using drivers newer than the Nvidia 510 series driver.

Under the covers

Although 22.04 is an LTS release, which shouldn't include any features that degrade the user experience, it does include a version of the Grub bootloader, which has deliberately disabled the OS Prober. This means that





The Software & Updates tool includes an Ubuntu Pro tab, which is a premium subscription service for business users.

when you install Ubuntu 22.04 LTS on machines which already have an existing operating system, such as Windows. Jammy Jellyfish won't add an entry to boot into that other OS in the GRUB menu. The Grub project decided to remove the package to counter potential security issues with the OS Prober component. You can, however, modify the `/etc/default/grub` file and toggle the `GRUB_DISABLE_OS_PROBER` setting.

Another change that won't be apparent is the switch to the Snap version of Firefox. This move is the result of the combined effort of Mozilla and Ubuntu, which both argue this will enable developers to push security updates faster and consistently. In our testing, after initial launch, the Snap version of Firefox performs just as well as the native .deb package did earlier.

Ubuntu 22.04 also boasts of improved hardware support beyond the x84-64 architecture. For starters, you can now run the distro together with a full desktop for the Raspberry Pi 4 with 2GB RAM. Note that since the release of Ubuntu 21.10, the distro already works on the 4GB and 8GB variants. But thanks to the use of zswap, Ubuntu 22.04 can now run even on the lowly 2GB variant.

Furthermore, starting with this release, the Ubuntu developers will also put out a live image designed specifically for the RISC-V architecture.

Prioritising stability

Ubuntu 22.04 is powered by the Linux 5.15 kernel, which itself is a LTS release, and brings some notable improvements. In particular, there's a brand-new implementation of the NTFS file system to read and write data to NTFS partitions and disks more efficiently.

However, had this been a regular Ubuntu release then the distro's developers would have gone with a newer

» ELSEWHERE IN THE UBUNTUVERSE

An Ubuntu 22.04 LTS release doesn't just include the main GNOME-based distro, but also pushes out new releases for the Ubuntu Server, along with updates to downstream distros such as Kubuntu, Lubuntu, Ubuntu Budgie, Ubuntu MATE, Ubuntu Studio, and Xubuntu.

One of our favourites is Ubuntu Budgie, which in the 22.04 release features a slightly tweaked default layout, with the icons in the panel now having additional spacing to ensure they don't appear cramped. The release also yanks the GNOME Control Center from the distro, and replaces it with the Budgie Control Center. In terms of usability, hot corners in Budgie sport an option to delay their activation.

Then there's Ubuntu Studio that uses the same KDE Plasma desktop environment as the official Kubuntu release. Note that due to the change in the desktop from Xfce (20.04) to KDE, Ubuntu Studio users don't receive an upgrade path from the older release. However, upgrades from Ubuntu Studio 21.10 work as advertised.



In 22.04, Ubuntu Budgie's welcome app offers the option to install the Microsoft Edge web browser, as well as the repository for the Brave browser.

kernel such as the 5.16 release, which packs in quite a lot of new functionality.

To continue what we touched upon at the start of this review, an LTS release isn't a natural upgrade path for those on the regular releases. An LTS release, such as Ubuntu 22.04, is meant for those running the previous LTS release, and the changes, even in the conservative release, will be much more dramatic for such users.

Remember, however, that if you're running the previous 20.04, which is still supported until at least April 2025, and perhaps even until 2030 under the ESM, you won't be prompted to upgrade to Ubuntu 22.04 until the release of its first update, namely Ubuntu 22.04.1 when any teething issues should have been ironed out. Of course, that doesn't prevent you from manually initiating the update process earlier as well. **EX**

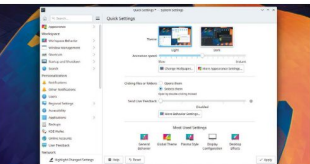
VERDICT

DEVELOPER: Canonical
WEB: www.ubuntu.com
LICENCE: Various

FEATURES	7/10	EASE OF USE	8/10
PERFORMANCE	8/10	DOCUMENTATION	8/10

A good distro for those looking for stability over freshness. If you need the latter, stick with the regular Ubuntu releases.

» **Rating 8/10**



KDE users can experience the advantages of the latest LTS release through Kubuntu 22.04, which ships with KDE 5.24.3.

A Musical Story

Fun, folk and funk are three things that Management hate with a passion, so **Tom Sykes** has turned down his speakers.

SPECS

Minimum OS: Ubuntu 18.04
64-bit CPU: 1.5GHz
64-bit Memory: 4GB
GPU: Integrated
HDD: 3GB

Guitar Hero is about the fantasy of being a rock legend, but *A Musical Story* is about the reality. There can be no whiffing of notes here, and no strutting around your living room like a rock god. This is a tough, unusual rhythm game that insists on perfection for each of its instrumental songs. That's perfection through repetition, through learning each rhythm and getting a feel for the music. It's probably a more accurate representation of the process of learning a song. Real rock stars don't get a timeline showing them when they need to hit each note.

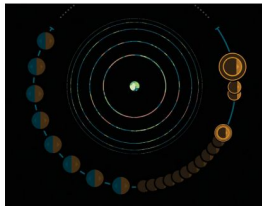
So it goes in *A Musical Story*, a game where the timeline has (mostly) disappeared. Instead, you need to learn the tempo and rhythm of each tune. It's an uncompromising rhythm game, and we like it for that. After passing each exam-like song section, you're rewarded with another story chunk, which is told via animation and (surely) the game soundtrack of the year.

We're going on a road trip!

While you'll flit between multiple instruments, the story follows the guitarist of a 1970s rock band, as the gang embarks on a road trip to the Pinewood music festival. But it's much more a game about the guitarist's drug addiction, as represented by the crows that begin to intrude upon his perception of reality.

There's no dialogue, but it's not needed thanks to the quality of the visual storytelling and the exceptional soundtrack. Funk rock gives way to folk and 70s synths, and extended maths rock sections that will really test your ability to tap, and hold, in time to invisible tempos.

Some help is offered in the form of a subtle position marker that activates if you fail repeatedly, and which disappears when you improve. You'll mess up a lot after the early chapters lull you into a false sense of security.



A rhythm section with concentric circles. Because why not?

Even if you master the music, you'll never get to play a full song. You only perform brief snippets in *A Musical Story*. You'll do a bass loop, then maybe the drums and guitar, before finishing things off with a dusting of lovely synth. To be fair, this is an approach that keeps the music fresh and the challenge level high. But it also means that when you've finally perfected that guitar riff or drum loop, mastery is essentially thrown away. You're not given the time to demonstrate, to enjoy your new skill before the game shunts you on to something new.

It feels like *A Musical Story* is a game caught between two worlds, something immediately evident when you look at the screenshots. There are the cutscenes, and the gameplay, and they don't meaningfully connect – so much so that the story fades out, so as not to be a distraction. There's a fantastic, varied soundtrack that gels perfectly with the animation, morphing from one genre to another, as best suits the scene.

Despite these issues the rhythm mechanics are strong, paring the genre down to its basics while still making it feel like you're playing a real instrument. *A Musical Story* wrings a lot of challenge out of just two buttons or keys. You're only ever tapping or holding one, the other, or both at once, but it takes skill to get the rhythm of each song down.

We just wish the game knew what it wanted to be: a story you experience, or a set of challenges you master. As it is, it's somewhere awkwardly in the middle. **LF**

VERDICT

DEVELOPER: Glee-Cheese Studio
WEB: www.glee-cheese.com
PRICE: £11.39

GAMEPLAY	7/10	LONGEVITY	7/10
GRAPHICS	7/10	VALUE	8/10

Great music, delivered in frustratingly bitty form. A clever rhythm game that doesn't mesh with its prominent story.

» Rating **7/10**

The Last Cube

Management isn't keen on sentient anything, it's bad for productivity, so **Neil Mohr** keeps the thinking to a minimum which isn't helping here at all...

SPECS

Minimum OS: Ubuntu 16.04 64-bit, SteamOS
CPU: Intel or AMD 2.2GHz dual-core
Memory: 4GB
HDD: 2GB
GPU: OpenGL 3.0+ compatible

Recommended

OS: Ubuntu 20.04 64-bit, SteamOS
CPU: Intel or AMD 2.7GHz quad-core
GPU: Nvidia GTX 970 or AMD RX 570

Exciting game characters of our time: Zelda, Lara Croft, Mario, Sonic and, erm, a cube? No, not any cube but *The Last Cube*! At its heart this is a straight-up logic platform puzzler, based on those classic top-down logic puzzles where you're using lasers and prisms to unlock the end block.

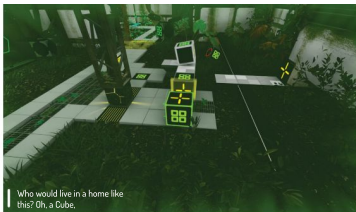
Here *The Last Cube* sees our cubic hero (ne?) rolling – for cubes can indeed roll – around a 2D world. You can add powers to each face of your cube through what the game calls stickers and there are Lore cubes that will drop narrative hints as to why all this is happening, but they feel more of an afterthought than core in any way.

Here lies much of the fun and frustration in *The Last Cube*, as you roll your cube around a level attempting to land that sticker face-down on the correct square. But why? The paper-thin plot of *The Last Cube* is as inconsequential as you'd expect: something about the Enormous Cube crumbling due to the internal problems destroying it. Only by completing each zone can The Cube be rebuilt and just by chance there's a zone for each side of a cube, that's six if it needed pointing out.

Mix 'n' match

Environmental obstacles such as rivers and electrical blocks in each zone will wipe or reset your cube, and switches will only activate with a matching symbol. Key to completing every level is planning your route, ensuring the correct stickers are active and that you land on the correct square.

Stickers are colour-coded, too. This element comes into play depending on what sticker is showing on top of a cube. Blue rotates your cube on the spot, green creates clones, yellow offers a slide move that can break obstacles, and red creates staircases.



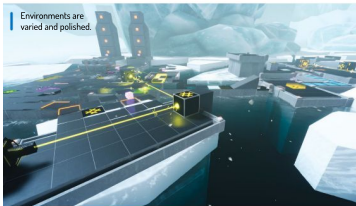
Who would live in a home like this? Oh, a Cube.

There are passing distractions with what the game calls relics dotted around levels. These are hard to miss, but gain you nothing other than a sense of achievement rather than something that aides you in-game.

This would be challenging enough, but the game throws in more devious tests. Once a level is complete you'll be given a challenge stage. It could be as straightforward as a time challenge, or ensuring that you don't lose a sticker. These are maddening because it feels as though there's little to no room for error, so you're more being tested than having fun.

The game is beautifully presented. The six biomes have a clear theme, from natural-looking ice, desert and lava stages to the more abstract and sci-fi TRON worlds. The zones are accessed from a central hub, so if you become frustrated with one area you can try another, and it's likely that you will at some point – if not many points!

If you enjoy puzzle games you'll find *The Last Cube* entertaining. It lacks the character of, say, *The Talos Principle*, its slim plot and the spartan world means many will find it lacking. Add with the sometimes challenging puzzles some might also find it too frustrating as you find yourself endlessly rotating a cube, trying to place a single side down on the right square. **8.7**



Environments are varied and polished.

VERDICT

DEVELOPER: Improx Games
WEB: www.lastcubegame.com
PRICE: £15.49

GAMEPLAY	7/10	LONGEVITY	7/10
GRAPHICS	7/10	VALUE	8/10

A clever, good-looking and rewarding puzzler, although it's exhausting and frustrating at times.

» **Rating 7/10**

Roundup

AppCenter » Bauh » Discover
» Gnome Software » HaikuDepot



Alexander Tolstoy

is going on a shopping spree, but this time it's to open source software stores!

Open source app stores

Many Linux flavours and open source OSES have their own software stores, **Alexander Tolstoy** wonders if it's worth your time browsing their wares.

HOW WE TESTED...

There are several criteria that enable us to make a fair assessment of software stores. The first and the most obvious one is whether it actually works and does the job it's meant for. This isn't as silly as it sounds because – as we'll find out later – some stores have difficulties in installing software.

Another criteria is how much information does a store provide on its 'product card'. For inexperienced or non tech-savvy users that can be a crucial feature that can tell a story about a program.

Next, we take a look at a store's product offering, which reflects our expectation of finding enough high-quality applications for work and play. Having several software sources and various package formats available in a store is certainly the right way to gain extra points in our test, and here we're looking at Flatpaks, Snaps and AppImages.

How an app store looks and feels was also taken into account because essentially it's a good indication of how user-friendly a store is. Let's dig into some details...



This is the first time that we've put software stores up against each other, to identify the most compelling option. This is because stores have started to play a significant role in modern Linux operating systems and have become a factor of their appeal, for those who either want to try Linux, or are thinking of switching their current OS.

Historically, Linux distros had package managers, and these still exist. However, a package manager normally lists thousands of packages that are mostly seen as internal technical 'stuff' by consumer-level users. That's why stores, with their curated and

visually appealing items are a better showcase of what the modern open source ecosystem has to offer. We're looking at the five most established stores. These are the two desktop-specific stores Gnome Software of Gnome and KDE Discover of Plasma KDE; two OS-specific – the AppCenter of ElementaryOS and HaikuDepot of Haiku; and finally one universal and distribution-agnostic store, which was *Bauh* (which we covered in *Hotpicks* from **LXF266**). Technically, *HaikuDepot* isn't Linux software (you can't run it on Linux), but it's a good example of how an open source app store affects the public's perception of an operating system.

First-launch experience

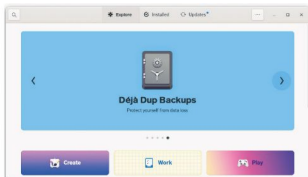
Do the stores load quickly, so that you're not left watching the clock?

We performed cold starts of all our stores to see how fast they become responsive and ready for action. Three out of the five stores use the *PackageKit* as their main backend: *Gnome Software*, *Discover* and *AppCenter*. We expected them to show equal start-up routines, but there were differences.

The *Gnome Software*'s integration with *PackageKit* was almost perfect and made the store work smoothly. On first launch it displayed the 'downloading software catalogue' banner, but after this the start-up time was significantly reduced. No delays, no annoyances. KDE's *Discover* was a bit slower: we had to give it some time and then restart it to make the store work as expected.

In the early days of *ElementaryOS 6* there was a major usability problem with *AppCenter*, which took too long to warm up and didn't react quickly to the user input. However, this is no longer the case, and the store in the latest *EOS 6.1* loaded up instantly, enabling us to install new programs without delay. We didn't touch the amount and variety of the available programs in this test, so we spotted no further issues with *AppCenter*. A small iOS-styled spinner rolled in the right-hand corner of the window, but the store remained ready for action.

Bauh's approach for starting up was significantly different. It refreshed its catalogue every time we launched the store so that no internal updates took place later on. Once *Bauh* has chewed through all the repo updates it became operational. It did take a



It's a pleasure to discover new programs using *Gnome Software*, once the software catalogue is downloaded. The user experience is sleek and distraction-free.

little while and the load time was longer than some of the other stores, although we were confident it would get there in the end.

On its very first start *HaikuDepot* politely asked if we wanted to permit the collecting and uploading of anonymous usage data online. After that it loaded the main interface window, which remained empty until the store fetched repository data. We were notified about the current progress in the lower part of the window. Although *HaikuDepot* provided us with good feedback about what was going on, it was the slowest of the five software stores. Subsequent runs of *HaikuDepot* still had the 'synchronising package data' phase, which was a bit of annoying.

VERDICT

GNOME SOFTWARE	10/10	BAUH	8/10
DISCOVER	8/10	HAIKUDEPOT	7/10
APPCENTER	10/10		

Gnome Software proved most responsive. Haiku's app manager is a little laggy.

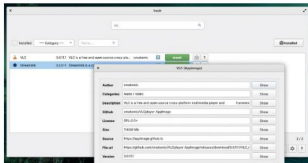
App description

How much useful information is in each product card?

We compared the details that could be found once we selected a program and wanted to know more about it. Each store provided some kind of description, but the amount of details, stars and reviews differed. We wanted to compare product cards of the same program across the five stores, and it worked out fine in all cases except for *AppCenter*, which only showed a limited selection of curated applications developed primarily for *EOS*.

As for *Gnome Software*, *Discover* and *AppCenter*, all these contenders relied on the same *AppStream* framework for diagnostic application descriptions. Yet their product cards were very different. *Gnome Software* had the most verbose and diverse description page that included size, versions and version history, ratings, users' comments, and even extra add-ons. The same program had a much smaller description in *Discover* with only some package details, screen shots and user reviews.

HaikuDepot featured a detailed program rating system that showed quite a lot of data in the lower panel below the main list of programs, including ratings, reviews and changelog. *Bauh* was less oriented on providing program data cards, instead operating like a convenient package manager, such as *Synaptic*. Still, we could



Package information in *Bauh* is available on-demand. Press the '?' button and an extra window with a detailed program description will show up.

press the '?' button next to the program name to access its details. A screenshot option was also available. Software pages in *AppCenter* were the most visually appealing, with large screen images and well-spaced out areas of text. Unfortunately, they provided less information than *Gnome Software*.

VERDICT

GNOME SOFTWARE	10/10	BAUH	7/10
DISCOVER	7/10	HAIKUDEPOT	8/10
APPCENTER	6/10		

Sometimes a well-filled program description entry in *Bauh* gave us more information than an *AppStream*-powered metadata document in other stores.

Variety of app sources

Do these stores give you enough choice?

The number of available applications is, of course, never cut and dry, because it depends on the Linux distributions. But a lot of things still happen on the store side. Thus we treated the *PackageKit* back-end of the three stores that depended on it as a one single entity, no matter how many programs it could retrieve using conventional packages (DEBs, RPMs and so on).

We tried to identify extra sources of new software that could bring us more cool stuff, and that indeed bore some fruit. Scoring more points in this test means that a software store brings a valid improvement over traditional package managers – not only in user friendliness, but in terms of the actual value. And so we tried to uncover the ways to use portable application formats such as Flatpak, Snap and AppImage, as well as seeing if it was easy enough to enable these options in our stores.

Gnome Software

6/10

Normally Gnome Software pulls the local *PackageKit* subsystem and enables users to install all programs that have properly configured AppStream metadata cards. This store also sports out-of-the-box support for Flatpaks (both technologies have strong ties with the Gnome project).

Depending on your distribution, *Gnome Software* will also display programs from third-party repositories based on Copr in Fedora or PPA in Ubuntu. However, that's about it for *Gnome Software*, which limits your choice to 'your distro's packages plus Flatpaks'. You can add the Flathub source and gain access to more exciting Flatpak applications, but you can't add Snap support unless you're running the Ubuntu-flavoured version of *Gnome Software*. The latter supports both portable formats, but it doesn't cut it for the vanilla version of the store, which is a clear downside. There's no way to manage AppImages or desktop extensions in *Gnome Software*, either.

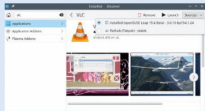


Discover

8/10

The official KDE Plasma store may have its own drawbacks related to stability or UI design, but when it comes to the variety of available software, it has few equals. *Discover* has three back-ends, namely for *PackageKit*, Snap and Flatpak sources. If you want to give all those three a try, you're better off with an appropriate Linux distro such as KDE Neon or Kubuntu. If the same program is available from more than one source, *Discover* will properly handle it and display just one program card, but will also show the Sources menu that lists the available installation methods.

Discover also has an integrated mechanism of obtaining Plasma-specific add-ons that include widgets, icons, themes and other bits from the KDE Store website. Lots of great content and mini tools can be found there, although the contents from store.kde.org desperately needs moderation because it's bloated with tons of stuff unrelated to Plasma.



System upgrade capability

Can our stores upgrade not just single programs, but the whole OS?

Most software stores performed reasonably well in this test because they rely on the mature and feature-rich *PackageKit* subsystem. We tested *Gnome Software* under Fedora, *Discover* under Kubuntu, and *AppCenter* – guess where – under EOS. We wanted to provide the best possible conditions for each store by running them in their native OS environments.

All three stores can handle system-wide upgrades and save you from running any CLI commands in the terminal. However, *Gnome Software* pushes its upgrade capabilities even further and knows how to handle full OS upgrades. That means it can perform an offline upgrade for switching to a newer major OS version once it's available – that is, of course, if you don't run a rolling release Linux OS. *Discover* can't manage full OS upgrades, and neither can *AppCenter*. Moreover, eOS itself doesn't support upgrading between major releases, so you can only install newer packages as you would with `apt-get`. *HaikuDepot* is generally in the club of upgrade-capable stores. It can search, display and install newer

system packages just fine. The Haiku OS has generally two release channels: one for the current beta3 version, and another for the 'nightly' rolling version. You can safely use *HaikuDepot* to keep both version updated.

Bauh doesn't have any system-wide upgrade features, although it was never meant to be used this way. It supports AUR and DEB packages, but it assumes that your system is a rolling distro that only receives 'normal' package updates. Therefore, *Bauh* and *AppCenter* don't score many points in this test.

VERDICT

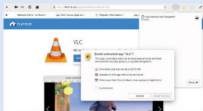
GNOME SOFTWARE	10/10	BAUH	8/10
DISCOVER	9/10	HAIKUDEPOT	9/10
APPCENTER	9/10		

All five stores can keep your system updated, although only *Gnome Software* can handle full release upgrades.

AppCenter**5/10**

Elementary's *AppCenter* is short of available software. It only displays a modest subset of Flatpak apps packaged for ElementaryOS. This wasn't the case for previous versions of EOS, but the team deliberately got rid of everything but Flatpaks. Therefore, *AppCenter* won't do a good job when searching for popular programs such as *VLC* or *LibreOffice* – they're not in the store, although you can still obtain them via *Apt* or *Synaptic*.

The good news is that you can go to the Flathub web page, click the Install button for any program you want and have it installed using the Sideload component of *AppCenter*. It's not obvious, but once you get at least one program installed via Sideload, all the rest of the Flathub starts to show in *AppCenter* with 'non-curated app' labels. It's better than nothing, but the almost empty default *AppCenter* is a problem that often drives away users who want to try EOS and its ecosystem.

**Bauh****9/10**

The concept of the *Bauh* store is to contain various distribution-agnostic software sources under one roof. As a result, *Bauh* has more back-ends than any other store in our *Roundup*. In particular, it supports Flatpaks, Snaps, AppImages, Arch packages, Debian packages and even web apps desktop integration. In most cases something from this list is missing, but if you hover your mouse over the question mark next to the missing source, *Bauh* will always explain why.

Bauh was also the only app store from this month's *Roundup* that supported AppImages management. Many great open source programs are provided as AppImages, such as *Krita*, and it's always good to have a tool for registering and tracking such packages. *Bauh* works very well on any Linux system, but if you want it to be your comprehensive software management store, it's best to run it on an Arch- or Debian-based distribution.

**HaikuDepot****5/10**

HaikuDepot is a classic-looking store that supports only native Haiku package repositories. You'll soon realise that there's a very short list of such repos thanks to limited take-up of Haiku OS itself. It isn't *HaikuDepot's* fault, but it also leaves little chance for it to outclass more substantial stores from the Linux world, perhaps with the exception of *AppCenter*, which still has fewer programs than *HaikuDepot*.

Both the official and third-party Haiku repos have regular packages plus some 'featured apps' that have their own well-described cards in the store. Don't underestimate the *HaikuDepot* assortment because it includes many popular tools that have either been built for or ported to Haiku, including *Telegram*, *VLC*, *LibreOffice* and *Krita*. Haiku is the only one non UNIX-like community-developed OS that has many open source applications available within a couple of mouse clicks from its software store.



Task queueing

Can you tell your software store to batch-install new programs?

Using a software store occasionally to install just one application is perfectly fine. However, some people prefer more intense shopping and would like to install many software titles at once. A good store would put all such requests in line and process them sequentially.

I quickly turned out that *Gnome Software* looked very uncomfortable during this particular test. It didn't fail, since all programs in the queue were finally installed correctly, but the store didn't provide any feedback about what programs were scheduled to be installed, what their progress was and what was the result. *Gnome Software* definitely doesn't want you to leave the product card before it's installed. KDE's *Discover* performed much better in that regard. After we pushed many Install buttons here and there, we saw the 'Tasks' section, which expanded into the list of progress bars for each program being downloaded and installed. As such, we were able to cancel individual downloads at any time while leaving the rest intact.

The *AppCenter* store was definitely better than *Gnome Software* because it displayed some helpful notifications once an application in the queue was successfully installed. There was no extra summary about the recent queue details and status, but at least we knew what was installed and when. Both *HaikuDepot* and *Bauh* would install only one program at a time and if you wanted more then you had to wait for the running task to complete. This effectively limits their value and makes you lose a lot of time in case you have many applications to install.

VERDICT

GNOME SOFTWARE	6/10	BAUH	3/10
DISCOVER	10/10	HAIKUDEPOT	3/10
APPCENTER	7/10		

Our test revealed that *Discover* is second to none when it comes to managing multiple installation tasks. You have better control over what's happening.

Reliability

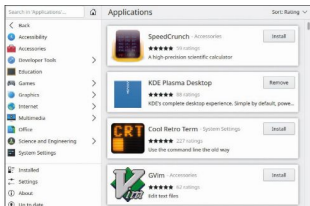
How stable are the software stores that you want to rely on?

No one wants a software store to misbehave, crash or freeze without any explanation. We generally had a good experience with Gnome Software but occasionally it reported 'something went wrong' messages. After restarting the store they were gone.

In contrast, *Plasma Discover* proved to be unreliable and even completely broken in some distributions. This store suffers from unstable and a seemingly only partial working PackageKit integration, therefore it's a big hit or miss depending on what Linux flavour you're using. *Plasma Discover* can freeze, display only part of the updates, or even try to update packages twice, and that's all rather sad. *AppCenter* was sleeker but we still can't call it stable due to numerous complaints from EOS users who suffer frequent *AppCenter* glitches. The EOS community it seems lacks the manpower to polish their store and make it more stable.

Bad news for *HaikuDepot* here too, because this store is subject to sporadic lock-ups and freezes. Part of the problem originates from Haikus itself, which has issues with networking, which in turn causes *HaikuDepot* to stall. Although we liked the *HaikuDepot* design, it was the only store we had to restart every now and then.

And this is where *Bauh*, our alternative store, shines. At the time of writing *Bauh* has the modest 0.10 version, but it is



The amount of packages available in Plasma Discover is huge. However, the catalogue needs moderation to prevent accidental deletion of the KDE desktop.

surprisingly stable and fail-proof. Whatever task we decided to do with it, *Bauh* always worked like a charm. It may be a bit slow and lacking some features, but it's a software store you can rely on.

VERDICT

GNOME SOFTWARE	8/10	BAUH	10/10
DISCOVER	6/10	HAIKUDEPOT	6/10
APPCENTER	7/10		

Bauh and Gnome Software are the two most reliable stores in our tests. The rest may work for some users, but not others.

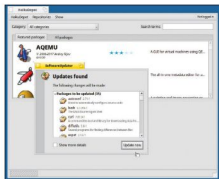
Extra features

What else can each store do and make you want to use it?

Both Discover and Gnome Software support the *fwupd* back-end for updating firmware for your devices. *Fwupd* is a software daemon that supports EFI system utilities, disk controllers, wireless accessories and more. See the full list at the Linux Vendors Firmware Service (LVFS) website. It's a great extra feature to have in a software store.

Did we mention Plasma extensions are available right from Discover? Extra widgets, icons, themes, decorations, wallpapers and a plenty of eye candy are available within one store! It's a pity *Gnome Software* doesn't have anything like that. Both *AppCenter* and *Gnome Software* have a similar set of features. These two stores are aimed at simplicity and therefore don't ship with too many extras. *AppCenter* also handles firmware updates, shows proprietary Nvidia drivers for those who may need them, and lists details on updated components. In other words, *AppCenter* enables you to quickly assess what exactly is being updated and why, which is a good design decision. Should we call payment methods integration in *AppCenter* an extra feature? That's debatable. Other than that, *AppCenter* keeps things simple.

We had some great moments installing web apps with *Bauh*. That was an interesting process thanks to the automated Nativefier feature that *Bauh* has to offer. Once you choose a web app, such as *WhatsApp* or *Netflix*, *Bauh* fetches all other dependencies for turning a web site into something that looks like a desktop application. It downloads *Electron*, *NodeJS*, the *Nativefier*



HaikuDepot can handle OS updates because it's a front-end to the pkgman utility in Haiku. However, this store has a limited set of features.

module and brings together all of these parts. *Bauh* also has a list of 'recommended' web apps to help you better choose what you want. And let's not forget about AppImagines, because *Bauh* is the only store that can carry out some integration and housekeeping for apps packaged in this format.

We finally come to *HaikuDepot*, which is a graphical package manager and a store in one tool. *HaikuDepot* has built-in user registration and this feature enables you to quickly get on board and start writing your app reviews. This feature is very well implemented in *HaikuDepot* and feels like an invitation to leave feedback that's hard to turn down.

VERDICT

GNOME SOFTWARE	8/10	BAUH	10/10
DISCOVER	8/10	HAIKUDEPOT	6/10
APPCENTER	8/10		

'Native' web apps available in Bauh is something that we've not seen before.

The Verdict

Open source app stores

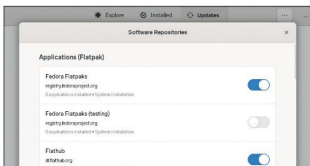
Gnome Software is strong in so many areas that we consider it a worthy winner. It's not ideal, of course, and it has some problems related to usability and interaction with its back-ends. Yet we can see that the GNOME Project dev team has invested a lot of effort into making *Gnome Software* a production-ready software store. The default store in Ubuntu was once based on *Gnome Software* for a reason: it took a solid foundation and added even more Ubuntu-specific features on top of it. It's a well-earned victory.

Plasma Discover is hot on the heels of *Gnome Software*. Our experience with *Discover* varied a lot depending on the Linux distribution that we used. Truth be told, proper integration of *Discover* and a distro-specific PackageKit/AppStream subsystem is a tricky thing that rarely goes to plan without hiccups. Apart from that, *Discover* is remarkably feature-rich and well-designed. It was the only store that enabled us to put many installation tasks in a queue and maintain control over it. It was more versatile too, thanks to the fact that it supported Flatpak, Snap, *Fwupd* and Plasma extensions at a time. Nice combo!

Third place belongs to *Bauh*, a high-quality distro-agnostic store that you should definitely try. *Bauh* has several unique features that might attract your interest. It has a clean interface that clearly shows the various package formats of the same application, it knows how to wrap a web app into a desktop one, and it can register and take control over your fragmented collection of downloaded AppImage files. *Bauh* could have challenged for first place in this Roundup if only it didn't lack multi-tasking. As of now, *Bauh* completely locks up and doesn't even allow browsing the catalogue while another application is being installed. You've got to be patient in this store!

Although we can't say that ElementaryOS's *AppCenter* is bad, it's certainly a mediocre store. It drives away users thanks to its noticeable lack of available applications. Adding the Flathub remote repository partly fixes this issue, but the procedure isn't obvious for many newcomers. The idea of supporting developers via "pay what you want" donations is good, but it looks a bit out of place in a store with such a modest selection of goods.

The *HaikuDepot* store turned out to be very laggy and unstable, despite the rich selection of various open source software packaged for Haiku. In many cases it makes more sense to use the command line *pkgman* tool to avoid frequent lock-ups of the official market application.



1st **Gnome Software** 10/10

Web: <https://gitlab.gnome.org/GNOME/gnome-software>
Licence: GPL2/LGPL Version: 41.5

The most mature and well-designed software store of the five on test.

2nd **Plasma Discover** 9/10

Web: <https://apps.kde.org/discover>
Licence: GPL2 Version: 5.24.3

Despite some distro-specific integration issues, this is a capable store.

3rd **Bauh** 9/10

Web: <https://github.com/vinifmor/bauh>
Licence: Zlib Version: 0.10

The best way to organise and maintain your collection of portable apps.

4th **AppCenter** 7/10

Web: <https://github.com/elementary/appcenter>
Licence: GPL3 Version: 3.9.1

It could have performed better if there were more programs available.

5th **HaikuDepot** 5/10

Web: <https://github.com/haiku/haikudepotserver>
Licence: MIT Version: 1.0.129

Looks promising, but suffers from performance and reliability issues.

» ALSO CONSIDER

We deliberately excluded various flavours and rip-offs of the popular *Gnome Software* store. Ubuntu has its own version, which can be configured to use Snap packages, while *System76* still relies on their own *AppCenter* fork known as *Pop!_Shop*. All those forks are interconnected and generally have minor differences. If you still miss Snap support, take a look at the Snapcraft web store, which is perfect for browsing and discovering apps. Once you decide to install something, it

is a matter of one or two commands to be copied and pasted to your terminal, so maybe there's no need in a separate GUI?

Nevertheless, there are many lesser-known app stores for Linux that we haven't covered here. How about *Souk*, a Flatpak-only store written in GTK4 and Rust? Or maybe the Flutter-based *AppImage Pool* that's designed exclusively for that package format? There's always a choice, so go ahead and find the best store for your needs and tastes. **BY**

BULLET-PROOF UBUNTU 22.04

Walpurgis Night is nearly upon us, so cast aside your old OS and begin life anew with Ubuntu 22.04, says **Jonni Bidwell**.

EIGHTEEN years ago Canonical, led by dot-com magnate-space tourist Mark Shuttleworth, unleashed the first Ubuntu release. It was nothing short of revolutionary. Suddenly Linux was a thing for human beings. Networking worked out of the box, as did a glorious – albeit brown – GNOME 2 desktop. It was built on Debian and inherited that reputation of stability, but it wasn't Debian. It was something special.

A huge community rallied around Canonical, which promised that it would listen. A bespoke bugtracker named Launchpad was set up, and the first bug filed was "Microsoft has a majority market share". For many, Linux's golden age was about to begin, and there was a palpable sense that Bug #1 would soon be fixed.

Flash forward to today, and you'll see that not all of those dreams came true. Windows still rules the desktop (though macOS and ChromeOS are swallowing that up). Casual desktop computing as a whole is becoming a niche hobby, because a great deal of our browsing and

communication is now carried out by smartphones (some of which run Linux, but not 'real' Linux). Desktop Linux is alive and well, but the ecosystem is still not perfect. An abundance of desktop choices, together with numerous forks of popular distros, have led to complaints about fragmentation (from people that don't understand open source software and free will). And Canonical copped plenty of flack when it abandoned Unity and the Ubuntu Phone.

But it's not all bad. Companies have embraced Linux, in particular Valve. Its work on Proton has enabled some 5,000 Windows-only games to be played on Linux. And Ubuntu is still a hugely popular Linux distribution that's great for playing said games, wrangling vital documents, or managing your clouds.

And now Canonical

is back with a brand-new release called Jammy Jellyfish. It incorporates parts of the latest GNOME 42 desktop. The switch to the Wayland display protocol has finally happened. The new PipeWire multimedia framework is woven into its fabric. And it's a Long Term Support (LTS) release, so you can keep on using it for five whole years.

You won't find earth-shattering user-facing changes here, but you will find a great, reliable OS. Read on to find out why...



Of jams and jellies

It's Ubuntu LTS time, so let's see what will be the shape of Ubuntu for the next few years...

We always look forward to trying out a new Ubuntu release. But this time around we're not expecting a wildly reinvented desktop paradigm or huge performance leaps. The previous Ubuntu LTS, Focal Fossa, after occasionally rocky beginnings, has been a loyal servant to many of our machines, and we're sure Jammy Jellyfish will be a worthy successor. We're looking forward to Gnome 42 (although there are some loose ends from earlier releases), a more polished Wayland experience and we want to see how Canonical is pushing ahead with its Snap initiative.

There's only one problem. At time of writing, it hasn't been released. But that's okay, because it will be by the time you read this. And if we're lucky there we won't have missed any last-minute additions or surprises. We've been testing the daily Jammy Jellyfish images for a couple of months prior to the official release.

Minor niggles, begone!

And we've seen quite a bit of change in that time, particularly as parts of the recently released Gnome 42 start to find their way in. Indeed, as we write this we're still about a week away from launch day, but since both the Feature and UI Freeze have passed we don't expect any drastic changes. We do rather hope some minor niggles (such as stuttering and occasional crashes while dragging windows between monitors) get sorted out, though.

If you've used either of the interim releases (21.04 or 21.10) since the last LTS then you'll be aware that now



CREDIT: @simonbutcher

Ubuntu uses Wayland and (maybe) remember that Active Directory can be set up from the installer. You'll be aware that there's light and dark versions of the Yaru theme, and you'll suspect (rightly) that these have been further tweaked. To be frank, if you've been using Ubuntu 21.10, then there's not anything groundbreaking in 22.04. But that doesn't mean you shouldn't upgrade. You should, because if nothing else your interim release is about to be EOL'd. Oh, and if you're of the ilk that gets excited by phrases like 'modern design trends', then check out the new logo. It's similar to the old Circle of Friends logo, but on a web3-friendly (stop baiting sensible readers! – Ed) rectangular background.

If you just want to see what Ubuntu is like, there's no need to install it at all. Just follow our handy three-step guide to downloading, writing and booting an Ubuntu USB stick, or DVD (Sorry!–Ed) if you must.

The official background is over the page, but these AI-generated jellyfish by Simon Butcher are something else.

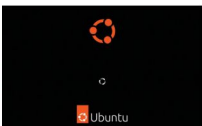
DOWNLOAD AND BOOT UBUNTU



1 Download an ISO
Go to <https://ubuntu.com/desktop> and download the ISO file. It's 3.5GB so you may want to fetch a cup of tea. If you're interested in trying another flavour, such as Kubuntu or Ubuntu, you'll find links at <https://ubuntu.com/download/flavours>. You'll also find links to the Server, Pi and Core editions here.



2 Write out a USB stick
Use your favourite image writer (or download Balena Etcher from <https://etcher.io>) to write the image to a USB stick. Don't remove the medium until you're told it's safe to do so. Bad media will cause problems later. You could also (using different software) make a DVD, but this will be slower than using flash media.



3 Boot Ubuntu
Your machine might enable you to bring up a boot menu by pressing F12 or F10 at boot time. If so use this to one-time boot from the Ubuntu medium. Otherwise you'll need to go into the UEFI/BIOS setup interface and change the boot order. See the official docs at <https://ubuntu.com/tutorials/try-ubuntu-before-you-install>.

Escape Windows

Whether you're a complete novice or Windows has driven you to seek out other operating systems, Ubuntu can help.

Windows 11 has been rolled out through the Insider program since late last year. All Windows 10 users will have been offered the upgrade, in all likelihood, by the time you're reading this. There's nothing like a new Windows release for motivating people to switch to Linux. So here's a quick guide for recent Windows apostates.

You may be tempted to dual-boot Windows and Ubuntu together. This might sound convenient, but it's rich with pitfalls so not something to rush into. Ubuntu will install alongside Windows, but there's no telling if down the line Windows Update will, on a whim, decide the Linux partition(s) is no longer necessary. For this reason we don't recommend installing both OSes to the same device. A 250GB SSD is ideal for your first Linux explorations, and you can get this new for around £25.

Our next prudent bit of guidance is perhaps overly cautious, and a little inconvenient, but it's the only way

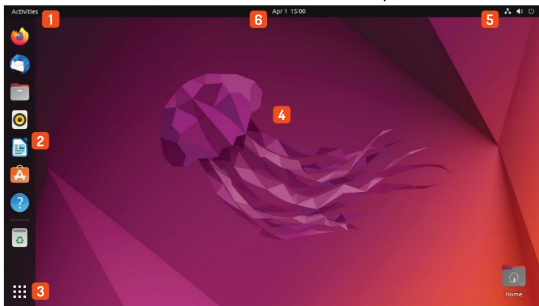
to be sure Windows won't touch your Linux: unplug the SSD prior to booting to Windows. Yup, it's hard to remember and potentially awkward to carry out (if your case is under the desk, say), but at least you can describe your install as 'airgapped from Windows'.

You might instead want to install Ubuntu to an external hard drive or USB stick, though if you're not using USB3 storage this won't be terribly performant. Ideally, you'd put it on a whole new machine, but not everyone has a spare, working machine.

BIOS, meet UEFI

Modern PCs use a newer firmware, the Universal Extensible Firmware Interface, rather than the traditional BIOS of yore. UEFI machines may have a classic BIOS emulation mode, but you almost certainly shouldn't enable it. Especially if you already have OSes installed in UEFI mode (they will stop working).

Get to know Ubuntu's Gnome desktop



1 Activities

Click here or press Super (Windows) to bring up the Activities Overview. This will show you previews of open windows, which you can drag over to the right, to move them to a new virtual desktop.

2 Dock

This provides easy access to popular programs. Running applications are indicated by a red dot. Right-click to pin or unpin applications from here.

3 Applications grid

This displays all currently installed applications. Type a few characters to narrow down the list. You can also find emoji this way, if they float your boat. Oh, and there's a virtual desktop switcher here too.

4 Desktop options

Right-click to change either the background or display settings. You can also create desktop folders.

5 Status icons

Network, volume and power indicators. Click to access Bluetooth, brightness and (for laptops) power profile settings. The logout and shutdown options are also here.

6 Notification area:

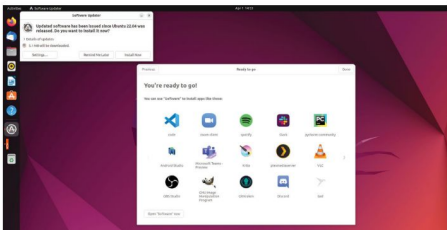
Alerts, such as new software being available or new media that's being played, are shown here. There's a calendar too - this can either be used locally or connected with a cloud service.

Note that PCs these days can be incredibly fussy about getting into the UEFI setup or summoning a boot menu. Precision timing, multiple reboots, as well as digging up manuals to find the appropriate shortcut key may be required. The Ubuntu EFI boot capsules are all signed by a Microsoft-endorsed key, so there should be no need to disable Secure Boot.

One thing to be aware of is that one EFI partition is required to boot a UEFI system. So if you plan on installing Ubuntu on a separate drive, make sure the "Device for bootloader installation" is set to the original drive, and that the EFI partition is selected. This drive will need to be plugged in to boot either OS, but you'll be able to choose which from the UEFI.

Once you've successfully booted Ubuntu, you'll be asked whether you want to try Ubuntu or jump right in and install it. We'd suggest trying it first, if you haven't already. This enables you to get a feel for the operating system without it touching any of your storage. So you can try out bundled software, install new things and see if it's right for you. The only downside is that it won't quite be as performant as the real thing. Oh, and any changes you make will be lost after a reboot, of course. The annotation (*below left*) shows you the rudiments of Ubuntu's GNOME desktop.

Just as in other OSes you'll find folders for your Documents, Photos and Downloads. But unlike at least one other OS you won't be bombarded with marketing or voice assistants trying to help you. The Dock area on



the left-hand side is a nod to Ubuntu's old Unity desktop. The new desktop has been based on GNOME 3 since Ubuntu 18.04, but with some usability tweaks. GNOME 3 was seen by some as too ambitious, occasionally buggy, and a memory hog when it was introduced in 2008 (this commentator even used the phrase "hypermodern"), but these days the fact it forms the basis for so many desktops is testament to its solidity. If you imagine the dock was gone you'll see what a lot of traditionalists' main problem with GNOME

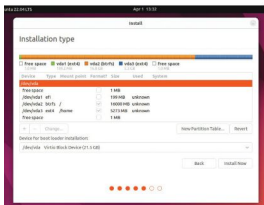
Don't know what to install first? Let the Snap Store inspire you. And don't forget your updates!

USING A ROCK-SOLID GNOME

"The new desktop has been based on GNOME 3 since Ubuntu 18.04, but with some usability tweaks."

is: There's no obvious menu to launch applications. The applications grid provided by the Dock isn't quite the same thing, but if you're in the habit of using a mouse to open a traditionally placed applications menu, then your muscle memory will more likely bring you here than to the Activities view.

On a clean install the dock area has shortcuts to *Firefox*, *Thunderbird* and *LibreOffice Writer*. The question mark icon will take you to the desktop guide, which hopefully answers any questions you may have. You'll also find links to Ubuntu Software (the shopping bag icon), in case you want to install more software, as well as the venerable *Rhythmbox* music player. Internal and external drives will also show up here, plus there is a Rubbish Bin from whence deleted files can be retrieved.



If you know what you want, partition-wise, then the Something Else option in the installer will help.

» BECOME A KEYBOARD WARRIOR

In an age of QHD screens and 4K displays, the mouse cursor's pilgrimage to the top-left corner can be an arduous one. This journey can be saved through the magic of keyboard shortcuts. Apart from the all-important Super (Windows)

key to bring up the Activities view, your life in GNOME may be improved (*May? – Ed*) with the following knowledge:
Ctrl+Super Left/right » tile window left/right
Super+A » show applications grid

Super+PgUp/PgDown » switch virtual desktops
Shift+Super+PgUp/PgDown » move window to prev/next workspace
Shift+Super Left/Right » move active window to prev/next display

Customise Ubuntu

Discover new software. Change settings. Install a new desktop (or three).

Ubuntu (and most other desktop Linux flavours) have been designed to be intuitive and easy to learn. However, sooner or later you'll probably want to change some things around. For example, we think *Rhythmbox* is great. It's been the default music player in Ubuntu since the very beginning (with only a brief sabbatical while *Banshee* took its place in 2011). But with its **Client Side Decorated window** it looks dated, and can't connect to popular (albeit proprietary) streaming services so we might want to look at alternatives. By this point we're assuming you've installed Ubuntu, and enjoyed its new look **Flutter-built installer**.

Fire up the *Ubuntu Software* application, scroll down to the list of categories and select Music and Audio. You'll see a selection of audio programs, most of which we've

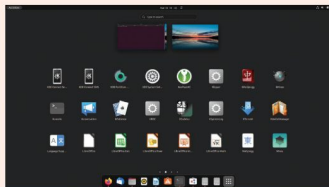
never heard of. You will, however, find the official *Spotify* and *Audible* programs, as well as unofficial players for *Deezer*, *YouTube*, *Google Play Music* and *Apple Music*. If you prefer something even more nostalgic, you'll also find *Foobar2000*, *DeadBeef-vs* (a minimal GTK player and glorious hex reference) as well as myriad text-based music players. Install *Spotify* (or whatever else takes your fancy) by hitting the green button.

Most applications in the *Software* application are shipped as Snap packages. You can see the delivery mechanism in the *Source* box in the top-right. Snap is Canonical's self-contained packaging format which (like Flatpak, which is a similar effort) enables developers to easily ship software without having to worry about distro-specific packaging and which versions of which libraries to ship. Snaps also run in a confined sandbox (unless you give them permission to otherwise) so they can't access any files or hardware they don't need to.

» DESKTOP CHOICES

There are multiple flavours of Ubuntu 22.04 that include the same rock-solid foundation as the flagship, but with a different desktop environment. If you like Ubuntu but don't like modern GNOME, then *Kubuntu*, *Ubuntu MATE* (inspired by *GNOME 2*) or the lightweight *LXQt*-powered *Lubuntu* are well worth your time. But rather than install a whole new *buntu, you might prefer to just add a new desktop to the current installation. This is unlikely to break anything, but the session packages we'll install include each desktop's core applications. So you might end up with two (or more) file managers, text editors and the like.

Some desktops come with their own login manager too. So for example if you install the **kubuntu-desktop** package you'll be asked if you want to stick with *GNOME's* *GDM3* or switch to *SDDM* (which is built using *Qt* so looks more *KDE-like*). There's no right or wrong answer, and you can change your mind with `sudo dpkg-reconfigure gdm3`. The other desktop packages are named similarly, so there's **ubuntu-mate-desktop** and **ubuntu-desktop**. Most of these have a more slimmed-down version – for example, **kubuntu-core** will install a more minimal set of applications.

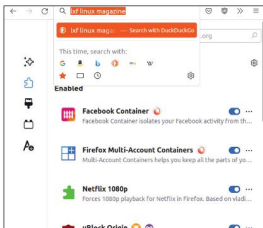


Installing the whole *Kubuntu* desktop package makes for a menu that, unsurprisingly, is rich in items that begin with *K*.

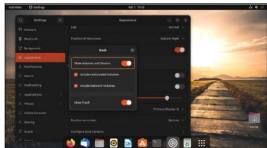
Life on the bleeding edge

Occasionally in the *Source* box you'll see a variety of different 'channels' are available for a given Snap. These often enable you to grab a beta or development release, in case you want to live on the bleeding edge. System packages are still installed as *.deb* packages and there are still tens of thousands of these traditional packages you can install from the command line with *Apt*. These no longer show up in the *Software* application, but if you install *Synaptic* you can browse these graphically.

Canonical has put a lot of effort into making sure popular applications are available in Snap form. Besides *Spotify* you'll find *Telegram*, *Slack*, *Blender*, *GIMP* and the *PyCharm* development environment for *Python*. There's also open source versions of some classic games, including *Prince of Persia (SDLPoP)*, *Open Jedi Knight* and *Widelands* (a *Settlers* clone).



An ad-blocker and Mozilla's container programs are essential for the modern Web. And switching the default search to *DuckDuckGo*.



What's up, dock? Here we've put the Dock at the bottom, removed various clutter, and made it shorter.

A big change in this Ubuntu outing is that *Firefox* is only available as a Snap. This comes directly from Mozilla, saving Canonical a packaging burden (and forcing derivatives such as Linux Mint to build and package their own *Firefox*). In our testing, there was a delay of about 10s each time *Firefox* was started from a clean login. This is mildly annoying since the web browser is often the first thing one opens post-login, but hopefully Snap startup times will be worked on in future.

If the slow-starting *Firefox* (or Snaps in general) bother you, then you can always use the Mozilla PPA to install a traditional package (see <https://launchpad.net/~mozillateam/+archive/ubuntu/ppa>). Or download a tarball from its FTP site. Or you could switch to the other side of the modern packaging formats debate and install Flatpak and *Firefox* from the Flathub. Your first will likely be to install *uBlock Origin*, as well as Mozilla's *Facebook Container* and *Multi-account Container* add-ons.

Back in LXF283 we looked at how *Firefox* worked on Ubuntu 21.10 (and Fedora 35), and found that the Snap version didn't work at all with VA-API video acceleration. Happily, we were able to get it working in the new version, though some extra configuration is required. Go to **about:config** (noting the warning) and set **media.fmpeg.vaapi.enabled** to true. Later you may also want to set **media.navigator.mediadatadecoder_vpx_enabled** as well, which will accelerate WebRTC (for example, *Zoom*, *Teams*, *Jitsi*) sessions. In our testing (in *Firefox* 98, 99 and 100 by way of Snap channels) we had to disable the RDD sandbox to make it work. Since this is a security risk we won't tell you how to do it here (but we're sure you can DuckDuckGo it).

In that feature we also saw that both Snap and Flatpak versions of *Firefox* (and *Chromium* and *Edge*) can't handle extensions which use Native Messaging. This is still true, so password manager extensions (as well as things like hardware authentication tokens) don't currently work here. Both packaging formats should soon see a host messaging portal soon, but until then these add-ons will only work with traditionally packaged browsers. On a related tangent, *KeePassXC* installed as a Snap (or Flatpak) will integrate with such browsers, but you'll need to run a script, as described on its website.

Another consequence of contained browsers is that the old <https://extensions.gnome.org> (EGO) website won't work correctly. Even if you install **chrome-gnome-shell** and the browser plugin. That's okay though, for now you can use a third-party tool, such as *Extension Manager*, to do this. This tool is available as a Flatpak, so we'll need to install that and set up the Flathub repo first. You might want to do this even if you don't care

about Gnome extensions, since it gives you a whole other avenue (and tool) by which more software can be accessed. So open a terminal and run:

```
$ sudo apt install flatpak gnome-software-plugin-flatpak gnome-software
$ sudo flatpak remote-add --if-not-exists flathub
https://flathub.org/repo/flathub.flatpakrepo
```

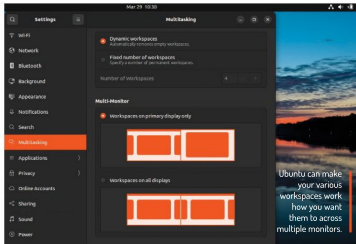
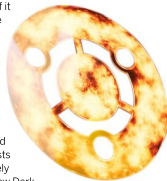
This add support for Flatpak programs in the Gnome Software GUI, also installed by the first command. So you'll be able to search for Extension Manager there after a reboot. Note that Gnome Software is distinct from the usual *Ubuntu Software* tool. It's just called *Software* and has a shopping bag icon. Alternatively, if you're enjoying the terminal the required incantation for installing and running (sans need to reboot) is:

```
$ flatpak install com.mattjakeman.ExtensionManager
$ flatpak run com.mattjakeman.ExtensionManager
```

You'll see that Ubuntu uses three Gnome extensions (for desktop icons, appindicators and the dock) and that two of these can be configured. And if you navigate to the Browse tab you can find many more. You might already have some favourite Gnome extensions, and hopefully most of those have been updated to support version 42. Extensions Manager will display "Unsupported" if not. The popular "Blur my Shell" is available. Likewise *GSCONnect*, a Gnome-centric take on the popular *KDE Connect* utility for talking to your phone from the desktop.

The shortcut bar on the left isn't to everyone's taste, although fans claim it is more efficient than having it along the bottom. You might prefer to get rid of it altogether and make the desktop more like the vanilla Gnome you'd find in the likes of Fedora. Wherever you want your dock, it can be configured by starting the Settings application (either from the menu at the top-right or from the Activities Overview) and navigating to the Appearance section.

The screenshot shows a slightly more orthodox arrangement, except there doesn't seem to be a way to move the Applications Grid shortcut to the left, which is where traditionalists might prefer to find the thing which most closely resembles a classical application menu. The new Dark Theme (which now should work universally) can also be enabled from the Appearance section.



Tweaking and rewiring

Some final edits to perfect your installation, plus a little Ubuntu nostalgia.

Wayland by default was tested in Ubuntu 17.10, but that was perhaps a little ambitious. Now the technology has matured and Canonical is confident that it's – to dredge up an irksome phrase – “ready for prime time”. Extensive testing has taken place and the team are confident that the Wayland experience will be good for all. Yep, even those using Nvidia hardware. If it's not, well, that's fine. The old X11 session is still there.

Wayland has been fairly misrepresented in the press. (who, me?—Ed) historically. The most egregious falsehoods were that remote desktop sessions, screen sharing and even humble screenshots are impossible with Wayland. Do not believe such myths. The problem wasn't Wayland, it was programs that didn't support it. All the screenshots in this feature would not be here if that were the case.

WE'RE IN GNOME'S GOLDEN AGE

“The stutters and memory leaks that dogged Ubuntu Gnome's performance for so long are well and truly gone.”

One change mulled for 22.04 but which in the end never made it is the replacement of PulseAudio with PipeWire. The latter is a whole new multimedia framework which, as it happens, enables desktop sharing and screen recording on Wayland. Programs may still depend on PipeWire (particularly web browsers), but venerable PulseAudio remains the default sound server. If you want to change this (for example, if you're having difficulty with Bluetooth

headsets), you can install the PipeWire session with `$ sudo apt install pipewire-pulse`. Then if you log out and back in and run the command:

```
$ pactl info
```

you should see (among other lines):

```
Server Name: PulseAudio (on PipeWire 0.3.xx)
```

Additional libraries may be required for some Bluetooth audio codecs. Try:

```
$ sudo apt install libfdk-aac2 libldacbt-[abr,enc]2 libopenaptx0
```

if you run into difficulties. Alternatively, seek more up-to-date documentation, we are unfortunately static!

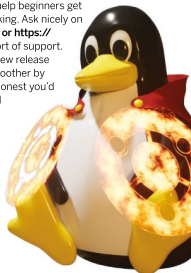
Ubuntu has used Gnome as its default desktop since 18.04 LTS. If you pine for the Unity desktop, then you might be interested in the Extended Security Maintenance (ESM) that's available for the previous LTS, Ubuntu 16.04 (Xenial Xerus). The official support period for this expired in May 2021, but since this version is still widely deployed Canonical offers paid-for support to organisations. This is achieved through its Ubuntu Advantage for Infrastructure program. Personal users are allowed ESM on up to three machines for free, so if you want to keep the Xerus alive you can now do so in a safe and (semi) supported manner.

We were feeling nostalgic, so we fired up Ubuntu 16.04 on our XPS. This hadn't been booted for some time, and had problems seeing our new-fangled USB-C dock (or the network cable plugged therein). But once we'd updated it, enrolled the machine in Ubuntu Advantage and updated again everything worked more or less fine. Don't let anyone tell you nostalgia is not a good reason for running old software. Especially when you're entitled to run three instances for your own pleasure. If you were looking for actual phone and ticket support, then this starts at \$150/year for a single desktop installation or \$750/year for a server. It's not really intended to help beginners get their printers or Wi-Fi working. Ask nicely on <https://ubuntuforums.org> or <https://askubuntu.com> for that sort of support.

Booting back into the new release was much quicker and smoother by comparison, which to be honest you'd expect after six years of UI development. This release might not have the kind ground-breaking features that we used to enjoy, but that's probably a good thing. All those features and breaking changes we used to love five to 15-years ago were a consequence of desktop Linux still being rather new. Now that Ubuntu's



One thing that was quite hard to screenshot (but for once not because of Wayland) was the new screenshot tool. Oh the irony!



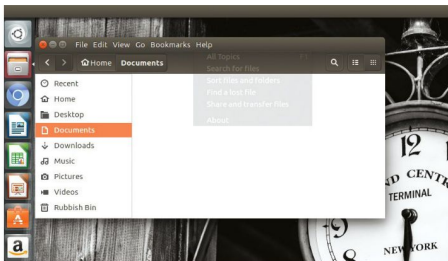
desktop is established, like it or not, it doesn't make sense to go changing it. Instead, we should take comfort in the fact that after four years of using GNOME for its flagship desktop, the experience is now second to none. The stutters and memory leaks that dogged Ubuntu GNOME's performance for so long are well and truly gone.

A common Theme

GNOME themes have come under the spotlight since the introduction of GTK4 (inaugurated with GNOME 40). Did we say themes? Ah, we meant theme, because custom theming of GNOME applications is now verboten. The default GTK3 theme was called *Adwaita*, a Sanskrit word often translated as 'the only one', (literally 'not two'). But it wasn't really the only one, because developers could happily write their own CSS stylings.

In GTK4 this theme has been promoted to a platform library, *libadwaita*, which GNOME developers say will guarantee conformance with their Human Interface Guidelines. And (like the characters often say in *Highlander*) there can be only one. GTK3 applications will still respect custom themes, but GTK4 ones will only support the limited changes (for example, background and accent colours) permitted by *libadwaita*.

For Ubuntu 22.04 this might be bad news, because at present it uses a mix of GNOME 42 applications (*libadwaita*-based) plus some from older releases (such as *Files*, which is based on GTK3 and *libhandy*). This may change prior to release, otherwise there are going to be some cosmetic inconsistencies. If this bothers you, then you might want to run away from GNOME 42 for the next little while, in which case there are some suggestions in the box (see below).



The old GNOME Tweaks tool is still available in the repo, but like the EGO website it can no longer manage GNOME extensions. That's okay, because it can do most everything else, including cleaning up the mess our GNOME fonts ended up in post installation of KDE Plasma. Tweaks also enables you to manage startup programs, change titlebar button visibility (or move them to the left, MacOS style) and adjust legacy theming. You can install Tweaks with:

```
$ sudo apt install gnome-tweaks
```

This will install a different Extensions tool, currently in beta form. At the time of writing this doesn't let you install new extensions, otherwise we could do away with the previous tool. For even more tweakability, try *Just Perfection*, found in Extension Manager. It allows for parts of the shell theme to be overruled (including removal of the top bar) to make matters more minimal. We don't go for GNOME extensions ourselves (despite having two programs for managing them), let us know what we're missing out on. Enjoy Ubuntu 22.04! **137**

Menus in titlebars. Amazon search results in the HUD. Ubuntu 16.04 had some crazy ideas!

» LOOKING ELSEWHERE?

Latterly there seems to have been a bit of a trend for Linux-leaning social media channels to announce they're "no longer recommending Ubuntu" or other such things. Reasons are varied, we suppose, but the triumvirate of Snaps, Wayland and GNOME don't seem to be to everyone's taste.

We'd still heartily recommend Ubuntu to anyone, beginner or otherwise – as it "just works". Even if you don't like it, as we've seen it can be customised, extended or otherwise bashed around to your taste. Lots of the distros these channels recommend in Ubuntu's stead are themselves based on Ubuntu – for example Linux Mint, Pop!_OS and Elementary OS. All great distros that

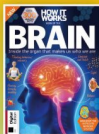
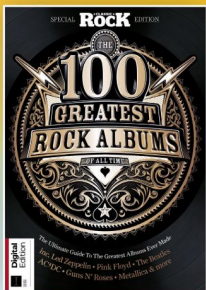
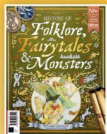
offer something which is hard to recreate on Ubuntu Linux, but ultimately distros that depend on its packages, infrastructure and documentation. Until now, perhaps.

Mint's latest Debian Edition (LMDE5) is rapidly gaining traction. Pop!_OS has moved its PPA repositories away from LaunchPad and is working on a new Rust-powered desktop environment (with a view to moving away from GNOME). And Elementary OS has had its own app store for ages and has likewise sided with Flatpaks over Snaps. In sum, if Ubuntu doesn't do it for you, there are plenty of derivatives you can switch to without having to learn a whole new way of working.

We're excited to see more people trying Fedora. It's now more accessible, particularly as regards installing non-free software. Together with its rapid release cycle this makes it a great platform for gaming. Well worth checking out if Ubuntu is no longer serving you.



Fedoras and the distribution of that name are all the rage right now.



DISCOVER OUR GREAT BOOKAZINES

From travel and history to gaming and photography, you're certain to find something you're passionate about



Follow us on Instagram  @futurebookazines



www.magazinesdirect.com

Magazines, back issues & bookazines.





Matt Holder
has worked in IT for
15 years, with 13
years' experience
in supporting
IT in schools.

» BITTEN BY THE FOSS BUG

Ever since my teenage years I've had an interest in programming and electronics. Many hours were spent sat next to my dad, learning how to program in QBASIC. I then happened upon a book, which showed how to use a parallel port to interface with the outside world. Dad and I spent many more hours devising a timing system that could be used to time cars as they hurtled around my Scalextric track. Following this, I carried on developing various electronic circuits.

I was bitten by the Linux bug two decades ago now. I can still remember being sat in a fast food restaurant, with the yellow logo with a friend of mine, during college free periods. My friend would talk excitedly about the culture surrounding free software and why more eyes on the code is a good thing as well as the technicalities, which, at the time, went over my head.

Then I installed Slackware at home, where I learned about the distribution, SSH, kernel options, enabling routing as well as many other technical things.

I couldn't have been happier when the Raspberry Pi was announced. The combination of an amazing community, readily available and cost-effective peripherals and support for multiple programming languages has turned this amazing little board into a world-conquering device!

At home I have Raspberry Pi Zeroes acting as baby monitors and temperature sensors for the kids' rooms that report back to Home Assistant, which is itself running on a Raspberry Pi 4, booted from a USB SSD.

Consumers last in the queue for Pi supplies

After warnings early in 2022, stock remains a challenge for the Raspberry Pi Foundation.

Despite baking half a million Raspberry Pi devices per month, demand is still outstripping supply and Raspberry Pi co-founder Eben Upton has taken to the company's blog (<https://bit.ly/xf289pi>) to talk about the continuing product shortages, and how it will be addressing the difficulty in purchasing new models. Commercial and industrial customers will get priority, leaving individual customer's struggling to buy Raspberry Pi for their own home projects.

Eben urges people to buy from an approved reseller, because they're getting priority to stock, but does add, "Right now we feel the right thing to do is to prioritise

commercial and industrial customers – the people who need Raspberry Pis to run their businesses – we're acutely aware that people's livelihoods are at stake." It remains the case that newer 28nm models – Pi 4-based options – are easier to source than older 40nm products such as the Pi 3.

Due to market scalpers many stores now limit the number customers can buy and offer two-factor authentication to limit or stop bots from auto-trading on scarce stock. Eben added, "Many Approved Resellers have implemented single-unit limits to combat this... we're encouraging other Approved Resellers to consider this route."

Eben advises that the Pi Pico and Pi 400 are generally in stock, while you can use <https://rpilocator.com> to locate stock, too.



CREDIT: Raspberry Pi Foundation

One Pi at a time, move along please, form an orderly queue.

Arm powered Not that sort!

Arduino has released a robotic-arm tool that's powered by its own Nano RP2040 Connect chip. The Braccio++ costs \$600 and has been designed for engineering-level classes. The arm is made from recycled food cartons with an aluminium layer for strength. It's powered by six Arduino RS485 Smart Servo Motors: four in the arm and two to control the claw. See <https://store-usa.arduino.cc/pages/braccioplusplus> for more details.



CREDIT: Arduino

Armed to the teeth.

It's another SBC New Portenta X8 board.

Coming in at approximately the same size as a Raspberry Pi Zero, Arduino's new single-board computer X8 contains a quad-core ARM Cortex-A53 processor running at up to 1.8GHz, an additional Arm Cortex-M4 core up to 400MHz, 2GB of RAM and 16GB of eMMC storage with a full suite of comms. With this SBC Arduino's 199 Euro device is aiming to take on the Pi Compute. Find out more at <https://store.arduino.cc>.



At least you can buy this one!

CREDIT: Arduino

Armbian 22.02 Jammy Xfce

Les Pounder takes a look at a distro that supports 64 different Linux single board computers, and now he has to buy them all.

IN BRIEF

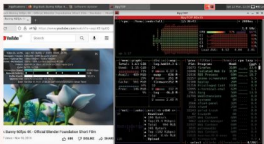
Armbian is a framework from which we can create our own distros for a plethora of Linux SBC. Supported by 64 different Linux devices, Armbian is a solid, no-frills distro that acts as a blank canvas for our projects. GPIO support is minimal and streaming video is a slideshow, but we get a rock-solid base to build from.

A simple, desktop environment affords us a blank canvas from which we can create our own custom distributions.

A rmbian is a Debian-based distro designed as a framework for many models of single board computers (SBCs). Armbian is available for 64 different SBC, some of which we've never heard of. No matter which SBC we chose, they all claimed to be lightweight, fast and secure. So with these bold claims in mind, we fired up our trusty Raspberry Pi 4 for a test. We chose the latest Armbian desktop release, Armbian 22.02 Jammy Xfce, which is based on Ubuntu 22.04 "Jammy Jellyfish" and uses the Xfce desktop environment. There's another CLI release that provides the bare minimum for a build. We tested Armbian on a 2GB Raspberry Pi 4 running at stock speed, which is 1.8GHz on our Pi because we have a later revision of the BCM2711 System on Chip (SoC).

The boot sequence is typical Linux: just lots of text scrolling up the screen as the services and device drivers are loaded into RAM. The first boot sees a short interactive sequence where we create a root password, an unprivileged user and a choice of shell: Bash or zsh. We chose the former. After a few moments the desktop appeared and we saw the familiar Xfce desktop. Armbian comes fully loaded. There's a web browser (*Firefox*), a choice of terminal emulators (including *Terminator*, an old favourite of ours), *LibreOffice* and typical desktop tools.

We ran a quick test of *Big Buck Bunny*, initially streaming the 1080p60 version that became a slideshow as frames were dropped. Dropping the resolution to 720p60 and it was a little better, 480p was great. This issue isn't linked to Armbian; even Raspberry Pi OS finds 1080p60 a challenge. The desktop is rather utilitarian, but that's part of the charm. Armbian isn't Raspberry Pi OS. Sure, we can use it as a desktop computer, and *Firefox* is responsive enough even on our 2GB Pi4. We also did a quick test of a 1GB Raspberry Pi 4, and the experience was not so stellar, with a lockup forcing us to reboot. A 2GB Pi should be the lowest spec to consider for the desktop OS. CLI users should be fine with 1GB of RAM.



As ever, YouTube playback on the Raspberry Pi 4 is bad. The best we will get with 720p60 video is a slight stutter.

After updating the software repositories we set to testing Armbian and because this is a Pi we wanted to test the GPIO. This is where we hit a problem. Raspberry Pi OS users will be familiar with RPi.GPIO, PiGPIO pin factories, but Ubuntu uses I2C, which talks to Linux gpiochip devices and isn't specific to the Raspberry Pi (remember, Armbian covers 64 SBCs). Using I2C with Python 3 we can flash an LED, make connections with I2C devices and PWM. But for HATs and other add-ons that require specific Python modules your mileage will vary. We used our default test HAT – Pimoroni's Explorer HAT Pro – which uses a mix of I2C, PWM and digital IO, and this is where things went wrong. The automated install script failed with multiple errors that we remedied, but Explorer HAT's dependence on RPi.GPIO killed our attempts.

Don't be put off by Armbian. It's a solid base for projects that don't require the GPIO. Your embedded systems will benefit from the stability. Build a server, appliance or kiosk with Armbian and you can port your OS to another SBC, and given the global chip shortage, this is becoming more of a necessity than an option. **BF**

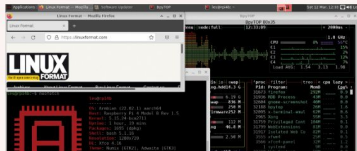
VERDICT

DEVELOPER: Armbian Team
WEB: www.armbian.com
LICENCE: GPL v2

FEATURES	7/10	EASE OF USE	7/10
PERFORMANCE	7/10	DOCUMENTATION	8/10

A solid framework for those wanting to make their own SBC distros. Not for the everyday Pi user, but a nice alternative.

» Rating **7/10**



Arducam Auto-focus 16MP Camera module

Quick-off-the-draw **Les Pounder** can't shoot shots faster than this camera.

SPEC

Sensor: Sony IMX519
Optics: Type 1/2.53
Sensor: 4,656x3,496, 16MP
Video: 1080p30, 720p60
F-stop: 1.75
Focal length: 4.28mm
Focus: 10cm to infinity
Size: 26x24x18mm

Arducam's latest crowdfunded camera features the familiar official Raspberry Pi camera form factor, but with an autofocus lens and a sensor: Sony's IMX519. This is even more powerful than the one used in the official Raspberry Pi HQ camera.

The case for the camera is sold separately, but if you can spare the extra £4.50 you'll discover that it features a 1/4-inch thread for tripods. The included CSI flat flex cable is the same as the one used with the official Pi cameras, and it can be changed for either longer or shorter cables, or used with converters if the Raspberry Pi Zero is your device of choice.

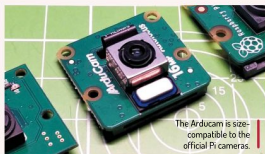
As you would expect from a Raspberry Pi camera, the unit connects to the Camera (CSI) port alongside the HDMI port. In order for the camera to work, we need to follow a few simple steps to download and install a driver and then a tweaked version of *libcamera*. This has since become the standard camera software on Bullseye, the latest Raspberry Pi operating system.

Manually operated autofocus

With the installation complete, we carried out a few tests, beginning with the "Hello World" test using *libcamera-hello*. We secured our camera into a tripod, pointed it at a suitable test subject, and saw that the image wasn't focused. It transpires that the camera focuses when the *libcamera* command is invoked with the `--autofocus` switch and it doesn't constantly "hunt" or change focus as objects move around the frame.

To refocus the camera we need to use an additional switch: `--keypress`. Using this command we can arrange the subject to be captured, then switch to the terminal window and press `f` then Enter to refocus the camera.


Out of the box Raspberry Pi OS Buster couldn't detect the camera and these commands fail. The same is true with the popular Python PiCamera library. We performed the same install process as we did with Bullseye and *libcamera* detected and used the camera with no issues.



The same can't be said for *raspistill*, *raspicam* and *PiCamera*. They just don't work with this camera, and that's a real shame. Apparently a fix is in the works...

Arducam's camera has much better focus than the fixed focus official V1 and V2 cameras, which have a 1-2m to infinity focus range. So anything greater than 1-2m is sharp, but close up we get a blurred mess (unless you hack the lens). With Arducam and its autofocus we have a focal range between 10cm and infinity.

During our comparison tests, we noticed an unfortunate side-effect of installing the Arducam software: it broke compatibility with the official Raspberry Pi cameras. We tested our Bullseye and Buster test installations, both of which were running Arducam's software, and sadly our official Raspberry Pi HQ camera wasn't detected. There was a simple workaround, with an official fix out by the time you read this, so we're told.

If you need the quality of the HQ camera and the size of the V1 and V2 official cameras, then Arducam High-Resolution Auto-focus Camera is for you. The size and shape of the camera mean that it can be a drop-in replacement for bird cams, wildlife trails and openCV robotics projects. Autofocus, the key selling point, is great and while it may not constantly set the autofocus, focus time is fast. In fact we accidentally took a focused image in 1/100th of a second, zoomoom! 



Just what the doctor ordered: a physical mount! Note that it is sold separately.

VERDICT

DEVELOPER: Arducam
WEB: <https://shop.pimoroni.com>
PRICE: £24

FEATURES	9/10	EASE OF USE	6/10
PERFORMANCE	9/10	VALUE	8/10

The hardware is sound and packaged well, so if you need to fit a camera to your Raspberry Pi, this is the one to get.

» **Rating 8/10**

PIBRELLA

Build a Python-based reaction game

Les Pounder goes back to the early days of the Raspberry Pi to look at a board that made a big difference to his career.



OUR EXPERT

Les Pounder is associate editor at Tom's Hardware and a freelance maker for hire. He blogs about his adventures and projects at bigl.es.

Pibrella is an awesome little board. We first came across it in 2014 while delivering a training course and since then we've used it with hundreds of learners. This £10 board may be eight years old, but it's still a great way to get to grips with electronics on the Raspberry Pi. In this tutorial we'll learn a little about this board, and create a reaction game to prove who's the fastest of them all!

Because Pibrella was designed for the original 26-pin GPIO Raspberry Pi it has to connect to the first 26 GPIO pins of any Pi. The first 26 pins are from the micro SD card. With the Raspberry Pi powered off, connect Pibrella so that it fits directly on top of the Pi. No part of Pibrella should "stick out" from the footprint of the Pi. The included rubber foot will need to be positioned so that Pibrella doesn't touch the Raspberry Pi. See the guide at <https://bit.ly/lxf289pibrella> for details.

For the project you'll need to place a push button into a breadboard, then connect the legs of one side to Input A of the Pibrella. It doesn't matter which leg goes to which part of Input A, as long as you only use one side of legs from the button.

When done, connect your keyboard, mouse, HDMI, Ethernet and lastly power which goes directly to Pibrella via the micro USB. Once everything's connected, boot to the Raspberry Pi OS desktop.

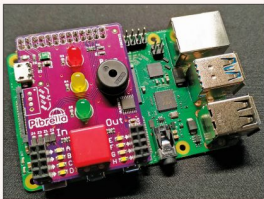
From the desktop, open a terminal and run this command to install the Pibrella Python library.

```
$ curl -sSget.pimoroni.com/pibrella | bash
```

Answer the questions and perform a full install and after a few minutes we're ready to create our project. But first, let's get to know Pibrella and the Python library. Pibrella was initially designed to make interfacing with the GPIO as simple as possible. Pibrella became the go-to board for educators in the early days of the Pi.

On the left of Pibrella are four inputs (A to D), and to the right are four outputs (E to H). The outputs can drive 5V DC motors, but only in one direction – there's no "flip flop" H bridge to change the polarity of the outputs. At the bottom of Pibrella is a large red button, and above that are three LEDs (red, yellow and green) and a simple piezo buzzer.

Pibrella's Python library is simple. Using high-level (human readable) functions, Pibrella is easy to use. Let's work through a quick example. Open your favourite



Pibrella will work with newer models of Pi. Just remember to place it on the first 26 pins of the GPIO.

Python editor – we chose *Thonny* because it comes pre-installed on the Pi. Create a new file, and for the first two lines import the **Pibrella** library and the **sleep** function from **Time**.

```
import pibrella
from time import sleep
```

Now let's create a for loop that will iterate 10 times.

```
for i in range(10):
```

Inside the for loop we'll use a function to turn all of the Pibrella LEDs (red, yellow and green) on at one, then wait for half a second, before turning them off and waiting for another half second.

```
pibrella.light.on()
sleep(0.5)
pibrella.light.off()
sleep(0.5)
```

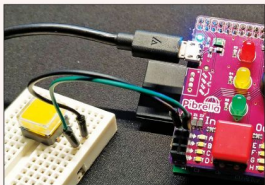
Save as **pibrella-test.py** and run the code. The LEDs will blink on and off. We can replace **pibrella.light.on()** with **pibrella.light.red.on()** to control just the red LED. Replacing red for yellow or green performs the same task for those LEDs. Armed with our new knowledge let's make a quick game.

Writing the project code

Create a new file and import three libraries. **Pibrella** and **sleep**, just like we did before, but we also import **random** that we shall use to generate a random number.

YOU NEED

- > Any Pi
- > Raspberry Pi OS
- > Pibrella breadboard
- > Button
- > Two M-to-M jumper wires
- > Code: <https://github.com/lesp/LXF-Pibrella/archive/refs/heads/main.zip>



Pibrella makes electronics simple. We can even use it with ScratchGPIO to make electronics projects in a block-based coding environment.

```
from time import sleep
import pibrella
import random
```

Create a variable, **time**, which will store a randomly chosen number between 5 and 10. This will make our reaction game a little more challenging because we don't know when the LED will light up.

```
time = random.uniform(5,10)
```

Using the randomly generated number we tell the code to **sleep** (pause) and then turn on the green LED, our trigger for the reaction game.

```
sleep(time)
pibrella.light.green.on()
```

Next, create a while True loop to continually run the code within.

```
while True:
```

Now we read the two buttons for player one and two, then store the values in variables **p1** and **p2**. Player one is connected to Input A, and player two is using Pibrella's button. For both buttons we use the loop to continually read the state of the inputs. By default they

will read "0" but when pressed they will read "1", as described by the following:

```
p1 = pibrella.inputa.read()
p2 = pibrella.button.read()
```

We next use a conditional test that uses two if conditions to check the value stored in the variables **p1** and **p2**. If **p1** (player one) variable is storing "1" then we know that they pressed the button first. If that is the case we'll blink the yellow LED using Pibrella's blink function. We set the duration on and off to be 0.5 seconds. Then the code will pause for five seconds before we break out of the main loop.

```
if p1 == 1:
    pibrella.light.yellow.blink(0.5,0.5)
    sleep(5)
    break
```

The same test is performed for player two, but this time the red LED will blink.


```
if p2 == 1:
    pibrella.light.red.blink(0.5,0.5)
    sleep(5)
    break
```

Pibrella's buzzer can produce simple sounds at varying frequencies. It won't produce high-quality music, but with a little practice you can get a tune out of it. Luckily for us the Pibrella Python library has two tunes – success and fail – that we can drop into our game. Now out of the while True loop, we're going to play a "success" tone to signal game over.

```
pibrella.buzzer.success()
```

To finish, we tell the code to pause for two seconds before turning all the LEDs off, which will signify that the game has ended.

```
sleep(2)
pibrella.lights.off()
```

Save the code as **reaction.py** and then run the game. After a random amount of time, the green LED will turn on and then it's "fastest finger first" to press your button and win the game! 

QUICK TIP

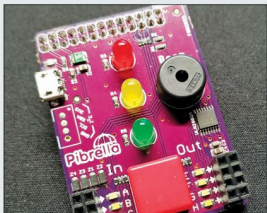
Powering Pibrella should always be via the onboard micro USB port. This provides power to the Pi and Pibrella, while also providing a small amount of protection when using high-current draw components such as motors.

» HATS, INSIDE AND OUT

Pibrella is from a time before the Raspberry Pi GPIO was expanded to 40 pins, and a time when boards had a specification to follow. The HAT (Hardware Attached on Top) standard was introduced with the Raspberry Pi B+ and as well as extra pins it brought a form factor standard which is still in use to this day.

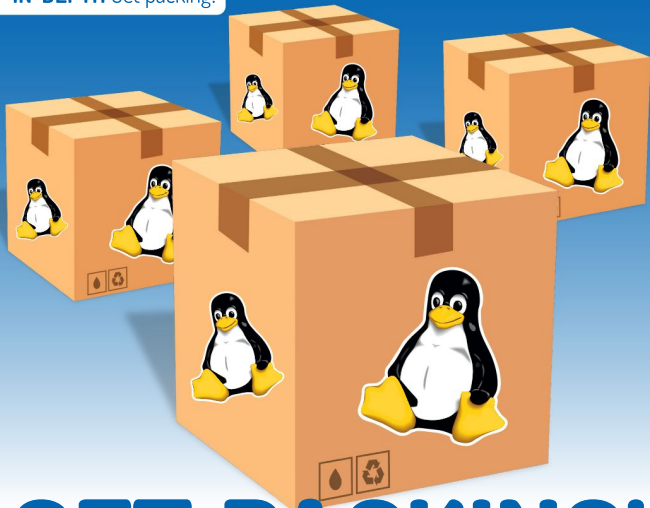
HATs must be 65mm wide, 56.5mm length and provide 8mm of clearance from the Raspberry Pi. The corner radius should be 3mm and at each corner there should be an M2.5 screw hole for compatible standoffs. There should also be cut-outs for the official Raspberry Pi camera (CSI) and display (DSI) ribbon cables to pass through under the HAT. An example of this is the Sense HAT, which has a cut-out for the camera.

To be officially called a HAT there needs to be a valid ID EEPROM at pins 27 and 28. This EEPROM identifies the board to the Raspberry Pi and tells the Pi how to set up the GPIO and other devices. Boards that follow the mechanical requirements of the HAT standard are not really HATs, but many are still called a HAT and the term has become part of the general language used by the community.



Four input, four outputs, some LEDs, a big red button and a rather annoying buzzer. A good deal for £10 some might say!

» GET YOUR Pi FILLING HERE Subscribe now at <http://bit.ly/LinuxFormat>



GET PACKING!

Keen to give back to the FOSS community, but don't know where to start?

Mike McCallister shows you one way to do a good deed.

Have you ever seen a nifty program in *Linux Format*, but couldn't find it in your favourite distro? If you were running a proprietary OS, you think that you'd have to ask the programmer to produce a version for you, but with FOSS that isn't how things work.

One of the wonders of the FOSS world is that you may not have to wait to run that program. Truth is that anyone can take any open source program and build a package from the source code. You can choose to run it on your machine. If it works well on your box, you can then submit your package to be included in the distribution of your choice.

If that sounds cool, if a little terrifying, here's a tool to make packaging easier: the Open Build Service (OBS). OBS enables anyone create a repository of source files

on a cloud server (or their own server) and build software packages for a variety of distros in popular formats: RPM, DEB, Flatpak and AppImage.

You don't really have to know much about code to make packages with OBS. If you have some experience installing and working with packages, or helped others with issues on a mailing list or forum, you might just be the type to be an OBS packager. Bonus points if you (a) have a GitHub (or similar) account, and (b) the command line doesn't make you queasy.

In these pages, we'll walk you through the OBS process. You may discover a new way of giving back to the FOSS community for all the fabulous software you use.

The OpenSUSE community created the Open Build Service to expand the pool of packagers contributing to OpenSUSE. They also wanted folks to make multiple

packages from a single source. Today OBS hosts more than 80,000 developers maintaining nearly 750,000 packages in 150,000 repositories (see System Status report 3/25/2022 at <https://build.opensuse.org>). Pretty successful project.

OBS makes it possible to import source code from another project into its repository. From there you give OBS the information it needs to build a package. In turn, you can install your newly created package on your system.

When you try this, follow these steps:

- Create an OBS account at build.opensuse.org and create your Project.
- Install the essential tools for packaging software and the osc shell client on your system and connect it to the OBS server.
- Fork and clone the source code of the project you want to package, and then connect to OBS.

- › Configure your packages (perhaps the hardest part).
- › Run the build.

Creating an OBS account isn't complicated. Head to the public instance of OBS at <https://build.opensuse.org>. Click Signup and fill out the form. Pick out a good username. Now login with your new credentials.

When you create your account, you receive project space on the server. Click Your Home Project on the left navigation bar to open your packaging dashboard.

Setting up your system

You can build packages strictly using the web interface, but if you'd rather work your magic on the command line on your own system, you'll download the osc client to your system.

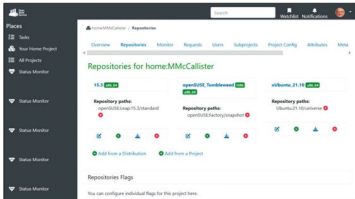
Install the client on any flavour of openSUSE or Ubuntu with your usual package manager. If you use another distro, try installing osc. If that doesn't work, try adding the `openSUSE:Tools` repo to your favourite package manager's source list. Chances are excellent you'll find the proper repository for your system. Heck, there's even a tool you can run on Raspbian. See http://download.opensuse.org/repositories/openSUSE:/Tools/<distro_version>/<architecture>.repo, where `<distro_version>` coincides with your system, and `<architecture>` is i386 or amd64.

When you have added the `openSUSE:Tools` repo, install the osc client for OBS. Make sure you also have these packages installed on your system: `gcc`, `make`, `python`, `bash`, `coreutils`, `diffutils`, `patch`. If you want to make RPMs, install these too: `rpm-build`, `rpm-devel`, `rpmflint` and `rpmdevtools`. If you've got Ubuntu and want to make DEBs, you'll need these: `gnupg`, `pbuilder`, `ubuntu-dev-tools`, `apt-file` and `dh-make`.

The first time you run osc, you'll connect to your Home Project on the remote OBS server with your OBS credentials (not your current system). The client will then create your `<OBS_username>.home` directory and add a configuration file, `.osrc`, to the system. You're now ready to try out OBS.

Pulling in the sources

To start packaging in the web client, return to your Home Project. Look for the Users tab. Click Add user, and declare yourself a Maintainer by checking the appropriate box. Click the Packages tab, and Create New Package. Add the Name of your package in this window. While not mandatory, you should also fill out a Title and Description for your package. The title (sometimes known as a Summary) is the one-line



description of the package that appears in the repo directory. The Description enables you more free rein to explain what the package does and how useful it is.

Next, find your program's development repository. You should be able to contact a developer or team via a program's website to learn where the repo is. If the program you want to package uses GitHub and GitLab, you can set up an account there, and fork the application to your repo. From there, you can clone the code to your computer. Before you leave here, copy the URL to this repository, for example <https://github.com/<username>/<appname>>.

Set up an Open Build Service (OBS) account, then get file storage and a Home Project.

RUNNING THE OBS NUMBERS

"Today OBS hosts more than 80,000 developers maintaining nearly 750,000 packages in 150,000 repositories. That's a pretty successful project."

Back to OBS>Project. Click the link to the package you created earlier. Click Add File. Paste the URL to your forked repo into the File URL box to upload the source files. OBS will wrap the code into a tarball (a `*tar.gz` file).

Click the Repositories tab to define one or more packages to make from your source code. You can select any or all of the available options, but you'll need any specific config files that each distribution requires.

OBS recommends checking both major openSUSE distributions if you're targeting the openSUSE project.



» BEHIND THE SCENES

Open Build Service does most of the actual package building on its own after you finish configuring the package. While OBS hides the machinations of package-building behind its web interface, here's what's actually happening:

In RPM-based distros, the RPM build system takes your program from its compressed archive (organised in the same directory hierarchy as you want to

install on the end-user's system) and places it uncompressed into the `/rpmbuild/BUILDROOT` directory. This directory is a "chroot jail," a protected sandbox that can't interact with the rest of the system.

When the build is complete, RPM wraps the contents of `BUILDROOT` into another archive with the `cpio` archiving tool. When the user installs the package,

RPM copies the files in the directories you specified.

For Debian and other DEB-based distros, you usually start with a "source package": just the tarball with the source code. The `debhelper`, or `dh-make` package reads the build files located in the `/debian` folder in the source package, especially the rules file, to produce the binary package that users can install.

so we've checked openSUSE 15.3 and Tumbleweed. Just for fun, let's pick Ubuntu too.

Here comes the paperwork!

Now comes the hard part: filling in the config files. We'll start with the RPM. How does OBS know how to build your software correctly? You tell it in the SPEC file.

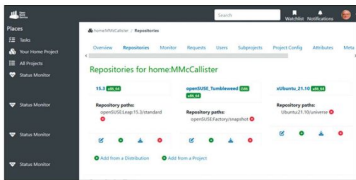
The SPEC file is a set of "directives" where you tell OBS about your program and how it should run. It has two sections: a Preamble with metadata, and a body containing the build instructions. You'll fill out the details of your package's SPEC file. This is best done on your system. SUSE contributed a way to generate a new blank SPEC file to use in packaging with OBS. Run `rpmdev-newspec <program name>` and open the SPEC file in an editor.

The top of the file (the "preamble") includes descriptive metadata information important to the user considering installing your software:

The first three fields, Name, Version and Release are required, and together form the elements of your package's file name.

- **Summary:** This single line offers a concise statement of what your application does.
- **Licence information:** What licence covers this package, whether it's the GPL, LGPL, MIT, or whatever.
- **URL:** A link to the package's website, where users can get more information on the program. There's also more technical information in the preamble.
- **Source0:** The path or URL that leads to your tarball.

Selecting a repository in OBS equals building a package for that distribution. The distributions you can choose from goes far beyond OBS sponsor OpenSUSE.



- **Patch0:** If the package includes a patch, include the URL for this patch.
- **Architecture:** The architecture of the processor your package will run on, such as `x86_64` for a 64-bit OS. Only if your code is compiled for a specific architecture.
- **BuildRequires:** The packages needed to create the build. If the program was written in a compiled language, like C, you may not need to fill this directive if you only need a shell/ubiquitous packages, to build the software.
- **Requires:** Another list of packages the user's system needs to have for the package to run.

The body of your SPEC file has just a few directives for the build system.

- **%description:** This is your marketing piece. Unlike the one-line Summary in the preamble, here you can include whole paragraphs of information.
- **%Prep:** A shell script or a set of commands to pull your project from its archive and prepare it for building.
- **%Build:** Assuming your program is not just a shell script, use this directive to identify the right commands to build the software to machine code or byte code.
- **%Install:** How to copy your build artifacts from where the build happens to `%rpmbuild/BUILDROOT` (more on this directory later).
- **%Check:** Commands to run unit tests and other sanity tests before releasing the package.
- **%Files:** The list of files to be installed.
- **%changelog:** If this package is an update/upgrade from a previous version, list the changes here.

Here is what a complete SPEC file for a C language program, from the RPM Packaging Guide: <https://rpm-packaging-guide.github.io/#working-with-spec-files>

```
Name: cello
Version: 1.0
Release: 1%{?dist}
Summary: Hello World example implemented in C
Licence: GPLv3+
URL: https://www.example.com/%{name}
Source0: https://www.example.com/%{name}/releases/%{name}-%{version}.tar.gz
Patch0: cello-output-first-patch.patch
BuildRequires: gcc
BuildRequires: make
BuildRequires: %description
```

The long-tail description for our Hello World Example.

```
%prep
%setup -q
%patch0
%build
make %? smp_mflags
%install
%make_install
%files
%license LICENSE
%{_bindir} %{name}
%changelog
* Tue May 31 2016 Adam Miller <maxamillion@
fedoraproject.org> - 1.0-1
- First cello package
```

Ubuntu (and Debian) packages are more complex to create than RPMs. Instead of a single SPEC file, you need a batch of text files that sit in a separate `/debian` folder of the package. In the directory containing your tarball, create a `/debian` folder. Open your editor and create these required files:

» OBS PACKAGING RESOURCES

Want to learn more about packaging and the Open Build Service? The OBS Documentation space (<https://openbuildservice.org/help>) has manuals, videos and slide shows to help you use the system.

OpenSUSE's Build Service portal (<https://en.opensuse.org>) **Portal:Build Service** offers information for new packagers and developers. Do check out the Build Service Tutorial at https://en.opensuse.org/OpenSUSE:Build_Service_Tutorial, too.

The RPM Packaging Guide at Github <https://rpm-packaging-guide.github.io> walks you through a standard process, minus OBS.

Red Hat offers a chapter of its manual on packaging and distributing software. See <https://red.ht/3vIOR3d>

Debian provides help at <https://wiki.debian.org/packaging/intro> for packaging newcomers. Seven ways to set up Debian Unstable (aka sid): <https://wiki.debian.org/Packaging/Pre-Requisites>.

Finally, Ubuntu will also help you with its Packaging Guide at <https://packaging.ubuntu.com/html/index.html>

Debian.control: This file contains all the metadata of the package. The organisation of the file is similar to a SPEC file, but many of the fields are different. More on this later.

PackageName.dsc: Pulls some of the metadata from the control file to help the OBS build tool find the tools they need to complete the job.

Debian.rules is the most complex file. If we were still doing things the old-fashioned way, you'd probably already know by looking at it: this is a Makefile which compiles the code and turns it into a binary package. Fortunately, most of the work is automatically done these days by the debhelper package we installed earlier. The universal % Makefile target just runs the dh script which will run everything needed.

The optional stuff

➤ **Debian.changelog:** only needed to describe changes in your package since the last version.

➤ **Copyright:** follows the licence of your program.

➤ **Docs:** contains any documentation files for your program that belong in the final package.

➤ **Source/format:** describes the version format of the source package and should always be 3.0 (quilt).

The control file

➤ **Control:** The first paragraph describes the source package.

➤ **Source:** The source package name.

➤ **Maintainer:** The name and email address of the person responsible for the package – presumably this is going to be you!

➤ **Priority:** The priority of the package (one of 'required', 'important', 'standard' or 'optional'). In general a Debian package is 'optional' unless it's 'essential' for a standard system to boot or connect with a network.

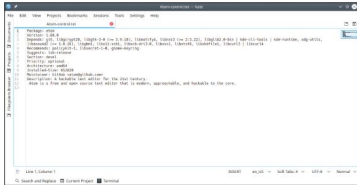
➤ **Build-Depends:** The list of packages needed to create the build, if your program's language is compiled. You may not need to fill this field if you only need a shell, or similarly ubiquitous packages, to build the software. The second and following paragraphs describe the binary package(s) to be built.

➤ **Package:** The name of the binary package. The name might be different from the source package name.

➤ **Architecture:** The architecture of the processor your package will run on, such as x64 for a 64-bit OS. Debian works on about a dozen computer architectures in total, so this architecture support is crucial.

➤ **Depends:** The list of packages that must be installed for the program in the binary package to work.

➤ **Description:** The full description of the binary package. As with its RPM counterpart, it should be



This **debian.control** file is critical to building packages for Debian and Ubuntu packages. You have to create, edit and upload it to OBS.

helpful to users, but is also a bit of marketing. The first line is used as the short synopsis (summary) description, and the rest of the description must be an independent longer description of the package.

When you're finished editing the **debian** files, upload them into in your OBS project

Building the packages

When you've completed the configuration files and uploaded them to your project, OBS automatically builds the packages and stores them in their respective repositories. This can take time, depending on the load on the public OBS servers. If you're anxious to see the results, building your package on your local system can go faster.

To ensure that your RPM file is up to quality standards, OBS runs your package through *RPMLint*, which will identify any problems.

Any package you make is now stored on the download.opensuse.org site, available to any openSUSE user through the package search. Others can find it too, and if you built a package for them, they can install it.

In openSUSE, add your repository to YaST Software Repositories:

- 1 Open YaST.
- 2 Click Software Repositories.
- 3 Click Add.
- 4 Choose Specify URL from the list. Click Next.
- 5 Type a name (My OBS Repo) and the complete URL. Note that you can't add the repo until there's a package stored there.

YaST will check your type of repository and check its license before adding it to your system. For Ubuntu/Debian users, you can add your repository to the apt sources.list.

First, open your browser to identify the mirrors where your repository is stored: http://download.opensuse.org/repositories/home://<username>://<project>/Ubuntu_22.04/Packages?mirrorlist

Pick a mirror from the list, and add it to **/etc/apt/sources.list**. To install your new package, run your package manager as you would for any other package.

And there you have it. You have joined the community of Linux software packagers, and taken the first step toward becoming a package maintainer. Maintainers commit to keeping that package updated for their distro by working with the upstream developers to keep the software up-to-date. They are important to any vital distribution, and often honoured members of the development team. **EXP**



The OBS shell client, **osc**, generates this SPEC file. However, you have to fill in the values and upload it to your OBS project.



Bringing stories to the command line

Shashank Sharma knows that the Linux CLI, once seen as the domain of the uber geek, also serves as the pathway to the myriad worlds of stories.



OUR EXPERT

Shashank Sharma is a trial lawyer in Delhi and an avid Arch user.

A popular quote, often attributed to Albert Einstein, goes, "If you want your children to be intelligent, read them fairy tales. If you want them to be more intelligent, read them more fairy tales." Teddy Roosevelt was just as much a fan of reading, and is quoted to have said, "Now and then I am asked as to 'what books a statesman should read,' and my answer is, poetry and novels – including short stories under the head of novels." With many seminal works now available in the public domain, and ebooks being churned out with incredible passion and frequency, for all this reading to be done, one needs the right tools for the job.

In addition to *epy*, which is an incredibly robust ebook reader that support a multitude of popular

formats, we'll also discuss *ebook-convert*, (see *below left*) a nifty utility that can be used to convert ebook files from one format to another.

Easy as pie

You might think that the choice of heading here has something clever to do with *epy*, but alas that isn't so. We settled on that heading because of the alliteration.

With the exception of Arch's AUR, you won't find *epy* in the software repositories of most popular desktop distribution. But that's no cause for worry because the installation is fairly straightforward. If you already have *pip* package manager installed, you can install *epy* with the `pip3 install epy-reader` command. Depending on your system configuration, you can either append `sudo` at the start of the command to install it for all users, or instead use the `pip3 install epy-reader --user` command to install the tool in the `~/local/bin` directory.

Released under the GPLv3 license, *epy* was forked by the author from his previous *epd* utility, which itself was released under the MIT licence. Unlike the parent project, *epy* boasts several features such as bookmarks, external dictionary integration and URL support.

If you already have ebooks at hand, you can start reading immediately by running the `epy /path/to/`

» CONVERTING EBOOKS

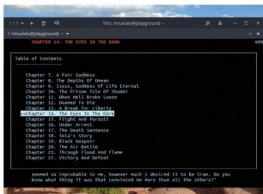
If you read ebooks on various devices such as tablets, desktop or Kindle, then it makes sense to retain ebooks in the format supported by all devices. While most desktop ebook readers support various popular ebook formats such as EPUB, MOBI and PDF, some devices such as Kindle are far more limited. It makes sense to have all your ebooks in the same file format so that they can easily be moved between devices per your convenience.

Thankfully, a nifty command-line utility does this trick. If you work with ebooks, you might already be familiar with *Calibre*, a popular graphical ebook manager. In addition to doubling as an ebook viewer, you can also use *Calibre* to download news and magazines from the web, share and backup your library, convert between ebook file formats, edit the metadata on your books and more.

You'll find *Calibre* in the software repositories of most popular desktop distributions. Once installed, you can use the included *ebook-convert* command-line utility to quickly convert files from one format to another.

To convert an EPUB file to MOBI format, run the `ebook-convert 2\ \gods-of-mars.mobi 2-Gods-of-Mars.epub` command.

Depending on the source ebook, you might wish to adjust the base font size, or move the table of contents (TOC) from the start of the file to the end. These adjustments and more can also be done with *ebook-convert*. Refer to the project's man page for more details.



The progress at the top-right of the screen is only visible if your terminal emulator window is wide enough. Press `s` to toggle it on or off.

ebookmobi command. Apart from the mobi format, **epy** also supports epub as well as azw and fb (fictionbook) formats.

When reading an ebook, you can access the table of contents (TOC) at any time by pressing **t**. The **epy** project boasts of a number of keybindings, which you can access by pressing **?**, but we'll list a few useful ones to help you get the most out of the reading experience:

Keybinding	Function
t	Open table of contents
c	Change colour
n	Go to next chapter
p	Go to previous chapter
d	Define word
b	Add a bookmark
B	List bookmarks
/	Regex search
M	View metadata information

Start reading

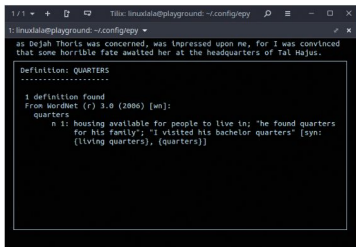
The **epy** tool remembers your reading history, so you don't have to specify the complete path to the ebook each time you launch **epy**. Instead, you can run the **epy** command, and the project automatically opens the last ebook you were reading, at the location where you left off. Even better, when you specify the path to a book, **epy** will open at the position where you left off. For instance, if you've just started chapter seven of book A, and then read three chapters of book B, when you open book A again, **epy** will open it at chapter seven.

You can access your reading history by running the **epy -r** command:

```
1 17% 11:29PM Apr 01: Works of Edgar Rice Burroughs - Burroughs, Edgar Rice...
2 3% 10:15PM Apr 04: Thuvia, Maid of Mars - Edgar Rice Burroughs (/media/L...
3 20% 03:27PM Mar 27: The Gods of Mars - Edgar Rice Burroughs (/media/linux...
4 31% 05:57PM Apr 01: Swords of Mars - Edgar Rice Burroughs (/media/linux...
5 10% 03:27PM Apr 07: Llana of Gathol - Edgar Rice Burroughs (/media/linux...
6 35% 03:27PM Apr 07: A Princess of Mars - Edgar Rice Burroughs (/media/lin...
```

Each book in history is allotted a number. You can quickly open a book from the library using the assigned number with the **epy <num>** command. For instance, the command **epy 4** opens the Swords of Mars book from our reading library. The **epy -r** command also shows the progress as well as the date and time when the file was last accessed.

If you don't like the idea of repeatedly looking up the reading history to confirm the assigned number before opening a book, you can also use a match string to inform **epy** of the book you wish to read with the **epy <string>** command opens the book that matches the specified search string. From our reading history, the



command **epy princess** opens the first book in the series, while **epy swords** opens the eighth book in the classic pulp fantasy series.

Configuration

One of the greatest joys of reading is expanding one's vocabulary. But this only works if you can quickly look up the meaning of new words in a dictionary. Thankfully, **epy** supports tools like **dict** and **wkdict**, which you'll find in the software repositories of most popular desktop distributions. Once installed, you can edit the **~/config/epy/configuration.json** configuration file and make the necessary changes. Look for the "DictionaryClient": "auto", line and change "auto" to the name of the dictionary tool you installed. Because we installed **dict** on our test machine, we changed the line to "DictionaryClient": "dict".

When you now press **d** while reading a book, you'll be prompted to enter the word that you wish to look up in the dictionary.

You can also change the default keybindings by editing the **~/config/configuration.json** file, which is also home to various other editable parameters. For instance, the **epy** tool utilises an animation when scrolling pages in the book. If you find the feature distracting, open the configuration file in your favourite editor and change the "PageScrollAnimation": true, line to "PageScrollAnimation": false.

By default, the tool displays the reading progress at the top left corner, but this too can be turned off by changing the "ShowProgressIndicator" line in the config file.

In addition to the variety of ebook formats, **epy** also supports working with URLs, so you can read books directly from Project Gutenberg without downloading them first. The command **epy https://gutenberg.org/files/1268/1268-h/1268-h.htm** enables you to read Jules Verne's The Mysterious Island without having to download an EPUB or MOBI format. Unfortunately, this feature only works for books that have been properly formatted in HTML, so you won't be able to indulge in fan fictions using **epy**. **157**

Even if you make no changes, go through the **~/config/epy/configuration.json** file for an idea of the features on offer, such as text to speech and mouse support.

QUICK TIP

In addition to Project Gutenberg, there are other communities that provide access to public domain works. See Global Grey Books (www.globalgreypebooks.com) or Standard Ebooks (<https://standardebooks.org>). If you're interested in scholarly or academic works, see HathiTrust (www.hathitrust.org), a not-for-profit collaborative efforts of various academic and research libraries.

» ENHANCE YOUR TERMINAL-FU Subscribe now at <http://bit.ly/LinuxFormat>

CZKAWKA

Credit: <https://github.com/qarmin/czkawka>

Quickly track down & delete unwanted files

Nick Peers takes a deep dive into this fast-evolving and brilliant tool for finding and removing redundant files from your PC.



EXPERT

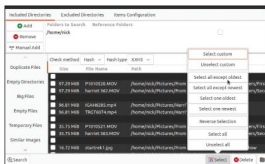
Nick Peers was unsurprised – but still unnerved – by the thousands of potentially redundant files that Czkawka has found on his hard drives.

Keeping your hard drive clean can feel like an uphill struggle as it fills up with detritus. Weeding through your files by hand is a painstaking task, and this is where a search-and-delete tool like Czkawka comes riding to the rescue.

Czkawka can do so much more than find duplicated files. It can also track down similar images and videos, music duplicates, broken files (and symlinks), empty files, empty folders, and more. It's written in Rust, uses caches to speed up follow-up scans, and is available both via the Linux desktop and as a CLI application.

Getting started

Czkawka can be installed in various ways: via snap, Flatpak (<https://flathub.org/apps/details/com.github.qarmin.czkawka>) or you can download the AppImage (<https://github.com/qarmin/czkawka/releases>). If you install it through snap, you'll need to give Czkawka access to all your drives:



Manually selecting files by hand can be time-consuming. Let Czkawka's automated select tool lend a hand.

```
$ sudo apt update && sudo snap install czkawka
```

```
$ sudo snap connect czkawka:removable-media
```

If you plan to use Czkawka's Similar Videos tool, you'll also need `ffmpeg`. If you get a warning when you attempt to use the tool, simply install it thus:

```
$ sudo apt install libgtk-3-dev ffmpeg
```

You'll also need to install an ALSA lib package to include music files in the Broken Files Finder search. On Ubuntu, use the following command to install the missing package:

```
$ sudo apt install libasound2-dev
```

Start your searches

If installed through snap, launch Czkawka via the application launcher. As the annotation (see right) reveals, it's split into various sections, covering the selection of files, which file-cleaning tool to use, reviewing your results and finally performing actions to clean up your files.

Most of these tools are self-explanatory: Duplicates, Empty Directories, Big Files, Empty Files, Temporary Files, Similar Images, Similar Videos, Music Duplicates, Invalid Symlinks and Broken Files. We'll cover them in more depth shortly.

First, select which directories to include in your scans. By default, only your home directory is included. If you've launched Czkawka without admin privileges, you'll be restricted to only those folders and files your user account has access to. This is a good thing: Czkawka should primarily be viewed as a tool for helping

» TERMINAL USE

If you'd like to incorporate Czkawka into batch scripts, or run it from the Terminal, you'll first need to install the `czkawka_cli` command-line tool, which requires compiling and installing separately:

```
$ sudo apt install -y curl git build-essential
$ curl -proto -https --tlsv1.2 -sSf https://sh.rustup.rs | sh
$ sudo apt install -y libgtk-3-dev
$ git clone https://github.com/qarmin/czkawka.git && cd czkawka
$ cargo run --release --bin czkawka_cli
```

Once done, you'll be able to use `czkawka_cli` by following the standard syntax:

```
$ czkawka_cli tool-flag options
```

The CLI version supports 12 tools, comprising all the functionality of the main program: `big` for large files, `dup` for duplicates, `image` for finding similar images, and so on.

The CLI doesn't offer any interactivity, so by default each tool will simply perform the required scan, then display the results of its findings in the Terminal window. You'll need to re-run the command using the `-D` flag to remove files (for example, `-D AEN` would delete all files except the newest version). You can also save the results to a file using the `-F` flag (so `-F <filename>`) for reviewing more carefully before making any firm decisions.

you streamline your personal files by weeding out duplicates and enabling you to finally tackle those folders packed full of similar images.

With this in mind, you may not need to extend your search any further than your home directory. Indeed, you might prefer to focus your search on specific directories (Documents, Pictures, Music and so on) within your home directory, in which case select your home folder and click Remove before adding your choice of subfolder(s) via the Add button.

On the other hand, if you're searching for duplicates, but want to designate a specific folder as the 'source' folder (in other words, files from this folder are kept no matter what), then you'll need to select this folder and tick the box next to it. Doing so identifies that folder as a 'Reference folder', which means its files are considered the 'originals' and untouchable.

Track down duplicates

As its name implies, the Duplicate Files tool enables you to search all specified folders for files that match exactly by file name, size or hash. Use the 'Check method' drop-down menu above the (currently empty) results pane to choose what you're searching for.

Choosing a file name is obviously the least reliable – files are matched by name only, not size or content, so Czkawka will pair completely different files that happen to share the same filename. File size is more accurate and quicker than matching by hash, but again there's no guarantee that two files of the same size are identical.

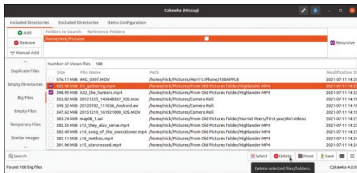
The safest option is to stick to the 'hash' option, which is the slowest – but most accurate – way of rooting out duplicates. The method combines the file size search to ensure only those files identically sized are checked with the hash, which determines that the contents are identical. Selecting this option also gives you a 'Hash type' drop-down menu offering three choices: Blake 3, CRC32 and XXH3. All three will identify duplicates, but XXH3 is quicker than the other two. On fast drives with small folders, the default Blake3 is fine, but otherwise consider using XXH3.

Once you've made your choices, click the Search button in the bottom pane. A list of duplicates will be presented in the results pane within seconds. You'll be shown three columns of information: the file name, its path and – be prepared to expand the program window to reveal this – the modification date of each file.

If you need more help identifying the file, click it if it's an image and a preview pane will appear. You can then click the other duplicates to verify they're the same file. Otherwise, try double-clicking a file to bring up an 'Open with' dialogue, enabling you to select a suitable application to review it (and its alleged duplicates).

Once you've reviewed the files, what then? Assuming you want to remove some or all of the duplicates, you have two basic options. Either you can manually go through the list ticking the box next to each item you wish to remove. Alternatively, suppose there's a lot of items to process, and they share similar characteristics. If they're the oldest or newest file in the selected group, or they share a specific path and filename, for example, you can automate the selection process.

To do this, click the Select button to reveal a pop-up menu: while the Select All button isn't relevant here, the other options are: you can quickly select the newest or



Looking for a quick and easy way to free up drive space? Czkawka's Big Files tool could help.

oldest file in each group or all but the oldest or newest file in groups of three or more duplicates. Most intriguing is the 'Select custom' option. This enables you to choose based on name, path or Regex Path + Name (for more complex suggestions).

Once you've made your choice, click OK and see which files have been selected – you can then manually fine-tune this list or use the Select button to reverse or even wipe clean ('Select none') the selection to start the selection process again.

Process duplicates

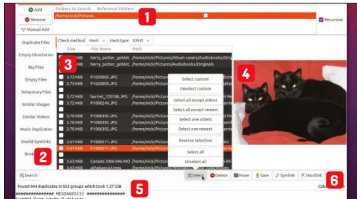
Czkawka is nothing if not flexible, so you have several options when choosing what to do with those duplicates. Delete will remove all selected files from your hard drive, or you could click Move instead to move them all to another directory or drive, just in case you need to restore one or more later.

The Save button creates a text file listing your search results, stored in the root of your home folder as **results_duplicates.txt**, enabling you to review the results at your leisure. If you plan to move files instead

QUICK TIP

Different cache files are used for different scan settings. You can access these for by clicking the spanner icon to open its settings, followed by 'Open cache folder', found under General.

EXPLORING CZKAWKA



1 Selection panel
Choose which folders to include in your hunt for unwanted files here. Tick Reference folders to protect files in that folder from being targeted for deletion.

2 Tools list
Select which of Czkawka's 10 tools you want to use from here.

3 Results window
After your scan is performed, duplicates and other potentially removable files are listed here ready for selection.

4 Preview pane
When reviewing image files, click one to bring up a preview, which will enable you to perform visual comparisons.

5 Notifications pane
The results of your searches and actions are displayed here, together with any helpful (or unhelpful) messages.

6 Actions
After using the Select button to help choose which items to process, use these buttons to perform specific actions.

QUICK TIP

Click the spanner icon to access Czkawka's settings. If you're nervous about accidentally deleting the wrong file, tick 'Move deleted files to trash' and click Save configuration to add a fail-safe step against a potentially devastating mis-click.

of deleting them, this file will help you put back those files later if necessary.

Suppose you're worried that deleting duplicates could have ramifications for other applications. In that case, another option might be to replace unwanted duplicates with symlinks or hard links, which will free up drive space, but ensure no application loses access to the file. You'll need to select at least two items in the group, and the first item in the list will be left unchanged, with any other selected items being replaced by symlinks or hard links as required. The good news is that you're not forced to apply one option to the entire list. Simply select your target items for one action, process them, then move on to the next set of items.

Locate empty items

Czkawka offers two options for locating – and removing – empty files and folders, one for each type. As with duplicates, a list of any offending items appear, and you can then double-click to open them to verify they're empty before proceeding to delete or move them. If an empty file is linked to an application you've not removed – such as an empty log file – use DuckDuckGo to make sure you won't break anything by removing it.

If you're looking to free up space in a hurry, you'll want to target the biggest files in your collection. Czkawka's Big Files search will list the 50 largest files it finds by default, but you can adjust this figure using the 'Number of shown files' field above the results pane

before clicking Search. Again, files can be deleted or moved only. The Temporary Files search works similarly, except it targets what the author describes as the 'most basic ones' only. See the box (below) for his recommended alternative for removing potentially redundant system files.

At the bottom of Czkawka's list of files it can fix are tools for rooting out invalid symlinks and broken files. The Invalid Symlinks tool will display the symlink file name and folder, its destination path and the error type.

The Broken Files tool locates corrupt files and those with invalid extensions. Three file types are supported: images, archives and (if you've installed `libasound2-dev`) audio. Czkawka tests each file by attempting to open it, assuming if an error is generated, the file is corrupt or unsupported. Where errors are found, try opening the file yourself because it's not a foolproof tool.

Weed out visually similar images

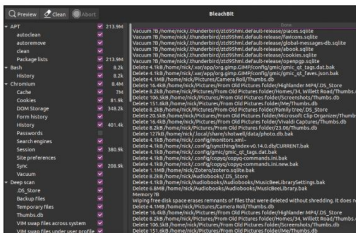
Digital photography may have eliminated weak photos, but chances are that's at least only partly true because you're smart enough to take four or five shots each time, ensuring at least one will be usable. But it's all too easy to leave the unwanted photos in place, even as your hard drive steadily increases.

At some point you should take the time to go through them all, eliminating the unwanted shots. And that's where Czkawka's Similar Images tool comes in, enabling you to group visually similar photos (such as resized versions or those containing a watermark) to choose which one to keep.

The walkthrough (opposite) reveals how it works: Czkawka generates individual hashes for each image file using your choice of hash type, hash size and resize algorithm. All but one hash type resizes the original image to produce a smaller cache according to the hash size (8x8 by default, but up to 64x64, which means more accurate comparisons, so fewer matches, but larger cache files and slower scan times).

There are also four resize algorithms to choose from when resizing your image hashes. Of these, the worst by far is Nearest, but otherwise there's not a huge difference. For most people, the defaults (Lanczos3, hash type gradient, hash size 8) are sufficient.

Czkawka enables you to experiment with different scan settings. It keeps separate cache files for each setup, so once the initial scan is performed for each, subsequent scans are much quicker. The latest version



▶ Giving your system files a proper cleanout? Then consider pairing Czkawka with Bleachbit.

» CLEAR TEMPORARY FILES

One thing that Czkawka (currently) doesn't do well is rid your hard drive of system and temporary files.

Its tool simply targets files based on their file type: **# thumbs.db**, **.bak**, **-.tmp**, **.tmp**, **.ds_store**, **crdownload**, **.part**, **.cache**, **.dmp**, **download** and **partial**.

If you're looking for a tool that can do a better job of removing such useless flopsam and jetsam, pair Czkawka with (see the grab above) Bleachbit, which can

hunt out temp files across your system and a range of programs, including your web browser(s).

Avoid the version shipped with your distro, and instead download the latest version as a `.deb` direct from the author (www.bleachbit.org/download/linux). Once installed, it can be run under your normal username as well as the root user (choose the 'as admin' version from the launcher). From here, tick the parts of your system you wish to clean and then

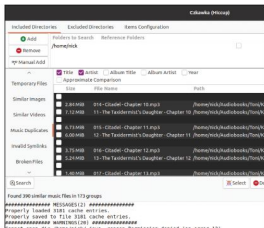
click Preview to see what it can find.

If you installed Stacer (<https://oguzhaninan.github.io/Stacer-Web/>) following our tutorial in LXF284, then its System Cleaner tool is another option worth trying. This focuses on package and application caches, crash reports, application logs and the trash. It's not as comprehensive as Bleachbit, but it's still a good step forward from what Czkawka's limited temporary files cleaner can achieve.

of Czkawka also debuts a tool for comparing videos to help you weed out similar ones similar to the Similar Images tool. Here, your options are much less: a simple similarity slider plus an option to exclude files of the same size (typically exact duplicates) from the results.

Beneath this is the Music Duplicates button, which scans MP3, FLAC and M4A audio files for duplicates based on your choice of tags: title, artist, album title, album artist and year. An 'Approximate Comparison' option uses AI to remove parentheses from phrases (such as remixes or live versions) to provide you with a list of multiple versions of the same song.

Czkawka continues to enjoy lots of care, and attention from its author – the recently released version 4.0 added similar video support reference folders (to protect a single folder from any changes) and revamped the tool's performance through support for multithreading. It's got a bright future, and your hard drive will be all the better (and emptier) for it. **LXF**

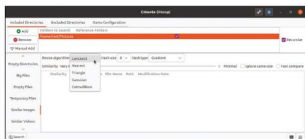


Czkawka's Music Duplicates tool is a little crude, relying on metadata to determine if files are matches or not.

QUICK TIP

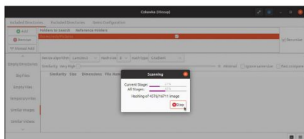
The Czkawka website (<https://github.com/garmin/czkawka>) is packed with useful information, including pages in the instructions folder that make up a manual of sorts.

FIND VISUALLY SIMILAR IMAGES



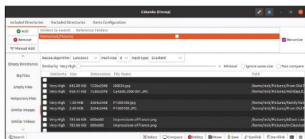
1 Set search options

Select Similar Images and choose the folder or folders containing all your images using the Add button. Above the results window you'll see a series of options – the three drop-down menus at the top affect the speed (and quality) of the scan process. Leave the defaults as they are for now (see the body copy for tips on tweaking these).



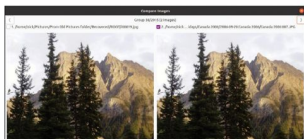
2 Fine-tune your search settings

The Similarity slider enables you to determine how closely matched each image must be. Leave it set to 'Very high' for now. Tick 'Ignore same size' only if you want to ignore exact duplicates, and leave 'Fast compare' unticked, too. Click Search and wait while Czkawka performs its initial scan of your files to determine potential matches.



3 Scan and tweak

The initial scan will take some time to complete, but a cache file is generated that will speed up future rescans with different settings if they're required (you can also tick 'Fast compare' if you move the similarity slider right to speed things up, too). If no results are returned, verify you've not ticked 'Reference folder'. Finally, a list of matches should be displayed.



4 Review results

Select each result in turn and you'll see a preview appear in the right-hand pane. Click the Compare button beneath to view the matched photos side-by-side in a separate window. You can then tick the ones you wish to remove or use the Select button for batch selections (you'll notice an additional option to select all but the biggest/smallest in addition to the usual choices).

Credit: www.seasp.info/Univ/Joyce

Emulate the classic Amstrad PCW

Les Pounder goes back to school, a time when his form room was full of Z80 computers and noisy dot matrix printers.



OUR EXPERT

Les Pounder is associate editor at Tom's Hardware and a freelance maker. He blogs about hacks and makes at bigles.

At high school we had a business studies room, next door to our IT department (Goldstar 286 PCs and the RM Nimbus!), which doubled as a form room. Everyday we sat there for registration and important messages, but sometimes we were let loose on the computers. However, these weren't x86 machines. Rather, they were Amstrad PCW8256 and 8512 powered by the mighty Zilog Z80 processor.

Amstrad, founded by Alan Michael Sugar (AMS-Trading) in 1968 was a well-known manufacturer of low-cost computers and consumer technology. Sometimes its consumer tech was maligned, but in the realm of computing Amstrad had success and gained a strong following. The company would later purchase the Sinclair brand from Sinclair Research (which saw new models of the ZX Spectrum being released).

Amstrad had a history of producing cheaper computing hardware, and in the case of 1985's PCW8256 the £300 asking price (adjusted for inflation this is approximately £1,000 today), which was a steal compared to Apple's \$2,600 Macintosh Plus. But in classic Amstrad fashion, the PCW range was initially business focused and this saw a few cutbacks, and proprietary additions to the package.

As you can probably guess, the PCW 8256 has 256KB of RAM, while the 8512 came with 512KB. This was plenty for the era and the industry standard CP/M operating system. CP/M (Control Program/Monitor) was created in 1974 for Intel 8080-based machines. Loaded via a three-inch floppy disk, incompatible with 3.5 inch disks of the time, CP/M provided a basic OS from which we could launch programs and manage files.

Going back to the disks, the choice for a three-inch drive was based on it having a simpler electrical interface. This meant that users had to purchase bespoke disks for their PCW, and for a short time these disks were hard to come by. The PCW 8256 and 8512



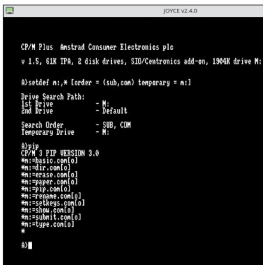
Designed for the office, the PCW8512 was a compact and cost-reduced business powerhouse that took on Apple for early DTP projects.

shared the same basic aesthetic: an all-in-one screen and disk drive (some with dual drives) with an 82-key keyboard connected via a DIN-type connection. The bundled dot matrix printer cemented the focus on home office/business use, but it also supported a popular desktop publishing scene.

The 8256 and 8512 were replaced with the 9256 and 9512, which saw a slight change in form factor to a more "PC" looking unit, and we saw 3.5-inch floppy disk drives. In 1995 we saw the PCW16 released at a time when PCs were claiming dominance over the home computer market. While the PCW16 was interesting, it wasn't backwards compatible with the older models,

QUICK TIP

Hidden in plain sight is Joyce's menu. To show the main menu press F9 and then navigate using the mouse or keyboard. To boot from an alternative disk press F3 and then follow the standard disk menu.



It may not look like much, but CP/M is an easy-to-use and powerful operating system for Z80-based computers.

and was under-powered for the time. This was the last model of PCW released, and Amstrad focused more on consumer tech until its purchase by Sky in 2007.

Emulating the Amstrad PCW

The best emulator that we found for this task was Joyce (www.seasip.info/Unix/Joyce). Installation is tricky because there's no installation candidate in the repositories and so installation involves downloading, extracting and compiling the emulator, but it does have a few dependencies that we need to address: SDL, libpng and libxml2. All of the files are in the Ubuntu repositories. Download Arculator 2.1 for Linux. We couldn't do this via a browser, so we used wget.

```
$ wget http://www.seasip.info/Unix/Joyce/joyce-2.4.0.tar.gz
```

Extract the files to a directory. Navigate to the extracted files directory via the Terminal, update your repositories and then install the dependencies:

```
$ sudo apt update
$ sudo apt install libsdl1.2-dev
$ sudo apt install libpng-dev
$ sudo apt install libxml2-dev
```

In the same directory as the extracted files we configure Joyce based on our specific system:

```
$ ./configure
```

Then build the software into an executable.

```
$ make
```

We must run a check to ensure our build is correct:

```
$ make check
```

Finally we install Joyce to our system. Note that we need to use sudo to do this.

```
$ sudo make install
```

Depending on your machine this can take a few moments to complete, but once done we'll have Joyce installed on our system.

There are two ways to invoke Joyce. The first is a windowed version:

```
$ xjoyce
```

The other is a full-screen emulator.

```
$ xjoyce -f
```

Using CP/M

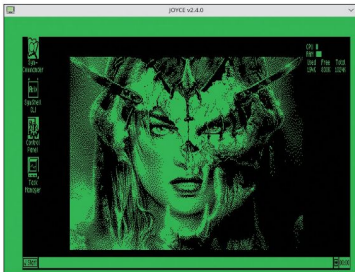
We used the standard `xjoyce` command to load a windowed version. On first boot Joyce will ask us to set up a boot disk. Select "Set up boot discs properly" and then in the next menu select Disc File... Our disc file is CP/M.

CP/M was the only operating system that came with the PCW8256/8512 machines. Given the low spec of the machines it's a natural fit. CP/M is still under copyright, despite being very old. You can find .disk images for CP/M to work with Joyce online, but finding them is left as an exercise for the reader. Copy the CP/M

QUICK TIP

In Mallard 80 BASIC we can tidy up our code, ensuring that line numbers are equally spaced by using the `RENUM` command. This also adjusts the line numbers that are referenced in other sections of your code.

A multitasking OS for CP/M-based computers. SYMBOS is fantastic. It eats a lot of RAM, but for the "cool factor" it can have all the RAM it wants.



» TAPPING INTO THE POWER OF SYMBOS

In the 1980s, the Z80 reigned supreme. It was everywhere, and the processor can still be found in industrial and commercial machinery. But while the Z80 was a beast of a CPU, it couldn't manage to run a graphical OS. Until now.

SYMBOS (Symbiosis Multitasking Based Operating System) is a multitasking OS for the PCW, CPC, MSX and Enterprise 64/128 computers. Considering that SYMBOS is meant for

4MHz Z80s, this is an awesome programming achievement. Taking the PCW shortcomings into consideration, any type of graphical output is amazing, but SYMBOS is truly multitasking, with games and applications running at once! Created back in 2000, SYMBOS is still being actively developed, with a community of coders tinkering on additions and enhancements to this free download.

SYMBOS comes with everything a desktop of the era should expect. A notepad, image viewer, control panel – all provided by a Windows 3.1-styled user interface. There's also a basic video player. Don't expect crisp 1080p *Big Buck Bunny*, more of a low-res slideshow, but achieving this on a Z80 is truly amazing.

SYMBOS can be downloaded from www.symbos.de and it works with the Joyce emulator.



It took us longer than normal to write this feature, because Sky War is such a good River Raid clone.

disk image to `~/Joyce/Disks/` and in the Joyce emulator navigate to that location and select the disk image. Click OK to load the disk. When prompted to give the disk a short name, type `CP/M` and press Enter.

Joyce will now open a general user interface, and we can see that `CP/M` is option 1. Press 1 to insert the `CP/M` disk and boot. After a few seconds the `CP/M` prompt will appear, and it has a look and feel of early MS-DOS. `CP/M` is a fun operating system to tinker with, and you can learn more about the commands at www.primrosebank.net/computers/cpm/cpm_commands.htm.

For now let's turn our attention to something more BASIC – pun intended!

Writing some BASIC

Amstrad's PCW range is not well known for their BASIC prowess, which is a shame given how much we loved hacking around with BASIC on the CPC 464. For the PCW range, specifically `CP/M`, we have Mallard-80 BASIC and to start the BASIC interpreter we need to type the following:

```
basic
If we ever want to leave the interpreter, type
system
and then press Enter.
```

Okay, let's flex a little BASIC muscles, we've done this a few times on many different machines but we start as always with the ol' `10 PRINT` project. Each line of BASIC code for a project will start with a number, `10`, `20`, `30` and so on. This tells the interpreter the sequence of code: it jumps from one line to the next in ascending order. But why do we do this? Quite simply, if we make a mistake and miss out a line of code we can insert another line of code without messing up the original code. Let's do the `10 PRINT` project to illustrate this.

```
10 PRINT "HELLO WORLD"
```

Next is line 20 and here we use a for loop, which is a loop that will iterate 10 times. Each time the loop goes round it counts the iteration, adding one to the total as it iterates.

```
20 FOR A=1 TO 10
```

Line 30 and we print that `LXF RULEZ!`. This is inside the for loop so it will print this line 10 times.

```
30 PRINT "LXF RULEZ"
```

Line 40 sees us instruct the for loop to iterate until it hits 10.

```
40 NEXT
```

If we RUN this code it will print `"HELLO WORLD"` and then `"LXF RULEZ!"` 10 times.

What if we want to add another line to our short sequence of code? The solution is to insert a new line between 30 and 40. Logically this would be 31, giving us many more options to expand or correct the code. But we are going to use 35 because this is just a simple test.

```
10 PRINT "HELLO WORLD"
```

```
20 FOR A=1 TO 10
```

```
30 PRINT "LXF ROOLZ"
```

```
35 PRINT "SO DOES TOMS HARDWARE"
```

```
40 NEXT
```

Now RUN this new code and you will see alternating lines of `LXF ROOLZ` and `SO DOES TOMS HARDWARE` (Tom who?—Ed) on the screen. The keen eyed among you will notice the typo for `TOMS`. This is intentional because the ' (apostrophe key) is mapped to `SHIFT 6` on the PCW keyboard, but despite our best efforts we couldn't replicate '. Instead we saw a garbled character.

To save our code to the disk, we simply use the Save command along with a filename:

```
SAVE "FILENAME"
```

Then press Enter. Change the filename accordingly – no extension is required. Our project code is now safe on the disk, so let's come out of the BASIC interpreter to see the file:

```
system
```

QUICK TIP

Bored of typing in line numbers for your BASIC projects? Using the `AUTO` command Mallard BASIC will add them for you, starting from 10, incrementing by 10 for each line. Need to start from a higher number? Just use `AUTO 100` to start from 100.

>> BARE-METAL Z80 MACHINES

We're lucky enough to own a small Z80-powered machine, in this case an RC2014 Micro kit. The kit itself is £54 and you need to solder it yourself, but that was part of the charm. We're building a computer from scratch, not just plugging in components.

The kit took around an hour to solder and we then connected it to our computer using a USB to Serial converter and the `ti0` serial console tool. We were

then presented with a classic BASIC interpreter and spent a fun few hours writing code.

If you want to take the project a step further, the RC2014 Mini is a slightly more powerful unit, which has an optional `CP/M` module that we can use to build our own `CP/M` computer. When connecting other modules we can add Wi-Fi via an ESP8266, Serial via a Raspberry Pi Zero and astonishingly we

can even add basic VGA output using a breakout designed for the £4 Raspberry Pi Pico.

If all of these retro computer hardware kits have inspired you to pick up a soldering iron, head over to <https://rc2014.co.uk> and grab yourself a basic (pun not intended) kit to start you off. We chose the Micro because it was the cheapest, but the Mini offers more expandability for not much more money.

Now look for the file, which has the extension BAS:

```
dir
```

To launch the code in BASIC all we need to do is use the BASIC command and call our file.

```
BASIC test.bas
```

We can also load the file inside the BASIC interpreter using the load command.

```
LOAD "FILENAME"
```

Then we need to use RUN to start the code. A quicker way is to just use RUN.

```
LOAD "test"
```

Using BASIC on the PCW is not as refined as the CPC 464, but it's still a great experience. If you'd like to know more then the complete manual can be found online at <https://bit.ly/1xf289-mallard-basic-pcw>.

Playing a game

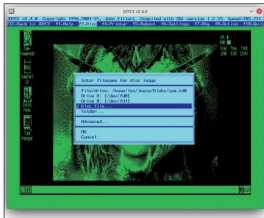
Let's be honest, the PCW range was not marketed as video games machines. However, the Zilog Z80 CPU and 256KB of RAM means that it could produce some decent games. Games can be found online, typically using the DSK file format. They can be loaded by pressing F3 from Joyce's main menu, then navigating to the disk image. Games will typically autoplay.

From the list of games we tested we found a version of 2048, the sliding tile puzzle game that was written in 2018. That was an addictive half-hour away from Wordle. We then spent far too long playing *Sky War*, a clone of *River Raid*. Even on the rather limited graphics of the PCW the sprite work was detailed and bold.

The Amstrad scene

If emulation has whetted your appetite for the real hardware, then there are some bargains to be had. If you don't mind a "fixer-upper" then you can pick up an 8256 for around £50. Models in better condition go for over £100. There are RAM upgrades available for the 8256, bumping it up to 512KB of RAM.

Because the supply of three-inch disks is finite, and bit rot is bound to claim more disks, there are floppy drive alternatives to be had. Gotek is the name to look for and these drives provide a means to load disk images from USB. They aren't cheap, coming in at



Swapping disks is easy. Press F3 to open the menu, F2 for Disc, eject the current and insert a new disk to load your application or game.



around £100, but they are worth the expense as the original disks become more fragile.

As ever with retro hardware, make an informed purchase. Do your research and learn what perils await those who make a purchase carelessly. A simple capacitor replacement is an easy job, but older tech often has hardware that's irreplaceable, so make sure to check before you apply the soldering iron.

The Joyce interface is a nod to the era it emulates. There's a hidden menu for tweaking our setup.

The Amstrad PCW legacy

It was never a gaming machine, but it never claimed to be. What the PCW range offered was a home office setup on a budget. They were a charming presence in home computing, where we had just enough power to realise our ambitions. Fanzines, magazines, spreadsheets and word processing may sound dull by today's standards, but just like BASIC provided a grounding for bedroom coders, *LocoScript* et al provided the foundation for writing and producing our own content. **LF**

» HISTORY OF THE PCW

The founder of Amstrad, Lord Alan Sugar, reportedly sketched out the original Amstrad PCW as a low-cost replacement for a typewriter. It was a single-box design with integrated portrait display and printer – largely because the biggest use for computers at the time was for word processing. PCW stood for Personal Computer Word-processor, while the printer was spun out as a standalone unit and a traditional (due to cost) landscape display ended up being used.

The original PCW 8256 shipped in 1985 with a separate keyboard, printer and a main unit with a 12-inch green monochrome CRT display with an unusual 32 lines of 90 characters. This diverged from the normal 25x80 character displays of the time.

The PCW 9512 was launched two years later. This device moved to a white monochrome display and offered a parallel port, enabling non-Amstrad printers to be used. This model survived until 1991 when the PCW 9256 launched with a standard 3.5-inch floppy drive. The last gasp was the PeWi6 in 1995, which still ran a Z-80 CPU but wasn't compatible with the PCW. It was a flop.

» **IMPROVE YOUR LINUX SKILLS** Subscribe now at <http://bit.ly/LinuxFormat>

BACK ISSUES » MISSED ONE?

ISSUE 288

May 2022

Product code:
LXFDB0288

**In the magazine**

From flashing lights to image recognition and even an aircraft tracking system – we show you how to get more from your Raspberry Pi. Elsewhere, we transform photos and video with the G'MIC plugin, emulate the Acorn Archimedes, process satellite imagery, find out how practical it is to use a Pi as your daily driver, and code memory-secure systems in the programming language Rust.

ISSUE 287

April 2022

Product code:
LXFDB0287

**In the magazine**

Discover the ins and outs of the Linux kernel. Then turn your hand at open source projects including ebook publishing, home automation and organising your research efforts. We test five alternatives to Ubuntu, preview Valve's exciting Steam Deck handheld console, and discover how the Emmabuntüs collective is distributing second-hand computers to communities in need.

ISSUE 286

March 2022

Product code:
LXFDB0286

**In the magazine**

Not all VPNs are created equally – find out how to avoid second-rate services and maintain your privacy online with our in-depth feature. We also test five GUI text editors, show how to build a distro from the ground up using Linux From Scratch, emulate an MSX, set up multi-boot USB devices, code a 3D game world, and reveal the tools for managing your passwords from the command line.

ISSUE 285

February 2022

Product code:
LXFDB0285

**In the magazine**

Stay one step ahead of the nefarious perpetrators of ransomware with our in-depth feature. We bring you tutorials on rock music effects, offline password management and creating a virtual network lab. Discover how to set up a temperature display for your Raspberry Pi and build web services with Go and the Gin framework. We also put five of the best GUI-based backup tools through their paces.

ISSUE 284

January 2021

Product code:
LXFDB0284

**In the magazine**

Discover how to turn your Raspberry Pi into an all-singing, all-dancing media hub. We put a spotlight on video conversion tools, take you on a tour of *Stacer*, the one-stop system management tool, and emulate the Oric-1 (thankfully without *that* keyboard). Discover how to build a Pi-powered NAS and learn more on how the world's top-500 supercomputers all harness the power of Linux.

ISSUE 283

December 2021

Product code:
LXFDB0283

**In the magazine**

We pit Ubuntu against Fedora to see which is the better GNOME-based distro. Five filesystems are put through their paces, we show how to run a Ghost blog, code a *Galaxian*-style shooter in Python, and use *Okular* to edit PDF files with ease.

DVD highlights

Ubuntu 21.10 and Devuan 4.0 "Chimaera" (both 64-bit only).

To order, visit www.magazinesdirect.com

Select Single Issues from the tab menu, then select Linux Format.

Or call the back issues hotline on **0330 333 1113**
or **+44 (0)330 333 1113** for overseas orders.

Quote the Product code shown above and have your credit or debit card details ready

READING IN THE USA?

UK readers
turn to
p18

SUBSCRIBE!

Don't wait for the latest issue to reach your local store – subscribe today and let *Linux Format* fly straight to you. Faster, cheaper and with DRM-free archive access!



3 GREAT
WAYS TO
SUBSCRIBE

Print, digital-only,
and print+digital
bundles!

» USA
From **\$132**
For 13 issues

» REST OF THE WORLD
From **\$132**
For 13 issues

» EUROPE
From **€100**
For 13 issues

IT'S EASY TO SUBSCRIBE!

Visit www.magazinesdirect.com/linux-format

Call **+44 0330 333 1113**

Lines open Monday-Friday, 9am-5pm, UK time

HOME ASSISTANT

Credit: www.home-assistant.io

Part Three
Did you miss
parts one or
two? Turn to
page 62

Make your home as smart as possible

Exploring NFC tags, energy monitoring and the Wireguard VPN to add further capabilities to your smart home with **Matthew Holder**.



OUR
EXPERT

Matt Holder has been a fan of the open source methodology for over two decades and uses Linux and other tools where possible.

The previous two articles of this series introduced a large number of concepts, which are important in gaining an understanding of how Home Assistant works and how devices can integrate with it.

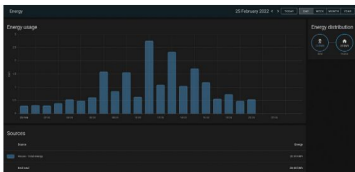
Other items that have been covered include using data from entities in automations, adding users to the system, customising Lovelace dashboards, adding hardware on the local network and adding integrations that pull in data from online sources.

This article will cover the Energy tracking dashboard in Home Assistant, how to use power-monitoring data from smart plugs to create automations, which can alert you when appliances such as the washing machine or dishwasher has completed a cycle. Also being covered is the WireGuard VPN, which can be used to securely access the system from outside of the home network, and the usage of RFID tags, which can be scanned by a smartphone and used to trigger automations.

As previously discussed, the Raspberry Pi is an excellent device to run Home Assistant. However, the unreliability of micro SD cards can let it down. It's possible to boot the Raspberry Pi directly from an SSD, which this author currently does.

More recent Raspberry Pi firmware makes direct boot from USB possible for some models. This was used with a USB SSD to enable Home Assistant to be installed directly to the SSD – no Micro SD card was needed. To configure this, a Micro SD card was written with the Bootloader->USB boot image file to configure the Pi in the correct way. This can be written to the SD card by using the *Raspberry Pi Imager tool* (www.raspberrypi.com/software): under Operating System scroll down to Misc Utility Images and click to open the next menu. From this menu, USB boot can be selected.

The target device can then be selected (be careful that the correct device is selected because this operation will wipe whichever device is chosen), followed by clicking the Write option. Once written, the Micro SD card can then be inserted into the Raspberry



Energy monitoring dashboard, displaying electricity usage.

Pi and it can then be switched on. Once completed the activity light will flash a uniform pattern and if a HDMI cable is connected the screen will turn green.

With this step completed, the Home Assistant image can be written to the USB SSD. This can be done by following the usual instructions, with special care being taken to select the correct device to write the image to. When booting from an SSD for the first time, Home Assistant will take the necessary steps to ensure that space on the entire drive can be used. If being used for a new installation, simply complete the welcome wizard. If replacing the Micro SD card of an existing installation, then a recent backup file can be used to restore everything to its previous state.

Access when away from home

There are a number of ways to access your installation when outside of the home. The most insecure method is to open a port on your router that is directed to the device running the installation. Alternatively, you could pay for a www.nabucasa.com subscription.

The third option is to use a VPN to connect to your home network and then it can be accessed as if connected from home. There are plenty of options available to choose from, including OpenVPN, WireGuard, TailScale and ZeroTier. The WireGuard add-on is excellent. It provides everything needed to set up the server and client and will be covered below. WireGuard provides a VPN at layer two, and the IP address of the server and client will need to be set manually. Clients exist for

QUICK TIP

While a lot of configuration takes place using the GUI, YAML is still required in places. See <https://bit.ly/txf289-yaml-ha>.

Linux, Windows, iOS, Android and Mac OS. Some router firmware also contains support for the solution.

To be able to create a secure VPN connection between the *Home Assistant* instance and another device, a few things need to be configured. First, the *WireGuard* add-on will need to be installed. To do so navigate to the Add-On store and install the *WireGuard* add-on, making sure to set the Start at boot, Watchdog and Auto-update options.

Start the add-on and then move on to the configuration. It's likely that your home network has an IP address range of something like 192.168.1.0/24. This means that there's scope for 255 IP addresses on the network, which should be fine for most. The *WireGuard* network will need a completely separate range to avoid any conflicts. The 172.27.66.0/24 range will be suitable in most cases and this is the range that will be described below. Part of the setup required for the VPN is to configure a DNS server to use for the remote clients. This could be your home router, Google's DNS server or perhaps even an instance of *AdGuard Home*, which is another add-on that is available and can provide DNS filtering to block ads and can support basic filtering for other users in the household.

Step two is to configure the "server" side of the connection. Copy the below YAML into the configuration box of the Add-Ons configuration page and adjust the text accordingly.

```
log_level: info
server:
  host: <DNSADDRESSHERE>
  addresses:
    - 172.27.66.1
  dns:
    - 1.1.1.1
    - 1.0.0.1
  peers:
    - name: <DEVICENAME>
      addresses:
        - 172.27.66.2
      allowed_ips: []
      client_allowed_ips: []
```

The peers section of the YAML above refers to a client and as many of these can exist as required. For each of the peers configuration is generated. Each peer or client needs an IP address assigned to it as well as a name. The `allowed_ips` and `client_allowed_ips` section provides a way of restricting IP ranges that the client can access and vice versa. These can be left blank to allow access to and from all devices. The server section of the configuration requires an IP address to be set and this needs to be on the same range as the clients. **DNS** refers to the DNS server used and can be your home router. Finally, the host section refers to an address that can be used to reach the installation from the internet.

Assuming a dynamic IP is provided by your ISP, there are multiple services that enable a client to report to a web-based service and this can then provide a DNS address to a changing IP. With this configured, the next stage is to restart the add-on. This will force the container to build a new config for any new peers that don't currently have the relevant information available.

Before configuring the client, port forwarding will need to be configured on your router so that traffic destined for your DNS name can arrive at the router and

be forwarded to the device running *Home Assistant*. This differs for all brands of router, so DuckDuckGo will be your friend for this step. A static IP address will be the best option for the device running *Home Assistant* so that there aren't issues when the DHCP address changes. Please note, that any port being forwarded into your home network can be a security concern, even if being used with a VPN. A more secure method would be to use Nabu Casa, but this is an option that requires a paid subscription.

Configure your client

The final step to getting the VPN fully configured is to set up your client. For an Android phone, for example, download and install the *WireGuard* app from the play store. From within the app, click the plus button and select the option to start from scratch. Now, from within *Home Assistant*, open the Studio Code Server add-on, navigate to the **SSL/Wireguard** directory and open the `client.conf` file within the directory, which is named after the client being configured. Enter the details and then

QUICK TIP

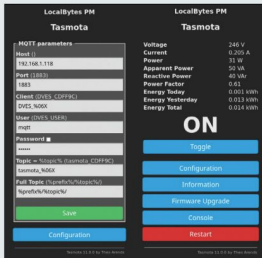
Visit www.mylocalbytes.com to buy smartplugs with open source firmware.

» TAKING THE TASMOTA ROUTE

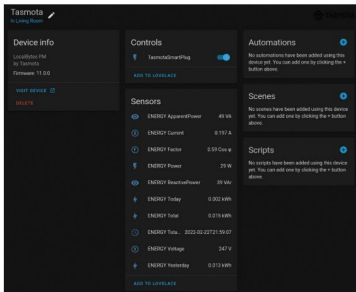
Tasmota is an open source project that provides alternative firmware for ESP8366 and ESP32 devices. The project provides a way to run firmware that ensures that all data is kept local to our networks. When installed, access is via a web browser, MQTT, web requests and serial. The support for MQTT allows for very simple integration with *Home Assistant*. Upgrades are provided Over The Air (OTA) and are simple to perform from the built-in web interface.

Support is provided for a wide range of hardware and the Tuya-convernt project can be used to reflash devices that are produced by Tuya and are sold as rebranded devices across the world. Tuya-convernt supported devices will most likely use the *Smart Life* or *Tuya Smart* mobile apps for configuration of the pre-installed firmware.

It's important to note that there can be serious safety issues when considering reflashing any devices that are connected to mains voltages. A safer option is to purchase devices pre-installed with the firmware, such as smartplugs (see *Quick Tip*, above). Tasmota has built-in support for GPIO pins (both analogue and digital), displays, IR communication, I2C interface, lighting of different types, temperature sensors, RF communication, Bluetooth and much more.



Tasmota MQTT configuration (left) and status page (right).



Home Assistant entities related to the Tasmota-powered smart plug.

save the config in the *WireGuard* app. To test all is working well, switch off the Wi-Fi on your phone, switch on the *WireGuard* VPN and connect to the internal IP address of the *Home Assistant* server. If all has worked well, the *Home Assistant* interface will load. This method can be used to connect from a browser or the mobile app. When using this method, by default all web traffic on the phone will use DNS provided by your home connection. The Android app and perhaps others too can be configured to only pass traffic from certain apps through the VPN.

In the past nine months or so an energy tracking dashboard has been added to *Home Assistant*. This provides support for the tracking of entities that supply data with units of kiloWatt hour (kWh). Support also exists for solar generation and battery energy storage and the net energy usage, taking into account multiple

» CONFIGURING MQTT

MQTT stands for Message Queuing Telemetry Transport and is a messaging service that can be accessed by a wide range of hardware. Libraries exist for devices all the way from x86 to microcontroller boards. The messaging protocol requires very little processing power and a tiny amount of bandwidth.

MQTT operates as a publish-subscribe system. An MQTT broker runs somewhere on the network, which devices can publish to. Other devices then subscribe to the broker and are able to act when certain messages are received. The recommended broker to use for *Home Assistant* is called *Mosquitto* and can be installed from the Add-On store (Configuration>Add-Ons, Backups and Supervisor).

Once installed, navigate to the users section of the configuration and add a new user for MQTT. This can be a limited user who doesn't need to be an administrator, and can be set to allow access from the internal network only. Now that the MQTT add-on has been installed, head to the Integrations page within Configuration and configure the MQTT integration which has been discovered. The hardware we'll be discussing in this article can then easily be added to the system. The *Home Assistant* team has developed a standard so that devices can automatically be added, when certain messages are received.

sources, is displayed. This can be shown for various time periods including the current day, week, month and year. While the cost of energy usage can be displayed, currently there's no option to add a daily standing charge as well as cost per unit, which can make UK usage of this feature a little tricky. Gas consumption can also be tracked, but this depends on the correct hardware/meter being present.

1 Energy monitoring dashboard

In this next part of the article a Tasmota-powered smart plug will be configured and added into our installation. Integrations do also exist for other brands, such as TP-Link, but note that these may need to be cloud-connected. This configuration will enable the plug to be switched on and off from an automation as well as the energy being used to be stored and automations fired based on the values.

The first stage is to add the device to the wireless network and therefore *Home Assistant*. Once the smart plug has been plugged in, wait for a minute or two and then connect to the Wi-Fi network that it provides. Open a web browser and visit the captive portal offered by Tasmota. Follow the setup wizard and set the details of your home network Wi-Fi SSID. The device will then connect to the network and report the new IP address. Swap Wi-Fi network and visit this IP address.

The next stage is to configure MQTT, so that *Home Assistant* can integrate as needed. Visit the Configuration page, select MQTT and then enter the IP address of the *Home Assistant* installation and the username and password setup for MQTT specifically (see *boxout*, below for further details). Other options can be left as default. The smart plug will then communicate with *Home Assistant* and it will be reported, under the Integrations section of the Configuration page, that a new device is available. Follow the wizard to adopt this device. The relevant entities will be added automatically.

2 MQTT configuration

Using the device page, linked to from the Tasmota integration's tile, the relevant entities can be seen and the relay in the smart plug can be switched on and off. The current, voltage and energy usage will also be displayed. An entity will also be provided with the unit of kWh and this can be added to the energy-tracking section of the configuration. During testing, the data provided from Tasmota didn't seem to be reported to *Home Assistant*. The issue here was that, by default, Tasmota will only send data, via MQTT, every five minutes. This can be changed: access the Console option within the smart plug's web-interface and enter **TelePeriod 30** which will change the reporting interval to 30 seconds.

3 Tasmota-powered Smart Plug

An automation can now be created using a helper and the power usage of the device to determine when a device starts and stops. For example, this can be used to alert us when the washing machine cycle has finished. In the Configuration>Automations and Scenes section add a Helper of type, Dropdown, call it **DEVICE_STATUS** and add options of **InUse** and **NotInUse**. This will then create an entity called **input_select.DEVICE_STATUS**. Two automations will be used for this and the

QUICK TIP

NFC tags can be used to interact with Home Assistant using the phone app www.home-assistant.io/blog/2020/09/15/home-assistant-tags/

first will detect when the power consumption rises for a number of seconds and it will then set the value of the Helper to **InUse**. The second automation will fire when the power consumption drops to a low value for a number of seconds and will use a condition of the Helper being set to **InUse**. When this second automation fires, the value of the helper can be set back to **NotInUse** and then a notification sent to Telegram or a Google Home/Nest device – see last month's article for further details about verbal and textual notifications.

Create the first automation and set the Trigger to be of type Numeric State. Add the relevant entity, which will be something like **sensor.tasmota_energy_power**. Set the For box to be 30 seconds. In the Above box, set a realistic value, such as 20. Set the condition to use the value of helper being set to **NotInUse**. In the Actions section set the Action type to Call service, the service to Input select: Select and the entity to the relevant Helper. The Option box can be changed to **InUse**.

Create the second automation and set the Trigger to be of type Numeric State. Add the relevant entity, which will be something like **sensor.tasmota_energy_power** and set the For box to be 30 seconds. Set the Below box to be a reasonable value, such as 3. Set the condition to use the value of helper being set to **InUse**. Under the Actions section, set the helper value to be changed to **NotInUse**. Now you can either set a further action to notify Telegram/Google Home/Nest device that the device is no longer operating or, if you want to place further conditions on when the notification is sent, create a third automation with a Trigger of the Helper changing from **InUse** to **NotInUse** with Conditions containing the times that the automation should fire.

4 Automation debugging

NFC has revolutionised a number of industries from payments to file transfers to retail. RFID tags are even used by some Bluetooth peripherals. With a smartphone or standalone reader these tags can also be used with Home Assistant. Tags are programmed and automatically registered with Home Assistant and automations can then be fired when devices are scanned. These tags can be used to arm an alarm, send notifications, start timers and more.

The first step to using tags is to decide which ones to buy. This author has used fairly basic MiFare tags before, but before heading to your favourite online marketplace, check which your chosen device supports. It's also possible to make your own standalone reader using a PN532 circuit board and microcontroller (<https://github.com/adonno/tagreader>).

Install the Home Assistant app from the app store and register with your installation. When your new tags arrive, make sure NFC is enabled on your phone and open the Home Assistant app. Open the Configurations> Tags option and add a tag. Give it a descriptive name, select Create and Write and scan the tag. This will then register the tag and when scanned a URL is visited in Home Assistant, which can then be used. For example, there could be a tag by the TV, so that when scanned, if the sun has set, an evening scene can be activated to dim the lights. Tags could also be used to play certain

albums, based on which tag is scanned. Imagine a book of album covers, each containing an NFC tag. Tags can also use QR codes, so experiment with these before purchasing NFC tags as this may be as convenient.

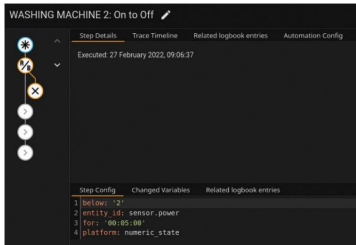
To create a QR code, navigate to Configuration>Tags >Add Tag and give the tag a name. Once generated, the QR code will be shown in the list and when the cog icon is selected a QR code will show, which can be printed and used anywhere around the home. An automation can be created by clicking the icon of the Android, next to the newly created QR code. When scanned, the QR code will call a URL within the Home Assistant system, which will trigger the relevant Automation.

For this example, a notification will be sent to a registered Google Nest device. When creating the Automation in this manner, the trigger section is already populated. Add any conditions and set an Action to call a service and assuming that last month's tutorial has been followed, add the service type to be **tts.google_translate_say**. In the Entity box select the relevant device and add a relevant message. Save the automation and then scan the QR code. Home Assistant will then perform its magic and the automation will fire and the message will be audible from the relevant speaker.

This is the final article in the series on Home Assistant itself. There have been a lot of concepts introduced over the past three issues, but this will hopefully have given a good grounding into such a large project. Hopefully now you'll find it straightforward to develop your own ideas into captured data and automations.

In the following two issues, projects will be discussed that integrate nicely with Home Assistant. The first is ESPHome (also developed under the NabuCasa umbrella) and this provides a way of generating firmware for small microcontroller devices, such as the ESP8266 and ESP32 devices. This allows for "satellite" data collection devices to send data to the central Home Assistant server. The second article will cover WLED, which is used to install on a microcontroller board and provide a way of controlling individually addressable sets of multi-colour LEDs. See you next month! **✎**

Using the clock icon next to each automation, it is simple to troubleshoot any problems.



» HOW TO AUTOMATE LXF... Subscribe now at <http://bit.ly/LinuxFormat>

WINE

Credit: www.winehq.org

Get more from Wine and Windows

Michael Reed teaches you everything you need to know about using Microsoft Windows compatibility system WINE with some handy tips.



OUR EXPERT

Michael Reed has Wined and dined almost every popular Linux distro out there. What a smoothie.

Wine enables the Linux user to run software that was designed to run under Windows directly on the Linux desktop. Wine stands for Wine Is Not An Emulator, and that gives some clue as to how it works. Rather than emulating a PC, as is the case with a virtual machine, Wine remaps the calls made by Windows software to equivalent Linux calls. This means that the Windows application is, effectively, running natively on Linux. The advantages of this approach, as opposed to virtualisation, are that fewer resources are consumed, performance is improved and Windows applications run seamlessly on the Linux desktop.

Installing the latest Wine

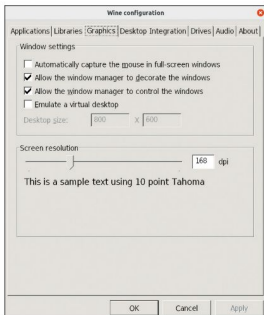
The standard Wine package in the repository of most Linux distributions is **Wine Stable**, and this is the branch that most users are looking for. **Development** is a more up-to-date branch and **Staging** is the branch that contains experimental patches and new features. Generally, you'd only use those last two if you knew for sure that you needed a bug fix or feature that was only available in either of those branches.

On Ubuntu, for example, **Wine Stable** can be installed by typing `$ sudo apt install wine`. At time of writing, the Ubuntu version was a bit out of date, so we followed the instructions on the Wine website (<https://wiki.winehq.org/Download>) to install a more recent version. In the case of Fedora, things were a bit more current, so we used the version in the official repositories instead.

Windows Software, EEK!

The installation procedure for most Windows software is different to that of most Linux software. Typically, one locates the desired software on the web, downloads an installer executable and then runs that. Often, the program itself requires additional resources such as DLL (Dynamically Linked Libraries) files and additional components, which the user has to locate and install. Alternatively, some Windows software, particularly older, commercial software, is installed via a medium such as a CD-ROM. Some smaller utilities are distributed in a .zip file to be extracted into a directory of your choice.

The starting point for the installation of most Windows software is to run a setup program, which can either contain the entire application or game or fetch



Our first port of call here is setting the font size by altering the Screen resolution setting.

the required files from the internet. In those cases, the executable can be run by navigating to the directory in a terminal using the `cd` command and running the following command:

```
$ wine [name and path of executable]
```

In the case of applications and games that have an installer, upon completion of the installation process, Wine should add them to the Linux system to be launched in the same way as other Linux programs.

We'll work with an example that uses Notepad++, a popular Windows text editor. Having located the website (<https://notepad-plus-plus.org>), we downloaded the installation executable. Once Wine is installed it's usually possible to run Windows executables by double-clicking them, but it's often worth running them from the command line terminal so that you get some feedback if there's a problem. We navigated to the download directory and then ran `$ wine npp.8.3.2.Installer.x64.exe` to run the installer executable.

QUICK TIP

There's never a good reason to run Wine with root privileges. Doing so will open up your Linux system up to security and stability risks, and confuse your configuration file privileges.

We didn't have to do much to complete the installation. We simply accepted the licence agreement and the default selections for optional plugins. We also accepted the default suggestion for installation directory, and it's worth taking a closer look at what this means. Microsoft operating systems such as Windows have always labelled hard drive partitions with a letter. This is incompatible with the Linux filesystem layout, so Wine creates a virtual drive C in a directory called **wine_c** within the directory hidden directory **.wine** within the user's home directory.

Wine Prefixes

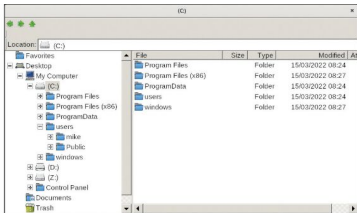
Wine gives some scope for taking advantage of its facilities for remapping directories by using the **wineprefix** environment variable. If we had run `$ WINEPREFIX=~/.wineprefixes/npppreffix wine ~/Downloads/npp.8.3.2.Installer.x64.exe` then a complete Windows file system would have been created in the **~/.wineprefixes/npppreffix** directory before running the executable. There's no need to make your custom prefixes hidden directories (with a dot at front of the name) just because Wine defaults to that.

The prefix feature is ideal for situations in which you want to set up a separate Wine environment with its own settings and support files for a program or a type of program such as games. Each prefix directory has its own configuration. You can browse the files within a prefix in the normal way by navigating your file manager to the prefix subdirectory, be it in the default **~/.wine** subdirectory or that of a prefix you created.

You can even place a Wine prefix on a completely separate drive. For example, you may have a speedy SSD as your boot drive coupled with a larger mechanical drive. To make a given prefix the default one, export the **WINEPREFIX** variable by adding a line to your **.bashrc** in your home directory such as: `export WINEPREFIX=~/.ext_drive/WindowsGames`

Running Windows Programs

Once an application is installed, it should function as a normal Linux application in terms of how you access it. On a real Windows system, once an application is



installed, a desktop shortcut is typically created along with an icon within the Windows application launcher.

Wine simulates a similar setup, but how this is presented to you depends on the desktop environment that you're running. Most of the time, the application that you've installed should be available from within the application launcher of your chosen desktop environment alongside your native Linux software. In addition, a desktop file is placed in the **~/Desktop** directory. This is a text-based file that's sometimes worth examining in a text editor if you need to find out where the program is located or how it's launched.

The command 'wine explorer' launches a simple file manager, which gives you a clear view of the file layout that Wine sees.

Command Line

Some Windows programs, such as some games, require command line flags to be passed to them. To do this, navigate to the directory containing the executable and then pass the command flags at the end of the Wine command. To discover the location and name of the executable of an installed Windows program, load the associated **.desktop** file from your **~/Desktop/** directory and look at the **exec** and **StartupWMClass** fields.

If you had a game executable called **game.exe** and you needed to pass the flag **-disablenetworking** to it, you would navigate to that directory from within a terminal and type:

QUICK TIP

Some Windows programs come in the form of **msi** files, which aren't executables. To run these installers under Wine use the **wine start [name of msi file]** command.

» ALTERNATIVES TO WINE

The most obvious alternative to running a Windows application or game under Wine is to use a virtualiser such as **VirtualBox** or **VMWare**. Virtualisers such as these emulate the hardware of an entire PC and can run Windows. In some cases, this gives a better chance of running an application successfully because you're technically running the application under the operating system that it was designed for.

In addition, aspects such as DRM can throw a spanner in the works when it comes to compatibility through Wine. You lose a lot of the integration convenience of running an application under Wine, and VMs are cumbersome

and resource hungry in themselves.

For playing video games, it's a toss-up as to whether performance and



ReactOS running in a VirtualBox virtual machine.

compatibility are better under Wine

because the intensive hardware requirements that games require expose the weaknesses of virtualisers. In addition, in many cases a virtualiser is unable to expose as extensive a selection of graphics card facilities as Wine.

Wine has been made possible due to the fact the Windows API (Application Programming Interface) is well documented. ReactOS (<https://reactos.org>) takes things a step further by reimplementing not only the Windows API, but also implementing a Windows-compatible operating system and has taken great strides in recent years!

QUICK TIP

Note that Wine's prefixes don't really constitute a sandbox, as such, because Wine applications can still access the Linux environment.

\$ wine game.exe -disableworking

Remember that Windows applications make use of a single dash character before a non-abbreviated command line flag, unlike Linux tools that would use a double dash instead.

Typically, you'll end up trying a few different things before you find a command line sequence that correctly configures a Windows game. Going back to the `.desktop` file, you'll notice that it actually launches the desktop link within the emulated Windows file system rather than running the executable file itself. Like a lot of Windows resources, these links aren't flat text files. For this reason, although you can alter the link to add the command line option, it's often easier to create a Bash script. For example, you might create a launcher script in a text editor along the lines of:

```
cd ~/wine/dosdevices/c:/Program Files/GreatGame/
wine greatgame.exe -skipintro
```

Save it with a `.sh` extension. You can leave it in your home directory for easy access, but we like the location of `~/local/bin/` for user scripts because they're available from any location, they are backed up with backups of your home directory and root access isn't needed to create or edit them.

Just remember to run `$ chmod +x [name of script]` on it to make it executable.

Configuring Wine

Rather than using text-based configuration files, Wine breaks with Linux tradition by storing configuration details in a binary file called the registry, a technological approach that it shares with Microsoft Windows. If you find a tip for getting a specific program running that involves editing the registry, you can invoke the registry editor by typing `wine regedit`. However, most configuration of Wine is carried out using a GUI utility called `Winecfg`. If you run `winecfg` from the command line or an application launcher, it edits the configuration located in the default Wine prefix. It's common to take advantage of that fact when creating a new prefix. You could create a prefix just for games by typing

```
$ WINEPREFIX=~/WindowsGames winecfg
```

Next, you'd place `WINEPREFIX=~/WindowsGames` before a command that launched the given Windows application. It's usually worth the hassle of creating separate prefixes for categories such as games. For one thing, it means that you can back up your Windows programs and carry them between different Linux installations without having to reinstall them every time.

The default font size tends to be a bit unpredictable when running Windows applications. This is something you can check and adjust by running `Winecfg`. Under the Graphics tab, alter the 'Screen resolution' slider until the example text provided is of comfortable size. Click Apply or OK when you're happy with your selection, but you'll have to restart `Winecfg` or running Windows software to fully test out the selection.

As we mentioned earlier, by default, Wine sets up a directory (`~/wine/drive_c`) that simulates the C: drive of a Windows system, and you can add extra drives under the Drives tab of `Winecfg`. This is handy for directories that you want to have within easy reach of your Windows applications and games.

A word of warning here: every time you add a drive, you're potentially giving Windows programs access to

that area of your system. Drive `z:` is linked to the root of the Linux filesystem, but with the same access privileges as a normal user. You can safely remove this drive, but you'll no longer be able to access Linux files outside of the given Wine prefix.

Winetricks

`Winetricks` is a helper script that has different facilities for improving the experience of running Windows programs under Wine. It's usually installed along with Wine, but if not, search for it in the package manager.

Typically, when you run a specific `Winetricks` option it will install libraries and other program resources under the currently selected Wine prefix. Therefore, all programs that run from that prefix will benefit from the resources you added with `Winetricks`. Most `Winetricks` usage examples run from the command line, but if you run `Winetricks` without any command line options, it'll launch into a GUI. From here, you can browse through its various options to start building up a picture of what `Winetricks` is capable of.

`Winetricks` comes into its own when fetching and installing resources needed to get programs working. It's even possible that Windows users would regard `Winetricks` with envy because they're used to having to hunt down packages by hand, downloading from what they hope is a safe website and installing them manually.

In many cases, you can find out what `Winetricks` command sequence is needed by Windows programs by searching the Wine Application Database (see box, opposite). As a simple example, one of the first things many users do to improve the look of Windows applications is to install the `Corefonts` package to their Wine prefix. To do this, run the following command:

```
$ winetricks corefonts
```

As often happens with `Winetricks`, this script takes a long time to complete, but it does hunt down the individual files that are needed and install them. Naturally, you can use a custom prefix with the `Winetricks` command so that you can have different resources available to different setups on your system, a difficult thing to set up on a real Windows system.

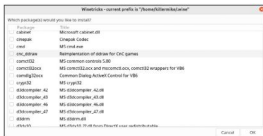
As an alternative to using the GUI, type `$ winetricks dlls list` to get a list of all of the DLLs and frameworks that `Winetricks` can download and install, a useful starting point if a program says it can't find something. For example, if a program complained that it couldn't find a DLL relating to Visual Basic, you could type:

```
$ winetricks dlls list | grep -i "visual basic"
```

for a case-insensitive search of the complete list. Type `winetricks list` for a list of all categories of things that `Winetricks` can install.



Notepad++ uses a fairly typical Windows installer executable. In this case, we largely accepted the defaults and kept clicking Next.



The WineTricks GUI version is a good way to acquaint yourself with what it can do and browse its features.

To investigate installing Windows games, we started with *Fallout Tactics: Brotherhood of Steel* (2001), which we purchased from the Good Old Games website (www.gog.com). Good Old Games makes an effort to provide the games with the copy protection and DRM (digital rights management) removed, and this is welcome on Linux because DRM often causes more headaches than any other issue when running games under Wine. In fact, ancient DRM systems often prevent older Windows games from running on newer versions of Windows.

In the case of that particular site, it's often possible to download a .zip file containing the files for a given game. In this case we could only find an .exe installer. Our experience with the GOG installer has been that it generally works under Wine, but tends to throw up the old error. Game installers often want to install various extra resources, adding Lord-knows-what to the system – a fact of life that Windows users just have to learn to live with – and this makes using a Wine prefix a good idea for games.

We started by creating a Wine prefix on an older, secondary hard drive to save space on our boot drive, and then we launched the GOG installer for the game: `$ WINEPREFIX=~/olddrive/WinGames wine setup_fallout_tactics_2.1.0.12.exe`

This installation took quite a while, and at one point, another window popped up requesting to install some Microsoft support files, which we allowed it to do. As ever, we simply clicked OK when the Good Old Games installer threw up some errors, with no obvious ill-effects. A tip we can offer you here is to change the default suggestions for installation directories as Windows installers love to add spaces to filenames, which can be awkward to deal with.

One problem with installing a game to a custom prefix is that the launch icon isn't added to the launcher of the desktop environment. If you think about it, there's no way it could because custom prefixes enable you to install more than one version of a program at once. This gave us some extra hassle of having to navigate to the installation directory within the custom prefix we'd set up. Once we in there, we ran:

```
$ WINEPREFIX=~/olddrive/WinGames wine POT_HiRes_Patch.exe
```

to apply the high-resolution patch that Good Old Games provided with the game. It was smooth running from there on, and we found that a resolution of 1,280x768 provided the best compromise of graphics size while fitting on a modern, 22-inch monitor.

» SOFTWARE COMPATIBILITY

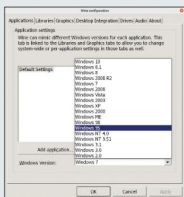
The Wine Project hosts a database where users can report their experiences running applications and games under Wine (<https://appdb.winehq.org>). In particular, it's important to check the database if you're considering investing money in a Windows program. Even if a program appears to launch properly when run under Wine, it's worth checking to see if there are any gotchas in store when you try to access certain features.

Some reports feature tips and tricks to get the program going, depending on the specific version too, and it's well worth expanding the Comments and How To/Notes sections at the bottom of the page for extra information about getting programs working.

If you're having difficulties running a particular program, particularly one of an older vintage, it's sometimes worth causing Wine to simulate an older, 32-bit version of Windows. Consider the following command sequence:

```
$ WINEARCH=win32
WINEPREFIX=~/wine32
winecfg
```

This creates a new prefix called wine32 in the home directory and specifies that this prefix should be configured to simulate a 32-bit version of Windows and then launches *Winecfg*. Subsequently, it's possible to select an older version of Windows, such as Windows 95 under the Applications tab in *Winecfg*.



Selecting a 32-bit prefix and then selecting an earlier version of Windows in *Winecfg*.

What we ended up with was a startup script that looked like this:

```
cd ~/olddrive/data/WindowsGames/drive_c/GOG/
FalloutTactics/
WINEPREFIX=~/olddrive/data/WindowsGames/ BOS.
exe
```

Given the tactical nature of the game, it was worth experimenting with running it in a window. Because it wasn't an in-game option, we ran `$ WINEPREFIX=~/olddrive/data/WindowsGames/ winecfg` and selected 'Emulate a virtual desktop' under the Graphics tab. This runs full screen Windows applications in a window. In our experience, setting the game resolution in the game itself overrides the resolution you've selected in this dialog, so don't worry about getting an exact match.

Stop wineing

Hopefully, we've given you an introduction to the basics of operating Wine. It can sometimes require a bit of experimentation to get a Windows program working, and of course, it's always better to use a real Linux program that does the same thing, if it's available. However, thanks to features like Wine's prefixes and the *WineTricks* script, Wine can sometimes make working with Windows applications and games more organised than real Windows does. **157**

» USE OUR INCOMPATIBLE SYSTEM... Subscribe now at <http://bit.ly/LinuxFormat>

Credit: www.rainloop.net

Take full control over your email

David Rutland does the impossible and sets up a VPS-based email server and a webmail front-end, then writes a tutorial about it – all in one afternoon.



OUR EXPERT

David Rutland hasn't moved for the past four hours. He's drenched in a pool of his own sweat and in need of a cuppa.

RainLoop is a pleasure to use, and unlike a client such as Thunderbird you can access it from anywhere. Remember to change the default login credentials.

Email is the king of online communication. Even today, with alternatives including an array of messaging services, video platforms, Slack, and of course, the Tok, email is where it's at if you want a secure, reliable, platform through which you can contact anyone in the world.

Signal, Telegram and WhatsApp are ideal for organising a night out, but problems arise when one friend uses Facebook Messenger, two are on Signal, and there are a couple of revenants from the 1990s who refuse to use anything more up to date than their own IRC chatroom.

Messages need to be relayed to everyone in the friend group, and getting anything done relies on some members having access to more than one services and acting as an intermediary. Honestly, it would be quicker and easier to deliver RSVP invitations by carrier pigeon.

Email isn't like that. Anyone with an email address can send messages, GIFs, audio clips and top-secret documents to those with an email address. *Fastmail* doesn't prevent you from interacting with people who prefer *Outlook*, and even if Google is your data-slurping behemoth of choice, you can still use *Gmail* to send messages to your Apple-crunching friends.

Email is also insanely resilient. If *Telegram* goes offline or is banned in your country, tough luck – you've lost contact with all of your *Telegram* buds. If you lose access to *Gmail* for some reason, you can just pick another

provider and carry on as normal.

As this article is being written, the inhabitants of a large, supposedly European country have lost access to *Instagram*, and it looks likely that both *WhatsApp* and *Facebook Messenger* will follow suit. It's impossible to know what will happen with the *Telegram* and *Signal* services, but we can practically guarantee that the email network will continue to function even if the countries involved no longer exist.

Who's reading your mail?

We've established that email is fantastic, reliable and the only sensible way to send messages if you want them to be received by everyone. But for most people it isn't exactly private, and the ability to use their email account to communicate with others exists at the whim of a corporation that has a nasty habit of shutting down services and changing the terms and conditions of use.

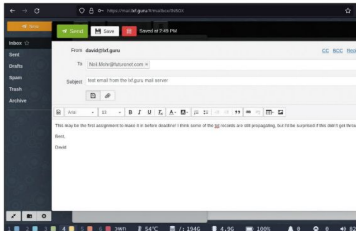
Around two billion people have *Gmail* accounts in 2022. That means that around one-quarter of the people alive on our planet have their mail routinely scanned and mined for details of their personal and professional lives. Such information is worth money. Google claims it stopped scanning emails for advertising purposes (we may or may not believe them) in 2019, but it still has its AI read through your ramblings for other purposes. Read the terms and conditions some time – they're about what you would expect.

Any email account, whether free or paid, is prone to surveillance for either commercial or legal purposes. Even *ProtonMail* – which vaunts its privacy credentials and holds itself as a paragon of snoop-free virtue and has long been regarded as the most secure and private email service in the world – has admitted that it can be compelled to log activity and IP addresses to assist police with their enquiries.

If you use email in the hope that your communications are private, by relying on a third party to provide the service for you, then you've already lost.

Email isn't that hard

Setting up your own email server is a daunting task. Visit any self-hosting community online and you'll find dozens of threads with people asking for help, only to be shut down by users saying not to bother. "It's too



difficult," they say. "You can't secure it properly," chimes in somebody else. "Your mail won't reliably get through," they add.

Such doom-sayers aren't entirely wrong. This writer's main email address is on a server running on a Raspberry Pi behind his couch, and took around a week to get working properly. It wasn't easy. A previous email-based tutorial was aborted after turning into a solid 3,000-word block of code.

A typical email server relies on a few pieces of essential software: *Postfix*, a free and open-source mail transfer agent that routes and delivers email; *Dovecot*, and open-source IMAP and POP3 server, from which email can be sent and retrieved using your favourite email client; *SpamAssassin* to assassinate spam (it's in the name); and *openDKIM*, which helps assure other mail providers that your email server is who it says it is, and is authorised to send mail.

These components, while easy enough to install, require a great deal of configuration, and can easily form the kernel of a severe migraine as you try to work out what went wrong this time.

If you want to set a mail server the proper way, we suggest consulting Jonni Bidwell's excellent tutorial which featured in **LXF284**. If you want to get up and running as quickly as possible, read on...

Record keeping

Before we get started in the terminal, we need to set up a few records. Head over to your registrar (Namecheap in our case) and set up a new A record with @ as the host and the IP address of your VPS as the value. You'll also need a record with mail as the host, and again, your VPS IP as the value.

Further down, the email settings drop-down will likely be set to either no email or email forwarding. Find the Custom MX entry and select it. Enter your @ as the host and **mail.your.domain** as the value. Set the priority as 10.

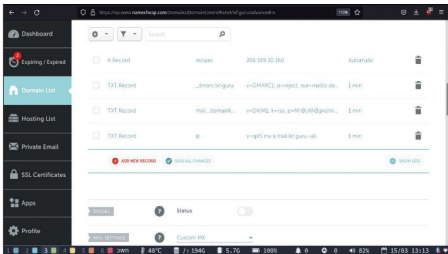
Now you need to change the rDNS of your actual VPS. This is the hostname that your server will give if the IP address is queried, and assures receiving mail servers that your email comes from a legitimate source.

You can check the hostname revealed by rDNS. Visit <https://dnschecker.org/reverse-dns.php>, and type in your IP. You'll need to change this, so that a reverse DNS query returns your current name. For the *Linux Format* VPS this would be **lxf.guru** and not **mail.lxf.guru**.

If you chose Digital Ocean as your VPS provider, changing the rDNS is as simple as renaming the machine in the Digital Ocean dashboard. Records will be updated automatically. If you use another provider, check their help docs.

Head back to the terminal and SSH in. You're going to create a configuration file so that Apache knows what to do with the base domain (**lxf.guru**) and your mail server subdomain (**mail.lxf.guru**)

```
$ cd /etc/apache2/sites-available/
and then: $ sudo nano your.domain.conf and enter:
```



```
<VirtualHost *:80>
ServerName your.domain
DocumentRoot /var/www/your_domain/
</VirtualHost>
```

Press Ctrl+O then Ctrl+X to save and exit. Next, do the same for your mail subdomain, using **mail.your.domain** instead: **\$ sudo a2ensite your.domain.conf mail.your.domain.conf** then restart Apache with:

```
$ sudo service apache2 restart
```

Now **\$ sudo mkdir /var/www/your_domain** and **\$ sudo nano /var/www/your_domain/index.html** and enter whatever text you want. At this point the content doesn't matter. Again, Ctrl+O then Ctrl+X to save exit, and then repeat for your mail subdomain.

At this point you should be able to reach your root domain and the mail subdomain using your browser. You won't see the broken padlock or insecure warning, which lets you know that SSL isn't enabled.

Dive back into the terminal and **sudo certbot**. Answer

DNS TXT records need to be set up correctly to assure receiving servers that you aren't a filthy spammer.

QUICK TIP

Read our VPS features in **LXF281** and **LXF282** at <https://bit.ly/lxf281lxfserver> and <https://bit.ly/lxf282lxfserver>.

» USE YOUR EMAIL SERVER FOR MORE

If you read through, rather than skipped, over the opening paragraphs of this tutorial, you'll have learned something about this writer's dislike of instant messaging services. When you use *WhatsApp* (just as an example), the app on your phone is just a front-end to a massive communications infrastructure, which comes with pitfalls around surveillance, confidentiality and lack of control. *WhatsApp*, which is really *Facebook* (aka *Meta*) in disguise, knows who you contact, when you contact, and can probably make an educated guess as to what you discuss. And then there's the fact that you're limited to contacting people within the *WhatsApp* network.

By using an app such as *Delta Chat*, your email server can take the place of instant messenger services, enabling you to communicate instantly and easily with your buds. *Delta Chat* is essentially a customised email client, with versions available for Linux, Android, Windows, MacOS and iOS. Chats are automatically encrypted using *Autocrypt*, and the layout and functionality is instantly familiar to anyone who's used *Telegram*, *WhatsApp* or *Signal*. Photos, videos and voice messages can be taken from within *Delta Chat*, and because it uses the email backbone rather than a proprietary walled garden, you can group-chat with whoever you please. Best of all, your messages are on your server and your devices – and under your control.



QUICK TIP

Set up different alias users to give as an address to shops, signups, and companies. You'll know who leaked, when the address starts catching spam, and can simply disable the user when you no longer need the account.

the questions and select the domains for which you want SSL certificates, and when prompted, choose redirect to ensure that all HTTP traffic is redirected to HTTPS. Check that this has worked by visiting your domains again and noting the presence of the tiny padlock in the URL bar.

The easy bit...

With the basic preparation out of the way, it's time for what would normally be the hard part – installing and configuring `dovecot`, `postfix`, `openDKIM`, and `Spamassassin` – but which, thanks to a handy setup script, is now the super-easy part.

From your home directory, run:

```
$ curl -LO lukemsmith.xyz/emailwiz.sh
```

If you want to see what the script does, and ensure that you're not dumping a bunch of malware on to your virtual machine, `cat emailwiz.sh` and take a look. We've read through it, and can assure you that at the time of writing, the script is exactly what it appears to be.

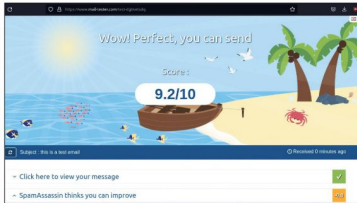
Make the script executable with `$ sudo chmod +x emailwiz.sh`, then run it with `$ sudo ./emailwiz.sh`.

Choose Internet Site as the general type of mail configuration, and for the System mail name, put your domain name, for example `lxf.guru`.

You'll find that there's just enough time to make a cup of tea as the emailwiz script does its thing, and when you return, you'll should find a success message, along with some ASCII art spelling out NOW in big red ASCII letters. Below this, you'll see three lines of text. These are records which need to be added on your registrar's DNS page, so head back over there, and select 'Add a new record'.

For one of our records, the emailwiz output gives us:

9.2 is a poor score and results from the lxf.guru being less than a fortnight old.



```
dmarc.lxf.guru TXT v=DMARC1; p=reject;
rua=mailto:david@lxf.guru; fo=1
```

This relates to the DMARC policies, and having a text record which states what should happen to emails which supposedly arrive from your server but don't pass certain validation tests, means that your mail is more likely to be successfully received.

In this record, we have set instructions that if the email doesn't match our DKIM (Domain Keys Identified Mail) record and SPF (Sender Policy Framework), it's to be rejected. Each email you send will cause the receiving server to generate a DMARC report giving you its evaluation of your SPF and DKIM, and whether the server is inclined to let your mail through or not.

You probably don't need to read these reports, but it's good to be able to see if anything is going wrong. If you do want to read the DMARC reports, change the `mailto` address to one that you'll actually use.

In our new DNS record, we set `dmarc.lxf.guru` as the host, `TXT` as the type and then

```
v=DMARC1; p=reject; rua=mailto:david@lxf.guru; fo=1
```

as the value. Note: depending on your registrar, you may need to delete the `your.domain` from the host field, and use `dmarc` instead of `dmarc.lxf.guru`. Repeat for the other two records and save. That's it. Your email server is set up and ready to roll.

Adding email accounts is as easy as creating a new user in the terminal and adding them to the mail group.

If, for instance your name was Fred, and you wanted an email account on your spanky new `themost.fun` server, you would pop into the terminal and type

```
$ sudo useradd -G mail -m fred
```

Create a password with `$ sudo passwd fred` and you're good to go.

If the fred user already exists then add it to the mail group:

```
$ sudo usermod -a -G mail fred
```

Remember that you'll need to log out and back in again to see new group memberships.

You've got mail!

To actually send or receive mail you need an email client or webmail app. You may already have a favourite email client, or if not, the odds are high that there's already a copy of *Thunderbird* hidden away on your system.

To get it up and running with your new email address and server, launch it and select Add Account. Fill in your name, email address and password, then click Configure Manually. For the incoming server choose IMAP: the

>> DON'T BE A SPAMMER

Official sources state that as of March 2022, spam emails constitute around 3,000 per cent of all internet traffic – encouraging users to purchase industrial-strength Indian Cialis and a range of other drugs, presumably to use while cavorting with all of the sexy singles in our neighbourhood.

Large email providers take measures to limit accounts on their servers from

being used as spam palaces by ne'er-do-wells. Gmail, for instance, will prevent you from adding more than 500 recipients for a single email and sending more than 500 emails in a single day. If you violate these limits then your Gmail account will be temporarily blocked, and if you repeatedly break the rules, you may lose your Google account entirely. ProtonMail limits are even tighter, with

both hourly sending limits and hourly BCC sending limits.

When you run your own email server, there are no such limits. You can send to thousands of people at the same time if you so wish. You can probably get away with this exactly once before your domain and IP address are blacklisted, meaning that you'll never be able to have anyone read your emails ever again.

server will be `mail.your.domain`. Choose SSL/TLS on 993. The username is your username.

For the outgoing server choose SMTP; the server will be `mail.your.domain`. Choose STARTTLS on 587. The username is your username.

Test the settings, and if everything is correct, then congratulations on setting up your new email server in under an hour!

But this tutorial series is all about getting the most out of your VPS and it would be remiss of us if we failed to provide you with VPS-hosted webmail so that you can retrieve your newsletters and chain mails from your grandma on any machine with a browser.

For our webmail we'll be using *RainLoop*, which is a good-looking app. It supports drag and drop, is easily customisable, and even integrates with Facebook, Google, Twitter and Dropbox – if that's your bag.

Remember the DocumentRoot you specified in your `mail.your.domain.conf` earlier? Head over there with:

```
$ cd /var/www/mail_your_domain/
```

Remove the file you created:

```
$ sudo rm index.html
```

then type the following:

```
$ sudo wget https://www.rainloop.net/repository/
```

```
$ webmail/rainloop-latest.zip
```

unzip the package

```
$ sudo unzip rainloop-latest.zip
```

You'll need to set the correct permissions:

```
$ sudo find . -type d -exec chmod 755 {} \;
```

```
$ sudo find . -type f -exec chmod 644 {} \;
```

And recursively set your apache user as owner for the application:

```
$ sudo chown -R www-data:www-data .
```

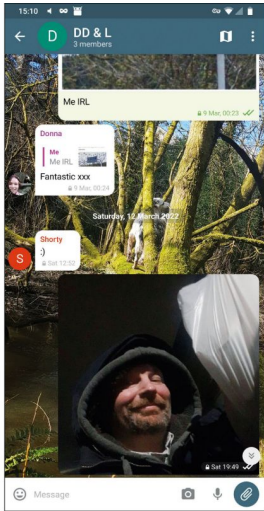
Now visit your `mail.your.domain?admin` and you'll be greeted with a login page. Hopefully no-one has managed to get there before you because the default login is admin, and the password is 12345.

Immediately after login, you'll be prompted to change the password. While you can ignore this prompt, you absolutely should not.

With basic security out of the way, you can set up *RainLoop* to interact with your email server. Locate Domains on the left hand menu, then click Add Domain. A pop-up menu will appear asking you to fill in the details. Fill them in exactly as we described for the *Thunderbird* setup above.

While you're in the admin panel, you can configure various other options such as the server name, the login and loading text, and set favicons. If everything is to your satisfaction, navigate to `mail.your.domain` and log in.

The layout is simple and easy to understand, with the usual sent, received, spam, drafts and trash folders on the left. A message list is on the left side of the main pane, which makes it possible to see the sender, subject and date. On the right is the message preview. Clicking a message in the list will cause the contents of the message to appear in the preview pane. You've used webmail before – we don't need to tell you this. For the sake of completeness, clicking the green New Message button enables you to draft a new message. But don't send one yet.



Using an encrypted email frontend like Delta Chat gives you all of the ease-of-use of a messenger app, but without the walled garden and with your messages under your control.

Email reputation is a funny thing, and if any part of your configuration is incorrect it can land you on an email blacklist, from which you'll find it difficult and time-consuming to escape.

Visit www.mail-tester.com and you will be given a unique address to which you can send an email for it to be evaluated. Copy the address down, and carefully compose an email to send to that address. Send it, then refresh the mail-tester page.

You should receive a 10/10 score. Anything less than that isn't great, and even a 10/10 doesn't guarantee that your mail will get through.

The email server at lxf.guru received a paltry 9.2/10, and further investigation revealed that although all of our TXT records were correct, the DKIM signature was valid and neither the VPS IP nor the domain name was blacklisted, we were losing points because we registered lxf.guru less than a fortnight before.

And with that, we were satisfied that the *Linux Format* mail server was fully operational. Yours should be too – and you didn't even break a sweat. Now go and email your friends and relations, and tell them subscribe to your favourite Linux magazine!

QUICK TIP

Unless you've encrypted your server, all of your emails are stored in plain text. If an attacker gains access to your server then they'll be able to read your emails and those of your users.

» ALLOW US TO SERVE YOU LXF... Subscribe now at <http://bit.ly/LinuxFormat>

MULTIPASS

Credit: <https://multipass.run>

Easily create instant virtual machines

Stuart Burns helps you to master Multipass, the virtualisation platform that runs on any desktop platform.



**OUR
EXPERT**

Stuart Burns is a cloud- and security-focused administrator specialising in large-scale virtual implementations.

Developers like to be able to launch virtual machines quickly for testing, but when developers write code on different platforms it can become a bit more complex to test if they have to use different virtualisation stacks.

Fortunately, Ubuntu provides a solution for quick, disposable Ubuntu-based virtual machines on Intel-based machines, and it's even supported on the new Mac M1 based devices (with a few limitations, naturally). This virtualisation software is called *Multipass*.

Multipass is basically a front-end tool to start virtual machines, but it can support many varied back ends including *VirtualBox*, *LXD*, *KVM* and others. *Multipass* is essentially an abstracted automation tool to create a VM on the underlying hypervisor, which is why it can work on most platforms because the underlying hypervisor works on those platforms.

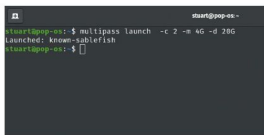
To install *Multipass* locally is straightforward.

Multipass on Linux does require *snap* to be installed first but that shouldn't be a problem with most modern Linux distributions. Install it and confirm installation using the commands below:

```
$ sudo snap install multipass
```

```
$ sudo snap info multipass
```

Instructions for Windows and Mac installation can be



Basic command line used to deploy a *Multipass* virtual machine with two CPU cores, 4GB of RAM and a 20GB hard disk.

found at <https://multipass.run>. *Multipass* comes with both a command line and a rudimentary GUI to boot. We're going to use the command line because it offers a lot more flexibility. After *Multipass* installation completes we'd strongly advise a reboot. Doing so will reload the configuration and start the GUI management widget as well.

First steps with Multipass

To start using *Multipass*, open a terminal window and use the following command:

```
$ multipass launch
```

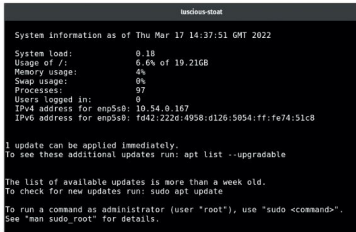
This command will download and create a new VM with a dynamically generated name. It may take a few moments to create a VM the first time, because it has to download the base Ubuntu image.

By default, *Multipass* will use the latest Ubuntu server image made available for *Multipass* use on the platform it's running on. There are additional optional VM base images available, which are discussed later. This first VM is special, and it's marked as the primary instance. As the name suggests, it's important. Out of the box (on Linux) the primary instance also mounts the users home directory. It shows up in the VM as a folder called **home** within the existing home folder, so files can be accessed as though they were part of the normal filesystem – so be careful what you remove or delete!

Additional virtual machines, however, don't have this facility, and require an additional *multipass* command:

```
$ multipass mount $HOME ifexample4 /home/home
```

It's possible to have several mounts on a VM and there's no need to export the mount to a variable. You



An example of the console that can be accessed with the *multipass* shell command. It can be treated like any other virtual machine.

can substitute paths instead of variables – for example, the `$HOME` could be `Downloads` to just expose the `Downloads` folder to the VM. Unmounting is done by using the `umount` command – `multipass umount lfxample4 /home/home`. The primary VM can also be accessed by a GUI console by going to the `Multipass` drop-down and selecting 'open shell'.

Other ways of getting an interactive shell on the VM include the `Multipass shell` command, which the reader can use to run an interactive BASH shell, like this:

```
$ multipass shell lfxample
```

This will open a console as though the reader was sat directly in front of the virtual machine.

Alternatively, a reader could set up a user from the console and then be able to SSH into it. Once inside the shell the VM can be treated just like any other VM. All the usual commands are available. It's literally a complete VM image. For example, to install Apache, just use:

```
$ sudo apt-get install apache2
```

It will install just like any other Ubuntu server. Using a browser to navigate to the virtual machines IP address should display the Apache default installation page.

Obtaining a list of all running machines is straightforward – just use:

```
$ multipass list -all
```

If the user doesn't need to have an interactive shell, then another way to run a command and return the information to the console is to use the `exec` option. For example, to obtain a configuration file, use

```
$ multipass exec lfxample
```

Adjust system resources

By default, `Multipass` can be rather modest in the amount of RAM and CPU given to a new virtual machine by default: just one CPU core and 1GB of RAM. Fortunately, it's easy to override these defaults and give the VM more resources, in this example we've supplied it with 4GB RAM and two CPU cores. Note that it's also possible to use these switches when creating the primary VM if desired.

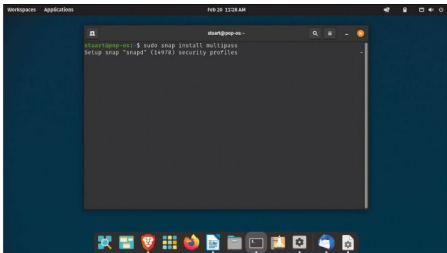
```
$ multipass launch -c 2 -m 4G -name lfxample2
```

The options should be easy to spot, with the `-c` referring to CPU core count and `-m` for memory. The memory does expect a unit of size after the number, be it `k` for Kilobytes, `m` for Megabytes and `g` for Gigabytes. This same custom sizing is available using the `-d` for disk sizing. To keep track of the created VMs, simply using `multipass list` will display its name, what's running, its IP addresses and the OS that it's running.

For those following along and creating the two example servers, to see more information about them there's the `info` parameter and its name, as shown below. This command will display the hardware and software details of the VMs:

```
$ multipass info lfxample2
```

Some more eagle-eyed readers may have noticed that the IP addresses are not on the local network.



Installing the Multipass snap in Linux. Doing this via the command line gives you more options, compared to taking the GUI route. Note that Multipass is only available as a snap install.

That's because they use NAT (network address translation) by default – in other words, in a private network that can reach the internet. They can also be seen and managed from the local workstation but by default they're invisible to everything else that's on the network.

In order to clean up the two examples we created (and cover stopping and starting) there are a few `multipass` parameters to use (`$ multipass stop <vm name>` or `$ multipass start <vm name>`). Once stopped, use the command `$ multipass delete <vm name>`. It won't ask for confirmation and will delete the virtual machine. To recover a deleted VM use `$ multipass recover <deleted VM>`. Alternatively, to purge the deleted VMs use `$ multipass purge`.

In summary, `Multipass` provides a way to automatically and speedily create disposable virtual machines for testing. Rather than having to manually recreate or redeploy a VM, it becomes as easy as one command to launch a new VM that can also see the host filesystem if required. **157**

QUICK TIP

Multipass often benefits from new features. One recent addition is a quick and easy Docker development environment. To deploy Docker in Multipass just requires `multipass -c 2 -m 4G docker`. docker and there will be a docker host installed to run containers in.

» ADVANCED ITEMS, MULTIPASS SET

options, including setting the VM to start automatically, selecting the hypervisor and advanced networking. Install `LXD` and all its requirements and then it becomes possible to use the different networks available within the virtualised environments. Change the hypervisor using the command `sudo multipass set local.driver="lxd"`. Make sure all the VMs are stopped; this may require a reboot. Once the VMs are running on `LXD`, to see the available network interfaces the VMs can utilise run `multipass networks`. To specify the VM to be launched is, use `--network switch` with the appropriate network name.

At the same time, the `set` parameter can be used to set the primary VM using `multipass set client.primary-name=new_vm`. Doing this makes the old one no longer the primary VM.



ALL HANDS ON DECK

Jonni Bidwell pries Valve's Steam Deck from PC Gamer's cold, anthropomorphised hands and gets his game on.

When the Steam Client came to Linux it was a watershed moment for gaming. At the time of its official release in January 2013, Linux gamers could run some 50 titles, most of which were Valve titles based on its Source engine. A year later, that number had risen to 300. Today, thanks to Valve's Proton (a fork of Wine featuring a DirectX to Vulkan transition layers), that number is over 6,000. Driving this recent push has been

the Steam Deck, Valve's all-new Linux-powered portable games machine.

This is undoubtedly one of the largest developments Linux gaming has seen since the turn of the millennium. So naturally we were keen to get hold of the hardware and see what it's capable of. The Steam Deck is powered by the third edition of SteamOS, which is based on Arch Linux. This gives it easy access to modern kernels, and the latest drivers for its (not inconsiderable) Zen 2 APU. This also

means we can to meddle with it in the same way we like to meddle with regular Linux PCs.

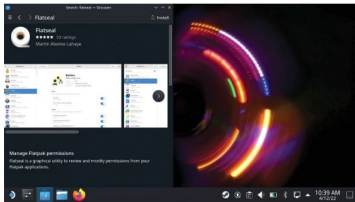
So gather round the Deck's crisp 1,280x720 display and see how you can customise and tweak this marvel of modern gaming. Also, it can do old games, DRM-free games and games that run in the terminal. And if you don't like games you can plug it into a dock and use it as a regular Linux device and starting at £350 it's not even that expensive!

With the benefit of hindsight, it's easy to see how Valve's previous innovations and efforts have informed the Steam Deck. First came the Steam Machines, but that never really worked out. Because who really wanted an overpriced gaming PC (however stylish) in their front room? Attempts were made to decouple the SteamOS operating system from the Machines, so that it would become a major league gaming distribution. That didn't really work either, since most Linux gamers were happy to install Steam on their current distro.

The choice of Debian as a base for SteamOS seemed reasonable from both a hardware and software stance. But the need for new libraries meant SteamOS was Debian Unstable with a custom kernel and various userspace modifications. Not the kind of thing people would want to be running as their daily driver. Maybe the plan should have been for Steam to work better on Steam Machines than regular Linux. Anyway, let's continue the story of Valve's hardware devices that should have done better.

Then came the Steam Controller and the Steam Link, also part of the Steam Machines initiative. The controller, with its unique haptic feedback and bounty of buttons/triggers/sticks, certainly had its fans – *Linux Format* among the most ardent – but it was discontinued in 2019. The Steam Link, a low-powered box that could stream Steam titles from a PC (ideally a Steam Machine), hoped to bring gaming into every room, but again it never really caught on. However, Valve did open source the code, so you can easily turn a Raspberry Pi into a Steam Link. There are also apps available for mobile devices and smart TVs.

The mid-2000s brought some notable games to Linux, including *Doom 3* (which was open sourced last year, *RBDOOM3BFG*) and *Unreal Tournament 2004*. Going back further there were other trailblazers too. Loki Software, founded in 1998, produced many ports in its short lifespan, starting with *Civilization: Call to Power* and going on to do dozens more, including the original *Unreal Tournament*, *Myth II* and *Sim City 3000*.



Ryan Gordon (aka icculus), Loki Software's founder continued to be involved in Linux ports after the company's demise. In particular, his work was instrumental to the early Humble Indie Bundles, ensuring such diverse titles as *Super Meat Boy* and *Dungeon Defenders* worked on Linux.

Add a drop of Wine

Later, porting houses like Aspyr and Feral Interactive would continue to bring big name titles to Linux (and MacOS), including the latter part of the *Tomb Raider* series, *Dirt Rally*, *Life is Strange*, *Mad Max* and more. Outside these native Linux ports (and some classic open source games and engine reimplementations) users' only gaming recourse was to play Windows titles through *Wine*.

Wine developers and the community put in a huge collective effort (and we don't want to know how long they spent staring at copious *Wine* debug messages) to make this viable. See <https://bit.ly/1xf289-fsckin-wine> to see how thorough people were. Today, lots of titles will run through *Wine* without modification, and if not then *Winetricks* will help you wrangle runtime libraries and .dll files without too much ado. In the past, it was not so simple.

There are great things to discover in the app store. Flatseal, for example, will help you wrangle Flatpak permissions.

» PROTON AND PROTECTION PROBLEMS

Valve has put a huge number of human hours into getting as many games as possible stamped with Playable or Verified status. But even Verified titles may have glitches or aberrations. And if you'd like to see them fixed then your best bet (as long as it's a title that plays through Proton) is to report the issue at <https://github.com/ValveSoftware/Proton/issues>.

Conversely many games currently don't work with Proton because of various anti-cheat technologies. On Windows these often work by installing a kernel module that monitors either the game's process or its communication with game servers. This prevents third-party cheats or hacks meddling with the game and granting players high scores,

enchanted boots or whatever. Unless these mechanisms are adapted for Linux (more precisely Proton) then titles that rely on them won't work.

Different publishers have different attitudes about this. For example, some games using *BattlEye* will work just fine with Proton, if the developer has opted in. Epic Games' Tim Sweeney has said that its *Easy Anti-Cheat (EAC)* technology, as used in the rather popular title *Fortnite*, could be circumvented in Linux.

Tim explicitly mentioned custom kernels, presumably because a hypothetical *EAC* kernel module (ignoring for a second the difficulties involved in creating and licensing such a thing) could easily be neutered by a homebrew effort.

EAC has (at least) two different modes though: the incompatible, low trust kernel module approach, and a higher trust mode used by some less world-dominating titles. This higher trust mode can be made to play nice with Proton, thanks to the *EAC* runtime introduced with Proton 7.0.

It's easy for developers that use *EAC* in their titles to flip a switch and get it working with Proton. That's how *Apex Legends* suddenly became verified. But hardcore *EAC* titles like *Fortnite*, *Halo* and *Rainbow Six Siege* won't see official Steam Deck Support anytime soon. There are a few ways to disable *EAC* on a per-game basis, but doing this will launch titles in 'offline mode', which for the likes of *Fortnite* isn't much fun.

Now a brief historical aside. In 2012, when Valve was doing its original Linux push (in which it made a native OpenGL version of *Left for Dead 2* that outperformed the Windows version), Gabe Newell described Windows 8 as a 'catastrophe'. He said that if Linux were better at games than more people would make games for it. With the *L4D2* demo, he hoped Valve would help.

In 2018 an exciting and seemingly unrelated project appeared. DXVK could take DirectX 10 and 11 code, and translate it into Vulkan, the new, close-to-the-metal, graphics language that would supersede OpenGL and compete with DirectX 12. This was much faster than previous efforts that had to target OpenGL, and had the pleasing consequence that AAA title *Nier:Automata* could be played on Linux, very close to native speed. DXVK appeared to be the work of a lone developer, albeit a rather prolific one, inspired by D9VK (which did the same thing for DirectX 9). Soon the VKD3D project (for DirectX 12) appeared. And seemingly overnight we were looking down quadruple barrels ready to blast

thousands of Windows titles onto Linux.

Not all Window titles would survive the ballistics of this metaphor. But pundits were amazed as DXVK continued to add support for more titles. And then came the big reveal: the developer (Philip Rebohle) had



We'd forgotten all about our Epic Games account. Now thanks to the Heroic launcher we can play them (and 60G titles) on the Deck.

been sponsored by Valve. DXVK, together with Valve's fork of Wine, would be at the heart of its new Proton offering, which would bring a raft of new Windows titles to Linux. Soon D9VK would be subsumed into DXVK and, VKD3D (originally a Wine project) was picked up by Valve after its founder passed away.

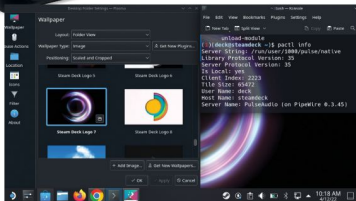
That takes us up to 2020, where in **LXF267** we really should have probed Collabora's Simon McVittie a little further about his and his employer's work for Valve. He talked about Pressure Vessel, essentially a container for Steam that enables titles (whether run natively or through Proton) to be run in consistent environments.

Pressure Vessel is similar in design to Flatpak and, it turns out, is instrumental to running games on Deck OS. So to form an awkward shell expansion, if we take the combination steam-{on linux, machines, controller} throw in Proton and cook inside a Pressure Vessel until there's more Steam, then we almost have a roadmap to the Steam Deck. Apart from the svelte form factor into which these components are squished.

We can tongue in cheekily speculate where this came from. The PC form factor was a non-event, a gaming phone is clearly a daft idea and Valve already has a XR project. This leaves a handheld as the only reasonable course of action.

The Steam KDE-ck

There are plenty of devices out there running Linux that don't appear to be doing so at first glance. If you'd never seen Android you wouldn't know there was a Linux kernel running underneath it. And the same is true for



The KDE install is pretty spartan, but not so spartan as to not include some pleasing backgrounds. Note that it uses PipeWire, too.

» ATOMIC UPDATES AND ENABLING READ-WRITE

Atomic updates are all the rage these days. ChromeOS, iOS, Android, Fedora Silverblue and EndlessOS (as well as most modern embedded devices) all feature an OS partition that's mounted read only. At update time, a new filesystem is downloaded, the partition is unmounted and the downloaded image is written out in a single transaction. This greatly reduces the probability of an update breaking the OS, especially if there are two such partitions that are updated alternately, ensuring a fallback (ChromeOS does exactly this). Having a read-only OS doesn't mean additional

software can't be added – it can – but it's added to a different partition (and optionally melded atop the root partition via UnionFS or some other voodoo).

SteamOS 3, as you've probably figured, uses atomic updates (complete with "A/B" shuffling, like ChromeOS) with additional software installed via Flatpaks (whose design is closely tied to OSTree images). Initially, there was some confusion around whether the OS partition could be unlocked, so that new packages could be installed and configuration files meddled with (it is based on Arch after all). Valve has

clarified the situation by saying this is possible, but not recommended. Any changes to that partition could be overwritten on the next update. Still, if you know Arch and don't mind picking up the pieces, read-only mode can be disabled by first setting a password (so that `sudo` doesn't complain), then running a script:

```
$ passwd
$ sudo steamos-readonly disable
```

If it all goes wrong then the Deck recovery image is your friend. See <https://help.steamowered.com/en/faqs/view/1B71-EDF2-EB6D-2BB3>.

all kinds of routers, smart TVs and other devices. The Steam Deck in gaming mode doesn't give away that it's running Linux, until you jump over into desktop mode and there's no mistaking the glorious full-fat KDE Plasma desktop.

One of many benefits of running Arch (apart from having cringey Redditors make cringey jokes about you) is that the latest versions of Plasma, Qt and the KDE Framework libraries all find their way into the official repos very shortly after release. So the version of KDE you find running on your Steam Deck might well be newer than the one that's running on your PC (if indeed you run KDE).

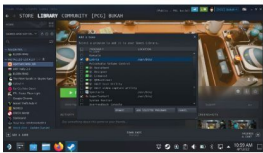
The layout is pretty standard, using the dark theme and classic applications menu. But desktop mode isn't just for show. Plug in a mouse, keyboard and monitor and the Deck is actually useful as a desktop PC. If (for some unholy reason) you don't like KDE then you can easily install something else. And if (and this one we really can't get our heads around) you don't like Arch Linux you can install a whole new distro on a different partition. Adding new software is easy thanks to the KDE Discover app store, which comes all hooked in to FlatHub. Software installed this way can even be added to the main Steam interface via the Add a Non-Steam Game option.

An official dock for the Steam Deck (wonder what it will be called?) has already been announced, which will certainly boost the Deck's utility as a regular PC. Until then you can happily use a regular USB-C dock with the Deck, but some people have reported issues powering the device from these. To be honest, given the number of problems these docks cause at *LXF Towers* we're looking forward to using one that works as advertised.

There are currently 1,000 odd verified titles and that same number of "playable" titles available for the Deck from Steam's massive catalogue. Probably these numbers will have increased by the time you read this (anything that already works on Linux, natively or via Proton, shouldn't need much work). But the Steam Deck doesn't want you to stop at Steam games. The Add a non-Steam option also enables you to install such wonders as the (Windows-only) *Epic Games Launcher*.

Lutris (the open source game manager) now has excellent Steam Deck support, but currently installing it requires us to break into that read-only sanctum that houses Deck OS. A Flatpak version of *Lutris* is in the works, so this will soon change.

DRM-free games from the likes of GOG and the Humble Store can also be shoehorned on to the main partition, or added via desktop mode. Again, Linux titles



Use the Add a Game option to add non-Steam things, like Lutris and the all-important SuperTuxKart.



With all the graphics settings turned up to 11, SuperTuxKart rarely dropped below 60fps.

from these sources should all work fine, excepting perhaps some controller-related quirks. We borrowed our Steam Deck from our digital friends at PC Gamer, who had taken it upon themselves to install Google Chrome as a Flatpak. It turns out that if you do this, and you have an Xbox Live account, then you can avail yourself of Microsoft's cloud gaming offerings. This will also work if you use Microsoft's *Edge* (is nothing sacred?—ED) browser, also available as a Flatpak.

The *Epic Games Launcher* also works fine with Proton, just switch to desktop mode, download and run. But why not try the open source *Heroic Games Launcher*

RESTRICTION-FREE GAMING AHOY!

“DRM-free games from the likes of GOG and the Humble Store can also be shoehorned on to the main partition, or added via desktop mode.”

instead, which supports both Epic and GOG. *Heroic* is available as a Flatpak, so you can add it to the Steam Deck from the Discover application. You can then add it to Steam as a non-Steam game in the usual way. The *Heroic Games Launcher* enables you to choose different versions of *Wine* and *Proton*.

But why stop there? If you're already a keen Linux gamer then you'll have already heard of *Lutris*, the open source game manager. In addition to GOG and Epic, it also supports Humble Bundle games, Steam itself (which might not be so useful on the Deck but who knows?), native Linux titles and many of the well-known Linux emulators including *Mame*, *Dolphin* and *RetroArch ScummVM*. Best of all a beta Flatpak of *Lutris* is available, so there's no need to violate the Deck's read-only sanctum. It'll probably be out of beta by the time you read this, so you won't need to add the FlatHub Beta repo as outlined at <https://github.com/flathub/net.Lutris.Lutris>. **BY**





The home of technology

[techradar.com](https://www.techradar.com)



HotPicks

FireDM » Mplayer » Rnote » Mailspring » Aircane
Fbcat » Topgrade » Superdud » Winterapples
Casper-fs » Zplora



Alexander Tolstoy believes that we should build better open source applications instead of a new iron curtain.

DOWNLOAD MANAGER

FireDM

Version: 2022.2.5

Web: <https://github.com/firedm/firedm>

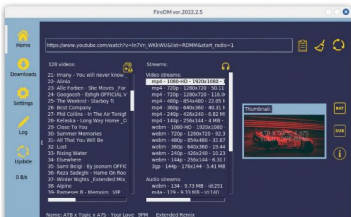
FireDM aims to be the ultimate all-in-one suite for downloading items to your hard drive. This application takes *Libcurl*, *Ffmpeg* and of course *Youtube-dl*, and wraps them up with a collection of Python scripts and a custom-made GUI. The result is a responsive and compact application that can download anything, be it an online video, an ISO image of a Linux distro you want to try, or just some random files.

As you might know, *Youtube-dl* supports dozens of popular video-hosting platforms, not just YouTube. *FireDM* works with *Youtube-dl*'s advanced features and enables you to download playlists, select video quality, download just subtitles or just the audio track, perform batch downloads and so on. Under the hood there are two versions of that tool: the original one, plus *Yt-dlp* – a feature-packed fork of *Youtube-dl*.

FireDM isn't particularly media-centric. If all you want to do is download a file, then *FireDM* is again at your service. The application will try to speed up your downloads by adding more workers and fetching a file using many streams at once. You'll see the progress bar with segments of each worker under the Downloads section. It's a step above the basic downloading method used by web browsers by default.

There are a range of settings for various non-standard cases. You can define *FireDM*-specific proxy settings, use custom website credentials, inject cookies, set custom user agent and even disable SSL validation. Just like the best download managers, *FireDM* monitors your clipboard and grabs links automatically, maintains all download logs and even auto-updates itself.

Everything should work out of the box, but you might be interested in tinkering with the *FireDM* settings, at least for switching to another colour theme, or setting the preferred way for minimising the program to its tray icon. We believe that anyone who regularly downloads files is going to have a good time with *FireDM*!



FireDM is a versatile download manager that can handle a wide range of files.

EXPLORE THE FIREDM INTERFACE



- 1 Start at the Home screen**
Once you have the link, review the download details under Home, set a target directory, manage batch tasks and more.
- 2 Control your downloads**
The Downloads section contains the list of files that were added either for downloading manually, or within a playlist.
- 3 Review download status**
FireDM usually fetches a file using several workers. It enables you to assess which segments are finished and which are pending.
- 4 Customise *FireDM***
The Settings section contains a lot of extras that may come in handy in special cases, such as when you need to authenticate your credentials within a website in order to retrieve a file from it.
- 5 Run the latest version of *FireDM***
The application can update itself. It can also check for new versions of *youtube-dl* and *yt-dlp*, and grab their latest versions.

MEDIA PLAYER

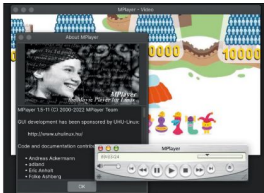
Mplayer

Version: 1.5

Web: <http://mplayerhq.hu>

Recently there's been an update to Mplayer, the video player for Linux and beyond. We settled down to read the changelog for version 1.5, but to be honest the changes aren't particularly ground breaking. There are some changes in the set of decoders, the code has been cleaned up, there's better localisation support (the GUI finally respects your locale!), and the new default skin enables you to run the Mplayer GUI even if there aren't any skins installed.

Mplayer 1.5 does a fairly good job of playing various types of audio and video content. It includes a modern version of *Ffmpeg 5* and adds some more drivers and codecs on top of it. Mplayer also provides its own CLI converter known as *Mencoder*. Obviously, transcoding any media file means writing a long command with several input and output arguments, which you may want to avoid by using some sort of a graphical transcoder such as *Handbrake* (see [LXF259](#)). However, it doesn't take long to get used to both Mplayer and Mencoder and in return you'll have access to robust and feature-rich playing and encoding capabilities.



Mplayer is a versatile player that supports almost any form of media.

Mplayer sports a wide range of supported media codecs. Historically, it comes from the early 2000s – the age of DVD ripping and video files down-sampling for use on tiny ancient phone screens. That said, you'll be surprised how many options there are in Mplayer if you just issue the `./configure --help` command in its source tree. Build them all, or perhaps go with a tiny and memory-efficient Mplayer installation. In a nutshell, this gives you plenty of optimisation options for your video library, if you have various tasks in mind. For instance, Mplayer is the perfect choice as a media player on Raspberry Pi and similar devices, whether you chose to run it in either a CLI or GUI mode.

By the way, exploring dozens of now naive-looking skins feels like a real step back in time. Most of them still work and, surprisingly, look good for their age!

NOTE-TAKING TOOL

Rnote

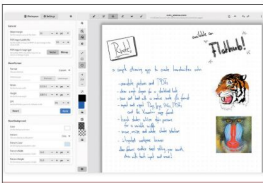
Version: 0.3.5

Web: <https://github.com/flxzt/rnote>

Running Linux on a device with touch controls reveals a whole set of extra usability requirements. One of those is a high-quality application for taking either hand- or finger-written notes. We've been looking for a capable open source version, and have come to the conclusion that Rnote is perhaps the best option that's available. It's a sleek graphical tool that provides you with an endless canvas and some drawing tools.

We didn't expect its set of tools to be that advanced, though. As well as a pen and a brush there are multiple selector tools, shapes, a dedicated marker tool, and of course an eraser. The brush tool has extra settings that you may be interested in adjusting. For instance, there's the cool 'textured' mode, which activates the gear button below it with extra options such as density, the radius of dots and their distribution. Such fine-grain controls seems more appropriate for a vector graphics package like *Inkscape*, or a drawing tool like *MyPaint*.

If you have a tablet with a digital pen, then you'll have an even better experience with Rnote and your notes



Take notes with ease and comfort, add more objects, scale and rearrange them any way you like.

will look as though they were made on paper. The program isn't solely about writing, though – you can add pictures, too. Use any of the available selection tools to move an object, scale or rotate it. Objects can overlap but Rnote doesn't enable you to either bring them to the front or push them into the background. To be fair, such functionality isn't usually called for when taking notes.

You can save your sheets in Rnote's own format, which isn't XML-based and can be read again only by Rnote itself. The latest version of the program is 0.3.5 at time of writing, and we failed to build it from source in Fedora because it required the latest GTK 4.6 library. Not a problem for a rolling distro with a bleeding-edge GTK4, but if you don't care about this then head over to Flathub and get Rnote up and running with no delay. It's a really nice tool that's worth a try!

MAIL CLIENT

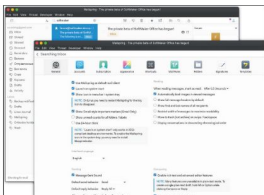
Mailspring

Version: 1.10 Web: <https://github.com/foundry376/mailedspring>

Mailspring is an email client that's been around for quite a while. It's accumulated a decent selection of new features since we last covered it in **LXF258**, so this is a good opportunity to revisit this outstanding software.

Mailspring is a direct alternative to *Thunderbird* and *Kmail*, and also to the web-based interfaces that we've all become used to. The reasons of choosing *Mailspring* over its rivals include various nuances that relate to productivity and ease of use. This application makes it possible to move several mail accounts into one workspace, and it supports *Gmail*, *Outlook*, *Office 365*, *Yahoo* and, of course, custom IMAP servers.

The interface is clean and full of numerous usability enhancements such as wizards, notifications and pop-up tips. As for more advanced features, there are routes for mail forwarding, a functional tray icon (something that *Thunderbird* still lacks), complex conditions in the search bar (for example, `in:inbox NOT is:unread`), a built-in translator that's available inside the new mail composer and more. Most of the aforementioned



Mailspring is compatible with popular mail services, and you can set up a custom IMAP server for all your email needs.

features are recent additions introduced in the past couple of years. The program is becoming more refined and flexible. On the latter point, the application no longer insists on signing up to the internal *Mailspring* account. If you don't want or need it then it's now possible to skip this step and proceed to add your target mail box.

The program also advertises the Pro version, which is available as a paid subscription. However, *Mailspring's* free version is still very good and doesn't leave you with the feeling that you're using a limited or a lacklustre mail tool. Every dialog or window conveys the efforts put into their usability by the dev team. If you're unsure about what standalone mail program to go with, give *Mailspring* a try. It offers RPM and DEB packages on the Github Releases section.

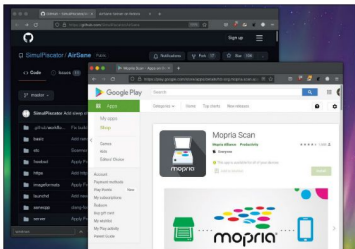
FRONT-END FOR SANE

AirSane

Version: 0.3.2 Web: <https://github.com/simplicator/airsane>

AirSane is an interesting piece of software designed for use with image scanners. The purpose is to let any device supported by SANE and plugged into a Linux machine to be accessible from MacOS and Android. In other words, if you have a flat-bed scanner or a multi-function device with a scanning feature that can be used with Linux, you'll also be able to scan over a network from other operating systems (although not Windows).

AirSane's target audience are people with several imaging devices shared across the home LAN using the 'net' backend of SANE. Indeed, scanning should work flawlessly between Linux hosts, but not for a Macbook or an Android phone. AirSane is a beautiful way to fix it. This software runs a server daemon, which publishes all devices detected by scanimage using the Linux-native Avahi daemon via mDNS. The technology includes parts of the reversed-engineered AirScan communication protocol used in Apple devices. Once AirSane is running, all detected devices are exported and available for all *ImageKit*-compatible tools on MacOS, and also for the



Mopria Scan app on Android. In addition, *AirSane* runs a web server at port 8090 for easy network scanning right from a web browser.

Not every Linux user will own a flatbed scanner. However, it's good to know that there's Linux support. *AirSane* is a great improvement on an already good level of scanning support in Linux. For example, the proprietary scanning tool *VueScan* for Linux won't work with SANE-shared scanners right away, but thanks to *AirSane* it will. Moreover, it's possible to tweak various colour settings by editing the `/etc/airsane/options.conf` file. Consult the *AirSane* documentation for details.

AirSaneMake your SANE-shared scanner work better with other operating systems.

SCREENSHOT TOOL

Fbcat

Version: 0.5.2

Web: <https://github.com/jwilk/fbcat>

The Linux text console may not look all that welcoming for running games or playing multimedia files, but it's still powerful and capable of performing lots of productivity tasks. We've discovered another handy tool you may find useful. *Fbcat* is a great catch for anyone seeking a way to take screen shots via the command line. It may first seem like another program in a long row of similar tools, but things are a little bit more interesting this time.

Fbcat is a low-level CLI tool capable of capturing a screen using a Linux framebuffer device. That means you can take screen shots directly from the TTY console, even when no display server is available. That's a great feature for remote servers' administrators and anyone dealing with a text Linux console.

The framebuffer output isn't formally compulsory in Linux, although most if not all distributions use it by default (check if `/dev/fb0` is present on your system). So we can assume there's hardly any Linux system that *Fbcat* won't run for. The software consists of two programs: one is a low-level *fbcat* grabber, and the

```

@stolsto@fedora ~$ LBRCN= fbgrab --help
/usr/local/bin/fbgrab: illegal option -- -
Usage: fbgrab [option... ] <filename.png>

Options:
-c 00      grab from /dev/tty
-C 00      grab from /dev/tty0 for slower devices
-d <dev>   use framebuffer device <dev>
-i         turn on PNG interlacing
-s 00      sleep 00 seconds before making screenshot
-?        display this help and exit

If the specified destination is "-", the PNG output is piped
to stdout.
@stolsto@fedora ~$
  
```

Take screen shots even if a black console mode is all you have to hand.

another is a high-level *fbgrab* tool. The former has almost no external dependencies apart from the `Make` and Linux kernel userspace headers; it writes the `/dev/fb0` contents directly to a PPM file that you specify. The latter supports PNG (via *Image- or GraphicsMagick*) and virtual terminal switching. As such, you can use *fbgrab* for more advanced tasks because it enables you to choose a virtual console (`/dev/ttyX`) and switch to a different 'fbdev' device. By the way, you can have as many as 64 virtual consoles in Linux. Explicitly specify the one you want to grab using the following template: `sudo fbgrab -c 1 screenshot.png`.

By the way, adding yourself to the 'video' group will eliminate the need to raise privileges for taking screen shots!

PACKAGE MANAGER

Topgrade

Version: 8.2.0 Web: <https://github.com/r-darwish/topgrade>

One of the well-recognised benefits of most Linux systems is the streamlined and unified update and delivery tools for managing software packages, which also enable good housekeeping with everything related to package management. Let's be honest, though: the situation isn't that clear-cut these days because we have Snaps, Flatpaks, Appimages, plus a bunch of parallel package managers used by software developers, such as *Pip*, *NPM* and *Cargo*. The problem is how to keep all that stuff up to date. The solution is *Topgrade*! We were a bit sceptical about the usefulness of this tool because unifying Linux distros, package managers, various frameworks and SDKs is a challenge in itself. However, *Topgrade* does a remarkably good job and works very well. This tool detects every supported package manager on your system and tries to run a network update against it. It even doesn't require any arguments: the simple `$ topgrade` command does its job completely in auto mode. For instance, *Topgrade* did a lot of upgrade steps on our test Fedora Workstation

```

@stolsto@fedora ~$ topgrade
Urgency: Critical
Vendor: unknown
Release Flags: + Is upgrade
Decompress:
  Remove ThinkPad P15 Gen 1/P17 Gen 1/P15q Gen 1/T15p Gen 1/P15v Gen 1
System Firmware Version: 1.23

The computer will be restarted automatically after updating BIOS comp
letely. Do NOT turn off your computer or remove the AC adaptor while update
is in progress.

Important updates: Update includes a security fix.

--- 16:18:18 - Summary ---
System updates: 0/0
config updates: 0/0
GNOME Shell extensions: 0/0
rustup: 0/0
pip3: 0/0
Containers: 0/0
Flatpaks: 0/0
Firmware upgrades: 0/0
@stolsto@fedora ~$ topgrade$
  
```

Upgrade every part of your system with just one single command. It's a terrific catch!

and saved us from a lot of manual operations. As such, it ran the updates for DNF, Flatpak, Rustup, Pip3, Docker, Firmware (*fwupd*) and GNOME extensions.

The list of items to update will vary depending on what's installed on your system. *Topgrade* can handle most popular package managers, and it's aware of dozens of modules, SDKs, managed repositories and more. All of those work beyond Linux too, so that you can unify your upgrade procedures by switching to *Topgrade* under Linux, MacOS and even Windows!

Topgrade is written in pure Rust, and therefore you'll need an appropriate toolchain to get it working. The installation is simple. Run the following command:

```
$ cargo install --path .
```

Then make sure you have `$HOME/cargo/bin` in your `PATH` and you're good to go.

VIDEO CONSOLE EMULATOR

Superdux

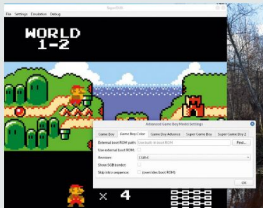
Version: GIT

Web: <https://github.com/snwmouse/superdux>

Superdux is a new project for providing a graphical interface for Sameboy, which is in turn an emulator for various retro gaming consoles produced by Nintendo. Sameboy is part of many Linux distributions, so if you're longing for your Nintendo games from around the late 90s, then this is the software for you.

However, Sameboy's barebones interface may put off some gamers, which is why we have Superdux. It's a Qt5-based front-end to Sameboy and provides quick access to its settings. Superdux enables you to quickly find out what GameBoy models are supported, increase video scaling, set up key bindings, adjust the sound, enable the FPS counter, apply colour correction and more. Even though Superdux is in its early stage of development, it's straightforward to start playing a game if you've obtained its ROM file.

Despite the fact that Superdux loads a tiny square black window, it's already configured with sensible defaults. Go to File>Open ROM and locate your game.



Make sure you have some gbc ROM files and then fire up Superdux.

By default, Superdux suggests you to use arrow keys and A, S, Z and X for recreating the buttons of the real Game Boy device. Of course, you can re-map them under Settings>Configure Controls. Each input device has its own configuration page with lots of extra actions that you might be interested in, such as Turbo, Slowmo and Rewind. The video resolution is relevant to the actual Game Boy screens and thus it's tiny (and square!). It makes sense to increase the virtual screen scaling ratio to either 4x or 6x.

We have had some happy Game Boy moments thanks to the broad set of Superdux features and the high-quality emulation. Superdux is a well-developed front-end to Sameboy with lots of tools available for both playing and debugging games.

ARCADE GAME

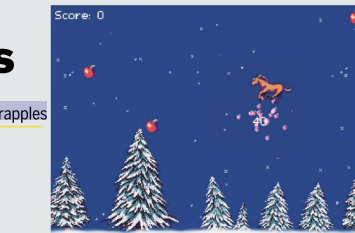
Winterapples

Version: GIT

Web: <https://gitlab.com/darkforce/winterapples>

Many of us consider apples to be good for horses, even though too many apples won't help their diet. However, in Winterapples many things are magical, and those delicious red apples don't cause harm to a horse. Instead, they give it even more energy and stamina, enabling it to make high jumps.

Winterapples is a fun game made with the famous Phaser engine. Phaser-based games usually run inside a web browser and feature high performance and smooth graphics. Winterapples is no exception. The game is a small effort, a one-man project – one of the thousands that we come across on Github and Gitlab every day – yet it's an entertaining game. The game features a nicely animated brown horse that can be controlled with a mouse and made to walk sideways. Left-click to make it jump and then try to catch a hanging apple. That will score some points and also make the horse jump again. The goal is to collect apples and keep ascending into the winter skies. Apples are placed in such a way that you always have a possibility to keep playing. Even if you miss one apple and the



horse slowly starts to descend, there's always a second chance to eat an apple and regain upwards momentum. Once you get 'high enough', don't miss a floating carrot – it doubles the score you get from apples. The more you play, the faster your score increases, but remember: the higher you go, the further you'll eventually fall.

Getting Winterapples up and running requires building the game using the supplied instructions (see [Readme.md](#)). After that you'll need to publish the game's files on a web server. It's enough to copy the assets to [var/www/html](http://www/html) and make sure your 'httpd' system service is running.

Help your horse reach new heights and eat more of those sweet apples!

LINUX KERNEL MODULE

Casper-fs

Version: 0.2 Web: <https://github.com/CoolerVoid/casper-fs>

Normally, you'd want to protect your sensitive information by setting read and write permissions using the classic Unix file attribute system that deals with users and groups. Alternatively, you'd use a password-protection method.

Neither of these two methods have anything to do with *Casper-fs*, which instead protects your file by simply not showing it. This is done via building a custom kernel module and injecting it into your running Linux system. The idea is to define a list of files you want to conceal in a *YAML* configuration file, build a tiny *casper-fs.ko* module and insert it to the running Linux kernel via *insmod*. From that moment on your system – and anyone else – won't realise those files exist.

To return visibility you can run `rmmod casper-fs` or reboot the system. However, *Casper-fs* has a built-in mechanism to turn files' visibility on and off on the fly.

The program has a few more features, such as protecting files without hiding them, and protecting the kernel module itself. The settings are stored inside the `module_generator/rules/fs-rules.yaml` file. You can



Casper-fs suggests a reliable low-level way of hiding files you don't want anybody to see.

change codewords for triggering the available protection features, as well as the name of the kernel module, and the fake device address.

We ran a series of tests of *Casper-fs* and found it to be interesting. It's not a replacement for fully fledged security tools. Rather, it's an auxiliary helper tool.

Casper-fs prevents unwanted attention to your secret files, and optionally prevents them from deletion.

These are two separate features that you can combine randomly. When a file is concealed, it's not shown by `ls` or any other command. Removing it using the `*` mask or a wildcard won't affect it, either. However, if you know the exact name of the file, you can still edit, move or delete it.

FILE EXPLORER

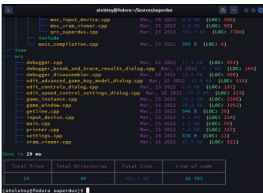
Xplora

Version: GIT Web: <https://github.com/hicodersofficial/xplora>

This isn't the first time we've come across a file management software intended to be used inside a Linux terminal. There are lots of full-featured file managers, like the classic twin-panel *Far2l* (LXF280) and the VIM-style *Ranger* (LXF253). This time we have a different tool that doesn't run all the time, but instead helps you on demand.

Xplora enables you to sort and then display your files and directories structure. It sits between a super-charged `ls` and a tree-based file manager. The goal is to let you quickly assess the number of files and subdirectories and identify what's using so much space.

Normally *Xplora* displays the tree structure of the current directory and then adds a table with some relevant statistics, such as the number of elements and their total size. However, *Xplora* is capable of much more once you become accustomed to its extra options. For instance, it can use filters based on file extension, file size, date and time, and also calculate code lines. The following example will reveal how many code lines are in header files:



Quickly filter your files and display their size, both in bytes and number of code lines.

```
$ xplora --extension .h --line-of-code
```

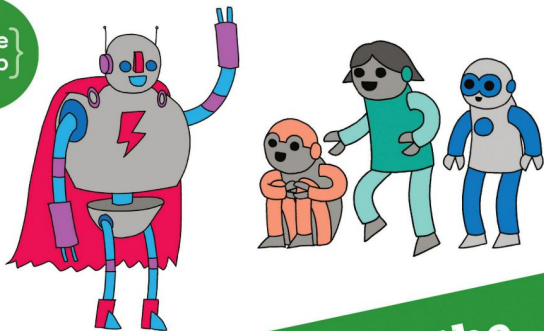
This command will locate not just the `*h` files in current directory, but also in all subdirectories (add `-nr` to run it in non-recursive mode). *Xplora* proved to be powerful and easy to fine-tune to specific tasks. You can explicitly instruct it to ignore certain files and even run a blacklist of such files (use `-igp` to point to an `ignorepath` file). *Xplora*'s output is neatly formatted and aligned. Subdirectories are displayed in green while files that match the filter are orange.

To get *Xplora*, simply install the respective NodeJS package from the public repository:

```
$ npm install -g xplora
```

Be sure to explore the built-in help function and discover even more features, including a custom start directory and more filtering options. **LF**

{code
club}



Can you help inspire the
next generation of coders?



Code Club is a nationwide network of volunteer-led after school clubs for children aged 9-11.

We're always looking for people with coding skills to volunteer to run a club at their local primary school, library or community centre for an hour a week.

You can team up with colleagues, a teacher will be there to support you and we provide all the materials you'll need to help get children excited about digital making.

There are loads of ways to get involved!

So to find out more, join us at www.codeclub.org.uk

RUST

Part Two
Missed part one? Turn to page 62 to get hold of it!

Develop Linux filesystem tools in Rust

Mihalis Tsoukalos explains how to manipulate and examine files and directories in Rust so you can write filesystem tools.



OUR EXPERT

Mihalis Tsoukalos is a systems engineer and a technical writer. Find him on Twitter using @mactsoouk.

The subject of this second Rust tutorial is working with files and directories as filesystem entities. This means that we're going to learn how to move, delete and copy files, explore directories, search directory trees and learn information about file permissions and file metadata. But first, we're going to learn about the `Result` data type.

To get started we're going to look deeper into the `Result` data type (`std::result::Result`) that we first saw in last month's Rust tutorial. `Result` is a Rust enum. An enum is a type with a list of predefined values. An enum variable can only have one of these predefined values at any given time. The definition of `Result` is as follows:

```
pub enum Result<T, E> {
    Ok(T),
    Err(E),
}
```

Given that, have in mind that we can change the signature of `main()` to return a `Result` value. So, because the `Result` enum has usually two values, `main()` can be rewritten to something like the following:

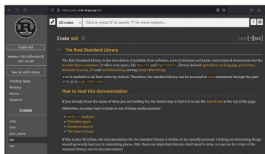
```
fn main() -> Result<, ParseIntError> {
    ...
}
```

You can replace `ParseIntError` with any error value you want, including the generic `std::io::Error`, or you can omit it entirely. You can learn more about `Result` by visiting <https://doc.rust-lang.org/std/result/enum.Result.html>.

Rusty Standards

Rust comes with a rich Standard library that can be extended with the use of external libraries, which in Rust terminology are called crates. This section presents the most useful functions and modules from the Standard library of Rust that are related to the subject of this tutorial – the standard Rust library is included under the `std::` crate. So, the list of handy Rust modules includes the following:

- > The `std::path` module contains functions and methods for cross-platform path manipulation.
- > The `std::env` module enables you to obtain



This shows the documentation page of the `std` crate, which is part of the Rust Standard Library and contains lots of handy functions, data types and macros.

information about the environment the process runs on. So, with its functionality you can learn about environment variables, the current directory as well as other important directories.

> The `std::result` module is about error handling with the `Result` data type, which was presented in the previous section.

The screenshot (above) shows the official documentation page of the Rust Standard Library – you can find more about it by visiting <https://doc.rust-lang.org/std>. Let's take what we've covered and use it to write a small command line utility that differentiates between regular files and directories.

File or directory?

Our first step will be to differentiate between regular files and directories. The Rust source file is named `ftypes.rs`. The logic of `ftypes.rs` can be found in the following code excerpt:

```
let md = metadata(&argument).unwrap();
if md.is_dir() == true { ... }
if md.is_file() == true { ... }
```

The first statement reads the metadata of the provided path in order to examine it. So, if the `is_dir()` method returns true, we're dealing with a directory whereas if `is_file()` returns true, we're dealing with a regular file. It's up to the developer to decide what code to write and actions to take when dealing with a

QUICK TIP

Get the code for this tutorial from the Linux Format archive: www.linuxformat.com/archives?issue=289

directory or a regular file. Our utility just prints information on screen. So, compiling and using **ftypes.rs** produces the following output:

```
$ ./ftypes ftypes.rs
ftypes.rs is a regular file.
$ ./ftypes /tmp
ftypes.rs is a directory.
$ ./ftypes /doesNotExist
The /doesNotExist path doesn't exist!
$ ./ftypes /dev/nvme0n1p2
```

The first two commands generate the expected output when processing a regular file and a directory, respectively, whereas the third command shows what kind of output to expect when the provided path cannot be found. The last command shows that you obtain no output when the provided path is neither a directory nor a regular file.

In the section that follows, we're going to write a simplified version of the *which* utility in Rust. If you don't already know about *which*, type **man which** on your Linux shell to visit its manual page.

Let's code which!

Which returns 0 on success and 1 on failure. We should be able to implement that in Rust to be fully compliant with a *copy-cat* which command. The logic of **which.rs** can be found in the next code excerpt:

```
for argument in env::args().skip(1) {
    let split_path = my_path.split(":");
    for v in split_path {
        ...
    }
}
```

The previous code processes all command line arguments apart from the first one because of the use of **skip**, and tries to find a match in the different paths of the PATH environment variable, which has been read previously and stored in the **my_path** variable. How you check each file path isn't that important (see the screenshot, top right, for more) – what's important is making sure that you process each command line argument the correct way. Bear in mind that if you give multiple command line arguments to **which**, and one of them isn't found, the return value is going to be 1 – our utility implements that as well. Using **which.rs** (remember that you have to compile it first) creates the following type of output:

```
$ ./which sed
/usr/bin/sed
$ echo $?
0
$ ./which sed123
sed123 not found
$ $ echo $?
1
```

We use the **echo \$?** *zsh* and *bash* shell command to examine the exit status of the previous Linux command. As desired, the first command returns 0 whereas the second command returns 1. This is really important when you want to include your Rust command line utilities into shell scripts, *Ci/CD* jobs or other tasks that check the exit code of a command to determine its success or failure.

The screenshot (top right) shows part of the code of **which.rs**. In order to return the desired value, we use

```
for v in split_path {
    let path = format!("{}", v, argument);
    let b = Path::new(&path).exists();
    if b == false {
        continue;
    }

    let md = fs::metadata(&path).unwrap();
    if md.is_file() == false {
        continue;
    }

    let mode = fs::metadata(&path).unwrap().mode();
    if mode & 0b111 != 0 {
        println!("{}", path);
        found = 1;
    }
}
```

the **std::process::exit()** function. Strictly speaking, **std::process::exit()** terminates the current process – at this point our program has a single process only – with the specified exit code. We also need a (mutable) variable that holds the final return value, which is called **ret_value**.

The final value of **ret_value** depends on the found variable: if an argument isn't found, then the value of **ret_value** changes from 0 to 1 and can't become 0 ever again. You can learn more about it by visiting <https://doc.rust-lang.org/std/process/fn.exit.html>.

How to copy a file

Let's create a tool to copy a file, just like the *cp* utility does, but without supporting all of the options of *cp*. This means that you copy a file without writing any File I/O system calls – this is handled by Rust. Using File I/O operations to process a file will be covered in the Rust tutorial of next month's issue of *Linux Format*. The name of the Rust source code file is **copy.rs**. The core functionality of **copy.rs** is presented in the following code excerpt:

```
match fs::copy(input,output) {
```

This shows the Rust code of **which.rs** that presents an implementation of the *which* utility in Rust. Rewriting UNIX command line utilities is a great way to learn a new systems programming language.

QUICK TIP

If you are not familiar with Go or Python, it is good to know that the use of **walkdir::WalkDir** is pretty similar to the **filepath::WalkDir** function found in the Go Standard library and **os::walkDir** found in the Python library.

» THE RUST STANDARD LIBRARY

In Rust terminology, **std** is a crate, and everything under it is called a module. It's helpful to have a look at the following modules:

> **std::string** is for working with the String data type. <https://doc.rust-lang.org/std/string/struct.String.html>.

> **std::vec** contains the Vector data type. You can learn about it at <https://doc.rust-lang.org/std/vec/struct.Vec.html>.

> **std::net** contains networking functionality related to TCP/IP. <https://doc.rust-lang.org/std/net/index.html>.

> **std::fmt** contains utilities for printing strings – see <https://doc.rust-lang.org/std/fmt/index.html>.

> **std::thread** module is about native threads. See <https://doc.rust-lang.org/std/thread/index.html>.

> **std::os** provides OS specific functionality. For Linux-related info see <https://doc.rust-lang.org/std/os/linux/index.html>.

> **std::fs** contains filesystem manipulation operations. Its help page is at <https://doc.rust-lang.org/std/fs/index.html>.

> **std::io**, which is going to be used in the next tutorial, is all about File I/O. See <https://doc.rust-lang.org/std/io>.

You can learn even more about the Rust Standard Library at <https://doc.rust-lang.org/std/>. Finally, visit <https://doc.rust-lang.org/core/index.html> to find out more about the core crate.

QUICK TIP

You can learn more about Rust at www.rust-lang.org/learn and you can ask questions at <https://users.rust-lang.org>. The Rust playground, which can be found at <https://play.rust-lang.org>, enables you to experiment with Rust from your web browser without having to install Rust first.

```
Ok(n) => println!("Wrote {} bytes", n),
Err(err) => println!("Error: {}", err),
};
```

Everything here is done by the `fs::copy(input, output)` call, which copies the file, provided that the input path already exists. After executing `fs::copy(input, output)`, we examine the return value of it, which can be either `Ok(n)` or `Err(err)`. In the first case, the call returns the number of bytes written, which is stored in variable `n`, and in the second case, we print the error message returned by the call.

The screenshot (bottom of page 92) shows the main code of `copy.rs`. Although `copy.rs` catches the case where the input file doesn't exist, `fs::copy()` might catch other types of error conditions such as insufficient Unix permissions to read the input file and not enough permissions to write the output file. Apart from that, bear in mind that most of the code is about catching error conditions and reading user input – `fs::copy()` is just a single statement.

Renaming files

This brief section presents a small utility that can rename files. The source code file is called `rename.rs`, which shares most of its code with `copy.rs` because they both require two command line arguments. The logic of `rename.rs` is found in the `fs::rename(input, output).expect("Unable to rename")` statement. The `fs::rename()` function does all the job of renaming and is also able to catch errors related to filesystem permissions. Bear in mind that `fs::rename()` requires that both file paths are in the same filesystem. Please look at `rename.rs` for more details.

Deleting files

Deleting a file is an essential task for an operating system. However, we should be very careful when deleting files. The code functionality of `delete.rs` is implemented using the `fs::remove_file()` method. Working with `delete.rs` generates the following type of output:

```
$ ./delete /tmp/copy.rs
$ ./delete /tmp/copy.rs
thread 'main' panicked at 'called Result::unwrap() on an Err value: Os { code: 2, kind: NotFound, message: "No such file or directory" }', delete.rs:15:31
```

If you can run this with the `RUST_BACKTRACE=1` environment variable to display a backtrace.

The first command was executed successfully. As expected, trying to delete the same file twice fails. In that case, Rust itself generates an error message that explains the error situation.

File permissions

In this section you're going to learn how to get the permissions of a file. The logic of `permissions.rs` can be found in the next code excerpt:

```
let metadata = fs::metadata(input)?;
let perm = metadata.permissions();
println!("{}", perm.mode());
```

First, we read the metadata of a file and then we obtain its file permissions by calling `metadata.permissions()`. Last, we print the file permissions as an Octal value using `{:o}` in `println!()`. Using `permissions.rs` produces the following output:

```
$/permissions permissions.rs
100644
$/permissions permissions
100755
$/permissions /tmp/DoesNotExist
The /tmp/DoesNotExist path doesn't exist!
```

The first three digits are not of interest at this point – the last three digits contain the information we're looking for. So, the file permissions of `permissions.rs` are 644 (rw-r--r--) and the file permissions of the executable file are 755 (rwxr-xr-x). If the file doesn't exist, we receive a descriptive error message. You can learn more about file permissions by visiting the man page of `chmod` (`man chmod`).

File details

Let's take a quick look into interrogating Linux files using various Rust functions and methods provided by Rust. All this new functionality is included in `details.rs`. Running `details.rs` generates the following output:

```
$/details details.rs
Is file: true
Is dir: false
Is Symlink: false
Len: 908
Last accessed: Ok(SystemTime { tv_sec: 1635349397, tv_nsec: 554953691 })
Last modified: Ok(SystemTime { tv_sec: 1635349396, tv_nsec: 714952586 })
Created: Ok(SystemTime { tv_sec: 1635349396, tv_nsec: 714952586 })
```

The screenshot (facing page) shows the Rust code of `details.rs`.

Creating a directory

This section will teach you how to create a new directory. The presented utility can create a directory structure – that is, all directories in a path, if some of them are missing. This also works for creating a single directory, if all parent directories already exist. The recursive(true) part is what makes `DirBuilder::new()` to recursively create any missing directories. Using `createDir.rs` produces the following output:

```
$/createDir /tmp/1/2/3/4/5/6/7
$/createDir /tmp/1/2/3/4/5/6/7
The /tmp/1/2/3/4/5/6/7 path already exists!
```

The output of `tree /tmp/1` is going to verify that `/tmp/1/2/3/4/5/6/7` was successfully created. Bear in mind that `DirBuilder::new()` isn't the only Rust function

```

20 // this should not exist
21 //
22 let c = Path::new(&output).exists();
23 if c == true {
24     println!("The {} path already exists!", output);
25     return
26 }
27
28 match fs::copy(input, output) {
29     Ok(n) => println!("wrote {} bytes", n),
30     Err(err) => println!("Error: {}", err),
31 }
32 } else {
33     println!("Not enough command line arguments.");
34 }
35

```

Here's the code of `copy.rs`, which shows how to copy a file in Rust. The core functionality is implemented by the `fs::copy()` function.

that can create directories. There are two other functions named `create_dir()` and `create_dir_all()` in `std:fs` that can be used for this purpose.

Finally, there are two functions for deleting directories: `remove_dir()`, which is used for deleting single directories; and `remove_dir_all()`, which is used for deleting directory structures, just like `create_dir_all()` is used for creating directory structures. Both functions can be found in `std:fs`.

A simple find

Let's pull all this knowledge together and code a simplified version of `find` that visits multiple directories and searches for Rust source files, which are files with the `.rs` file extension. We're going to use a Cargo project because we need an external crate. The first task is to create the Cargo project:

```
$ cargo new find --bin
$ cd find
```

Bear in mind that by default, Cargo assumes that you want a GitHub repository for your project. If you don't want to use that capability, you can delete the `.git` directory and the `gitignore` file from the root directory of your Cargo project. We need to edit `Cargo.toml` to add a dependency – the final version of `Cargo.toml` is going to be as follows:

```
[package]
name = "find"
version = "0.1.0"
edition = "2018"
[dependencies]
walkdir = "2"
```

So, the name of the utility is `find` and its only dependency is the "walkdir" crate – this crate contains `walkdir::WalkDir()`, which is very powerful and accepts many options. By default, `walkdir::WalkDir()` processes everything under a given directory root specified with `new(<root path>)`. If you want to filter the output, you should use `filter_entry()` or `filter()` or both. If you examine the contents of `./src/main.rs`, the `filter_map()` part shows what to do with any errors that might come up – in our case, we silently ignore all file entries with errors.

Remember that if you don't want to take any actions on the entries other than printing, you can just use `walkdir::WalkDir` as follows, which is the simplest form of `walkdir::WalkDir` usage:

```
for entry in WalkDir::new(<root path>){
    println!("{}", entry?.path().display());
}
```

However, in our case we use `WalkDir::new()` in a more advanced way. For each entry returned by `WalkDir::new(argument)` that isn't a directory, we examine whether the file name ends with the `.rs` file extension or not. The `filter(|e| e.file_type().is_dir())` part makes sure that we're not examining directory entries and the `if !name.ends_with(".rs")` statement checks the file extension. If it has the `.rs` file extension, then we print the relative path of that file.

Apart from the use of `walkdir::WalkDir`, the utility makes the necessary actions that you would expect from a system utility: reading user input as a command

» TYPES OF UNIX FILES

In Linux and Unix, everything is a file, even your printer! As a result, there are many types of Unix and Linux files, including the following.

- ▶ **Plain text files:** these are used for storing text used for system configuration files, YAML files, source code, etc. Easily processed line by line, character by character or even word by word.
- ▶ **Binary files:** from an OS perspective, these are just like plain text files. The reason for a separate entry is that usually there's no point in processing binary files the same way as plain text files.
- ▶ **Directories:** these filesystem entries contain lists of files and references to other files instead of any kind of user stored data.
- ▶ **Links:** there exist two kinds of links, which are symbolic links and hard links. Please visit the man page of `ln` for more information.
- ▶ **Unix domain sockets:** this kind of file is used for IPC (Inter Process Communication). Put simply, they enable processes on the same machine to communicate with each other, which saves you from having to create a TCP/IP network connection.
- ▶ **Character device files:** this kind of file includes devices such as terminals, keyboards, printers and mice.
- ▶ **Block devices:** devices like hard disks, DVDs, tape drives and so on.
- ▶ **Named pipes:** used for IPC and is an extension of the regular pipe.

line argument and making sure that the user input is an actual directory. You can build the project with `cargo build`. Using the `find` utility – its executable is `./target/debug/find` – generates the following output:

```
$ ./target/debug/find .
./src/main.rs
$ ./target/debug/find ..
./rename.rs
./find/src/main.rs
...
./permissions.rs
```

You can discover more about directory traversal at <https://rust-lang-nursery.github.io/rust-cookbook/file/dir.html>. In this tutorial, we learned a lot of handy information about working with the Linux filesystem in Rust. The next Rust tutorial is going to be all about File I/O (File I/O operations are an essential part of all operating systems), which includes reading files, writing files, appending to files and more.

You can learn more about Rust at www.rust-lang.org/learn and you can ask questions at <https://users.rust-lang.org>. Finally, the Rust playground, which can be found at <https://play.rust-lang.org>, enables you to experiment with Rust from your web browser without the need to install Rust on your machine. **👉**

```
10 let mut entries = Vec::new();
11 let k = PathBuf::from(<argument>).canonicalize();
12 if !k.is_dir() {
13     println!("The {} path does not exist!", argument);
14     return;
15 }
16
17 let file_extensions = &[".rs"];
18 let mut file_extensions_iter = file_extensions.iter();
19 while let Some(file_ext) = file_extensions_iter.next() {
20     println!("Searching for {} files...", file_ext);
21     let entries_iter = WalkDir::new(k).into_iter().filter(|e| e.file_type().is_dir());
22     for entry in entries_iter {
23         println!("Found {} file: {}", file_ext, entry.path().relative_to(k));
24     }
25 }
26
27 println!("Done. {} files found.", entries.len());
28 }
```

QUICK TIP

If you want to experiment, you can try writing a utility that copies entire directory structures including their files. Additionally, you can add a flag in order to only duplicate a directory structure without the included files.

This screenshot shows the source code of details.rs. The details.rs utility illustrates how to get detailed information about file system entries.

» **GET MORE RUSTY WRITTEN THINGS** Subscribe now at <http://bit.ly/LinuxFormat>

PYTHON

Credit: <https://github.com/bashkirtsevich-llc/PyBlackJack>

Update and improve old blackjack code

Updating old projects can be fun and educational. **Andrew Smith** ensures that your cards are dealt correctly at any resolution.



OUR EXPERT

Andrew Smith is a software developer at NHS Digital, and has qualifications in software engineering and computer networks.

We're going to take a look at an old *Blackjack* project and see how we can update and modify it. *Blackjack* is a card game where the aim is to get a value total of 21 (or closest to 21) across the cards you're given, to win a round of the game.

The original project contains some interactive features that we can experiment with, such as buttons and events, along with some pleasing visual resources that we'll look at in this tutorial. The original project that this tutorial is based on was created by Allan Lavell, and the original source code can be retrieved from <https://github.com/bashkirtsevich-llc/PyBlackJack>.

During this tutorial we'll cover how the game program works and what resources are included for the project to work. We'll then look at positioning elements on the screen in relation to the screen resolution that the program is running in. We'll also add a button to the program, which will be used to exit the game.

Shuffling your cards

For this tutorial we'll install and set up the latest version of Python (3.10). For those that have Python/PyGame already installed, Python 3.8+ should be fine. Type the following code to install Python 3.10 and PyGame.

```
sudo apt-get install python3.10
sudo apt-get install python3-pip
python3.10 -m pip install pygame
```

Check both the Python and PyGame versions. Next, `git clone` from repository, `git clone https://github.com/asmith1979/lxf289_blackjack/`

The project has been put into a folder called **PythonProjects**, which was created before downloading the project. Alternatively the source code and project can be retrieved from the **LXF289** archives at <https://linuxformat.com/archives>.

This tutorial will focus on the source code located in the **lxf289_blackjack** folder. Type `cd lxf289_blackjack` to open the folder and gain access to the Python source code. You'll see two Python source code files: **blackjack.py** and **lxf289ans.py**. The Python file **blackjack.py** is the Python script file that you'll be editing and **lxf289ans.py** contains the full tutorial code.



A new game of Blackjack. The dealer's hand is shown at the top half of the screen, while the player's hand is at the bottom half of the screen.

To edit and view the source code you can either use your distro's default text editor or more specific programs such as *Notepad++*, *PyCharm* or *VS Code*. We'll be using *gedit* to view and edit the source files. If and when using this method to view/edit source files, it may be helpful to open two console windows. One for editing and viewing source files, and the other terminal window for executing the PyGame code.

Project overview

This project requires a lot of graphical resources for the program to work. For example, you'll find the full 52 deck of cards in the **/images/cards** folder. Navigate to this location on your machine to have a look at the card images that have been stored. You'll see that there's an image for each card in the deck as well as a back-facing card image which is used in our *Blackjack* program. Card images have been created for each suite: Hearts, Diamonds, Clubs and Spades.

In the folder that's above this one – **images** – you'll see all the other image resources that are used in the game program. This includes the background image for the game and images of various buttons that the player can interact with during gameplay.

This Python project also uses sound to enhance the player experience. The sound effect file that we'll be using is located in the **sounds** folder. Inside this is the file **click2.wav**.

QUICK TIP

When choosing names for classes, functions or variables, make them meaningful so that someone else reading the code can see what they're intended for

The code for the project consists of independent functions and object orientated programming techniques (classes). It's not the aim of this tutorial to cover all the code in-depth, but we will provide an overview of the main important parts that make the game program work and will bear some relevance with what's required later in this tutorial.

Because there are a lot of images used for this project, you may or may not have caught on to the fact that a lot of image loading takes place initially so that all image resources are ready for the game program to use. The function to use to load all images for the project is called `imageLoad`, which can be seen near the top of `blackjack.py`. Its code looks like the following:

```
def imageLoad(name, card):
    if card == 1:
        fullname = os.path.join("Images/cards/", name)
    else:
        fullname = os.path.join("images", name)

    image = pygame.image.load(fullname)
    image = image.convert()

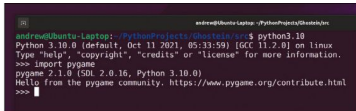
    return image, image.get_rect()
```

Overall, this function takes in two arguments: one is the file name and the other is an argument to identify whether or not a card image needs to be loaded. After loading the image into memory, the function returns both the image as an object as well as a rectangle surrounding the image. However, the function doesn't account for different screen resolutions that the program might be run in. This would result in, for example, the background image looking out of proportion with the rest of the program's visual elements. To correct this, we can make some changes to our program as well as to the `imageLoad` function as described below.

First, near the top of the program, after where the Python libraries are included, type the following code:

```
BACKGROUND_IMAGE = 0
CARD_IMAGE = 1
BUTTON_IMAGE = 2

SCREEN_WIDTH = 800
```



```
SCREEN_WIDTH = 600
```

```
# SCREEN_WIDTH = 1280
# SCREEN_HEIGHT = 718
```

We now have constant values for the background image, card images and button images, which saves us having to use numeric values such as 0, 1 and 2. We can refer to them by name instead.

The next step is to rename the second argument passed into `imageLoad` from `card` to `imageTypeIn` because now we're trying to differentiate between three types of image – the background image, card image and button image – the details of which can be passed into the `imageLoad` function. Values for the screen dimensions have also been added here, where one set of screen dimensions have been added but commented out. This enables us to experiment with different screen resolutions for the program to run in.

Not too far down (or up) from this, is a line that reads `screen = pygame.display.set_mode((1280, 718))`. Replace 1280 with `SCREEN_WIDTH` and 718 with `SCREEN_HEIGHT` (or commonly known as substituting).

The next step is to add the following code into the `imageLoad` function itself after the line `image = pygame.image.load(fullname)`:

```
if imageTypeIn == BACKGROUND_IMAGE:
    image = pygame.transform.scale(image, (SCREEN_WIDTH, SCREEN_HEIGHT))
```

This code will expand any image loaded in as a `BACKGROUND_IMAGE` to the screen dimensions that the program has been told to run at. To test this, write the following code just after where you specified `BACKGROUND_IMAGE, CARD_IMAGE`, etc. The function now should look something like the following:

```
def imageLoad(name, imageTypeIn):
    global SCREEN_WIDTH
```

The screenshot shows the setup and configuration of Python and PyGame after being correctly installed.

QUICK TIP

It's useful to look at existing examples of classes and functions as a guide as to how write your own.

» OBJECT ORIENTATED PROGRAMMING (OOP)

There have been quite a few object orientated programming techniques used in the project. For example, classes have been used to create all the buttons we interact with in the game program: Hit Button, Deal Button, Stand Button... etc. Each button class contains various properties and functions for the necessary operations to take place. Below we look at the `dealButton` class as an example. Only the main parts of the class are shown to demonstrate the main concepts and ideas.

```
class dealButton(pygame.sprite.Sprite):
    def __init__(self):
```

```
        pygame.sprite.Sprite.__init__(self)
        self.image, self.rect = imageLoad("deal.png", 0)
        self.position = (1155, 425)
```

```
        def update(self, mX, mY, deck,
                    deadDeck, roundEnd, cardSprite, cards,
                    playerHand, dealerHand, dCardPos,
                    pCardPos, displayFont, playerCards,
                    click, handsPlayed):
```

```
            # Get rid of the in between-hands
            chatter
            textFont = pygame.font.Font(None,
            28)
```

—

A class is first declared using the class keyword as demonstrated above, class `dealButton` in this case. The next step is to write out the class constructor identified by `def __init__(self)`, which is used to initialise properties (or variables) to certain values when an instance of the class is created. In the above example, the class constructor is used to load in an image of the deal button and also set the position of the button. Also in this class is an update method (or function) that's been implemented to update positions during gameplay.

```

global SCREEN_HEIGHT

if imageTypeIn == CARD_IMAGE:
    fullname = os.path.join("images/cards/", name)
    else:
        fullname = os.path.join("images", name)

image = pygame.image.load(fullname)

if imageTypeIn == BACKGROUND_IMAGE:
    image = pygame.transform.scale(image, (SCREEN_WIDTH, SCREEN_HEIGHT))

image = image.convert()

return image, image.get_rect()
    
```

Because `SCREEN_WIDTH` and `SCREEN_HEIGHT` are global variables, we include them into this function by using the `global` keyword. To test that our code works, run the program with the existing screen settings and then run the program again, commenting out the existing `SCREEN_WIDTH` and `SCREEN_HEIGHT` and removing comments from the other `SCREEN_WIDTH` and `SCREEN_HEIGHT` before the program is run.

What you're looking for in the program output is that the background image loaded in looks the same as in each screen settings. The buttons may appear funny and out of position when running at a different screen resolution, but we'll fix this later. To exit the program you'll have to press `Ctrl+C` because there's no way to formally exit the game. Later in this tutorial we'll be adding a button that will enable you to quit the game.

Next we'll add code that will position the buttons correctly relative to the screen resolution that's being used. However, before we add this code we could do with amending some of the existing code to prepare for this. Each of the buttons that have been created each have a class of their own (an object oriented programming concept), which you'll see declared in the program as `hitButton`, `standButton`, `dealButton`, `doubleButton`, `betUpButton` and `betDownButton`. Each of these classes contain three calls to the image load function. As an example, let's briefly look at the class `betDownButton`.

```

class betButtonUp(pygame.sprite.Sprite):

    def __init__(self, positionXIn, positionYIn):
    
```

```

pygame.sprite.Sprite.__init__(self)
self.image, self.rect = imageLoad("up.png",
BUTTON_IMAGE)

...

def update(self, mX, mY, bet, funds, click, roundEnd):
    if roundEnd == 1: self.image, self.rect =
imageLoad("up.png", BUTTON_IMAGE)
    else: self.image, self.rect = imageLoad("up-grey.
png", BUTTON_IMAGE)
    
```

You should see something like the following when you view the code in your chosen IDE. Where `BUTTON_IMAGE` is, there should be a number 0, so replace this with `BUTTON_IMAGE` as in the example just shown. Do the same for the other classes described above (`hitButton`, `standButton`, `dealButton`, etc.). After you've done this, save and run the program and you should see that buttons are now to scale (but could be positioned off-screen depending on screen resolution you are using). Even though the scaling issue with the button images is now fixed, the position of the buttons is not, because they'll only appear correct if using the screen resolution 1,280x718. We'll do something about this next.

We're now going to implement a class of our own called `buttonPositioningValues`. The purpose of this class or (data structure) is to position all the buttons relative to the screen resolution that's been set. This means regardless of what screen resolution is used, the buttons will always appear in a constant position.

Near the top of `blackjack.py` after the constant variables have been defined (`SCREEN_WIDTH`, etc.), write out the following code:

```

# UI Button Positioning variables
class buttonPositioningValues:

    # Class Constructor
    def __init__(self):
        self.myvalue = 1

    # Get the screen dimensions
    screenWidth = SCREEN_WIDTH
    screenHeight = SCREEN_HEIGHT

    # Set Deal Button Position
    dealButtonPosition_X = (screenWidth-125)
    dealButtonPosition_Y = (screenHeight-193)

    # Set Hit Button Position
    hitButtonPosition_X = (screenWidth-125)
    hitButtonPosition_Y = (screenHeight-243)

    # Set Stand Button Position
    standButtonPosition_X = (screenWidth-125)
    standButtonPosition_Y = (screenHeight-280)

    # Set Double Button Position
    doubleButtonPosition_X = (screenWidth-125)
    doubleButtonPosition_Y = (screenHeight-317)

    # Set Up Arrow Position
    upArrowButtonPosition_X = (screenWidth-150)
    upArrowButtonPosition_Y = (screenHeight-370)
    
```

Here, the dealer has won a round at Blackjack and that the player has lost and has also lost \$10. Once a round is over, the player can only deal.





We've added a Quit button to the game so the player can leave at any time. This saves the player having to press Ctrl-C to exit the program.

```
# Set Down Arrow Position
downArrowButtonPosition_X = (screenWidth-100)
downArrowButtonPosition_Y = (screenHeight-370)
```

```
# Quit Button Position
quitButtonPosition_X = 100
quitButtonPosition_Y = (screenHeight-100)
```

You'll notice from the code we've just written that the X and Y position values are relative to the screen width and screen height, and are offset by a certain value to position them on the screen. Now that our class has been written out we now need to declare what's known as an instance of a class, so that the properties of the class can be accessed. Before we do this, scroll down to just under where there's a comment that says **INITIALIZATION BEGINS** and type the following code: `uiButtonPosition = buttonPositioningValues()`

The next step from this is to alter the existing class constructors of the buttons used in the game program. The below is example of what you'll need to do to each button class.

```
bbU = betButtonUp(uiButtonPosition,
upArrowButtonPosition_X, uiButtonPosition,
upArrowButtonPosition_Y)
bbD = betButtonDown(uiButtonPosition,
downArrowButtonPosition_X, uiButtonPosition,
downArrowButtonPosition_Y)
```

As can be seen from this code, the class constructors of `betButtonUp` and `betButtonDown` have been altered to accept X and Y positions for the relevant buttons declared in our button-positioning class. Before continuing with the tutorial, do the same for each of the other button classes. The button classes themselves also need to be altered for each button featured in the game program. Below is an example of how it's done for the `dealButton` class:

```
class dealButton
...
def __init__(self, positionXIn, positionYIn):
    pygame.sprite.Sprite.__init__(self)
    self.image, self.rect = imageLoad("deal.png",
    BUTTON_IMAGE)
    self.position = (positionXIn, positionYIn)

# 2. Set the position values as part of class
```

» GAMEPLAY FUNCTIONS

Even though object oriented programming has been implemented in this project, these individual functions have been used to help with the gameplay of *Blackjack*:

- » `imageLoad` for loading an image resource.
- » `soundLoad` used to load a sound resource.
- » `display` this displays text on the screen.
- » `playClick` for playing a sound resource once loaded.
- » `gameOver` used to display the game-over screen.
- » `shuffle` use this to shuffle the deck of cards using the Fisher-Yates algorithm, which helps eliminate the element of predictability.
- » `createDeck` this creates the logical structure of deck of cards.
- » `returnFromDead` when the main deck of cards has been emptied.
- » `deckDeal` used to shuffle the deck.
- » `hit` for taking cards from the deck.
- » `checkValue` this checks the value of the hand of player and dealer.
- » `blackjack` for seeing if blackjack has been achieved either by the player or dealer.
- » `bust` used when player is bust.
- » `endRound` determines what happens with the cards at the end of a round.
- » `compareHands` this is used at the end and beginning of a round to compare hands of player and dealer.

```
self.xPos = positionXIn
self.yPos = positionYIn
```

```
def update(...)
# 3. Pass the position values into the position
self.position = (self.xPos, self.yPos)
```

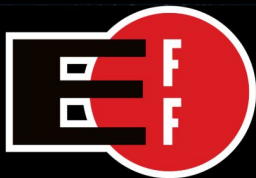
The first step in altering the class is to change the constructor of the class as above to take in `positionXIn` and `positionYIn`. The rest of the changes are identified by comments in the above code example. Before continuing with the tutorial, do the same to the other button classes for the game program.

When you've done all the changes to each class, save your changes and execute the program. You'll see that all the buttons have been positioned relative to the screen resolution. Because there's quite a bit of code to alter, check the python file `lx289ans.py`. The answer source code is written in there to see if your code is consistent with the answer... no cheating though! The text that's displayed in the game program is positioned relative to the button buttons (see `lx289ans.py`).

Finally, we need to add a button to quit the game program rather than just pressing Ctrl+C. The image resource for a quit button has already been created in the image folder location. In the answer Python file, `lx289ans.py`, there's a class created called `quitButton` that's used for the display and operation of the quit button in the game. Going over what we've covered in this tutorial, see if you can work out how it's been written, before looking at `lx289ans.py`. To help you, have a look at the existing button classes that have been written to see if it'll provide you with any clues.

Instead of scrolling through code in an IDE, use the search facilities available to go direction to the function or variable directly. **177**

» **WE'RE OLD AND NEED UPDATING** Subscribe now at <http://bit.ly/LinuxFormat>



The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

EFF.ORG

ELECTRONIC FRONTIER FOUNDATION

Protecting Rights and Promoting Freedom on the Electronic Frontier



PAPER POWER

60% of the energy used to produce paper and paper packaging in Europe comes from renewable sources.

Discover the story of paper

www.lovepaper.org

Source: Confederation of European Paper Industries (CEPI), 2018
CEPI represents 92% of European pulp and paper production

