

RASPBERRY Pi DESKTOP DISTROS

Get a full desktop experience from the single-board system

LINUX FORMAT

The #1 open source mag



Easy keyboard & mouse sharing



Clone your entire drive in minutes!

FAST VIRTUAL MACHINES

Accelerate your system to the next level with the best virtualisation platforms



PLUS: HOW TO

- » Jump on board the Web 3.0 revolution
- » Compose algorithmic music in open source
- » Run the first personal computer: the IBM 5110

CODE GIT TOOLS

Write a custom status checking tool with Qt

RUST PROCESSES

Get to grips with Unix signals and controls

DISTRO INFERNO

How to kickstart your own smash-hit project

LXF August 2022





**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

Nvidia has finally committed to an open source driver stack, but did the company's lack of support put off our contributors from buying its cards?



Jonni Bidwell

I sorta wish the Nvidia open sourcing happened because it was cowing to hacker demands (it really didn't). But open source (plus firmware blobs) or no, the EGLsteams/GBM binary will ensure that people still have something to argue about, graphics-wise. Sooner or later my AMD RX480 will need replacing, but I'll probably stick with Team Red.



Nick Peers

While I welcome Nvidia's announcement, my sole reason for buying its GTX 1660 Super for my PC was its support for the Turing Nvidia encoder. I'd hoped it would turbo-charge my media rips – and it did – but the resulting h264 files were too large to be usable.



Les Pounder

I'm a pragmatist at heart. I prefer Nvidia's cards over AMD and this all stems from an ATI card that I had in a Dell 1737 back in 2009. Linux gaming at that time was a minefield. Now I can play my games on my Kubuntu system running a 3060 Ti.



David Rutland

With the exception of the odd Raspberry Pi, I exclusively buy used hardware, and my purchase decision is based on what's available on eBay local in my hour of need. Yes, I do use Nvidia, and I even use the proprietary Nvidia drivers. I don't feel good about it, though.



Mayank Sharma

I'd like to say that I put down the keyboard, and picked up a pitchfork when Linus raised his finger. But the truth is that I'm too busy to get worked up about a company making a mess of its open source drivers. I take the proprietary drivers, thank them for supporting Linux, and meet my deadlines.

First step...



It seems that Tux can safely visit Hell, because last issue it froze over when Nvidia announced it's starting on the long journey to producing a fully open source driver stack.

It's clear this is going to be a drawn-out affair. Just look at what happened when AMD bought ATI back in 2006. A year on (mid-2007) it announced it would be open sourcing its driver stack, and the SUSE engineering team began work on the Radeon HD driver.

I'm not even going to try and catalogue the saga here, but by 2012 there was the promise of same-day driver support for the HD 8000 range release. It never came, although by this point performance of the open source driver was getting competitive with the closed-source Catalyst driver. It wasn't until nine-years after the initial announcement in 2016 and the release of the AMD Radeon RX 480 GPU that Linux saw day-one support through the AMDGPU kernel driver we still enjoy. The AMD open source driver hasn't looked back, offering better performance than the proprietary one.

It's a classic example of just how long these things can take. While it's evident this has been the result of enterprise-level pressure, we look forward to the day we can use Nvidia hardware with fully featured open source drivers!

It's hardly like we're lacking in open source to play with though, while we're not reviewing graphics cards this issue, we do have a pile of Raspberry Pi OSes for you to try, Jonni's making faster virtual machines, we utilise software KVMs, compose algorithmic music, try out *Clonezilla* and loads more, so keep on enjoying!

Neil

Neil Mohr Editor
neil.mohr@futurenet.com



**Subscribe
& save!**

On digital and print –
see p16

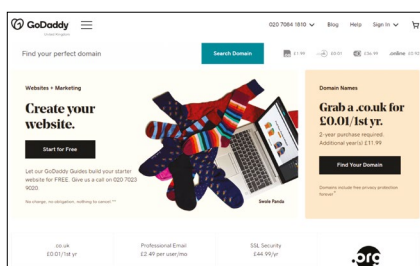
Contents



REVIEWS

GoDaddy19

Purchase and bid on domains with the daddy of all services. **Stefan Ionescu** is rather spoilt for choice, but doesn't have bottomless pockets.



OpenSUSE 15.420

If someone were to compile the definitive list of the mysteries of life, **Mayank Sharma** would ask them to include openSUSE at number 42.

Tails 5.121

This distro is one of those things **Mayank Sharma** would rather have and not need, than need and not have, so he fires up the Debian 11-based privacy distro for a test.

OpenMediaVault 622

Old wine in a new bottle is such a cliché, but that's exactly how **Mayank Sharma** likes his software to grow and evolve, especially when it can save your bacon.

OnlyOffice 7.123

Usually **Mayank Sharma** will run everything that'll fit inside a browser, but today he'll try something he's petrified to launch even as a native program.

The Stanley Parable: Ultra Deluxe24

Management doesn't like self-referential things. It just reminds them they're a simple strapline-based comedy construct courtesy of **Christopher Livingston**.

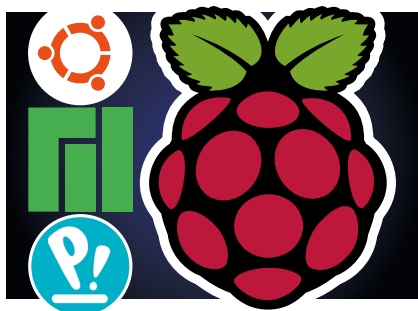


FAST VIRTUAL MACHINES

Jonni Bidwell transcends meat space, ventures virtuously into the valleys of the virtual and returns laden with actual knowledge. Find out more on **page 32!**



ROUNDUP



Raspberry Pi OSes26

The top-end Raspberry Pis can be used to build a formidable desktop. **Michael Reed** finds there's a fair bit of choice when it comes to Linux distros.

IN DEPTH

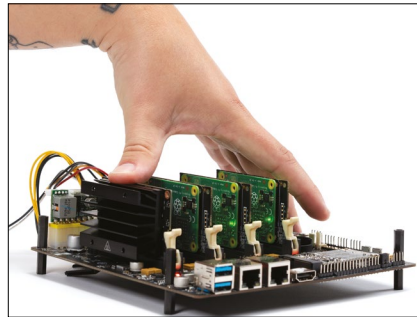


World Web 3.048

Battle lines are being drawn up as technologies for a Web 3.0 are primed and loaded. **Mats Tage Axelsson** helps you prepare for a user-owned world.

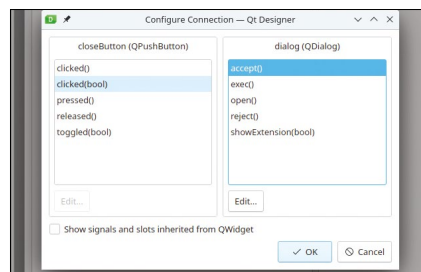
PI USER

- Raspberry Pi news** 40
Introduced by the robot crazy **Kevin McAleer**. The Turing 2 Pi cluster computing Kickstarter gets all the funding, learn about a new sensor-packed RP2040, and the Pi Education Research Centre to launch.
- Waveshare CM4-Nano-B** 41
Les Pounder looks at Waveshare's Compute Module 4 carrier board that offers Raspberry Pi 4 performance in a near Zero-sized package.
- Create a custom LCD menu** 42
Les Pounder loves to get hands-on with a good button, and if doing so can save him time and looks great, then it's a win-win situation!
- Using Systemd and rc.local** 44
Sean Conway is struggling with change – specifically, what system administration knowledge to keep and what to throw away as he explores Systemd in RiP OS.



CODING ACADEMY

- Unix processes and signals in Rust** ... 90
Join **Mihalis Tsoukalos** as he demonstrates how to use Rust to work and manage child processes, and handle Unix signals.
- Updating a GIT monitoring tool** 94
John Schwartzman adds new features to an existing PyQt5 program that finds your *Git* repositories and displays their statuses.



REGULARS AT A GLANCE

- News** 6
What's with all the hate around snaps? Why is everyone loving the Framework laptop? Is that Linux getting installed in Turkey? And is Gnome landing on phones?
- Kernel watch** 10
- Answers** 11
Issues with VM display sizes, creating password-protected network shares, fantasy partition sizes and missing files, configuring your own function keys and shell scripts.
- Mailserver** 14
You're loving Linux from Scratch, but you're not loving Mihalis's screenshot contrast, the lack of terminal help or governments spying on your emails.

- Subscriptions** 16
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
Mayank Sharma is staying cool under the pressure of the mighty *HotPicks* but also the 50°C Indian summer, finding top FOSS such as *Kooha*, *ApplImage Pool*, *Blanket*, *Universal Media Server*, *Beaker Browser*, *superProductivity*, *Safe Eyes*, *Mudlet*, *StackAndConquer*, *Bottles* and *Filmulator*.
- Next month** 98

TUTORIALS

- TERMINAL: Ventoy** 52
Shashank Sharma can't parallel park, but he knows how to copy-paste live distributions into a USB drive to boot 'em up.
- KSNIP: Better screen grabs** 54
Nick Peers reveals how easy it is to take screengrabs and annotate them using Ubuntu's built-in tools and third-party apps.
- EMULATION: IBM 5110** 58
Rounding off our emulation series, **Mike Bedford** revives the first portable computer that ran the bizarre APL language.



- SONIC PI: Algorithmic music** 64
How to create royalty-free music either automatically or with coding. **Mike Bedford** explores both composition options...

- CLONEZILLA: Drive back up** 68
Michael Reed shows you how to tame *Clonezilla* – powerful software that can help copy and restore hard disk partitions.

- LXF SERVER: Bitwarden** 72
David Rutland has learned from past mistakes and endeavours to keep all of his passwords firmly under lock and key.

- BARRIER: Networked KVM sharing** 76
Use *Barrier* to replace a physical KVM switch and enable a keyboard and mouse to be shared between multiple devices. **Matt Holder** explains all.

TOP OF THE FOSS!



- Seeing the light** 78
Three humble developers of Bodhi Linux enlighten **Jonni Bidwell** on the ways of Zen and the art of distro maintenance.

Newsdesk

THIS ISSUE: Vocal Snaps critics » Framework Laptop » Local government adopts open source » Mobile GNOME » Mesa 22.1

CONTAINERS

Ubuntu Snaps under fire

The release of Ubuntu 22.04 has prompted more people to complain about the usefulness of Snaps.

Opinions remain divided over the effectiveness of Snaps, Canonical's sandbox software format. Many Ubuntu users complain about Snaps' poor performance compared with competitors such as Flatpak. While Flatpak, which used to be known as xdg-app, is widely supported by many distros including Debian, Ubuntu has stuck with Snaps, despite it offering similar functionality, naming packaging and being able to distribute software that can be run in a 'sandbox', safely away from important operating system files.

To mark Ubuntu 22.04's launch, Canonical founder Mark Shuttleworth took part in a video Q&A session on the Ubuntu OnAir YouTube channel (which you can see at <https://bit.ly/lxf291ubuntuonair>). During the session Mark was asked if Ubuntu would ever come with Flatpak support out of the box. Perhaps unsurprisingly, he was pretty adamant that it wouldn't, saying that, "Flatpaks wouldn't work for us... I don't think they have the security story and I also don't think they have the ability to deliver the same integrity of execution over time that Snaps have."

This is disappointing news for anyone who was hoping for Flatpak support in Ubuntu. That includes the team behind KeePassXC (<https://keepassxc.org>), an open-source password manager. In a Twitter post (see <https://bit.ly/lxf291keepassxc>) they described Mark's defence of Snaps as "a bad joke," and went on to state that, "There is nothing 'simple' about Snaps and we are moving away from them for exactly that reason."

While Mark did say that Snaps offered simplicity thanks to its integration into Ubuntu, to

his credit he did also explain that, "There definitely are places where we need to improve the Snaps experience on the desktop. Startup performance times seem to be really, really important, so that's something we can focus on." He also was keen to make it easier to securely take an application out of the sandbox.

So, it looks like Snaps isn't going anywhere, much to the chagrin of an increasingly vocal group of people who seem to have taken a big dislike to the format. It's also not a good look that an ex-Canonical employee has made a tool,

The Canonical logo is displayed in white text on a dark purple rectangular background. The word "CANONICAL" is in all caps, with a trademark symbol (TM) to the right.

Canonical has a reputation for sticking with its own programs and services, and Snaps is particularly divisive.

THE SNAPS PROBLEM IN A NUTSHELL

"There definitely are places where we need to improve the Snaps experience on the desktop."

known as *unsnap* (<https://bit.ly/lxf291unsnap>) that makes it easy to port Snap applications over to Flatpaks.

Despite Mark mentioning that "clearly developers like the publication experience," it may not even be that popular within Canonical. His claims that Flatpaks doesn't offer the same level of security protection as Snaps will also raise eyebrows in some quarters. This could be a controversy that runs and runs.

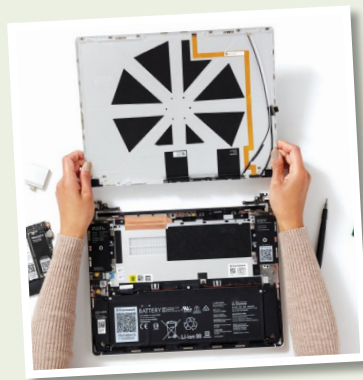
LAPTOPS

Framework Laptop receives 12th gen processor upgrade

Popular modular laptops with Linux support now available with 12th gen Intel processors.

We're big fans of the Framework Laptop. Thanks to the device's modular nature, users can easily swap and fix various parts of these laptops – a frustratingly rare approach these days.

It looks like Framework Laptops are here to stay, because the company has announced that new models are available to pre-order (from <https://bit.ly/lxf291framework>) with the latest 12th generation Intel processors. These offer a significant upgrade in performance and efficiency, with Intel embracing the 'big.LITTLE' approach to CPU design. Such chips make use of both high-performance and lower-power efficiency cores, with the processor switching between the two depending on the computing tasks you're performing. So, there's improved performance when managing heavy workloads, while also improved battery life when you're doing less-intensive work such as browsing the web.



The flagship model now comes with an Intel Core i7-1280P CPU, which has been specially designed for thin and light laptops, and comes with 14 cores and 20 threads. Of course, because previous Framework Laptops are modular, you can also purchase an Upgrade Kit (<https://bit.ly/lxf291upgrade>) that enables you to replace the 11th Gen Intel CPU with the new 12th Gen.

Prices start at £538 for the Core i5-120P, and go up to £1,138 for the aforementioned Core i7-1280P. While these upgrades cost around

the same price as a new laptop, it's great to see the ability to upgrade the laptop's CPU, and bodes well for the future-proofing of these devices.

The Upgrade Kit enables you to add a 12th gen Intel Core CPU to your Framework Laptop.

OPINION

DECISIONS, DECISIONS



Matt Yonkovit is the head of open source strategy at Percona

“When you have two options to decide between, your choice should be easier. For us in the world of open source, the decision between closed and open software has been an easy one for years.

But choices in the real world tend to be affected by multiple factors. This can make it hard to know how to move forward. The challenge here is how to make the decision easy again, which can involve looking at issues in a new way. Today, the problem is less about open versus closed software, and more about easy versus hard for the developers that have to pick these projects.

For open source to stay relevant, it has to move beyond being open from a technical perspective, and focus on being open to use, too. This means making open source projects just as easy to deploy, run and use as those cloud-based options. At the same time, open source has to remain open to everyone, rather than closing off options and moving to less-permissive or closed licenses.

To achieve this will mean looking at more automation. Replicating cloud's ease of use and developer experience while staying true to open principles will take time, but it's the future for the open source movement. ”

GOVERNMENT

Turkish district reaps Linux rewards

Local government saves \$1m by going open source.

We always like to hear success stories of how governments and public bodies have switched to open-source alternatives that bring both accessibility and financial rewards, and Turkey is home to one of the most impressive recent examples.

As the Fosspost website reports (<https://bit.ly/lxf291turkey>), the large Eyüpsultan municipality to the west of Istanbul has been migrating software used by its employees to open source since 2015. This includes running the Turkish Linux distro Pardus, which is based on Debian 11 (you can download it from <https://bit.ly/lxf291pardus>), as well as LibreOffice, VLC and GIMP.

Software licences in Turkey are expensive and so moving to free and open source

alternatives has helped to significantly lower administration costs that are ultimately met by the taxpayer. The lightweight Pardus distro also means that ageing hardware doesn't need to be replaced as regularly, and that has enabled the municipality to save a very impressive \$1 million.

The migration project isn't stopping any time soon, either, with the team behind it looking to move from *Microsoft Active Directory* to an open-source alternative such as *Samba*. While the migration wasn't without its obstacles, particularly around how *LibreOffice* handled Microsoft's proprietary .doc and .docx formats, these impressive savings are great to see, and hopefully they'll be noticed by other governments and institutions, which could tempt them to make the leap to open source.

OPINION

BE A GIT



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

Wikipedia lists 14 purely Open Source version control systems, but there's one that stands head and shoulders above the rest: *Git*. Developed initially by Linus Torvalds and used to manage the Linux kernel source for the past 17 years, *Git* achieves that elusive goal of being simple to use in everyday cases and powerful enough for just about any situation that you can think of.

If you don't use a version control system at all, then I'd suggest you consider doing so. If all you're maintaining is a few scripts, it'll be easy to get started. Just the 'diff' facility alone makes life easier, and once you're using that you'll quickly find other advantages that you can make use of.

If you're using something other than *Git* then consider changing. Change can be painful, but in this case it may help to bear in mind that *Git* can achieve pretty much anything version-control related that you can think of – and, as I've recently discovered, quite a few things I didn't realise I wanted.

There are plenty of *Git* tutorials online, but if you're not sure where to start then search online for "the git pro book". A complete book online and, like *Git*, completely free. What's not to like?

DEVICES

Convergence lives on with Gnome Shell

The dream of a Linux desktop for mobile devices isn't dead in the water.



This concept shows how GNOME 40 could look on a touchscreen mobile device.

Nearly five years have passed since Canonical ditched its Convergence dreams and axed plans to bring Ubuntu to smartphones, but a promising project looks like it could finally bring a Linux distro to mobile devices with an appropriate interface.

On the Gnome Shell & Mutter blog (<https://bit.ly/lxf291gnomeblog>), developer Jonas Dressler explained how the Gnome 40 design team has been working on several experimental ideas, some of which explore how feasible it would be to fully port the Gnome Shell to a smartphone or tablet. Thanks to numerous new features in Gnome 40, Jonas argues that there's

some decent groundwork laid already, such as customisable application grids, and support for touchscreen gestures that are similar to those used on mobile devices.

GNOME developers are also looking to improve notifications and the onscreen keyboard (among other things), which could make GNOME even more suited for smartphones and tablets. With support from the German Ministry of Education's Prototype Fund (<https://bit.ly/lxf291prototypefund>), a team of developers are working on bringing Gnome Shell to mobile devices. There's a long road ahead, but it's certainly an exciting prospect.

GRAPHICS

Mesa 22.1 brings better ray tracing

Open-source graphics driver stack receives a big update.

Mesa 22.1, the latest version of the open-source OpenGL/Vulkan graphics driver, is now available to download and it comes with some exciting improvements.

The headline feature is improved ray tracing support for AMD GPUs, which brings realistic lighting and atmospheric effects. It's the hot new thing in PC game graphics, so it's good to see it receive continued support for Linux. RADV, the Vulkan community driver, also gains Dynamic Variable Rate Shading (VRS) support, a nifty power-saving feature that's used by the Steam Deck handheld console and will hopefully extend the battery lives of laptops.

This release also helps pave the way for Intel's Arc Alchemist graphics cards, which is the company's first foray into consumer GPUs to take on AMD and Nvidia. There's also OpenGL support for Nouveau (the open-source alternative to Nvidia's graphics drivers) and various other improvements. Check out the release announcement at <https://bit.ly/lxf291mesa> for more information.

FUNDING

Thunderbird donations increase

Revenue of the email client rose by 21 per cent in 2021.

Thunderbird, the open-source email client from Mozilla, saw its revenues rise in 2021 to a healthy \$2.79m, compared to \$2.3m in 2020. With the revenue almost entirely driven by donations, it means they rose by \$500,000 to a new all-time high. Considering the tough financial situation that many organisations and individuals are facing, this is encouraging to see.

The rise in donations is partly attributed to a newly focused donation appeal, with the *Thunderbird* team also making sure to engage with donors more directly. These funds will mainly go towards paying the wages of the 18 employees working on the project, who are gearing up for a major *Thunderbird* v.102 release later this year.



Thunderbird may not be as popular as Firefox, but its finances are certainly healthy.

Distro watch

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

LINUX LITE 6.0

This simple-to-use Ubuntu-based distro has a new version available, and it's looking to encourage Windows users to make the switch by offering accessibility options that rival Microsoft's admittedly very good offerings. New features include a screen reader, desktop magnifier and a virtual keyboard that aim to make the distro as easy to use as possible, no matter who's using it. There's also a new Materia windows theme, and Google's *Chrome* is now the default web browser. Find out more at <https://bit.ly/lxf291linuxlite>.



Linux Lite 6.0 brings some welcome accessibility features.

NIXOS 22.05

NixOS, a distro that's built around the Nix package manager, has been updated with a range of general fixes and improvements. As the release announcement that can be read at <https://bit.ly/lxf291nixos> points out, a highlight of this new release is "experimental flakes support for the default installed Nix version" and a new graphical installer that should make setting up the distro much easier for new users.



NixOS now comes with an improved graphical installer that should make initial setup much easier.

LXLE FOCAL

Another lightweight distro, has been updated, with LXLE now based on Ubuntu 20.04 LTS, and comes with the LXDE desktop. This release continues that careful balancing act of providing plenty of fully featured applications, while keeping the distro as a whole as lightweight as possible. Various programs have either been updated or replaced with alternatives, and you can find out more at <https://bit.ly/lxf291lxle>.



LXLE continues to offer plenty of tools and features, while remaining impressively lightweight as well.

ALMALINUX OS 9.0

AlmaLinux OS 9.0 is now out, with the 5.14 Linux kernel and support for Intel/AMD (x86_64), ARM64 (aarch64), IBM PowerPC (ppc64le) and IBM Z (s390x) CPU architectures. Security and compliance features have also been boosted, and SELinux performance is improved as well. Python 3.9 is included, and the latest versions of compilers such as LLVM, Rust and Go have also been added, which the release announcement (<https://bit.ly/lxf291alma>) claims makes "modernizing the applications faster and easier."



AlmaLinux OS is a clone of Red Hat Enterprise Linux.

OPINION

SPOTLIGHT ON MESON



Xavier Claessens
is a senior software engineer at Collabora

When developing an application or a library, it's common to want to run it without installing it, or to install it into a custom prefix rather than on the system.

Meson has always helped with that by setting `rpath` in the built executables so they can find their shared libraries within the build directory tree. However, it has some limitations: Windows doesn't support `rpath`, DLLs are searched in `PATH`, and applications often need to find other resources: plugins, config files and so on.

Many applications solve that with custom shell scripts that set various environment variables, like `GST_PLUGIN_PATH`, `GI_TYPELIB_PATH`, etc...

Since Meson 0.58.0, this can be done by using the `meson devenv -C <builddir>` command, and adding the needed environment variables in your meson build files using `meson.add_devenv()`. Doing this directly in Meson instead of relying on custom shell scripts has two advantages. First, it's generic, so users don't need to know which custom script to use for each project. Second, Meson already knows most of the needed environment and sets it for you. Another example of how the Meson build system is improving the lives of developers!

OPINION

MULTIMESS



Jon Masters is a kernel hacker who's been involved with Linux for more than 22 years, and works on energy-efficient Arm servers.

Linux 5.19 will mark the passing of 12 years since the Arm “multiplatform” effort began with the merging of the final few fixes required to complete it.

In 2011 Linus criticised Arm for being a fragmented mess, with many vendor SoCs (System-on-Chips) either not having upstream support, or code being in the kernel in such a manner that it had to be compiled to target a specific platform. It wasn't possible to have a generic distro kernel that could run across many different devices as with x86.

To address the situation, Arm created Linaro, a cross-vendor industry consortium. One of the big projects Linaro began was support for multiplatform kernels.

The Arm systems of a dozen years ago were ARMv4T, ARMv5, or even ARMv7. The 64-bit (Armv8) architecture wasn't even released, although by the time it was things were much smoother for 64-bit systems. But people still run legacy hardware, and the ability to have true common kernel images remains as important today as it was back then.

Arm would never willingly get into such a situation today, yet another emerging architecture (RISC-V) faces the very real risk that it might. It's important that lessons are learned and heeded.

Kernel Watch

Jon Masters summarises the latest happenings in the Linux kernel, so that you don't have to.

Linus Torvalds announced the release of Linux 5.18, noting that there had been no “unexpected surprises” at the tail end of the development cycle. The new kernel includes support for Intel's Indirect Branch Tracking (IBT), improved scheduling performance on AMD Zen processors, and switches to a more “modern” C language standard (C11) as a baseline (previously C89!). **KernelNewbies.org** contains the usual great detailed breakdown.

IBT comes to x86

Among the many other features landing in 5.18, comes support for Intel's Indirect Branch Tracking feature that adds landing pads to the x86 architecture. The new ENDBR instruction can be inserted into code, particularly at the entry to functions that are intended to be called from other code. The new instruction on its own doesn't do anything, but on suitably enabled hardware platforms it'll prevent a function from being called unless it begins with such an instruction. If it doesn't, this could indicate an attempted attack to hijack program flow.

Of course enabling such a feature that's available on Arm systems requires a software lift. Code needs to be recompiled to add these instructions, and for the moment that includes only kernel code beginning with

Tiger Lake (2020). One additional and perhaps counterintuitive feature of IBT is that it can be used to prevent proprietary kernel modules from calling into kernel functions that aren't exported.

With the release of 5.18 came the opening of the merge window for what will be Linux 5.19. And after the requisite two weeks of merging, Linus announced Linux 5.19-rc1, and noted two things. First that “pretty much all of the pull requests were signed tags. I still don't technically require signatures for pulls from **kernel.org**, but I've been (not very subtly) encouraging people to use them.” Second, that this was the first merge window where Andrew Morton, maintainer of the memory management code, “participated all through *Git*” (as opposed to quilt-based patch series).

Linux 5.19 will include support for the China-driven LoongArch, a “new RISC ISA, which is a bit like MIPS or RISC-V. LoongArch includes a reduced 32-bit version (LA32R), a standard 32-bit version (LA32S) and a 64-bit version (LA64).” Reduced 32-bit version refers to a compressed form of the instruction set designed to fit in memory (similar to Arm THUMB).

Like other RISC ISA, it has 32 GPRs (registers), a hard-wired zero register and appears to support conditional flags, atomic instructions, as well as a fairly well-formed set of supporting apparatus. There's more details on LoongArch at <https://github.com/loongson>.

» ONGOING DEVELOPMENT

A potential security hole was discovered in which someone with local console access (including a remote console available over the network) “could trigger the debugger”. The kernel debugger can be used to read/write to arbitrary memory, which isn't usually a problem. But it can become a big problem if Linux is in “lockdown” mode due to Secure Boot. In such a situation, the kernel isn't supposed to be able to run unsigned code, but if the kernel memory can be arbitrarily written then this is effectively the end result. Patches were issued to limit debugger behaviour under lockdown.

Speaking of security, the “security-bugs” documentation describing the process for reporting and subsequent incident response

by the kernel community was overhauled. It aims to be easier to read for those who need to report problems. In particular, this document covers which lists handle creating and preparing patches in collaboration with vendors, compare to how distros coordinate their release.

Work continues to prepare for the upcoming Linux Plumbers conference, including many different microconferences. Among those are RISC-V and CXL (Compute Express Link), a new standard built in some aspects upon PCI and used to attach disaggregated memory and coherent accelerators. CXL patches have been landing in Linux, most recently including support for native CXL hotplug. **LXF**

Answers



Neil Bothwick
gives your Tux a total tune up to keep it ticking along.

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

Q Small screen VM

I've been running Ubuntu and Linux Mint happily in *VirtualBox* for years. A couple of weeks ago, I purchased a PowerSpec G902 gaming PC. I haven't been able to run Ubuntu 20.04 LTS full screen within Hyper-V. The choice of 2560x1440 screen resolution is never an option. I'm at my wits' end. I don't know if it has anything to do with the fact that this PC has a dedicated GPU (Nvidia GeForce RTX 3080), not an APU?

Henry Adams

A This may simply be a case of not allocating enough video memory to handle this size of display. Go into the Display section of the settings and make sure the Graphics Controller is set to VMSVGA, then crank up the Video Memory setting to at least 64MB, although playing safe with 128MB while trying to get it to work would be better. In addition, disable 3D acceleration – at least while getting things to work – because this has been known to cause problems.

If that doesn't work, you may need to set up the mode within the guest OS using *xrandr*, which is the command line program that manages displays. First use *cvt* to generate a suitable modeline:

```
$ cvt 2560 1440
```

This will give something like:

```
# 2560x1440 59.96 Hz (CVT 3.69M9)
hsync: 89.52 kHz; pclk: 312.25 MHz
Modeline "2560x1440_60.00" 312.25
2560 2752 3024 3488 1440 1443 1448 1493
-hsync +vsync
```

Plug your modeline data into *xrandr*:

```
$ xrandr --newmode "2560x1440_60.00"
312.25 2560 2752 3024 3488 1440 1443
1448 1493 -hsync +vsync
$ xrandr --addmode VGA-1
"2560x1440_60.00"
$ xrandr --output VGA-1 --mode
2560x1440_60.00
```

The mode in the second and third lines is the first item in your Modeline output. You may need to change VGA-1 to whatever your display is identified as, you can see this in the output from the command:

```
$ xrandr --query
```

Once you have this working by running the commands manually, put them in a short shell script and add it to the desktop's autostart list.

Q Stealth files

I have a root partition that *gparted* reports is 228GB with 1.5GB free. Using the *Disk Usage* tool in Linux Mint 20.3, the files total only 118.8GB, which I believe to be the true size.

Harrison Davis

A There's a distinction between partitions and the filesystems that are on them. If you enlarged the partition, the filesystem on it remains the same size. You don't gain the extra size until you've also resized the filesystem. You can check how full the filesystem is with *df*:

```
$ df -Th
```

If the size of the filesystem reported by *df* doesn't match the partition's size, you need to resize it. If the filesystem really is full, it may be that you've copied files to directories that are used as mount points. Once you mount another filesystem on a directory, the previous contents of that directory are invisible, but still occupy space. Let's say you plug in a USB stick. It's mounted at */media/usbstick*, but then it's unmounted and you still copy files to */media/usbstick*. When you plug in the stick again, those files will be hidden.

You can't go unmounting everything just to check on disk usage, but there is another way. Mount your root partition somewhere else, as well as the original mount. This is called bind mounting. For example:

```
$ sudo mkdir -p /tmp/root
```

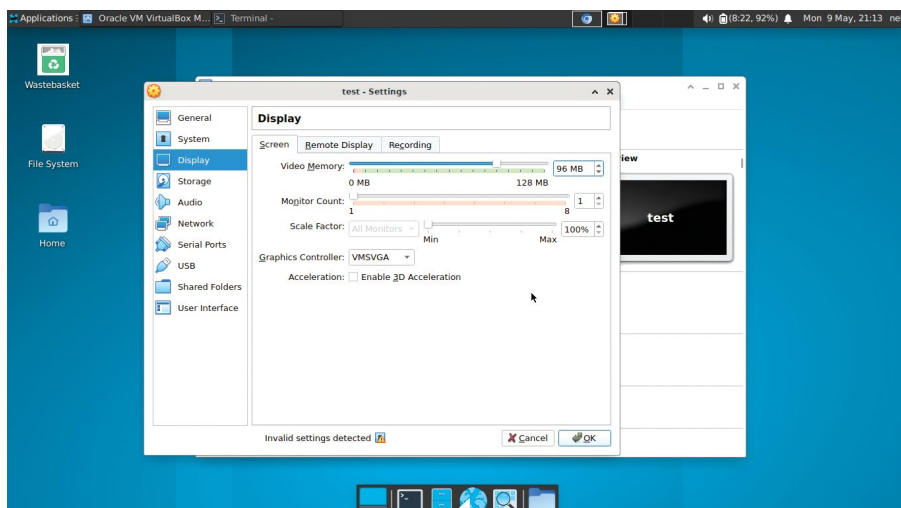
```
$ sudo mount --bind / /tmp/root
```

Now you can look at */tmp/root* and you'll see the same files as in */*, but you'll see the true contents of mount points, so places like *dev*, *proc* and *sys* should be more or less empty with most distros.

Compare the usage figures for */* and */tmp/root* to find rogue files. These usually occur because of copies to the mount point of a removable device or network share that wasn't mounted at the time. This can also happen if you've switched to a separate partition for */home*, but still have the old home contents present. You can use the bind mount access these files to delete them. If you have doubts about a file, move it rather than delete it until you're sure it's not needed. You should be fine to remove files under */mnt* or */media* because no system files are kept here.

Q Inverted keys

I recently bought a cheap netbook-style laptop. For the price, I'm really happy with it, but for one thing. As with



Large screens require plenty of video memory, so make sure that you allocate enough resources to a virtual machine if you want a high-resolution display such as 2560x1440.

many laptops, the F-keys double up as media control keys, using the Fn key to switch between. However, this laptop works the opposite way round to others I've used. I have to hold down the Fn key to access the F-keys, otherwise I get the media key's effect. This makes pressing, for example, Ctrl+F2 to switch desktops a cumbersome three fingered action. Is there a way I can switch this behaviour back to the arrangement I'm used to?

David Coates

A There's an easy way and a less easy way to do this. First check your BIOS menu (or the UEFI if you prefer). There may be an option to control the default action of the Fn key. If so, you're in luck, job done. Otherwise, read on.

The `xmodmap` command can remap the actions of keys and buttons to suit your needs. It's most commonly used to remap mouse buttons for left-handers or swap Ctrl and Caps Lock keys on awkwardly laid-out keyboards, but it can be used to alter mapping of all keys. The first step is to get a listing of the current layout with

```
$ xmodmap > ~/.Xmodmap
```

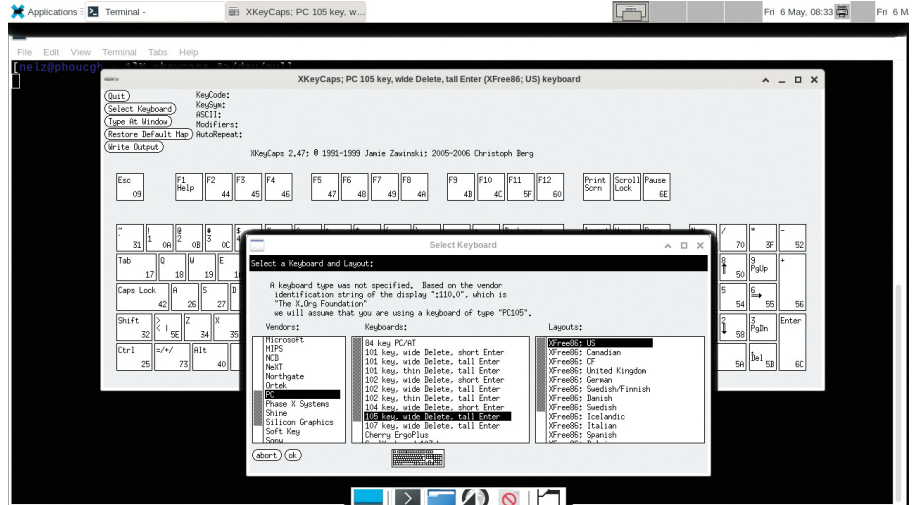
This generates the current configuration that we can then modify. To see the keycodes used by the keys, run:

```
$ xev -event keyboard
```

in a terminal, then press some keys – you'll see the details of each key as seen by X. Now load `~/.Xmodmap` into your favourite text editor and look for the lines you need. For example, on a netbook F1 is the mute key and in the file are:

```
keycode 67 = F1 F1 F1 F1 F1
XF86Switch_VT_1
keycode 121 = XF86AudioMute NoSymbol
XF86AudioMute
```

The first is from Fn+F1 while the second is F1 alone. The items after the = are what are returned with various modifier keys



I Take a trip back in time to edit your keypad. XKeyCaps is as old as it looks, but it does the job.

pressed, the first with no modifier. All we need to do is swap these definitions, change the 67 to 121 and vice versa. Repeat this process for the other F-keys. Save it and then apply it with:

```
$ xmodmap ~/.Xmodmap
```

You should now find the function keys work as you desire. All we need to do is make this change permanent. Try rebooting and see if your desktop picks up the `.Xmodmap` file. If it doesn't, add this line to `~/.xinitrc` (create the file if it doesn't exist):

```
[[ -f ~/.Xmodmap ]] && xmodmap ~/.Xmodmap
```

This checks for the existence of the `.Xmodmap` file and loads it if found. There's a graphical editor for `xmodmap`, called `xkeycaps` (www.jwz.org/xkeycaps), but it's old and editing a text file is easy enough.

Location, mlocation, plocation

I have a desktop computer running Ubuntu 22.04 LTS 64bit. I've tried to

use the `locate` command but it wasn't found. The terminal suggested installing `plocate`, but elsewhere the advice is to install `mlocate`. This is so confusing – which one should I install?

Lauren James

A The original GNU `locate` command is rarely used nowadays. Most distributions use `mlocate` instead. It has faster indexing of files but works the same as the original from a user perspective: you use `locate` to find files and `updatedb` to update the database. The latter is generally run automatically from a `cron` job or `systemd` timer. Now we also have `plocate`, which uses a different method of indexing that produces smaller and faster index files. `mlocate` mainly differed from `locate` in the speed in which it generated the index, though it's a more efficient method, rather than a different format.

Because `plocate` has different command name for the `locate` function (`plocate`) but the same `updatedb` command to build

» A QUICK REFERENCE TO: SHELL SCRIPTS

You'll often see the phrase "put it in a shell script" when instructions contain a series of terminal commands, but what is a shell script? At its simplest, it's a collection of commands that are executed in sequence that you put in a file, like this:

```
#!/bin/sh
first command
second command
...
```

The first line tells the system what command to use to run the script, in this case the standard `sh` shell, or you could use `bash`. Save this as a file. It's a

convention to use a `.sh` extension, but this isn't required. Set the executable bit to run it like any other program.

```
$ chmod +x myscript.sh
$ ./myscript.sh
```

The leading `./` is needed because the script is in the current directory and not in your command path. If it's something you'll use often, copy it to a directory in your path, either `/usr/local/bin` or `~/bin`. Don't use `/usr/bin`. This should be reserved for commands installed by the OS or its package manager.

It's possible to achieve more than just running a sequence of commands.

Loop and conditional command are well supported. For example:

```
for FILE in *.jpg; do
  convert $FILE ${FILE}/.png
done
```

This loops over all the `.jpg` files in the current directory and converts them to PNGs. Scripts like this can save an awful of time when you repeat an operation. As Douglas Adams said: "I am rarely happier than when spending an entire day programming my computer to perform automatically a task that would otherwise take me a good 10 seconds to do by hand."

the database, it's not possible to have both installed for a side-by-side comparison. You need to uninstall one to try the other. The good news is that as both run an `updatedb` command, so you don't need to change your `cron` setup. The other file used by both programs is `/etc/updatedb.conf`, which controls what to index, or not index. Both programs use the same format for this file, so switching between the two keeps the same settings. Although both programs use an `updatedb` command to maintain the database of indexed files, they store that database in different locations, separate directories in `/var/lib`, so trying one doesn't break the other.

That just leaves the problem of your muscle memory insisting on typing `locate` when you should use `plocate`, which can be solved by adding an alias to your profile: `alias locate="plocate"`

Q Hiding in plain sight
I need to create a simple (not encrypted) password protected folder not shared on LAN. I'm using Linux Mint.
Bob

A The only way to password protect a folder without encrypting it is to make it readable by only your user, so it's protected by your login, but once you're logged in, the folder is open to read. Also, anyone booting your computer from a live distro would have root access and be able to read all of your files. So you really do need to encrypt the folder to protect its contents. There are a couple of ways to do this. One is to use the kernel's `ecryptfs` filesystem. You can do this manually, but if you only want a single encrypted directory in your home, Ubuntu provide tools that make this easier. Because Linux Mint is derived from Ubuntu, you get the same tools there – just make sure the `ecryptfs-utils` package is installed. Run this command in a terminal:

```
$ encryptfs-setup-private --nopwcheck --noautomount
```

and follow the instructions. Give a passphrase when asked for one, but just hit Enter when asked for a mount passphrase and let the system generate a random one. This creates an encrypted directory at `~/Private`. Mount it with:

```
$ encryptfs-mount-private
```

to mount it at `~/Private` after

giving the passphrase. Any files you write to `~/Private` will be readable there, but are stored on your disk at `~/Private` in encrypted form. Run:

```
$ encryptfs-umount-private
```

to unmount it. Now all that can be seen are the encrypted files. This approach requires you to mount the directory when needed, the most secure way. Alternatively, you can have `~/Private` mounted automatically when you log in by omitting the `--nopwcheck` and `--noautomount` options when setting up and giving your login password as the passphrase. This does mean your private files will be visible whenever you're logged in, but won't be readable at any other time, even by root.

There's also a graphical option called *VeraCrypt*, which is a fork of the old *TrueCrypt* project. This is a graphical program that creates encrypted containers to store sensitive files. It's easy to use, but the chequered history of the program means we wouldn't want to keep the only copies of important data in one of their containers. *Ecryptfs* is in the kernel and won't be going away like *TrueCrypt* did. Having said that, it has some good features and is easy to use. It's especially handy for encrypting removable drives as there are versions of it for Windows and MacOS, too.

Q Wakey, wakey
I have a computer running Ubuntu 20.04 LTS that I access with SSH. I can suspend it by running:

```
$ sudo systemctl suspend
```

I need to know how I can wake it up again over SSH.

Ryan Dean

A The short answer is that you can't, because when the computer is suspended no software is running, including the SSH daemon or even the network stack it needs. However, there's a way to do what you want, provided the computer is using a wired Ethernet connection – this isn't possible with a wireless (*are you sure?–ED*) connection



An alternative to setting up a folder with `ecryptfs`, you can use *VeraCrypt* to create encrypted containers that can be read on Windows and MacOS, too.

because the wireless is suspended. The answer is a protocol called Wake-on-LAN (often shortened to WoL). This is a pure hardware implementation so it works with computers that are in suspended or hibernation modes. You should first check your BIOS menu to ensure WoL is enabled (this isn't always the case by default). Then you need the hardware MAC address of the remote computer. You can find this with the `ifconfig` or `ip` command, depending either on which is installed or your personal preference:

```
$ ifconfig eth0 | grep ether
```

```
$ ip link list eth0 | grep ether
```

Replace `eth0` with your network device. The MAC address is six hexadecimal numbers separated by colons, for example:

```
$ ip link list eth0 | grep ether
```

```
link/ether 1c:1b:0d:00:1f:b9
```

Install the WoL package on the computer you'll be using to do the waking up, and all you need to do is run the `wol` command with the MAC address:

```
$ wol 1c:1b:0d:00:1f:b9
```

You don't want to have to type in the MAC address each time, let alone remember, so write it to a file, say `hostname.wol` and run wake up with:

```
$ wol -f hostname.wol
```

If you're concerned about someone waking up this host without your knowing, bearing in mind they'll need access to your LAN in the first place. It may be possible to require a password to wake up, but this depends on the hardware of the computer you're waking up. You can read more about the SecureON password feature in the WoL man page. **LXF**

GET HELP NOW!

We'd love to try and answer any questions you send to lxf.answers@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
```


Mailserver

WRITE TO US

Do you have a burning Linux-related issue that you want to discuss? Write to us at [Linux Format](mailto:linuxformat@futurenet.com), Future Publishing, Quay House, The Ambury, Bath, BA1 1UA or email linuxformat@futurenet.com.

From Scratch

What a great article! It inspired me to have a go at it. I have been a Linux enthusiast since the late 1990s and I have used Gentoo as my main distro for well over 20 years. As you can see I'm familiar with the command line compiling stuff, and you might say I must be a bit of a masochistic to continue to enjoy it still at the age of 75!

Anyway I was VERY impressed with the *Linux From Scratch!* book. It is informative and the instructions are clear. I had the basic system up and running from scratch in about eight or nine hours, and am pretty chuffed to say that it booted with a working network at the first attempt. I'm now moving on through BLFS with a plan to get a full desktop system up and running very soon.

I just love the magazine and have done for many years. There are always interesting things to read about, and it does keep me up with progress.

Francis, Ireland

Neil says...

Thanks for the kind feedback, glad to keep on feeding the entertaining and informative articles. Do let us know what we should be covering too!

You're booked

I'm using the Linux Mint bundled *xReader* and it opens many types of files so from that point of view

I'm happy. I have noted, though, that for PDFs it takes more and more RAM with browsing through the document (displaying each new page). I've tried installing *Evince* and it doesn't do that. The issue here is that when I try to open EPUBs, which I need to do a lot, *Evince* doesn't open them and *xReader* uses 100 per cent CPU (one core and more) for just displaying EPUB. What ebook readers enable you to read books without taking more and more RAM or CPU resources?

Susan Wright, email

Neil says...

Our go-to option ebook reader is usually *Calibre*. It's pretty lightweight just as a viewer. You can open it from your file manager without going into *Calibre*. On the system here it takes about 240MB while opening a pretty big PDF or epub file.

A reader we've not tried is *Foliate*. It's available as a .deb from <https://github.com/johnfactotum/foilate/releases>. It looks like a decent reader and enables you to set a lot of options

Option options

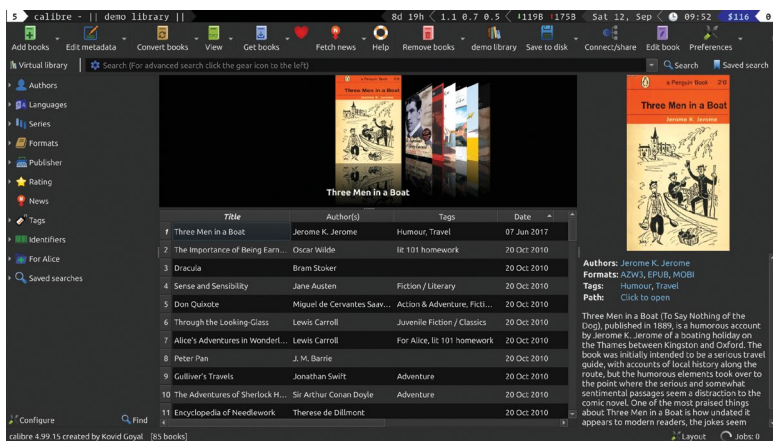
How do I discover the available flags for commands and programs? For instance, when launching *vscode* I can set `--enable-features=UseOzonePlatform --ozone-platform=wayland` and that's all fine and dandy.

But what I can't figure out is where do I go or what do I type to figure out what options are available for me to pass along with my command? In the case of *vscode*, there's no man page I could call from the shell. But for something like the environment variables, calling the man page (for *env*) only lists command options and nothing nearly as esoteric as scaling options.

Ian Smith, email

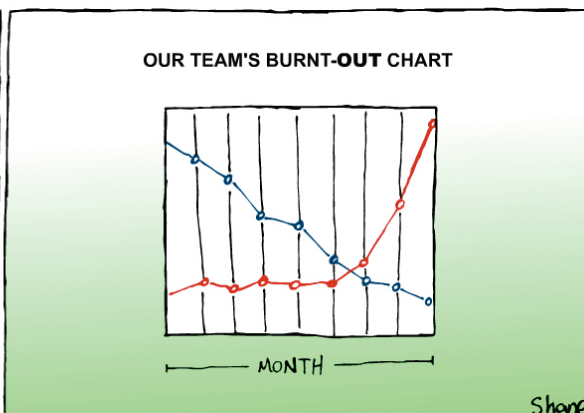
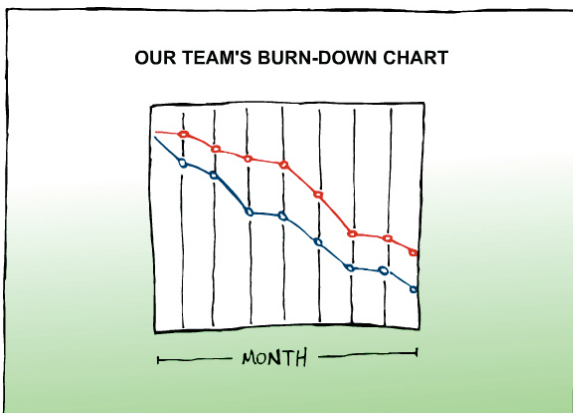
Neil says...

The developer doesn't have to provide any form of documentation or method of discovering command switches. The standard is that typing `--help` should return a list of the available options, but only if the developer documents them. Otherwise their use would be lost in the source code, which should be open for your perusal, though that's hardly convenient!



Now not attempting to support Python 2.0 all on its own, Calibre is an excellent ebook manager.

Helpdex



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

In contrast

This is not so much a burning Linux question as a comment about the readability in the screenshots in Mihalis Tsoukalos’s articles, both those in LXF288 and in many earlier tutorials. The dark colours on a black background are hardly readable (at least to someone of my age!) the contrast between the background and text is too small. I know I can download the code but it’s not always convenient when I read the articles. I assume that the screenshots are put in the articles for the reader to read and as far as I am concerned they’re not fulfilling that purpose and hence spoil some good and interesting articles.

John Huckle

Neil says...

Apologies all round on this one, I think the advent of Dark Themes and 4K monitors, while amazing for personal productivity, are something of a nightmare for us ancient dead-tree publishers. I’ve asked Mihalis to “cut it out lad” while we try and keep in mind that certain colour-blind issues can make white text out of coloured backgrounds hard to read too. **LXF**

```
code cat errorHandling.rs
fn main() {
    let a: Option<i8> = None;
    // This is not going to panic because we use println!()
    match a {
        Some(v) => println!("x: {}", v),
        None => println!("Error but continue!"),
    }
    let b: Result<i8, &'static str> = Ok(123);
    // This might panic depending on the value of b
    // In this case, it is not going to panic
    match b {
        Ok(v) => println!("x: {}", v),
        Err(e) => panic!("Error: {}!", e),
    }
    // This is going to panic because a = None.
    match a {
        Some(v) => println!("x: {}", v),
        None => panic!("Error!"),
    }
    println!("Ping!");
}
code rustc errorHandling.rs
code ./errorHandling
Error but continue!
x: 123
thread 'main' panicked at 'Error!', errorHandling.rs:18:17
note: run with 'RUST_BACKTRACE=1' environment variable to display a backtrace
code export RUST_BACKTRACE=1; ./errorHandling
Error but continue!
x: 123
thread 'main' panicked at 'Error!', errorHandling.rs:18:17
stack backtrace:
```

Having trouble reading this? You're not the only one!

» LETTER OF THE MONTH

What rights?

I believe that email should be treated just like snail mail. If the government can show a good reason why they need to look at emails, they should have to go to a court and get a warrant to look. Random scanning of emails is reminiscent of 1984.

Social media like Twitter or Facebook is a different situation. From a US perspective, when our Constitution and Bill of Rights was created, anonymous opinions were difficult or impossible to make. If you have a strong opinion, you should have the guts to stand up and acknowledge it and not hide behind a pseudonym. Our Bill of Rights guarantees the freedom of speech. It does not guarantee the freedom of anonymous speech.

All social media should be required to have the names and addresses of all their contributors on file and accessible. That would cut down on the outrageous and false claims being made.

On another note: I'm glad to see that I can subscribe to a Kindle edition of *Linux Format*. I'm getting to the point of having trouble with the small print, particularly with pastel backgrounds.

Bob Ewart, Florida, USA

Neil says...

Random scanning of emails is the least of your worries. The entire contents of the trans-Atlantic ocean fibre optic backbone (as one example) is spliced off and copied as part of the Tempora project alongside Prism, Muscular and other cyber surveillance programmes run by the Five Eyes, and yes it's entirely 1984, though I suspect the Powers That Be would say it's all justifiable for state security etc etc.

As for online anonymity that's a thorny subject. Many say killing online anonymity kills free speech, especially in repressed regimes, but also it wouldn't stop rogue anonymous services springing up or people spoofing accounts. And as a curve ball, who's to say any Bill of Rights is sound in the first place?

And we do try and avoid pastel colours but some slip through. It's true the Kindle zoom function tends to be better than the human eye version...



It even has an official logo, which we're allowed to copy as it's public domain because it was made by the US government!



shane_collinge@yahoo.com

SUBSCRIBE Save money today!

SUBSCRIBE

to *Linux Format* and get your

PowerKick

wireless charger

**YOUR
GIFT!**

**WORTH
£50**

Don't miss out,
subscribe now!



Product features

- » The PowerKick wireless power bank delivers functionality and convenience
- » Charge your devices wirelessly or via cables (either USB-A or USB-C)
- » Built-in kickstand enables you to keep your device in landscape or portrait orientation while it's charging
- » Suction cups secure the PowerKick to your phone for consistent charging
- » 10,000mAh lithium polymer battery
- » Fast charging capability

SUBSCRIBE NOW!

www.magazinesdirect.com/lin/a45k

Call **0330 333 1113** and quote **A45K**

Save money today! **SUBSCRIBE**

1) Only available to www.magazinesdirect.com subscribers

» **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

OUTSIDE THE UK?
Turn to page 63 for more great subscriber deals!

» CHOOSE YOUR PACKAGE!

SIX-MONTHLY PRINT EDITION



Only £33.75

Six months of *Linux Format* in print by Direct Debit

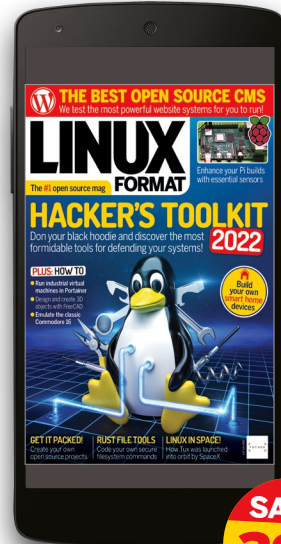
SIX-MONTHLY PRINT & DIGITAL EDITION



Only £46.25

Six months of *Linux Format* in both print and digital by Direct Debit

DIGITAL EDITION



Only £27.50

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

Terms and conditions: Offer closes 31 July, 2022. Offer open to new UK subscribers only. Pricing is guaranteed for the first 12 months and we'll 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.

SUMMER SAVINGS

Subscribe to your new summer read today



3 issues for £/\$/€3



6 issues for £6



3 issues for £/\$/€3



6 issues for £6



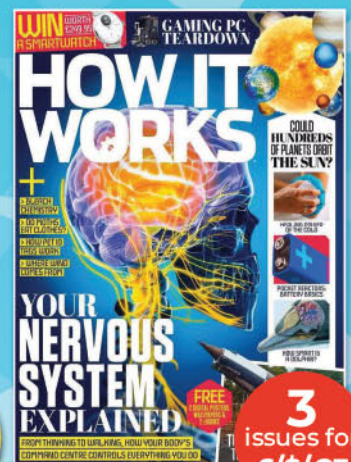
6 issues for £6



3 issues for £/\$/€3



6 issues for £6



3 issues for £/\$/€3

For great savings on all of our magazines, see the entire range online www.magazinesdirect.com/sun or call **0330 333 1113** and quote **SU22**

*TERMS AND CONDITIONS: Offer closes 31st July 2022. Offer open to new subscribers only. After your trial issues, your subscription will continue at the price shown at the point of purchase. 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). Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. For full terms and conditions, visit magazinesdirect.com/terms. For enquiries, please call: +44 (0) 330 333 1113. Lines are open Monday-Friday 8:30am-7pm, Saturday 10am-3pm UK Time (excluding Bank Holidays) 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.

REVIEWS

GoDaddy

Purchase and bid on domains with the daddy of services and **Stefan Ionescu**.

IN BRIEF

GoDaddy is a domain registrar that's easy to use and offers solid customer support for users. The company has one of the broadest selection of domain extensions, but its domains are expensive compared to rivals, especially for individuals and small businesses.

GoDaddy is an American domain registrar and web-hosting provider founded over two decades ago. It's one of the biggest companies in its sector, serving over 20 million customers and employs over 9,000 people worldwide. GoDaddy offers domain name registration, cloud hosting and a website builder. It also provides complimentary digital marketing tools for customers.

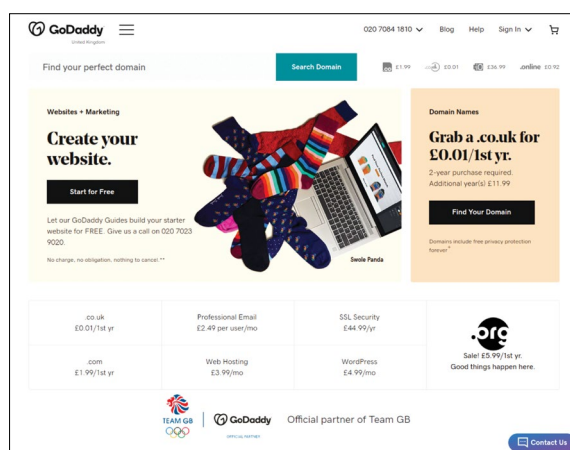
GoDaddy charges varying fees for different domain names. The common domain extensions such as **.com**, **.net**, and **.org** cost between \$20 to \$25 per year, while uncommon ones like **.cloud**, **.blog**, and **.today** cost between \$30 and \$40 per year. However, some select premium domains can run up to hundreds or thousands of dollars. GoDaddy's prices are significantly higher than what you can find on rival platforms.

You can register any available domain name through GoDaddy. One of the great things about the platform is that it offers a range of domain extensions. Its domain names are classified into two categories: generic top-level domain and country-code top-level domain.

Name of the game

Registering a new domain name is easy. The first step is to visit the official homepage (www.godaddy.com). Once the home page loads, you'll see a search bar where you can check to see if your desired domain is available. If it is, you can add it to the cart. Otherwise, GoDaddy will prompt you to search for a new domain name or select a different variation.

If the domain name you want is already taken, don't give up. You can attempt to acquire it but in a different way than buying a new domain. The first step is to find out who owns the domain. If you type in the domain name in GoDaddy WHOIS database search tool you should see the email address used to register it. In some cases, domain name owners enable privacy protection so that their information doesn't appear on the WHOIS database. Such people are likely not entertaining any offers for their domains.



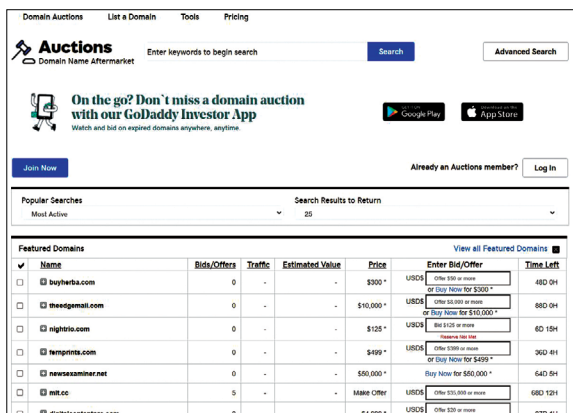
■ The number one destination for registrations.

GoDaddy Auctions is a marketplace for domain sellers to find potential buyers. It's an auction-based model, so the buyer with the highest bid wins. You can also transfer a domain name registered elsewhere to GoDaddy. Likewise, you can buy complementary services for your GoDaddy domains, such as professional emails and an SSL certificate.

Signing up and creating a GoDaddy account is simple. You can create an account using your email or a Facebook, Google, or Amazon account if you have one. GoDaddy's tools are easy to use. The interfaces are typically neat, uncluttered, and easy to navigate.

GoDaddy offers customer support through telephone and live chat 24/7. They're different mobile numbers to contact depending on your country of residence. GoDaddy has over 20 million customers, so the support lines can occasionally clog up when there's high demand.

GoDaddy is a reliable domain registrar. It's easy to use and offers excellent customer support in times of difficulty. However, the main drawback is that it's significantly costlier than rival domain registrars. **LXF**



Name	Bids/Offers	Traffic	Estimated Value	Price	Enter Bid/Offer	Time Left
buyherba.com	0	-	-	\$300	USDS Offer \$10 or more Buy Now for \$300*	48D 0H
theeggmail.com	0	-	-	\$10,000*	USDS Offer \$0.00 or more Buy Now for \$10,000*	88D 0H
nighrio.com	0	-	-	\$125	USDS Bid \$25 or more Buy Now for \$125	6D 15H
mapprints.com	0	-	-	\$499	USDS Offer \$100 or more Buy Now for \$499*	36D 15H
newsammar.net	0	-	-	\$50,000*	USDS Buy Now for \$50,000*	64D 5H
mtc.co	5	-	-	Make Offer	USDS Offer \$20,000 or more	68D 12H
digitalcontentpro.com	0	-	-	\$4,999*	USDS Offer \$20 or more	27D 14H

Try and pick up a domain bargain at the auction house.

VERDICT

DEVELOPER: Go Daddy
WEB: www.godaddy.com
PRICE: From \$0.19 (for .com domains)

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

Being the biggest offers a lot of benefits, but it also means that you can charge a premium for your services.

» **Rating 9/10**

OpenSUSE 15.4

If someone were to compile the definitive list of the mysteries of life, **Mayank Sharma** would ask them to include openSUSE at number 42.

IN BRIEF

One of the oldest Linux distros, openSUSE is to SUSE Linux Enterprise (SLE) what Fedora is to RHEL. Still popular, the distro has lost ground to its peers, as its corporate sponsor changed owners a couple of times in the past decade. Since 2018, the distro has aligned its version numbers with SLE 15.

SPECS

CPU: 2GHz dual-core
Memory: 2GB
HDD: 40GB
Build: x86_64, aarch64, PowerPC

We don't give openSUSE the attention it deserves, but that's partly because of the project's off-center release cycle. SUSE's enterprise edition and the community-supported openSUSE are now being developed in tandem from the ground up.

The openSUSE project maintains two main branches of the distro. In 2014, the developers split off a stabilised instance of its development branch named Factory into a rolling release update called Tumbleweed, pitched at developers and experienced users. Then comes Leap, which is the regular release that aligns with SLE, and is aligned with the sensibilities of long-term support (LTS) releases.

Leap 15.4 is based on the Jump release concept that was introduced in the previous 15.3 release, and sees the developers combine openSUSE Backports with binaries from SLE. Leap puts out one minor release once a year that's aligned with SLE Service Packs, and one major release between 36 and 48 months, aligned with the major SLE releases. This rather elongated release cycle helps openSUSE Leap pitch itself as a more stable option, making it suitable for everything, such as powering enterprise and beginner desktops.

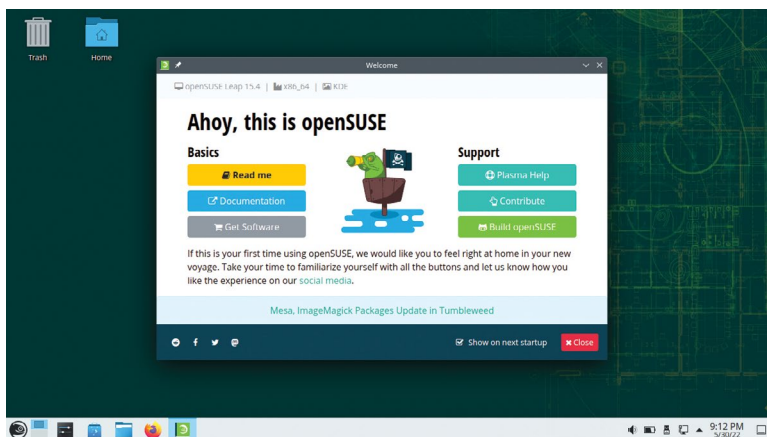
The 15.4 release we're reviewing has entered release candidate stage, which is pretty much what you'll get when the final release goes live around the time this issue is out. Unlike a regular release though, which includes installable Live CDs, openSUSE 15.4 at this stage only offers an install-only medium.

More of the same

OpenSUSE 15.4 will include the KDE Plasma 5.24 desktop environment, which is a long-term support (LTS) release, and will continue to receive updates and bug fixes until the desktop moves to Plasma 6.

Being an LTS release, 5.24 is all about ironing out the wrinkles, rather than introducing revolutionary new features. That said, the release does include a new Overview effect similar to the Activities Overview in Gnome. KDE developers also make a note about the changes to Plasma's default Breeze theme, which now aligns better with the Breeze style for programs, thereby improving visual consistency on the desktop.

For Gnome users, the release includes Gnome 41. While it does include some interesting features, not only did Gnome 41 come out last year, many of openSUSE's peers – most notably Ubuntu and Fedora – have moved to Gnome 42 in their latest releases. However, it's the Xfce users who'll feel they've got a raw deal because



The Welcome utility in Leap 15.4 is a little too crude for our liking. It has an unhealthy bias towards the project's documentation, rather than administration tasks.

15.4 bundles Xfce 4.16, which is what it carried in the previous 15.3 release as well, and is the last stable release from Xfce's stable.

Another component from last year is 15.4's 5.14.21 kernel. This is the same one that's used in SLE 15 Service Pack 4, and is maintained by SUSE.

OpenSUSE Leap is much more than a desktop distro, and the inclusion of various AI libraries and frameworks indicate that the release serves more than just desktop use cases. In fact, the System Roles section in openSUSE's elegant installer can easily customise the entire installation based on several different use cases, from a KDE or Gnome desktop to all-purpose server.

However, the most interesting option in the System Roles section remains the Transactional Server option. This uses a read-only root filesystem and makes it easier to track updates and perform rollbacks.

Like all LTS distros, openSUSE Leap releases aren't the most exciting in terms of new features, and the latest 15.4 release is no different. However, it does an impressive job as a stable enterprise desktop, or a rock-solid server or development workstation. **LXF**

VERDICT

DEVELOPER: OpenSUSE Project
WEB: www.opensuse.org
LICENCE: Various

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

A must-have upgrade for existing openSUSE Leap users, the 15.4 release will suit anyone looking for an LTS distro.

» **Rating 8/10**

Tails 5.1

This distro is one of those things that **Mayank Sharma** would rather have and not need, than need and not have.

IN BRIEF

The Amnesic Incognito Live System, popularly known as Tails, is one of the leading distros that's built to ensure maximum privacy, and anonymity for its users. The Debian-based distro is built around the Tor anonymous network, but also has a few other tricks up its sleeve to protect you as you browse the web.

SPECS

CPU: Any dual-core CPU
Memory: 2GB
HDD: 8GB
USB stick
Build: x86-64

Many people don't like the idea of trusting a third-party VPN provider to keep their online activity private, and instead rely on the open source Tor anonymity network. And no one does Tor better than Tails.

If you're new to Tails, the distro boots to a welcome tool that you can use to tweak some settings, such as the language and region settings. Furthermore, the screen also rolls in some advanced settings under the + icon, such as the ability to define a password for the default admin user, disabling MAC address spoofing (which could cause connection issues on some computers) and more.

When you're done with this initial setup and configured your Tails instance, the distro brings up another screen to help you connect to the Tor network. Tails will do all the heavy lifting – all it needs to know from you is how you'd like to connect to the Tor network.

The first option labelled 'Easier' is ideal for users who are browsing the web from a public computer or over a public hotspot. You can also optionally toggle the Tor bridge mode if you know that your ISP actively blocks connections to the Tor network. The option is there to save a couple of seconds because Tails will configure a bridge automatically if it can't connect you without using one, irrespective of whether you've toggled the bridge option or not.

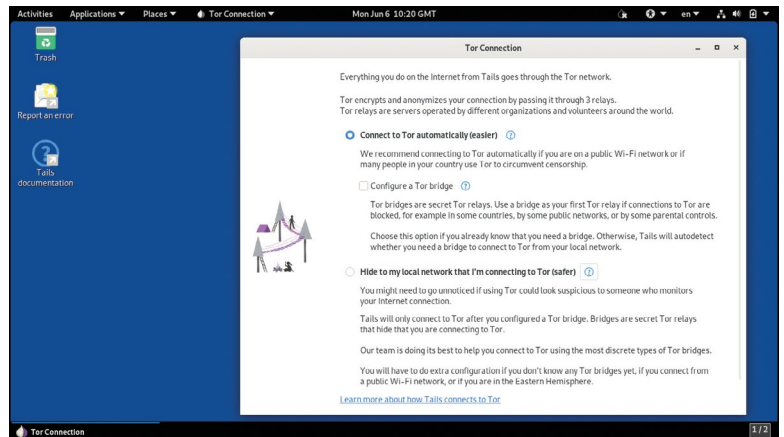
The second option labelled 'Easy' will take a few more steps to hide the fact that you're connecting to the Tor network, even from your local network.

Once you complete the setup, Tails will establish a connection to the Tor network and drop you to the Gnome desktop. You can click the Tor icon in the status bar to view the Tor circuits and streams. Once connected you can use the Tor browser and surf anonymously. There's also the *OnionShare* anonymous file-sharing application that leverages the Tor network.

Favourable tailwind

Tails is primarily offered as an image for the USB stick, although you can find an ISO image as well. Soon after Tails 5.0 was released, the project asked people to avoid using it because of a security vulnerability in the JavaScript engine of the *Tor Browser*. Instead, Tails 5.1 released on 5 June became the first official release of the Tails 5.x branch.

Tails 5 is based on Debian 11. While you can connect to Tor from any distro, Tails' USP is that it goes the extra mile to protect your privacy as well. For instance, it



Tails ships with a decent set of default applications that enables you to use your anonymous, privacy-focused distribution productively from the get-go.

ensures it stays clear of the storage disks on your computer, doesn't even try to look for a swap partition, and has no option in the file manager to mount the local disks. Moreover, it uses scripts to wipe your RAM when you shut down the distro, to guard you against any forensic recovery techniques.

Regular users of Tails can choose to create an encrypted persistent partition to store files, and settings like network configurations and SSH keys. Tails 5 adds another function to Persistence Storage called Additional Software. This makes it easier for users to configure additional software packages.

Another highlight of the release is the inclusion of the *Kleopatra* certificate manager and unified crypto GUI. This is borrowed from the KDE project to replace the Easier tools for managing passwords and keys that weren't actively maintained anymore.

Remember, though, that while Tails does all it can to keep you anonymous on the internet, you'll have to be equally prudent. Thankfully, the project's documentation does a good job of educating you on using the internet without leaving a trace. **LXF**

VERDICT

DEVELOPER: The Tails Project

WEB: <https://tails.boum.org>

LICENCE: GNU GPL v3

FEATURES	8/10	EASE OF USE	8/10
PERFORMANCE	7/10	DOCUMENTATION	9/10

Tails is one of those distros that we recommend all road warriors have on a bootable USB stick.

» **Rating 8/10**

OpenMediaVault 6

Old wine in a new bottle is such a cliché, but that's exactly how **Mayank Sharma** likes his software to grow and evolve.

IN BRIEF

OpenMediaVault is a specialised Debian-based distro to build and administer NAS devices. It can easily be scaled from serving a single user to a small office. The distro has modest hardware requirements and supports virtually all the popular features you'll need in a NAS, which are accessed via an intuitive, browser-based interface.

SPECS

CPU: Any dual-core processor
Memory: 1GB
HDD: 4GB
Build: x86-64, ARM

Nobody who's ever lost data will argue about the usefulness of a backup, and anyone who's ever kept one will know the importance of good backup software. Your distro probably ships with one, and it does a good job of retaining copies of your important files. But if you need more protection for your data than just duplicating it, you need to invest in a dedicated network-attached storage (NAS) device.

While there's no dearth of plug-and-play NAS boxes out there, building one from scratch doesn't take as much effort as other DIY tasks. That's because OpenMedia Vault (OMV) does all the work for you, reducing the process to essentially just tinkering with some settings to set things up as per your needs. In fact, the biggest task in creating your own NAS with OMV is assembling the hardware. Once you've cobbled together a bunch of disks inside an old, unused computer, you're pretty much done.

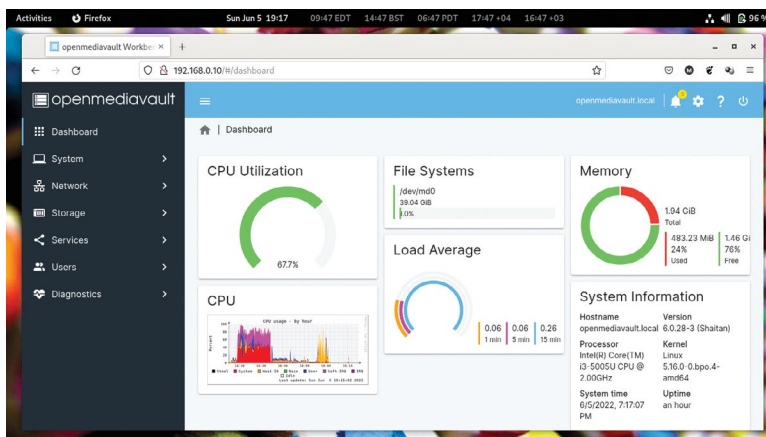
OMV's ease of deployment is only rivalled by its rich set of features. OMV supports all the popular deployment mechanisms, including several levels of software RAID, and you can access the data it holds using all the popular network protocols such as SSH, SMB/CIFS, FTP, rsync and more.

OMV also features an extensive permissions system that can help you control access to the shared volumes and folders, which means you can scale it to handle a lot of users. It supports all the file systems in vogue including EXT3, EXT4, BTRFS and XFS, and enables you to keep tabs on your rig with a comprehensive monitoring system. On top of it all, OMV is modular and can be extended with official and third-party plugins, which helps you do things with your custom NAS box that aren't always possible with an off-the-shelf option.

Same ol', same ol'

OMV's latest release marks the beginning of the v6 branch that's based on Debian 11. One of the noticeable differences in this release is the new user interface that's been written from scratch.

The new UI hasn't adversely impacted usability. You still step through the menu linearly, stringing your disks in a RAID, creating a filesystem, before mounting it and creating as many shared folders as you want. You'll need to configure the network services before you can access the NAS over the network, and OMV still offers reasonable defaults to handhold first-timers, making the process straightforward.



Raspberry Pi users can use a script to install OMV 6 on top of a Raspberry Pi OS Lite installation.

Another highlight of the release is the addition of a bunch of useful new plugins. For instance there's one that'll sync a shared folder with *Microsoft OneDrive*, another that'll use the AI-powered *PhotoPrism* to organise, and share pictures in your NAS, and others to give users the ability to browse files on the NAS using a graphical interface or from a web-based terminal.

Besides the visible changes, there are several enhancements in the distro's plumbing. The ISO installer has been enhanced to support USB-to-USB installations, and the *omv-update* tool has been replaced by the *omv-upgrade* CLI tool to keep the current installation updated. We also had no issues upgrading our previous 5.x OMV installation to the new 6.x branch using the `omv-release-upgrade` command.

OMV is one of the shining examples of open source software that hasn't lost any of its usability over the years. In all honesty, there's nothing it can do that can't be done by other NAS distros. However, it's OMV's intuitive interface and its dexterity thanks to the wonderful plugins ecosystem, which makes it a winner in our books. **LXF**

VERDICT

DEVELOPER: Volker Theile
WEB: www.openmediavault.org
LICENCE: GPL v3

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

The latest OMV 6 release enhances the distro's best bits and removes the shortcomings; everything an update should be.

➤ **Rating 8/10**

OnlyOffice 7.1

Usually **Mayank Sharma** will run everything that'll fit inside a browser, but today he'll try something he's petrified to launch even as a native program.

IN BRIEF

A free, open source office suite that runs inside a web browser, and can be installed on a private server. OnlyOffice is cross-platform and can be set up on Windows, Linux and MacOS. It provides all the functionality you'd expect from an office suite, along with the convenience of collaboration.

SPECS

CPU: Dual-core 2GHz
Memory: 2GB
HDD: 40GB
Build: x86-64, ARM

OnlyOffice's biggest USP is that it's a web-based editor you can host on your own servers. However, the office suite does have a desktop presence as well. The biggest difference between the web-based and the desktop instance of *OnlyOffice* is collaboration. While the web-based version can be used to collaborate on documents after installation, the desktop version is designed as a standalone editor. That said, you can connect the desktop version with an online cloud server to add-in collaborative editing features.

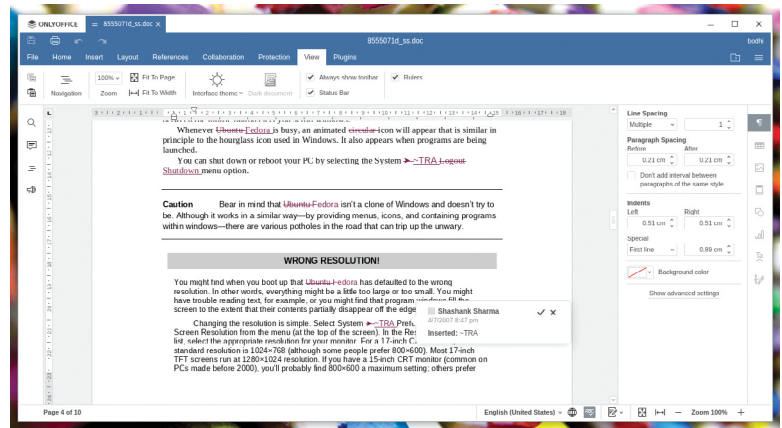
OnlyOffice includes tools for editing text documents, spreadsheets and presentations. Its developers tout its compatibility with *Microsoft Office's* Open XML's formats (.docx, .xlsx and .pptx) as one of its major attractions. The primary means of consuming *OnlyOffice* is via its web-based version. These are available either as a cloud-based software-as-a-service (SaaS) instance or a self-hosted on-premise server. The latter has several tiers based on scalability and usage requirements, with a zero-cost Community option for non-commercial use.

All installable versions of *OnlyOffice* are cross-platform, and the developers offer several avenues of installing them on Windows, Linux and macOS. For instance, the self-hosted version offers a convenient script that'll help pull-in the required components inside a Docker container. However, if you want a scalable option, you can set up *OnlyOffice* using its pre-compiled binaries. Similarly, the desktop edition of the office suite is bundled as a convenient AppImage, Flatpak and Snap packages in addition to the binaries.

New and noteworthy

Starting with v7.1, *OnlyOffice* can now be installed on Arm devices as well. One of the headline improvements is its ability to convert PDF files to docx, and other formats with a couple of clicks. All the tools starting with this version now include a View tab that packs in various options to help you alter the appearance of the document for better readability. The developers have also improved the built-in viewer, which now offers conveniences like a page thumbnails panel, a navigation bar, and support for external and internal links. Another useful addition is the ability to filter comments by groups of users.

The various tools in *OnlyOffice 7.1* have also been improved individually. One of the most useful additions in the spreadsheet editor, for instance, is the addition of tooltips for formulas, which displays all available



OnlyOffice's user interface bears a striking similarity to a certain proprietary office suite, which is further reaffirmed by its choice of default file format: docx.

formulas as you begin typing one. Similarly, the presentation editor in this release sports a new Animation tab that you can use to add and tweak all kinds of animations to help make your slides stand out.

While *OnlyOffice* is a wonderful open source collaborative office suite, it shares this space with others including *Collabora*. While *OnlyOffice* is an independently developed suite, *Collabora* is based on *LibreOffice*. *Collabora* also scores for being the primary document server in Nextcloud. However, *OnlyOffice* more than makes up by providing connectors for not just Nextcloud, but for various other services including Alfresco, Confluence, Liferay, OwnCloud, Seafile, SharePoint and more.

Except for its claimed superior compatibility with *Microsoft Office*, *OnlyOffice* doesn't offer anything you can't already do with the other options, particularly *LibreOffice* and *Collabora*. While we wouldn't recommend its web-based options to inexperienced users, due to its involved installation process, we'd encourage you to give its desktop version a shot, especially if you regularly handle documents in proprietary formats. **LXF**

VERDICT

DEVELOPER: Ascensio System SIA
WEB: www.onlyoffice.com
LICENCE: AGPL v3

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

If handling MS Office documents is a priority, *OnlyOffice* will give you more bang for the buck than its peers.

» **Rating 8/10**

The Stanley Parable: Ultra Deluxe

Management doesn't like self-referential things. It just reminds them they're a simple strapline-based comedy construct of **Christopher Livingston**.

SPECS

Minimum OS: 64-bit
CPU: Intel Core i3 2GHz
Memory: 4GB
HDD: 5GB
GPU: Nvidia GeForce 450, 1GB Vram

You'll never have given much thought to the "skip dialogue" button in a videogame, but after playing *The Stanley Parable: Ultra Deluxe* you won't stop thinking about it. The button (it's a physical button in the game world, so you have to be standing in a specific place to use it) is just one of several new features you can take for a spin in the "expanded reimagining" of 2013's *The Stanley Parable*. Once again, stepping into Stanley's shoes turns the act of playing a game into a hilarious, surprising and at times deeply thoughtful examination of games and game development, players and player choice, and yes, even the consequences of pushing a button.

Let's get this out of the way early: It feels like a trap to review *The Stanley Parable: Ultra Deluxe*, considering that part of this expanded version takes place in a museum of memories, where the narrator reads aloud from several reviews of the original game. Not just professional reviews from *Destructoid* and *GameSpot*, framed and hanging on walls and lit by candlelight (our sister title *PC Gamer's* own 90 per cent review is missing), but also Steam user reviews unceremoniously dumped in piles and scattered



Brace yourself for the most sarcastic sequence in videogame history.

around a rainy dockyard, including one that suggested a skip dialogue button was needed because the narrator talked a bit too much. It really gives you something to ponder while you're pressing the new skip dialogue button because the narrator is talking a bit too much.

It's a trap... but that's okay

Back to the point. Reviewing a game so willing to shine a light on its own reviews seems a bit like stepping on to a trap door clearly labelled "trap door." On the other hand, walking into traps you've been warned away from, and doing things you're not supposed to do before finding out the game really does want you to do them, is how you play *The Stanley Parable*. So, why not review it? Maybe it'll wind up framed in *The Stanley Parable* re-re-release someday.

But what even is *Ultra Deluxe*? It's not just a remaster, although the original first-person game has been so faithfully rebuilt it took us several hours to realise it was no longer in the *Source Engine* but in *Unity*. You can play through it once again as office worker Stanley, who one day realises he's the only one in the building and sets out to discover what happened to his coworkers as a gentle storybook-style narration guides him through the empty corridors. The simple act of disobeying your instructions and making your own choices leads to numerous branching paths, a range of reactions from the narrator, multiple endings, and the pure joy of doing something unexpected and discovering that the game fully expected you to do it.

But at some point while replaying the new version of the old *The Stanley Parable*, the new content of *Ultra*



Remember this? Now you can remember it with a bucket.

Deluxe begins to intrude. It's not a subtle introduction – a door labelled "New Content" appears in the familiar office corridor you've walked down a dozen times already. Step through it and a bunch of new features are trotted out for your approval. For instance, there's a bucket that Stanley can pick up and carry around with him.

Buckets of fun

That's it. It's just a bucket, a little joke based on the idea that some players found the original game confusing, and so this "Reassurance Bucket" can now be carried around as a companion to stave off any feelings of discomfort or uncertainty. But of course, a simple joke in the hands of developer Crows Crows Crows (the studio founded by one of *The Stanley Parable's* original creators) never stays simple for long. The bucket draws new commentary from the narrator, which made us wonder if anything else changed while we were carrying it. What if you climbed out a window with the bucket, or carried it to the boss's office, or took it along to one of the original game's many endings? What if you started the game over with the bucket and then brought the bucket back to the place where you're originally given the bucket? Could you get two buckets?

The Stanley Parable: Ultra Deluxe does a brilliant job anticipating anything and everything a player might do while playing *The Stanley Parable* while holding a bucket, and the rewards for experimenting are more humour, confusion, absurdity and thoughtful explorations about game design and player choice.

At one point deciding to try and get rid of the bucket, a lengthy task that rewards you with a "replacement bucket", which you can also stubbornly get rid of, after which if you restart the game you'll find that... there's no longer any bucket.

And as silly as it sounds, we have to admit we genuinely felt a bit of panic. Had we legitimately become emotionally attached to the bucket? No, no. It was just that carrying the bucket around with us had unlocked a number of new endings (and bucket-based revisions of the game's original endings) so we worried that without the bucket, we might wind up missing some secret ending, a bit of narration, or one of the silly or thought-provoking moments that are the collectibles of *The Stanley Parable* experience.

(*Ultra Deluxe* does add literal collectibles, by the way. We'd definitely suggest collecting them all, even though



you're told repeatedly there will be no reward for collecting them. And that's true, there isn't. But then again, there is. Was that a spoiler?)

We did, eventually, get the Reassurance Bucket back, and dammit, as stupid as it feels to say this, we felt reassured to once more be clutching it. That's really the genius of *Ultra Deluxe*: it gets you to laugh at a joke and then slowly makes you realise how much truth lies within that joke.

There's far more than just a bucket in *Ultra Deluxe*. We just don't want to spoil the rest of the new features, or even reveal how they're revealed because it's just about perfect for a videogame about videogames. We've played for about 10 hours and are pretty sure we still haven't uncovered all of *Ultra Deluxe's* tricks and treats.

As with the original game, there's a fair amount of fruitless wandering through parts of the office you've been through many times before, trying to discover something new where there's nothing new to be found. And there are still a few patience-testing moments where the narration carries on just a bit too long (*Don't say that! – Ed*) and found ourselves restless to keep exploring rather than standing in place listening. (We definitely don't think the game needs another skip dialogue button, however. One is plenty.)

But mostly, *Ultra Deluxe* is an adventure full of delightful surprises and sharp, funny observations about games, how we play them, what we expect from them, and what they expect from us. If you missed the original 2011 *Half-Life* mod or even the original 2013 commercial release, this is a thought-provoking and unusual twist on the twisty world of gaming. **LXF**

Hav'n't things moved on at all from the original? Just awful!



Something tells me this ending won't suffice.

VERDICT

DEVELOPER: Crows Crows Crows
WEB: <https://crowscrowscrows.com>
PRICE: £20

GAMEPLAY	9/10	LONGEVITY	8/10
GRAPHICS	7/10	VALUE	8/10

A delightful update that fills the original game with even more humorous and thoughtful rabbit holes to get lost in.

» **Rating 9/10**

Roundup

Raspberry Pi OS 2022-4-4 » Twister OS 2.1.2
» Ubuntu 22.04 » Pop!_OS 21.10 » Manjaro KDE 22.04



Michael Reed

once left his Pi to cool on the windowsill. He soon got a bigger heat sink though.

Raspberry Pi distros

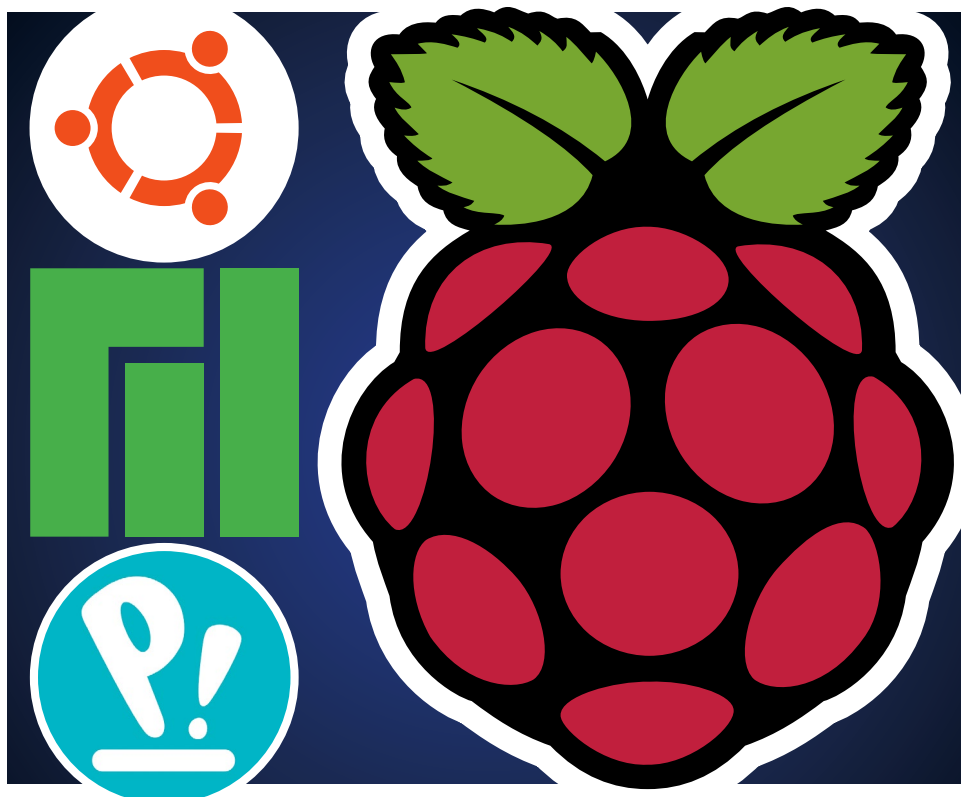
The top-end Raspberry Pis can be used to build a formidable desktop. **Michael Reed** finds there's a fair bit of choice when it comes to Linux distros.

HOW WE TESTED...

We spent some time with each distro and carried out typical tasks such as launching and running programs, copying files and installing new software. We also made sure that updating the system worked as it should.

Because we're looking at desktop usage rather than some of the other niche roles that a Raspberry Pi might be employed in, we're looking for good performance on the higher-end Pis.

For our tests we've used a Pi 400, the all-in-one model with a four-core 1.8GHz ARM chip and 4GB of RAM. At the time of writing, only the Pi 4 Model B has more memory, with 8GB of RAM, but we tend to find that for typical desktop use, we rarely hit the limit of what a 4GB can offer. The distributions tailored to the Raspberry Pi tend to be more memory efficient, and in general use, we tend not to pile on as many heavyweight applications when using a Pi.



The Raspberry Pi series of single board computers has evolved from a quirky, low-powered platform for experimental and educational projects to a machine that's powerful enough to take over many of the jobs that would have once required a traditional desktop PC. The Pi 400 and the Pi 4 are the top-end of the Pi range and they make light work of typical desktop tasks such as word processing and text editing, and they have enough power to be competent web browsers.

We're going to look at five distributions specifically aimed at turning the Pi into a

competent desktop computer. For many, Raspberry Pi OS (formally Raspbian) is the default face of desktop Pi usage. We've chosen the 64-bit edition because it's closer to standard Debian and more representative of the future of this distro. Manjaro uses Arch as its base, and we've chosen the KDE edition. Raspberry Pi Ubuntu is somewhat similar to the normal desktop PC edition, and it's also a 64-bit OS. Pop!_OS is derived from Ubuntu, but it's a highly personalised existence. Twister OS is a customised version of Raspberry Pi that comes with added features and sports a bespoke Xfce desktop.

The default software choices

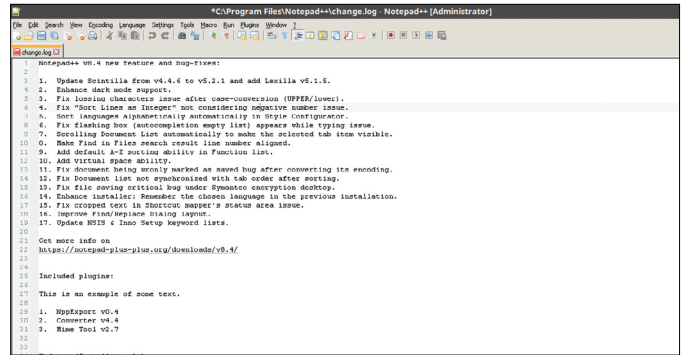
There's more than one approach that a distro can take to its default tools.

There are two approaches to the software loadout when it comes to a Raspberry Pi desktop operating system. Some distros offer ultra-light applications that make the most of limited hardware, and some go for full-fat choices at the expense of outright speed. Because these are full Linux systems, it's possible to replace all of the default application choices. We're scoring more highly if the default application selections can cover the typical use cases for a business, school or home desktop user.

Raspberry Pi OS has to be all things to all Pi users, and the choices are both minimal in number and lightweight. To get any real work done with it, you'd have to carry out some installations. We were keen to replace *PCManFM* with a heavier file manager such as *Dolphin*, which the higher-end Pis can handle.

Unlike its parent distro, Raspberry Pi OS, Twister OS has plenty of pre-configured software packages, including some that are difficult to get working on a Pi such as *Steam*, *Wine* and a few games and emulators. *Wine* was impressively fast on the Pi because it was having to emulate an x86 CPU. However, some choices such as the *Thunar* file manager are a bit lightweight and feel out of place next to the more adventurous inclusions.

Unsurprisingly, the Manjaro application selection tends towards the expected KDE stalwarts, meaning some lush choices, which will please higher-end Pi owners. For example, we were happy to see the *Dolphin* file manager and the *Kate* text editor featured because it's worth the delay of a few seconds on launch



We installed the 32-bit version of Notepad++ with the .exe file and double clicking on it. We were impressed with the problem-free smoothness of this operation.

for the extra features. However, the default application selection was sparse, which seems unnecessary and means that additions will be required for most desktop use cases.

Ubuntu has a minimal selection of applications that you'd expect from a Gnome-based distro. *Firefox*, *Thunderbird*, *Rhythmbox*, *Shotwell* and *LibreOffice* are present as expected, offering a fairly standard desktop.

Pop!_OS is a customised version of Ubuntu, and like Ubuntu it keeps up the Gnome tradition of giving programs generic names such as *Files* and *Document Viewer*. The default application selection is light, but it covers the basics such as office duties thanks to *LibreOffice* and a video player (which is actually *Totem*).

VERDICT

RASPBERRY PI OS	6/10	MANJARO	6/10
TWISTER OS	8/10	POP!OS	7/10
UBUNTU	7/10		

Twister OS has a large selection of pre-configured software. It's a shame that the basic tools aren't fully-featured versions.

Installation procedure

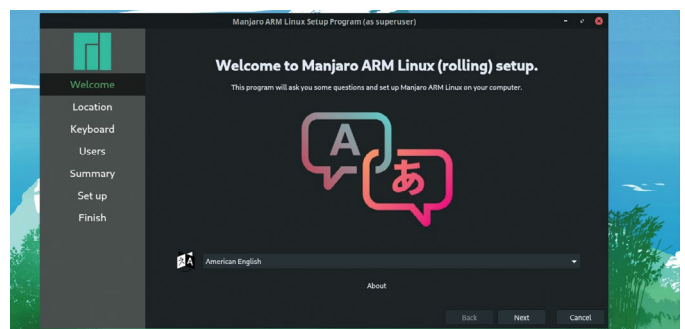
We're on the look-out for any speed bumps at the first stage.

All of the distributions that we've looked at use an installation method typical of Raspberry Pi operating systems – there's no installation procedure as such – instead, an external utility is used to write an image file to the SD card, which is then inserted into the Pi. On the first run, a setup routine takes place.

The first-run dialog of Raspberry Pi OS now advises against using the standard Pi username and password that had been the default in years gone by. It chooses to do the updates on the first run before restarting.

On first run, Twister OS pops up a welcome dialog that offers links to documentation followed by a Raspberry Pi configuration application, but this could make things a tiny bit less beginner-friendly. Because Twister OS is based on an older version of Raspberry Pi OS, we didn't approve of the use of the traditional Pi username and password. These can be changed, though.

Be quick when installing Manjaro. If you leave the machine unattended, the screensaver is activated, but because you haven't



The first-run dialog of Manjaro is typical of what Pi operating systems tend to offer. In many ways, it's simpler and faster than a typical desktop Linux distro.

set the username and password, there's no way of getting back in, requiring a hard reset! The user and admin password can, optionally, be set independently. Pop!_OS first run was a slick and friendly procedure and concluded with some UI options. The Ubuntu installation took longer than the others, and it's visually similar to a full desktop Ubuntu installation. On the first boot, there are some options for getting connected to social media accounts.

VERDICT

RASPBERRY PI OS	7/10	MANJARO	6/10
TWISTER OS	6/10	POP!OS	9/10
UBUNTU	7/10		

The Ubuntu installation is slower than the others. Manjaro's had a hiccup.

The default desktop experience

What's the personality of a distribution?

For many, the desktop sums up the personality of a distribution. Some of these desktop environments were once only considered suitable for high-end computers, but the Pi hardware has caught up, and none of them are too taxing for a 1.8GHz, four-core Pi 400. That said, there were some minor variations in overall responsiveness from one desktop environment to another, and we've noted such occurrences where appropriate.

Hopefully, the Super key is assigned to a useful function such as the launcher. That enables application searching and launching without the user having to take their hands off the keyboard.

We're a bit finicky at times, and we want to be able to customise the overall look of the desktop, and it's a point that falls in the desktop's favour if the settings for that are well organised and easily accessible.

In summary, we're looking for a desktop environment that looks smart and appealing, but that also supports an efficient workflow.

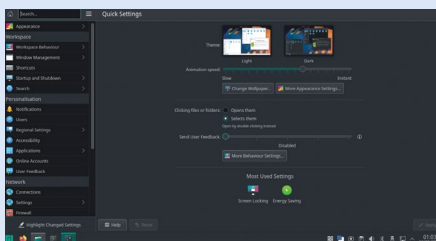
Manjaro

8/10

The default look is dark and understated. It's funny to consider that KDE was once a high-end desktop, but it was snappy on our Pi 400 despite offering some mild eye-candy in the form of transparent areas.

At the bottom of the screen we have a fairly standard bar with status area, quick launch icons and a status area. In the bottom left corner, we have the standard application launcher. This is searchable and can be invoked using the Super key giving excellent keyboard-based workflow.

The Settings panel is the full KDE one and this means that Appearance is just one of several categories that extend to general system settings in areas such as Hardware and Networking. Thankfully, the panel is searchable and there's a setting to switch between light and dark themes. Keeping all the settings together in this manner is probably better than setups that spread them around the entire system.



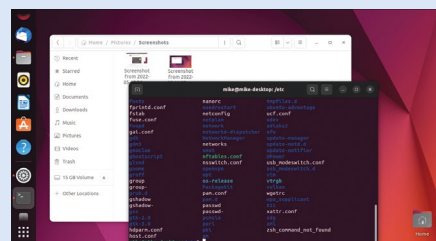
Ubuntu

7/10

The Ubuntu user experience on a Raspberry Pi is similar to that of a normal desktop Ubuntu, to the point that it becomes difficult to identify any differences in layout.

The standard Ubuntu Gnome setup means that an icon bar is present at the left side of the screen. This means that you lose a strip at the side rather than at the bottom of the screen, which makes sense on a widescreen monitor. The bar itself combines application launching and switching, and the Super key launches a combined graphical switcher and searchable launcher.

A click in the bottom right of the screen or a press of the Super key opens an application launcher. As is the case with all versions of Gnome, customisation of the layout is tricky to carry out. It's also a bit slower than the fastest desktops we've experienced on a Pi 400.



Speed and efficiency

Resources on a Pi are never sky high, so it's important that a distro be efficient.

Raspberry Pi OS was the lightest of the five distributions on test in this month's *Roundup*. It took 29 seconds to boot into the desktop. At this point, we opened a terminal and typed `code free -h`, which showed that 266MB was in use. We then launched *Chromium* and browsed to the *Linux Format* (which is a somewhat feature-free) website. At this point 485MB was in use. Overall, that's pretty impressive.

Twister OS offers similar characteristics to its parent OS, Raspberry Pi OS. Booting to the desktop took 33 seconds, and at that point, 304MB was in use, and this climbed to 436MB when the included *Chromium* browser was navigated to the site, a similar overall outcome to Raspberry Pi OS.

Performing the same test on Manjaro gave a power-on-to-desktop time of 43 seconds with 539MB in use. By the time we had pointed *Firefox* to the *Linux Format* site 919MB was in use. This still left a lot of memory free on a 4GB machine.

Ubuntu uses 698MB on a fresh boot, and it took 52 seconds to get there. That's not bad when you consider how similar this

edition is to regular desktop Ubuntu. This increases to 1.1GB in use when *Firefox* is pointed at our site.

Booting into the Pop!_OS desktop took 71 seconds and showed 748MB in use before any applications were loaded. Loading up *Firefox* navigation to the *Linux Format* website took that to 1.3GB used, which was more than two and a half times as much as Raspberry Pi OS.

It's super-subjective, but we felt that Pop!_OS and Ubuntu were a bit slower in use than Twister OS and Raspberry Pi OS. Manjaro was in the middle.

VERDICT

RASPBERRY PI OS	9/10	MANJARO	7 /10
TWISTER OS	8/10	POPI_OS	6/10
UBUNTU	7/10		

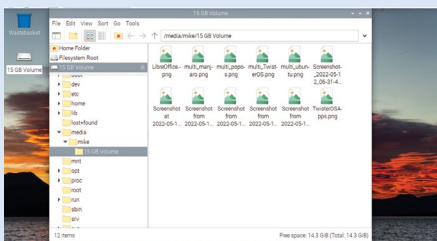
Kudos to the distros that save memory use where they can, but the difference isn't vitally important on a 4GB Raspberry Pi.

Raspberry Pi OS

6/10

Overall, the desktop has a bare-boned-but-functional feel to it, which isn't surprising because this is a front end that has to service the entire Raspberry Pi line in a variety of roles. Raspberry Pi OS places the main taskbar with application launcher and switcher at the top of the screen, in contrast to the standard bottom placement that many users will expect. In the corner we have a launcher button, but it's a shame that this isn't searchable or launchable via the Super key.

It's a functional desktop that can carry out standard desktop duties, but it's short on innovation or flair of a kind that draws users in and makes the desktop itself enjoyable to use. Beyond an overall lack of pizzazz, the lack of more modern accoutrements can slow down an experienced user, despite the fact that it's probably the fastest and most responsive desktop on a technical level.



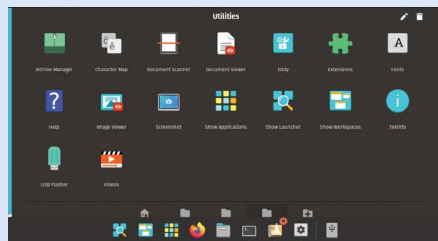
Pop!_OS

9/10

Pop!_OS uses a customised Gnome desktop. At the bottom of the screen is a dock for launching favourite applications and switching between running applications. The Pop!_OS desktop has two application launchers. An icon on the icon bar or a press of the Super key launches a minimalist, list-based, searchable application launcher. There's also a syntax for launching things like search engine queries from here.

Clicking Applications in the top launches an icon-based launcher reminiscent of Android's launchers. It has categories and it's searchable. A right-click menu enables you to add these programs to the dock. The desktop appearance settings are part of the masses of overall system settings, but again, it's searchable.

We'd rate Pop!_OS as offering the most carefully honed desktop experience of a Pi operating system.



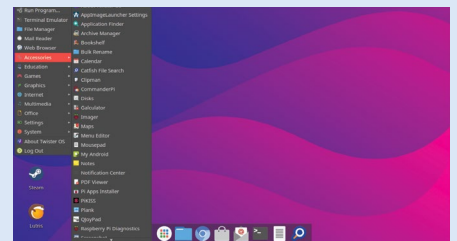
Twister OS

7/10

Expect to be hit with a swirly mixture of purple and blue when you first launch the Xfce-powered Twister OS desktop. At the top of the screen we have the usual taskbar with a categorised application launcher, and this is reminiscent of the Raspberry Pi OS screen layout.

Things start to deviate from that standard with the dock that's placed at the bottom of the screen. The first icon brings up a more modern, full-screen launcher that's searchable, but it's a shame that this isn't connected to the Super key. Another disappointment is that this dock is just a launcher rather than a combined launcher and switcher, as some other docks are.

The backdrop has a graphical system monitor section that shows variables like CPU speed and temperature. Performance was very good, although this is a desktop that takes features from different sources, giving an inconsistent feel.



Documentation and troubleshooting

How easy is it to obtain help when you run into difficulties?

Most of the official Pi documentation assumes that you're using Raspberry Pi OS, which gives the distro a head start in that department. Documentation for the OS has its own section of the website, forming a fairly traditional manual. Note that a lot of solutions for Debian problems will work with Raspberry Pi OS.

Ubuntu itself is extraordinarily well documented, with forums and manuals available and extensive support on the internet in general. The Raspberry Pi version is discussed within the main forums. Looking around the file layout shows that it's more or less identical to the desktop editions. This means that the majority of desktop Ubuntu solutions will work here.

To troubleshoot Twister OS problems, the usual path would be to search for Raspberry Pi OS solutions. Beyond a brief FAQ there doesn't seem to be any official documentation or a forum, although it's sometimes discussed on the Raspberry Pi OS forum. There is a Discord server, if you prefer a chatroom interface, and this has a good level of activity.

Manjaro covers the traditional support bases with a forum and a full PDF manual, but you might run into troubleshooting problems because Arch Linux on Raspberry Pi hardware is a less-common combination than Debian- or Ubuntu-based distributions. The forum has sections dedicated to the ARM builds of the operating system and these have good levels of activity.

There is official documentation on the System76 website that covers the specifics of Pop!_OS, which itself runs on an Ubuntu base. Therefore, solutions intended for Ubuntu will typically work for Pop!_OS if they're not specific to the Raspberry Pi.

VERDICT

RASPBERRY PI OS	9/10	MANJARO	6/10
TWISTER OS	6/10	POP!_OS	7/10
UBUNTU	8/10		

Raspberry Pi OS is the most common Raspberry Pi operating system and Raspberry Pi Ubuntu is similar to normal Ubuntu.

Adding software

Is it simple to install programs?

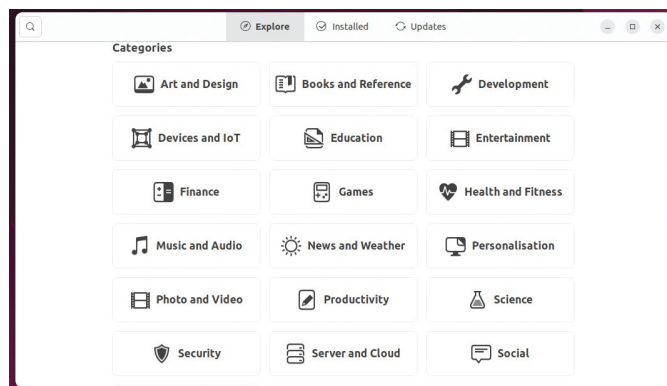
All of the distros that we've looked at are connected to a large repository of software, either of their own or a parent distribution. We're only taking into account the default software channels, even though – in theory – you could either add additional ones in some cases or adapt packages from a foreign system.

Raspberry Pi OS is connected to the Debian ARM64 repositories. This means that you can use the standard `apt` commands to install software. On the GUI front, 'Add / Remove Software' is a searchable front end to the repository itself that mixes components such as libraries and themes with actual applications, making it not so beginner-friendly. There's also a *Recommended Software* application, but there is no 'store'-type application built in.

Twister OS offers similar facilities to Raspberry Pi OS, but because Steam and Lutris are ready-installed, it could be argued that these constitute software channels themselves.

Manjaro's underlying distribution, Arch, uses its own packaging system, making it difficult to add the more common `.deb` and RPM format packages. However, typical installations are achievable using the *Pamac* package management front end to the Arch User Repository along with Snap and Flathub, which should cover most situations.

The Pop!_OS software centre, Pop!_Shop is hooked into the Ubuntu repositories and Flathub. The shop application ran well,



The Ubuntu Software Center is a fully featured software shop that includes user ratings and reviews. Shops give preference to fully fledged applications.

and the only way we could fault it is that applications don't have community reviews attached.

Ubuntu's store is the most featureful because it is connected to the Ubuntu repositories along with Flathub and Snap. The store application features community reviews and scores for the application, and so Ubuntu came out top in this section.

VERDICT

RASPBERRY PI OS	6/10	MANJARO	7/10
TWISTER OS	7/10	POPI_OS	7/10
UBUNTU	9/10		

Ubuntu has the best store application and ticks the most boxes in terms of software channels.

Distro editions

Alternative editions and variants to hone the experience and adapt to special requirements.

We're mainly interested in variants of a distribution that are available for a Pi, but we're also interested in editions that are available for other hardware such as desktop and laptop PCs. It's always handy if you can maintain some consistency between all of the machines that you manage.

Raspberry Pi OS is the only distribution that we'd recommend for typical desktop use on anything less powerful than a Pi 4 or 400. As well as editions that are tailored to running on a Pi 4/400, Raspberry Pi OS also offers a Lite edition without a desktop. Raspberry Pi Desktop is a version that runs on regular desktop PCs while offering much the same environment.

Twister OS is the only OS that we've looked at which doesn't offer a 64-bit edition. 64-bit OSes offer slightly better performance on a Pi 4 or 400 at the cost of some software compatibility. Twister OS has a Lite variant with less software included. Twister UI can be installed on regular desktop PCs.

Manjaro is the only one of the distributions we've looked at that's a rolling distribution, which some might prefer. As well as being a major desktop PC Linux distribution MATE, Gnome and Xfce editions are available for the Pi in addition to the KDE Plasma edition that we looked at here.

Ubuntu has editions to support most platforms and many variants with different desktop environments on top. For the Pi, Canonical's Desktop edition is 64-bit only and features Gnome as



The Raspberry Pi Imager can write images to an SD card. It has links to some operating systems built in, but it can also write an image that you've downloaded.

the desktop, which is the version that we've looked at here. There's also a server edition that runs without a desktop and is available in 32- and 64-bit variants. The same is true for the Pi versions of its Core edition that are designed for embedded use.

Pop!_OS keeps things simple, with a main edition of the distro available for desktop PCs and a Raspberry Pi edition. The releases are synchronised with Ubuntu releases. At time of writing, the Pi edition appears to lag slightly behind the desktop PC edition in terms of release speed.

VERDICT

RASPBERRY PI OS	7/10	MANJARO	8/10
TWISTER OS	6/10	POPI_OS	6/10
UBUNTU	6/10		

Manjaro has the most variations for the Raspberry Pi.

The Verdict

Raspberry Pi desktop distros

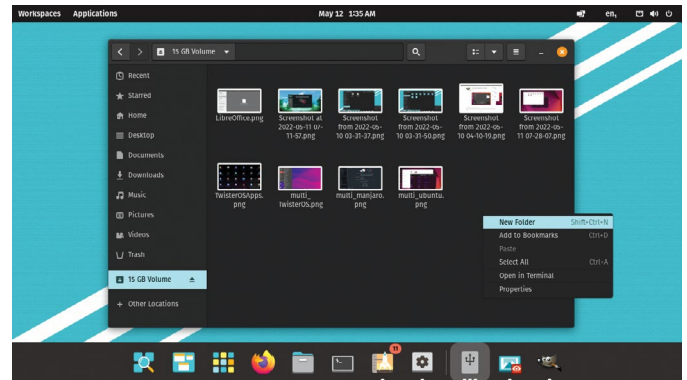
General desktop usage on a top-end Pi was the scenario that we imagined, and Pop!_OS has the most refined desktop. For that reason, we've decided to make it the overall winner. It's also the heaviest distribution in use. However, the amount of memory consumed by the base operating system and desktop is of lesser importance once you have 4GB of memory or more to play with. That said, running them side by side, some of the other distros did feel a little bit snappier in use. However, that's not to say that Pop!_OS is slow. Another thing to consider when it comes to speed is that Pop!_OS enables a fast workflow.

Manjaro offers a full desktop Linux experience on the Pi. The desktop environment is snappy and fully featured. Because this is basically KDE on top of an Arch Linux base, it's a fairly standard Linux under the hood, which is important when it comes to expanding and troubleshooting the system. We liked the choices in terms of desktop aesthetics and workflow, and a top-end Pi can handle most typical KDE applications well. It's a shame that the default application loadout isn't more extensive, though.

Ubuntu is another distribution that sticks closely to the normal desktop edition in its Raspberry Pi offering, and once again our Pi 400 handled it extremely well. The performance was acceptable. The default application selection covered all of the usual bases for desktop usage.

Twister OS has the solid, if a little unexciting, Raspberry Pi OS as its base with an extensive software collection. We'd seriously consider handing this distro to a Linux newcomer who wanted to see what a Raspberry Pi desktop could offer, particularly if they were interested in gaming and running Windows PC applications on a Pi. On the downside, the desktop feels like it was cobbled together from various sources and lacks some of the polish that we'd expect for a desktop that was to be used for business tasks.

Raspberry Pi OS is a safe choice for a desktop OS on a Raspberry Pi, and we can see why it's a popular choice for classroom usage, for example. It's stripped down with fast performance. However, for day-to-day usage most users would probably want to spend quite a bit of time swapping out components, adding software and configuring the desktop.



1st **Pop!_OS** **9/10**

Web: <https://pop.system76.com> **Licence:** various

Version: **21.10** This distro offers a carefully honed desktop experience on top of a solid Ubuntu base.

2nd **Manjaro** **8/10**

Web: <https://manjaro.org> **Licence:** various

Version: **KDE Plasma 22.04** A slick desktop that combines KDE with Arch Linux for robust technical foundations.

3rd **Ubuntu** **7/10**

Web: <https://ubuntu.com> **Licence:** various

Version: **22.04** A good, reliable distro in the desktop world that's been transferred to the Pi. Solid performance and few surprises.

4th **Twister OS** **7/10**

Web: <https://twisteros.com> **Licence:** various

Version: **2.1.2** An easy-to-use, if slightly messy desktop with some well-configured software.

5th **Raspberry Pi OS** **7/10**

Web: www.raspberrypi.com/software **Licence:** various

Version: **2022-04-04** The safest choice for those new to the Raspberry Pi, with excellent performance and support.

» ALSO CONSIDER

If you want to go super-lightweight while retaining a full desktop, consider DietPi (<https://dietpi.com>). Note that setup requires a bit of technical expertise. Gentoo Linux (www.gentoo.org) has a Raspberry Pi edition. Once again, you'll need to handle some technical issues, in contrast to the distros we've been looking at that are relatively plug and play.

If you're an experienced Linux user and you fancy dipping your toes into something really alternative, you could try one of

the FreeBSD offerings (www.freebsd.org). BSD is similar enough to Linux that you should be able to find your way around and build a desktop operating system with a difference.

If Red Hat-based distributions are more your style, it might be worth checking in on the Fedora project (<https://getfedora.org>). At the time of writing, the Pi is still listed as unsupported as a desktop operating system, but the developers are definitely working on bringing it up to speed. **LXF**

FAST VIRTUAL MACHINES

Jonni Bidwell transcends meat space, ventures virtuously into the valleys of the virtual and returns laden with actual knowledge.



Virtualisation... where would we be without it? We don't actually know, but life would certainly be harder. Thanks to virtual machines (VMs), home users can effortlessly boot all manner of other operating systems. Those might be exotic flavours of Linux or (since you surely don't really need an actual install of this monstrosity anymore) a Windows VM for those odd times where it's handy. This can be taken to extremes, too. Thanks to PCIe passthrough, which we covered back in LXF273, you can sacrifice

a whole GPU to your VM, enabling it to run games (or CUDA simulations) arbitrarily close to native speed.

This time around we'll look at a different but equally impressive technology in the form of Intel GVT-g. This makes it possible for us to segment newer Intel GPUs into virtual ones that can be seamlessly connected to our VMs. Unlike the passthrough method, this still enables us to use the GPU on the host.

Businesses, too, have a lot of love for the virtuals. VMs are highly portable and much easier to reinstantiate than regular

machines in the event that the host running them catches fire. There's also an efficiency improvement, in that a single machine can take on the role of several servers, and still enjoy the benefits of having those workloads properly segregated. Containers, of course, achieve this with even more efficiency, but at the cost of losing some of that segregation.

There are all kinds of tools for managing virtual machines at scale, so we'll have a brief look at one of those, too. Specifically, we'll show you that *Vagrant* makes it trivially easy to install Arch Linux.

Go beyond VirtualBox

Get your first VM running in seconds, with the simple elegance of Gnome Boxes...

Back in the mid-naughts, commodity CPUs learned a new trick. The idea had been around for a while, but the urge to make data centres ever more efficient really brought it to the fore. That idea was virtualisation, and on x86 platforms it first appeared in the form of Intel's VT-x and AMD's V on their early 64-bit CPUs. These instructions weren't groundbreaking in themselves, but they did enable simple software (such as Microsoft's *Virtual PC* or Sun's – now Oracle's – *VirtualBox*) to spin up virtual machines easily, thanks to being able to run a (largely unmodified) guest OS in a deprived environment.

It's important to avoid confusion with emulation. Emulation is imitation, and to emulate properly everything must be imitated pretty much exactly. That means processor bugs, weird firmware logic and even the rotation of floppy disks all have to be simulated. And this is a slow process, even on a fast CPU. With virtualisation some hardware may be emulated, but the processor instructions from within the guest are run more or less verbatim on the host CPU. Anything can be emulated, but only machines of the same architecture can be virtualised. So this tutorial won't help you create a virtual Apple M1 on your desktop box.

The Linux Kernel has its own hypervisor (a thing that runs virtual machines) called *KVM*. Others include VMware's *ESXi*, *Xen* and the ubiquitous *VirtualBox*. *VirtualBox* is the de facto hypervisor for many a home user, because it's easy to get started, it's cross-platform and if you need something out of the ordinary there's

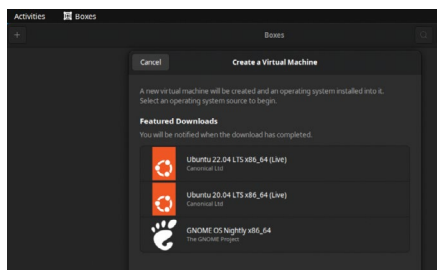
```
jenni@ubuntu2204:~$ sudo apt install vulkan-tools
[sudo] password for jenni:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed
vulkan-tools
0 to upgrade, 1 to newly install, 0 to remove and 5 not to upgrade.
Need to get 241 kB of archives.
After this operation, 1,124 kB of additional disk space will be used.
Get:1 http://gb.archive.ubuntu.com/ubuntu jammy/universe amd64 vulkan
1.3.204.0+dfsg1-1 [241 kB]
Fetched 241 kB in 8s (4,326 kB/s)
Selecting previously unselected package vulkan-tools.
(Reading database ... 197991 files and directories currently installed.)
Preparing to unpack .../vulkan-tools_1.3.204.0+dfsg1-1_amd64.deb ...
Unpacking vulkan-tools (1.3.204.0+dfsg1-1) ...
Setting up vulkan-tools (1.3.204.0+dfsg1-1) ...
Processing triggers for man-db (2.10.2-1) ...
jenni@ubuntu2204:~$ nvidia-smi
NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA driver.
MESA-INTEL: warning: Performance support disabled, consider sysctl dev.1915.perf_
s_trean_paranoid=0
Selected GPU 0: Intel(R) HD Graphics 620 (KBL GT2), type: 1
```

We'll see later how you can get 3D acceleration working in your VMs, like the Vulkan cube demo.

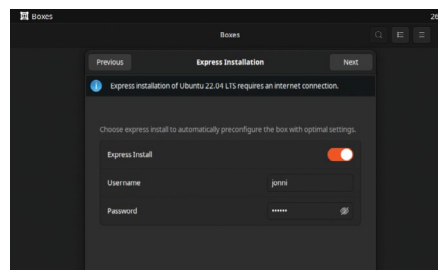
probably an easy option to get that working. But *VirtualBox* is made by Oracle, and while it's open source (both the core and the guest additions are GPL2-licensed) that company doesn't have a stellar reputation as stewards of FOSS.

Instead, we'd urge you to try *Gnome Boxes*. It's a front-end for libvirt (a platform for managing VMs), which behind the scenes uses *KVM* and *QEMU* (an emulator that can also virtualise). Libvirt can happily talk to other hypervisors too, but let's not get ahead of ourselves. *Gnome Boxes* is available as a Snap or Flatpak; however, our experience with the Snap wasn't positive. So we'd recommend installing from your distro's repositories. On Ubuntu you can find a Deb from the *Software* application. Click the drop-down labelled Source in the top-right to find it. Libvirt allows for all kinds of customisations, but *Boxes* is a delightfully simple application. So simple we've condensed set up of our first virtual box to a simple three-step dance (*below*).

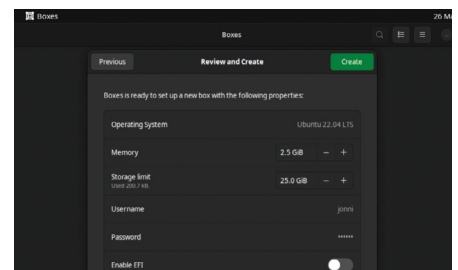
MAKE A VM WITH BOXES



1 Start Boxes Click the + in the top-left corner to begin the creation process. We'll create an Ubuntu 22.04 VM, which might even be suggested to you in the Featured Downloads area. If not you can find it (or all kinds of other distros) from the Download link at the bottom, or you can use any old ISO you happen to have lying around.



2 Download an OS Click the link and once your desired image has been downloaded you'll be offered an express installation. This enables us to preconfigure our Ubuntu VM with everything it needs, so let's do that. Express installation sidesteps interacting with the installer, so we'll need to specify a username and password first.



3 Choose resources Your VM will need memory, but so will your actual machine so don't assign more than around half your RAM to VMs. Ubuntu can run with 2GB of memory, but if you can afford to spare more, go for it. You need to specify a storage limit; the default 25GB is good for desktop Ubuntu. Don't worry about enabling EFI at this stage.

Tuning our VM

We have a virtual machine that's ticking over nicely. But what exactly can it do? And how might it do those things faster?

Once you hit **Create** (following on from the walkthrough on the previous page) you can watch the installation spring into life and in no time you'll be able to start your Ubuntu VM. As usual you'll be prompted to connect online accounts and perform any pending system updates.

So now we've got our first virtual machine up and running it's time to see what it can do. The Ubuntu VM should have set up the Spice (Simple Protocol for Independent Computing Environments) agent automatically. This means we should have a shared

THE SPICE MUST FLOW... DATA

“Spice is much more than shared clipboards and folders. It's what's used to get video and audio data out of the VM and display it on your screen.”

clipboard between the guest and host. Try copying and pasting in both directions to verify that this is working. If it's not playing ball, try running `systemctl start spice-vdagentd` on the guest. On other distributions this package may have to be installed separately.

But we can do much more than copy and paste (*I hope you're not referring to our other contributors there – Ed*). Try starting a file manager, or updating via Ubuntu's *Software* application. You should find it behaves exactly like the real thing (including asking you to reboot before you've even started). And if your machine's innards aren't too old you should find things run pretty smoothly. Hit the fullscreen button in the toolbar. The VM should automatically adjust to

Glxgears has never been a great benchmark. The cogs are really taxing our first VM.

accommodate the new resolution. If you want to exit fullscreen, just move the cursor to the top of the screen to reveal a hovering toolbar.

For now let's challenge our VM a little. Open up a terminal and run:

```
$ sudo apt install mesa-utils
$ glxgears
```

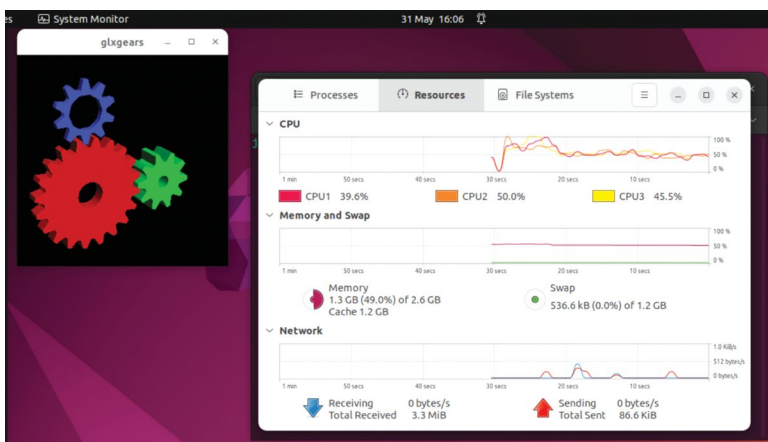
The familiar tricolour cogs should appear, and the terminal might tell you they're rendering at several hundred frames per second. But your eyes and ears will probably tell you otherwise. To wit, the animation will appear jerky and your machine's CPU will most likely be getting thrashed. Open another terminal tab and run `top`, and you'll see the *Glxgears* process is indeed taxing our VM's vCPU. On a regular Ubuntu machine (with an even quarter-decent graphics card) *Glxgears* would be accelerated by the Mesa drivers. But we have no such luxury (yet) on our VM, so it's all a little arduous. If you ventured into your VMs settings before powering it up, you might have noticed there was an option to enable 3D acceleration. Unfortunately, this is the subject of a reasonably annoying bug (at the time of writing, see <https://bugs.launchpad.net/ubuntu/+source/osinfo-db/+bug/1969524>) wherein the option will either not work, disappear or both. Don't worry – we'll come back to this later. For now let's focus on *Boxes*'s more functional options.

Transferring files

Just like a regular machine, your VM has an IP address. And just like a regular machine you might like to set up SSH on it and use that to run commands or transfer files. The default configuration in *Boxes* is to use NAT networking, so your VM will receive an internal address of the form 10.0.2.x. This enables the VM to talk to the outside world (including the host), but doesn't allow for host-to-guest communication. So if you want to use SSH to transfer files, you'd have to do that in the guest-to-host direction. Alternatively, you could let *Boxes* do it for you. If you navigate to the machine's preferences (via the three-dots menu on the toolbar) and go to *Devices & Shares*, you'll see that there's an option to enable *Shared Folders*, together with some advice for getting that to work. Let's follow that advice and run (on the VM, not the host):

```
$ sudo apt install spice-webdavd
```

Note this is actually one character more than the advice *Boxes* gives, but no matter. Return to the *Shared Folders* settings, hit the + button and you'll be able to share any folder on the host with the guest. Try sharing your home folder for example (give it a name like `hosthome` to avoid confusion). Now open up a file manager (on the guest), go to *Other Locations* and in the *Networks* section you should see the *Spice Client*



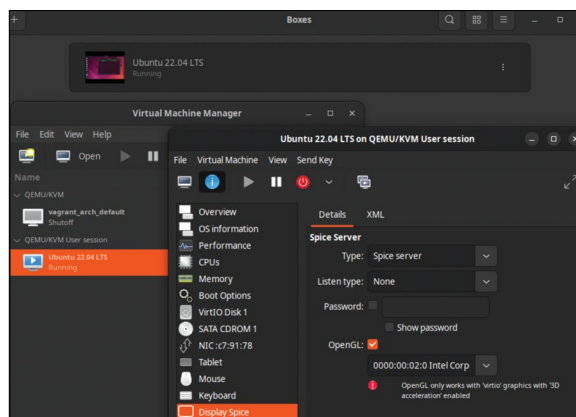
folder. Click this to mount it, and then inside you should see your home directory on the host machine. This is a simple and quick way to share files.

Spice enables this magic to happen. It's much more than shared clipboards and folders. It's actually what's used to get video and audio data out of the VM and display it on your screen (or possibly someone else's screen). It's used for remote desktop connections as well, and *Boxes* used to double as a remote desktop client. Now that functionality is handled by a separate Gnome program (*Connections*), so *Boxes* is about as simple as it can be.

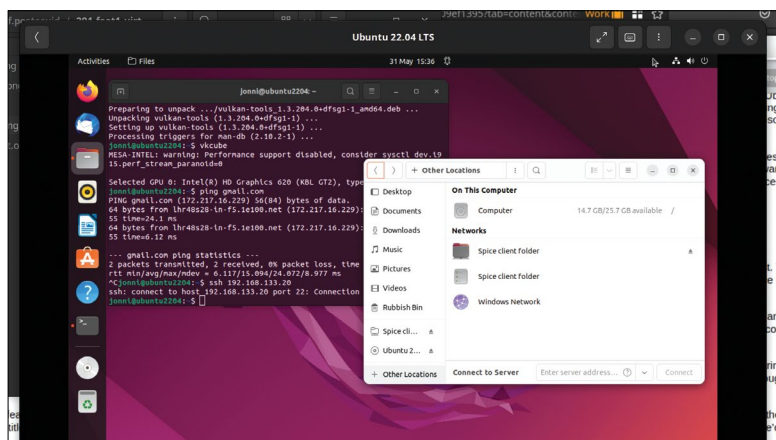
Besides sharing data with the VM, we can also share hardware. More accurately, *Boxes* makes it possible to easily pass USB devices through to the guest. This renders them useless on the host (so it's not really sharing at all). This is handy for things like USB hard drives, and if you ever go down the rabbit hole of GPU passthrough (giving a virtual machine its own actual graphics card) you'll need this for passing keyboards and mice through. More on that later.

We've actually covered most of *Boxes*'s features in this quick run-through, but we haven't yet got a satisfactory *Glxgears* experience. Recall that *Boxes* is using libvirt behind the scenes. There's a more fully featured frontend for libvirt that's imaginatively titled *Virtual Machine Manager*, or *virt-manager* for short. You can install *virt-manager* from the *Ubuntu Software* application, or from the command line with

```
$ sudo apt install virt-manager
```



Virt-manager avails us of all kinds of frobs and settings, so let's see if we can make our cogs behave better.



Close *Boxes* without shutting down the VM and then start *virt-manager* from the Activities view. You'll see a single connection labelled *QEMU/KVM*, which is the system's libvirt connection and is currently empty. *Boxes* uses the *QEMU* user session, so if we go *File>Add Connection* and select *QEMU/KVM* user session from the hypervisor menu, then our VM from before should show up (in exactly the state in which we abandoned it). You'll also see a warning that pre-existing (system) guests will be unavailable, and that networking options will be minimal. Nevermind that. Instead, open up the VM's hardware information using the second button from the left. Behold, unbridled configuration options. So let's see if we can make *Glxgears* fly.

Boxes can make portals from virtual to actual machines, all through the magic of Spice.

In the *Display Spice* section, tick the *OpenGL* box. Your GPU(s) should show up in the drop-down, along with a message that we need to enable *VirtIO* graphics. Let's do that by selecting the *Video QXL* section (agreeing to save the previous changes), changing the *Model* to *VirtIO*, and checking the *3D acceleration* setting box. Heed the advice and restart the VM for the changes to take effect. You might notice that using the *fullscreen* button no longer auto-adjusts the VM's resolution, in which case do that manually by right-clicking the desktop and choosing *Display Settings*.

Now let's see what those cogs are saying. Run *glxgears* from the terminal once again, and it should appear much smoother. And it should certainly use much less CPU usage, which you can check for yourself on both host and guest.



» KEEP TALKING WITH VIRTIO

VirtIO is part of the magic that enables faster communication between the virtual machine and the hardware. Using a technique called paravirtualisation the VM talks through the thinnest of driver layers with the actual hardware.

By default, *Boxes* uses *VirtIO* for storage and network purposes, and we've seen that we can enable it for graphics as well. Hopefully that part worked for you, but if not then we suppose it makes sense that the default

QXL virtual video card from Red Hat is still the default. Most Linux distros include the *VirtIO* drivers by default, so they should enjoy that acceleration as guests without further tweaking. But if you wanted to virtualise another OS – say, *Windows* – then these drivers need to be added manually.

Download them as an ISO file from <https://github.com/virtio-win/virtio-win-pkg-scripts/blob/master/README.md> (go for the *Stable virtio-win* offering first).

You'll have to struggle with quite a slow *Windows* VM at first, but with a bit of patience you should get the *ISO* mounted inside.

Then you can either install the drivers individually (there are about five), or (easier) run the *virtio-win-gt-x64.msi* wizard to install them all. There's an additional *virtio-win-guest-tools.exe* package that contains the *QEMU* and *Spice* guest agents (for VM control and shared clipboards).

Make use of Vagrant

Vagrancy is generally not a trait to be encouraged, but we can heartily recommend Vagrant for your VM pleasure.

Setting up virtual machines from scratch is all well and good, but if you wanted to set up a whole fleet of them (or even if you keep breaking one and having to set it up again from scratch), then some degree of automation is desirable. There are a few options here. One would be to take regular snapshots of the VMs state, so that if it breaks you can revert it to a known good state. *Boxes* has a simple Snapshot option in the options menu and *Virt-manager*'s is available from the toolbar. VMs (and snapshots of VMs) in *virt-manager* can also be cloned, so it's possible to set up several copies once you've got one set up just the way you like it.

However, none of that obviates the initial effort of setting up a VM, tweaking the OS and updating everything. So if you find yourself often in this cycle, don't fret – let *Vagrant* take care of all this grunt work instead. Hashicorp's *Vagrant* is a tool for provisioning and managing VMs, with an emphasis on setting up development environments. *Vagrant* VMs are known as Boxes (which hopefully we won't confuse with Gnome

Boxes here) and are easy to set up, either from scratch or from a prepared offering from www.vagrantcloud.com.

There are official boxes from Hashicorp and distro teams, as well as unofficial boxes from the community (available from www.vagrantbox.es). There are *Vagrant* boxes for almost any use case you can imagine, but be aware that the unofficial ones aren't vetted or guaranteed. We'd advise exercising a degree of caution when dealing with these.

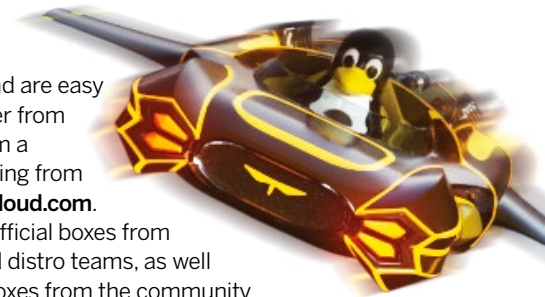
Get Vagrant up and running

Vagrant is a largely frontend-agnostic tool, in that it can create VMs for *VirtualBox*, *Hyper-V* and *Docker*. Additional providers are available as plugins. These include *VMware* (recommended where performance is critical), *libvirt* (which we're about to make use of) as well as cloud providers such as *AWS*. Our first task will be to install *Vagrant*, which on Debian-like distros (including *Ubuntu*) is achieved as follows (don't worry about the message that *apt-key* is deprecated):







```
$ curl -fsSL https://apt.releases.hashicorp.com/gpg |
sudo apt-key add -
$ sudo apt-add-repository "deb [arch=amd64] https://
apt.releases.hashicorp.com $(lsb_release -cs) main"
$ sudo apt update && sudo apt install vagrant
```

Shockingly, *Curl* wasn't present on our *Ubuntu 22.04* install, so you may need to install that first. For other distros (or cloud installs on *Amazon Linux*) see the instructions at www.vagrantup.com/downloads. *Vagrant* and its (sprawling) dependencies will only occupy about 150MB, but beware that VMs have a habit of growing and filling up your disks. So make sure that you have plenty of space on the partition which they're stored.

Vagrant's default provider is *VirtualBox*, which we're shunning slightly for this particular adventure. So let's



A box for all seasons. Whatever you need there's likely a Vagrant Box that does it.

Provider	any	virtualbox	vmware	libvirt	more	Sort by	Downloads	Recently
	ubuntu/trusty64	20190514.0.0				virtualbox	Downloads	30,592,541
Official Ubuntu Server 14.04 LTS (Trusty Tahr) builds (End of standard support)								
	laravel/homestead	12.1.0				hyperv parallels virtualbox vmware	Downloads	14,418,715
Official Laravel local development box.								
	hashicorp/precise64	1.1.0				hyperv virtualbox vmware_fusion	Downloads	6,797,457
A standard Ubuntu 12.04 LTS 64-bit box.								
	centos/7	2004.01				hyperv libvirt virtualbox vmware	Downloads	5,459,906
CentOS Linux 7 x86_64 Vagrant Box								
	ubuntu/xenial64	20211001.0.0				virtualbox	Downloads	3,495,129
Official Ubuntu 16.04 LTS (Xenial Xerus) builds (End of standard support)								
	puppet/ubuntu1404-x64	20161102				parallels virtualbox vmware_desktop	Downloads	2,510,988
Ubuntu Trusty 14.04 LTS x64								

» FURTHER VAGRANCY

We've barely touched on what *Vagrant* can do. And given nary a mention to other capable orchestration tools such as *Chef*, *Puppet* and *Ansible*, which can all work harmoniously with it.

Vagrant is quite good at taking the pain out of virtual networking (*spoken from experience?* – *Ed*). For example, if you run `vagrant ssh` from the directory where we started our Arch Linux VM

then you'll magically be SSH'd inside. Not a made-up IP address in sight.

If you want to resume your VM you can `vagrant up` it again, but that will recreate its original hardware configuration (and pull in any changes to the base image). Instead, get into the habit of using `vagrant suspend` to halt your image. After a clean boot, if you try and start a *Vagrant* box from *Virt-manager* you'll

likely see an error because *Vagrant*'s networking hasn't been set up. We can remedy this using *libvirt*'s command line powers:

```
$ sudo virsh net-start vagrant-libvirt
```

You can also use `virsh edit` to edit a machine's XML definition, for ultimate control. You can even enable this editing in *Virt-manager* and *Boxes* itself, as we'll see over the page.

get the libvirt provider installed, so that we may continue to use *Virt-manager*:

```
$ vagrant plugin install vagrant-libvirt
```

Arch Linux has a reputation of being tricky to install and, once installed, the distro takes days to set up to one's liking. Without commenting on the veracity of this assertion, let's get the official *Vagrant* Arch box spun up in mere seconds:

```
$ mkdir vagrant_arch
```

```
$ cd vagrant_arch
```

```
$ vagrant init archlinux/archlinux
```

You'll see a message saying that a **Vagrantfile** has been set up in this directory, and that you're now ready to go. Have a look at said **Vagrantfile** if you want (it's somewhat minimal because we're using an off-the-shelf configuration, but there are some useful comments to help you understand how to tweak things). Then activate the VM with the following:

```
$ vagrant up
```

After a few moments' setup and a request for a password you'll be ready to go. Or hopefully. You might instead see a message about the NFS version or transport protocol not being supported. This happened to us, and if it happens to you too then edit (as root) **/etc/nfs.conf**, adding the line **udp=y** in the **[nfsd]** section. Then run:

```
$ sudo systemctl restart nfs-server
```

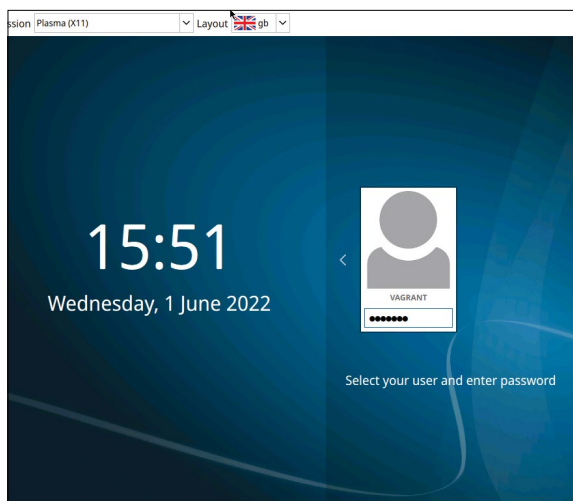
to effect the changes, and try again.

Access the Plasma desktop

All being well you should see a new VM appear in *Virt-manager* (in the standard KVM/QEMU connection this time, rather than the user session) called **vagrant_arch_default**. And if you open it up, you can log in with using **vagrant** as both username and password. As you can see, it's quite a lightweight VM, but there's no need for it to stay that way. Let's install the barebones KDE Plasma package and see how easy it is to achieve a fully featured GUI.

```
$ sudo pacman -S plasma-desktop
```

The installation will ask you lots of questions – everything from audio systems to fonts – and there aren't any right or wrong answers, so choose what you like. *Pacman* will then jump into action and set up almost everything we need for our desktop. We still need some



! You'll need your best Arch-fu to install SDDM themes from the AUR.

```

[~] bash — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Copy Paste Find

[vagrant@archlinux ~]$ neofetch

vagrant@archlinux
-----
OS: Arch Linux x86_64
Host: KVM/QEMU (Standard PC (i440FX + PIIX, 1996) pc-i440fx-jammy)
Kernel: 5.18.0-arch1-1
Uptime: 2 mins
Packages: 505 (pacman)
Shell: bash 5.1.16
Resolution: 1280x1024
DE: Plasma 5.24.5
WM: KWin
Theme: [Plasma], Adwaita [GTK3]
Icons: [Plasma], Adwaita [GTK3]
Terminal: konsole
CPU: Intel (Skylake, IBRS) (1) @ 2.903GHz
GPU: 00:02.0 Cirrus Logic GD 5446
Memory: 265MiB / 466MiB

```

more (fairly critical) packages, in the form of a display (login) manager, X.org itself and a terminal emulator:

```
$ sudo pacman -S sddm xorg-server konsole
```

There are still some important bits missing, but we're now in a position to start and login to our desktop with:

```
$ sudo systemctl start sddm
```

You'll notice that only the US keymap is available at the login screen, and that SDDM's default theme is rather spartan. Once logged in, you'll see that very few applications are installed, and that some shortcuts are missing their applications. It's quite hard to remedy this without a terminal, so it's just as well that we installed *Konsole* earlier. We can fix our keymap by editing **/etc/locale.gen** (you may wish to install *Nano* too) and uncommenting the line **en_GB.utf-8** (or whichever keymaps you require) and running:

```
$ sudo locale-gen
```

```
$ sudo localectl set-x11-keymap gb
```

You'll also need to update **/etc/locale.conf** with the new locale (but really you should read the Arch Wiki for all these set up details). We'll leave you to find some nice themes for SDDM. If you want to auto start the GUI on each boot then run:

```
$ sudo systemctl enable sddm
```

By this point you've probably noticed that our VM is much slower than our simple *Gnome Boxes* box. But don't lose hope. We're using the emulated Cirrus video card and have zero graphical acceleration. Fix this by shutting down the VM, going to its Hardware Information page, selecting Add Hardware>Graphics and setting up a Spice Server. As before, we need to check the OpenGL box and make sure that Address Type is set to None.

Now click Finish and navigate to the Video Cirrus section. Change this to VirtIO and check the 3D acceleration box. You might have also noticed that the **Vagrantfile** only assigned 512MB of memory to our box. This certainly will not do, so change this to something bigger.

Behold, Arch Linux with KDE Plasma, set up in no time and with no cursing, either. Unreal. Virtual.



Running a virtual GPU

Discover how to harness Intel's GVT-g technology to virtualise your GPU.

We've seen that virtualisation (with appropriate hardware support) is much faster than conventional emulation. And we've seen that when using paravirtualised VirtIO devices we can speed that up even more. But we can go further. What if, for example, we gave a virtual machine its own physical graphics card?

This technique, known generally as VFIO, has been around for a while. For the particular case of using PCI passthrough with graphics cards, the result is that VMs can run graphically intensive applications to within a hair's breadth of native speeds. This enables Linux users

TAKING THE VFIO APPROACH

“Using PCI passthrough with graphics cards, VMs can run graphically intensive applications to within a hair's breadth of native speeds.”

to run Windows VMs and play games without taking a performance hit. This is an alternative to *Proton*, albeit one which is a little tricky to set up and requires that the host machine has (at least) two graphics cards. We covered this in **LXF261**.

This time around we'll look at a slightly different technique for users of Intel hardware. This is known as vGT and enables your actual GPU to be segmented into virtual GPUs that behave exactly as though they were connected via PCI passthrough. Equivalent technologies do exist for Nvidia and AMD cards, although they're not commonly found on their consumer cards.

The first thing you need to do is ensure that your onboard graphics are supported, which requires at least a Haswell (sixth-generation) processor. Then we need to

Our VM guest now has the same model of GPU as the host, but the host is still displaying graphics. Amazing.

change some kernel options. Start by running the following instruction:

```
$ sudo nano /etc/default/grub
```

and find the line that looks like

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
```

The right-hand side may not be identical, depending on your setup, but no matter. Add the parameters:

```
intel_iommu=on i915.enable_guc=0
```

before the closing quote. Then save, exit and run:

```
$ sudo update-grub
```

to update the bootloader.

Next, we need to change the GPU module options. Edit (again using *sudo*) the file `/etc/modprobe.d/i915.conf` and add the following line to it:

```
options i915 enable_gvt=1
```

We need to also update the `initramfs` so that the `i915` driver is loaded (and respects our settings) early in the boot process. On Ubuntu this involves adding `i915` to `/etc/initramfs-tools/modules` and then regenerating with the following:

```
$ sudo update-initramfs -u -k all
```

Finally, we need to ensure some modules are automatically loaded at boot. Create a new file with:

```
$ sudo nano /etc/modules-load.d/gvt.conf
```

and populate it with:

```
kvmgt
```

```
vfio-iommu-type1
```

```
mdev
```

Now reboot so that the changes take effect.

Graphics card details

We need to find the domain number and PCI address for our graphics card. This is to be found in the output from the command:

```
$ lspci -D -nn
```

Look for the numbers before `VGA compatible controller` of the form `0000:00:02.0`. The first six digits are the domain number and, confusingly, the whole thing is the PCI address. If we now run the following (replacing the domain number and address as appropriate, noting that colons must be escaped with backslashes, and that tab-completion is your friend):

```
$ ls /sys/devices/pci0000\:\00\0000\:\00\:\02.0/mdev_supported_types
```

then you should see a few directories. Each of these represents a specific virtual GPU configuration, which you can find out about by looking at the `description` files within, for example:

```
$ cat /sys/devices/.../mdev_supported_types/i915-GVTg_V5_4/description
```

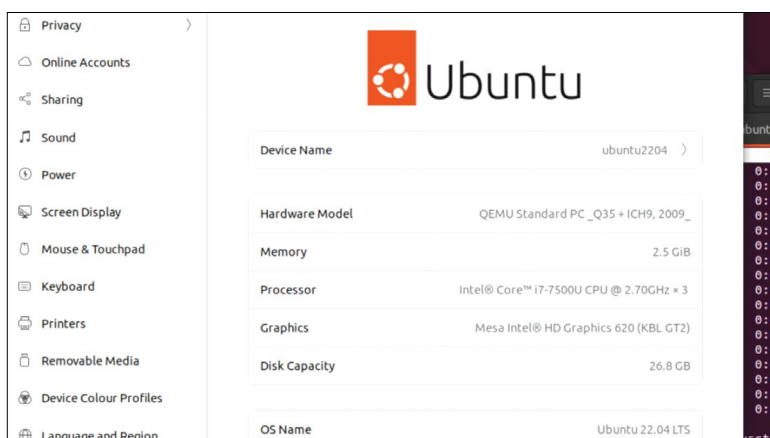
```
low_gm_size: 128MB
```

```
high_gm_size: 512MB
```

```
fence: 4
```

```
resolution: 1920x1200
```

```
weight: 4
```



In general the smaller the final number, the more resources the vGPU will have.

Next, we need a Globally Unique Identifier (GUID), which is a string of 32 hex digits. You can generate a random one (and store it in a variable) by running

```
GVT_GUID=$(uuidgen)
```

or by typing `guid` into *DuckDuckGo*. Since GUIDs are essentially 128-bit numbers, there's a very low probability of a collision between randomly generated ones (which does mean that they're very likely unique, rather than actually unique).

We'll use this GUID to create a new vGPU. For ease of reading (and consistency with the Arch Wiki without which this section wouldn't be possible), we'll also substitute `$GVT_DOM`, `$GVT_PCI` and `$GVT_TYPE` for the various identifiers we've come across so far. For clarity these might look like:

```
GVT_DOM=0000:00
```

```
GVT_PCI=0000:00:02.0
```

```
GVT_TYPE=i915-GVTg_V5_4
```

Now we use these variables so that the below command is easier to type:

```
$ echo $GVT_GUID | sudo tee /sys/devices/pci$GVT_DOM/$GVT_PCI/mdev_supported_types/$GVT_TYPE/create
```

All going to plan, this will have created a new device. Let's check this by looking at the PCI bus:

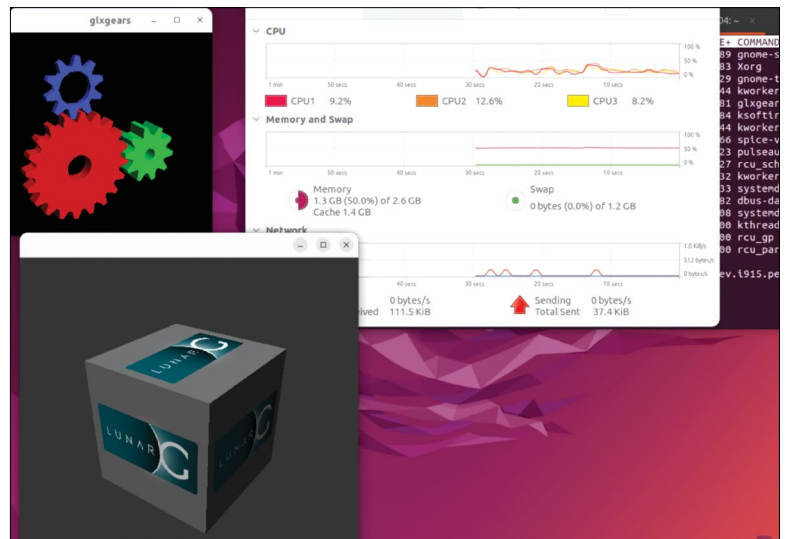
```
$ ls /sys/bus/pci/devices/$GVT_PCI
```

There should be a subdirectory matching the `$GVT_GUID` generated earlier. It's possible to create more vGPUs by repeating the process with different GUIDs, though precisely how many is limited by your video RAM. You can remove them by echoing `1` at the `remove` node inside the subdirectory.

In order to have a libvirt virtual machine use the virtualised GPU, we need to tweak its configuration. This is quite easy to do with *Boxes* (Edit XML), so we'll go back to the first VM we set up there and endow it with graphical superpowers. We must add the following stanza inside the `<devices>` section (and replace `GVT_GUID` appropriately, remembering the quotes).

```
<hostdev mode="subsystem" type="mdev"
managed="no" model="vfio-pci" display="off">
<source>
<address uuid="GVT_GUID"/>
</source>
</hostdev>
```

When you start the virtual machine it probably won't work, and if you study the logs you'll see it's because of

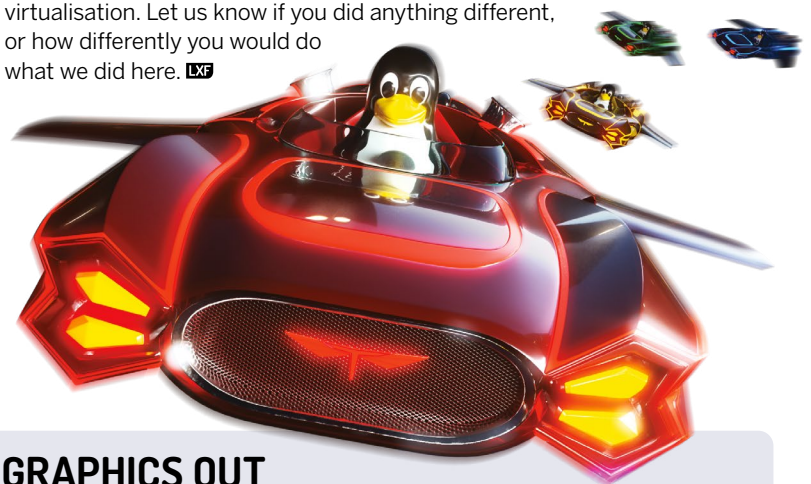


a rather parochial permissions error. You can fix this with the following:

```
$ sudo chmod o+rw /dev/vfio/0
```

You might have noticed that *Boxes* makes it possible to have our VM boot via EFI, using files from the Open Virtual Machine Firmware (OVMF) project. This needs to be done when the machine is first created, otherwise we'd end up with a paradoxical VM. We've stuck with a classic BIOS setup here, partly because some GVT-g setups currently don't work with UEFI guests. If you want a UEFI virtual machine then you'll need to install the `ovmf` package. The reason why we haven't bothered with this is that this particular brand of GPU virtualisation does require extra effort to work with UEFI guests.

And so concludes our glorious foray into virtualisation. Let us know if you did anything different, or how differently you would do what we did here. **LXF**



Now we have lots of spinning things and less CPU activity, thanks to a little Intel magic.

» ACTUALLY GETTING VIRTUAL GRAPHICS OUT

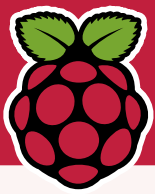
There's a lot that can go wrong with all of this so don't be disheartened if it doesn't work. Also, we haven't quite finished because we need to tell libvirt where to display the graphics. There are a number of options here. The most performant would be to not use *Boxes*'s (or *Virt-manager*'s) Spice display, but that brings its own problems. It's also possible to use a remote viewer, such as *Looking*

Glass, but that's CPU-intensive. We'll instead leverage Spice with EGL, which is a reasonable compromise and enables us to keep using *Virt-manager*'s viewer.

The first thing we need to change is the rather telling `display="off"` in the `hostdev` line we added before. Then we need to remove the remaining graphics and video devices in the XML, and replace them with

```
<graphics type="spice">
<listen type="none"/>
<gl enable="yes"/>
</graphics>
<video>
<model type="none"/>
</video>
```

If you run into problems, check out the relevant Arch Wiki page at https://wiki.archlinux.org/title/Intel_GVT-g.



Kevin McAleer loves bringing robots to life. Find him on Instagram and Twitter (search for @kevsmac).

» 3... 2... 1... ACTIVATE!

I'm Kevin McAleer and I make videos about robots (<https://youtube.com/kevinmcaleer28>). Each week I invite the audience to learn with me via my blog (<https://smarsfan.com>) as I build robots and then bring them to life with code.

I love the Raspberry Pi and bought my first in 2012. What I hadn't thought about at the time was how important the GPIO pins would be for this device. It would take a couple of years for me to master Python and connect the Pi to real-world sensors, motors, servos and lights. Ultimately, this led to me creating robots! I now create new robots every week for my growing YouTube channel.

I use Raspberry Pis in most of my projects, such as The Clustered-Pi. This is a Raspberry Pi Zero Cluster modelled on the famous Cray-1 Supercomputer (<http://clustered-pi.com>).

Another of my projects is Explora, a Raspberry Pi Zero-powered robot. I've also been working on a build-your-own-AI-Assistant in Python. Finally there's PicoCat, which is a Raspberry Pi Pico-powered cat robot that uses servos to move around.

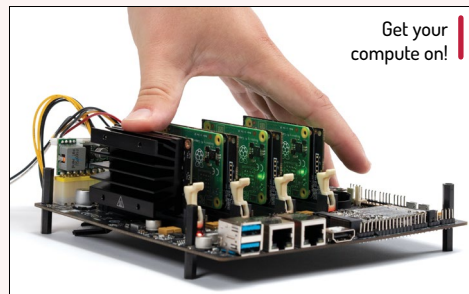
I choose the Raspberry Pi because they're cheap (£5 for a Zero), and there's plenty of documentation, tutorials and videos available for free. Unfortunately, they're a bit harder to come by now, but you'll get one soon enough if you're passionate about owning one.

I also love the Raspberry Pi Pico because it runs MicroPython and makes it very easy to build and program robots!

Turing 2 Pi raises \$1 million in a day

The Pi-based cluster computing Kickstarter project smashes its own targets for full funding.

Cluster computer Turing Pi 2 has smashed its modest funding target of \$64,000 by a country mile. It cruised through the \$1 million barrier in just one day, ending up with a final total over \$1.5 million. The Turing Pi 2 is a four-node mini ITX cluster board with a built-in Ethernet switch that runs Raspberry Pi CM4, a new Turing RK1 module or Nvidia Jetson compute modules in any combination.



CREDIT: Turing Pi

The new Turing RK1 features the Rockchip RK3588 octa-core processor and up to 32GB RAM. The RK3588 sports four Arm Cortex-A72 cores and four A55 cores. It also carries a neural processing unit (NPU) that's capable of six TOPS, making such a cluster more attractive to the machine-learning crowd.

The modules are connected via a Gigabit Ethernet switch that's built into the mini-ITX board, with enough bandwidth left over for a pair of Gigabit RJ-45 ports on the edge of the board. These are complemented by a pair of USB 3 ports (with more available via a header), two mPCIe sockets including one with a SIM card slot for connecting third-party extension boards such as wireless networking or home automation transmitters, two SATA III ports, an HDMI and a MIPI DSI header. There's a 40-pin GPIO header, too.

Find out more at www.kickstarter.com/projects/turingpi/turing-pi-cluster-board.

Sensor Pi

Fully loaded RP2040 unit.

Sferalabs has released a RP2040-based sensor array packing temperature, humidity, air quality, light, movement, audio, plus expandable digital I/O with edge-computing networked features. It's all packed into a 8x8cm, fully programmable package. See www.sferalabs.cc/product/exo-sense-rp.



CREDIT: Sferalabs

Education time!

Join the launch.

As reported in **LXF280** the joint Raspberry Pi Computing Education Research Centre, in conjunction with the University of Cambridge, is set to launch on 20 July. If you're interested in computer education research you can register for an in-person ticket! Get yours by visiting <https://bit.ly/lxf291centre>.



CREDIT: Christian Richardt, CC BY-SA 3.0, https://commons.wikimedia.org/wiki/File:University_of_Cambridge_Computer_Laboratory.jpg

Waveshare CM4-NANO

Les Pounder looks at Waveshare's Compute Module 4 carrier board that offers Raspberry Pi 4 performance in a near Zero-sized package.

IN BRIEF

The CM4-NANO-B is designed for the Compute Module 4 and provides impressive performance in a small package. The connectivity is curated, although we're missing USB3 and multiple CSI/DSI connectors. The Raspberry Pi GPIO is squeezed into the board and this is most welcome for electronics projects.

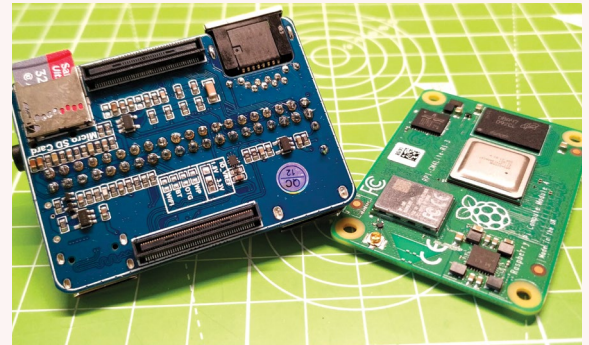
The Compute Module 4-NANO-B is an enticing prospect. It's roughly the same size as a Raspberry Pi Zero 2 W, but with the power of a Pi 4. Disposing of the SODIMM form-factor found in previous modules, the CM4 has connectors for HDMI, GPIO, power and storage. To get the most from the CM4 you need a breakout board. Waveshare's CM4-NANO-B is a breakout board for the CM4 that provides many of the features found in the Raspberry Pi 4, but at half of its size.

The CM4-NANO-B is designed to accept the CM4 via connectors on the rear of the board. It works with all variants of the CM4, including the CM4 Lite. The key difference? The Lite lacks the onboard eMMC storage, but the CM4-Nano-B hides a micro SD card slot sandwiched between the CM4 and the board. Speaking of ports, the CM4-Nano-B has a USB C port for power and data, analog audio output, mini HDMI, USB 2.0, and CSI and DSI slots for the camera and official display. In the middle of the board is the GPIO. Yes, Waveshare has managed to squeeze in the full Raspberry Pi GPIO. One caveat though: you'll need to solder it yourself.

Getting up and running

We connected up a CM4 with 4GB of RAM and 32GB eMMC to the board, flicked the BOOT switch to ON and fired up our PC to write Raspberry Pi OS to the flash storage. The CM4 requires a carrier board to flash the storage, and we were easily able to do this, with a few tweaks. The online instructions are for Windows – not ideal – but we followed the official Raspberry Pi guidance and all was good.

Another tweak is to enable the USB port (which isn't enabled as standard for the CM4) and enable the CSI/DSI. All of these tweaks were made in `config.txt`. Ten minutes later and we were at the Raspberry Pi OS desktop, with everything behaving as if we were sitting at a typical Raspberry Pi. Ethernet worked, HDMI was correctly detected on our 1440p monitor and USB worked, albeit at 2.0 speeds.

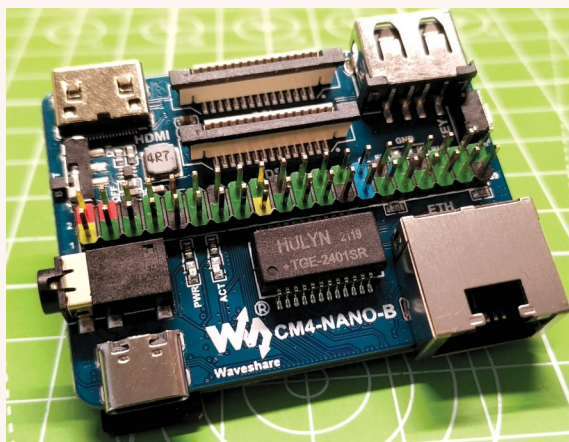


The two connectors enable the CM4 and CM4-Nano-B to communicate with one another. Sandwiched between them is a micro SD card slot.

Moving to the camera, we had to follow another tweak on Waveshare's website to enable the camera, but only with legacy support. In recent releases, Raspberry Pi OS has started migration to a new **libcamera** driver for its official cameras. In the past we would use **raspistill** and the PiCamera Python library for stills/videos. We have to use this again for the CM4-Nano-B, which is disappointing but not a deal breaker. Analog audio is another tweak, and involves using GPIO 18 and 19 for audio output. This means that any HATs using these pins won't work correctly unless HDMI audio is used.

Talking of GPIO, there are the full 40 pins present, but you'll need an extension for the pins to ensure your HAT clears the Ethernet jack. That aside, squeezing the GPIO into such a small board is a feat. After soldering up the headers we tested a few HATs (Explorer HAT Pro, Sense HAT and Display-O-Tron) and all of the HATs worked. The hiccup was two of our Sense HATs refused to work with the CM4 and 32-bit Raspberry Pi OS. Yet they both worked with a CM4 Lite running 64-bit OS. Strange.

Waveshare's CM4-NANO-B is a great choice for those who want Pi 4 power in a small package. We can manage without USB 3.0. If you're after an alternative to the official carrier board, or want to squeeze a Pi 4 into the smallest of projects, add this to your shopping list. **LXF**



This is an extremely packed board, but we have everything we need in a board that matches the CM4's rather neat footprint.

VERDICT

DEVELOPER: Waveshare

WEB: www.waveshare.com/wiki/CM4-NANO-B

PRICE: £17

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

After tweaks, this is a solid performer. Its key draw is the Pi 4 performance in a slightly-larger-than Pi Zero 2 W form factor.

» **Rating 8/10**

DISPLAYOTRO

Credit: <https://github.com/pimoroni/displayotron>

Create a custom LCD menu system

Les Pounder loves to get hands-on with a good button, and if doing so can save him time and looks great, then it's a win-win situation!



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 bigles.com.

LCDs are great for quick and simple information feeds, and one particular model, Display-O-Tron from Pimoroni is especially fun. An easy-to-use Python library enables us to create menus, stream text and generate colourful backlight effects with very little code. Around the board are six capacitive touch buttons that make up a simple user interface. We're going to use this board to create a device that will open websites without touching the keyboard.

Screw the M2.5 standoffs into the two holes under the LCD screen. With the Raspberry Pi powered off, place the Display-O-Tron HAT on to all 40 pins of the GPIO. Now connect all of your accessories and power up the Pi to the desktop.

Hardware installation is simple. Open a terminal and type the following to use Pimoroni's automated installer. During the installation choose to do a full install, as later we'll need a directory from **Examples**.

```
$ curl -sS get.pimoroni.com/displayotron | bash
```

The installer will set up the hardware and software, but during our installation (the latest 32-bit Bullseye release on a Pi 4 8GB) we had to manually install a Python library for the touch interface. The installer used an older version of *Cap1xxx*, yet requires version 0.1.4. We've reported this issue to Pimoroni.

Open a terminal and use *pip* to install version 0.1.4

```
$ sudo pip install Cap1xxx==0.1.4
```

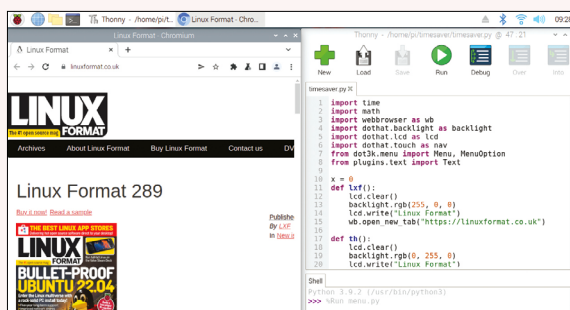
Make a new directory in your home directory called **timesaver** and change directory so that you're in it.

```
$ mkdir timesaver
```

```
$ cd timesaver
```

YOU NEED

- > Any Pi
- > Raspberry Pi OS
- > Pimoroni Display-O-Tron HAT
- > Two M2.5 standoffs and screws
- > Code: <https://github.com/lesp/LXF-DotHAT-Timesaver/archive/refs/heads/main.zip>



Thonny is the default Python editor for Raspberry Pi OS. It comes in two modes, a simple version (as per the image) and a more advanced version. You can easily change this as your skills grow.



It may be an old board, but Display-O-Tron HAT still looks great. The RGB backlight is a delight for such a display, while the touch interface is a neat addition to the board.

Copy the plugins directory from `~/Pimoroni/displayotron/examples/dotthat/advanced/` to your **timesaver** directory.

```
$ cp -R ~/Pimoroni/displayotron/examples/dotthat/advanced/plugins/ ./
```

With the installation complete we can now start writing the project code. Open the *Thonny* Python IDE, found under the Programming menu. Our first task is to import a number of Python modules to make our project work. The first three are **time**, **math** and **webbrowser**. **time** is used to control the pace of the project, **math** will be used to create an appealing backlight effect, while **webbrowser** is used to open web browsers or tabs.

```
import time
import math
import webbrowser as wb
```

The next group of imports handle Display-O-Tron's RGB backlight, LCD display, user interface and handle creating a text-based menu.

```
import dothat.backlight as backlight
import dothat.lcd as lcd
import dothat.touch as nav
from dot3k.menu import Menu, MenuOption
from plugins.text import Text
```

We next create a variable, **x**, which stores an integer value of 0. This variable will later be used for the backlight effect.

```
x = 0
```

We now need three functions – objects that run a sequence of code when we call their names. These functions will be called when we select the corresponding menu item. The first is called `lxf()`.

def lxf():
The code inside of the function is indented, to show that it belongs to the function. First we clear the LCD screen, and then set the RGB backlight to red (255,0,0).

```
lcd.clear()
backlight.rgb(255, 0, 0)
```

We then write “Linux Format” to the LCD screen before opening a new web browser tab to the *Linux Format* website. Note that this will open a new tab in an existing browser. If there’s no open browser, it’ll open the system default browser.

```
lcd.write("Linux Format")
wb.open_new_tab("https://linuxformat.co.uk")
```

We now create another function, this time called `th()`, with a green backlight (0,255,0) and it opens a new browser tab to the *Tom’s Hardware* site.

```
def th():
    lcd.clear()
    backlight.rgb(0, 255, 0)
    lcd.write("Linux Format")
    wb.open_new_tab("https://tomshardware.com")
```

A final function, `bgl()`, sets the backlight to blue (0,0,255) and opens a tab to the author’s website (<https://bigl.es>).

```
def bgl():
    lcd.clear()
    backlight.rgb(0, 0, 255)
    lcd.write("Linux Format")
    wb.open_new_tab("https://bigl.es")
```

We now move on to constructing the menu. We shall create an object, `menu`.

```
menu = Menu()
```

The menu has three options, one for each of the functions we’ve just created. Each menu item needs text that will appear in the menu, and then the name of the function that we wish to use. The structure of the menu is stored in a Python dictionary, and the key for each item in the dictionary is the menu text. The value returned when using that name are our functions.

```
menu = Menu(
    structure={
        "Linux Format": lxf,
```



```
        "Tom's Hardware": th,
        "Bigles": bgl
    },
```

Finishing off the menu construction we configure the LCD, and set an idle timeout of 30 seconds which will automatically run the *Linux Format* function.

```
lcd=lcd,
idle_handler=lxf,
idle_timeout=30,
input_handler=Text()
```

The last line of config sets up the touch interface using the default bindings set by the Python module.

```
navbind_defaults(menu)
```

The final section of code, and here is where we use a `While True` loop to continually check for user input, and run a special backlight animation. We redraw the menu, ensuring that it’s updated every 0.05 seconds.

```
while True:
    menu.redraw()
    time.sleep(0.05)
```

The value of variable `x` is updated by 1 and then used in a function called `sweep`. This will create a plasma rainbow effect using a little math involving the value of `x`. We then pause for 0.01 seconds before the loop repeats.

```
backlight.sweep((x % 360) / 360.0)
time.sleep(0.01)
```

Save the code as `timesaver.py` inside the `timesaver` directory that we created earlier. Click Run to start the code and then use the up and down buttons (left of the LCD) to select the option. Press the centre button (just above the HDMI ports) to select and run the code. A new browser window should appear. The backlight animation will continue to run as we use the project. **LXF**

The up and down buttons scroll through the list of menu options, and the centre button is used to select the website that you wish to visit.

QUICK TIP

The RGB backlight uses a tuple (a non-editable data storage object) to store the intensity values of red, green and blue light. By mixing these values between 0 and 255 we can reproduce a great deal of the spectrum.

» LCD DISPLAYS

There’s no shortage of LCD displays for the Raspberry Pi, Arduino and other maker boards. We’ll start at the top end of the spectrum, with IPS displays. These are still LCDs, but they have rich and colourful panels that look great for media projects. The problem is that they’re more expensive, and quite fragile so should be kept away from heavy hands. They can also eat a large number of GPIO pins, so plan wisely.

In the middle we have single-colour OLED displays. These panels are vibrant, clear and more robust. They come in I2C, and SPI versions, so they don’t consume too many pins and they can be driven via a microcontroller. They are cheaper than IPS, but lack the wow factor of fast moving media with high colour content.

At the bottom of the spectrum is the humble LCD: 16 columns by two rows, 20 by four, and so on. These screens are

built to last, hence they’re used in vending machines and industrial projects. Typically, these displays connect using a large number of GPIO pins (HD44780 we’re looking at you) but with an I2C “backpack”, a board that sits between the display and the I2C bus of your chosen device, we drop the pins down to four. Here’s an example of one in use:

www.tomshardware.com/how-to/lcd-display-raspberry-pi-pico.

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

SYSTEMD

Replacing rc.local and using Systemd

Sean Conway is struggling with change – specifically, what system administration knowledge to keep and what to throw away.



OUR EXPERT

Sean Conway is a former IT security specialist for a national telecoms company who uses Pi-based projects to get his much-needed technology fix now that he's finally retired.

When was the last time you changed? Okay, that question was pretty ambiguous. Let's try again. As a systems administrator, you need to decide what knowledge is worth keeping and what force or energy you need to replace old knowledge? For you as a senior administrator, with years of support knowledge for systems that are still in service, having to decide what knowledge stays and what goes it can be extremely difficult.

If you've been using Raspberry Pi's single board computers since they were first introduced in 2012, then change should come as no surprise. The single-board computer hardware has gone through several revisions as it offers more and more features. If you've been using the Raspbian operating system (OS), new releases have brought in their changes. In 2022, the OS changed its name to Raspberry Pi OS. Under the new name, it also released a beta 64-bit version.

Operating system upgrades resolve known problems, improve existing offerings or add new features or enhancements. Release notes are an extremely important piece of documentation. When upgrades make changes that render old knowledge obsolete, it's important for users to gain the new knowledge before it's needed.

Let's look at two changes that affect your ability to administer your Raspberry Pi. The aim is not to argue the merits of the changes, but to raise awareness so your knowledge can be updated, hopefully minimising the impact. The first is the Advanced User Options

Password change will be forced if either Enable SSH or Set Username is selected.

added to the *Raspberry Pi Imager* software and the second is changes to the Raspberry Pi OS that removes support for `/etc/rc.local`.

A new imager

Are you an administrator who installs headless (in other words, no attached keyboard, video monitor or mouse) Raspberry Pis? Any modification, additions or deletions to software or configuration changes to the OS are achieved through a console via some type of connection. If the connection is wireless and the communication protocol is SSH, then the services would need to be enabled before the operating system starts or there is no connection.

To enable the services, change the operating system configuration. Wireless network support and enabling SSH are two services that can be enabled by installing specific configuration files in the `/boot` directory of the SD card before the first boot. This tells the OS, "Hey I want these services available on the first boot." Placing the files in the correct place enables the OS to accommodate them.



If Raspberry Pi OS is selected then the gear icon will appear.

In release v1.6 of Raspberry Pi, *Imager* software developers started supporting Advanced User Options, which were accessible using Ctrl+Shift+X. Some of the Advanced User Options make the changes before the first boot, to get services enabled. *Imager* v1.7.2 release continues to enhance the Advanced User Options. It now provides a gear icon option rather than the three-key salute.

/etc/rc.local was a file system that administrators could use to execute scripts after all the typical operating system service had started. It could start custom services, under superuser privileges, without requiring a login. Raspberry Pi OS is derived from Debian, which no longer provides support for System V init that **/etc/rc.local** was predicated on. The move to Systemd over System V init is still creating division some 12 years since it was first introduced. What is an administrator to do for running custom scripts after bootup if **/etc/rc.local** is not available?

Launching the *Imager* application presents the user with three buttons. You select the OS, pick the storage device and then have the software write to the SD card. After completing the Operating System selection, the software screen presents a gear icon in the lower right that has no label. This is used to select Advanced User Options for the Pi OS. Clicking the gear takes us to Image Customization options. These can be set to apply for this session only or to always be used. Because each customisation option radio box is selected, we can change sub-options. If the radio buttons Enable SSH or Set username and password are selected, they require you to change the default password. Once we've completed the option selections, clicking SAVE takes us back to the main screen. With the storage selection complete, the write button becomes enabled to write an image to the SD card.

Imager provides the ability to select the most current Raspberry Pi OS – there's no need to check what's available. With an internet connection, the software finds the most current version and applies it to the SD card. No need to download images and keep track of the version numbers. There are some options available for selecting legacy versions as well.

Choosing Advanced User Options removes the need to install files in the **/boot** directory in order to enable

```
pi@piTL:~$ systemctl list-units sshswitch.service --state=inactive
UNIT                                LOAD    ACTIVE SUB    DESCRIPTION
sshswitch.service loaded inactive dead Turn on SSH if /boot/ssh is

LOAD = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of
SUB = The low-level unit activation state, values depend on unit t
1 loaded units listed.
To show all installed unit files use 'systemctl list-unit-files'.
```

SSH and wireless access. It reduces the need to use the Raspberry Pi OS configuration tool *raspi-config* to make changes. Advanced User Options reduces the level of knowledge that's required for someone to prepare an SD card for a Raspberry Pi. Reducing the knowledge level required to configure a Raspberry Pi is a good thing if we want to continue to encourage individuals to take part.

Let's reboot the tutorial for a look at another change. Now we will invest a few cycles in Systemd, the mechanism that brings the change.

Systemd

This will not be an exhaustive look at Systemd. Previous tutorials have done this (see [LXF199](#), [LXF191](#), [LXF188](#), [LXF184](#) and [LXF167](#) for example) and will continue to be written on the topic (*Ooh, now there's an idea... – Ed*). The aim is to gain some knowledge to replace that which has changed. The supporting website describes Systemd as a suite of basic building blocks for a Linux system.

Systemd brought with it the concept of Systemd units. These units are resources (see *boxout, below*) used within the system. A unit configuration file represented them. A unit file is a plain text file. The name of the file typically reflects the resource it manages. We categorise Systemd units according to the type of resource they manage. The suffix appended to the unit filename can determine the unit file type. For this tutorial we'll focus on the suffix **.service** type.

A service unit provides the information to manage a service or application that's to run on the operating system. The file holds the dependencies that are required for the operation of the service. It provides details on how to stop and start the service. This includes automation if used. A service unit file is what we'll need to create and install, in order to replace the entries in the **/etc/rc.local** file. Before starting the unit

Systemd service watching for an ssh file drop.

QUICK TIP

Watch the video on installing Raspberry Pi OS using Raspberry Pi Imager at www.raspberrypi.com/software.

» YOU WILL FIND THE ANSWER IN THE SUFFIX

Systemd categorises units according to the type of resource they describe. Look to the type suffix appended to the end of the resource name to determine the type of unit here are a few key examples you'll likely encounter:

- > **.automount** unit file describes a mount point that will be automatically mounted.
- > **.device** unit file describes a device that is recognised by the kernel and has been designated as requiring systemd management.

- > **.mount** unit file describes a file system mountpoint to be managed by Systemd.

- > **.path** unit file describes a path to a file or directory in the system.

- > **.scope** unit files are created automatically by Systemd from information that's received through external processes.

- > **.slice** unit files describe hierarchically organised Linux Control Group nodes that manage system processes.

- > A **.snapshot** unit is a saved state that's

automatically created by the `systemctl snapshot` command.

- > **.socket** unit files describes a network of inter-process communication sockets.

- > **.swap** unit files describe the swap device or swap space on the system.

- > A **.target** unit file describes a group of systemd units.

- > A **.timer** unit defines a timer that will be managed by Systemd, in a similar manner to a cron job for delayed or scheduled activation.

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



QUICK TIP

Details on Imager software releases and notes are at <https://github.com/raspberrypi/rpi-imager/releases>.

file build, let's become familiar with some software tools that will be needed by looking at some service files that already exist.

Get some Systemctl

Systemctl is a Linux command-line utility for the management of Systemd. The utility is the interface to communicate with Systemd and perform operations such as viewing the status, start, stop, restart, enabling, or disabling system services. *Systemctl* options that just list Systemd units can be performed by non-root users; options that make changes will require root user level access (such as `sudo`).

For example `sudo systemctl list-units -all` will provide a column-based output listing of all the unit files available on the system. Press the Space bar to step through full screens of data while pressing the letter q will quit *Systemctl*. At the bottom of the command output there's a brief description of the columns.

The command has attributes that can be added to filter the output. To show all installed unit files, use the command `systemctl list-unit-files`. That's a lot of information! Let's narrow the search down to `.service` type files because we plan to create one later:

```
$ systemctl list-units --type=service
```

We can write the command with shorter options (`systemctl list-units -t service`). This can lead to some confusion for individuals just starting out. We advise avoiding using the shorter form until you're more familiar with the command. When multiple attributes are appended to the command, the longer form improves readability and for a novice, that's important.

Recall we used the *Imager* software Advanced User Option to start SSH on the first boot. The options menu replaced the cumbersome method of installing a blank file with the name SSH in the `/boot` directory of the SD card in order to have the operating system turn on SSH on the first boot. The detection of the blank `ssh` file in the `/boot` directory was done by a Systemd service. Since Systemd detected no file on boot, the service state is inactive.

For this next Systemd tutorial exercise, let's assume there's a python script (`mypython.py`) in the user Pi's home directory that we would like to run when the Pi boots up. Since unit files are at the core of Systemd let's create the file to hand over starting the python script.

Use your favourite text editor to create the `local-mypython.service` unit file in the `/etc/systemd/system/` containing the following entry:

```
[Unit]
```

```
Description=local user python script run on start-up
After=multi-user.target
```

```
[Service]
```

```
Type=idle
```

```
User=pi
```

```
ExecStart=/usr/bin/python3 /home/pi/your-Python-script.py
```

```
Restart=always
```

```
[Install]
```

```
WantedBy=multi-user.target
```

We find unit files from distributed packages in `/usr/lib/systemd/system`. We find unit files that aren't packages in the `/etc/systemd/system/` directory.

The `[Unit]` section provides basic information about the service and condition dependencies before the service is to start. The condition following ensures that the system boot has reached a runlevel before we started the configured unit.

A runlevel is the mode that a legacy Unix-based OS would run at. A specific runlevel had several services stopped or started, providing control over the behaviour of the machine. In Systemd the legacy term runlevel has been replaced with what's referred to as a target. Conventionally, seven runlevels exist, numbered from zero to six. They translate to the following targets:

runlevel	target
0	poweroff.target
1	rescue.target
2,3,4	multi-user.target
5	graphical.target
6	reboot.target

The command `systemctl get-default` lists which targets the OS is currently turned on. *Systemctl* can change targets.

`sudo systemctl set-default multi-user.target` will bring the system up in text mode target, while `sudo systemctl set-default graphical.target` will boot the OS to graphical mode (GUI) target. Each command would require a reboot to establish the mode.

The `[Service]` section provides details on how to control the service. `Type=` categorises a service by its process and demonising behaviour. `Type=idle` shows it won't run the service until all other jobs are dispatched, meaning the system is up and running. `ExecStart=` specifies the path and the program used then executes the process. Since this is a Python script, Python 3 is required to execute the script. It's necessary that the full

```
pi@piTL:~$ systemctl list-unit-files --type=service local-mypython.service
UNIT FILE STATE VENDOR PRESET
local-mypython.service disabled enabled

1 unit files listed.
pi@piTL:~$ sudo systemctl enable local-mypython.service
Created symlink /etc/systemd/system/multi-user.target.wants/local-mypython.service → /etc/systemd/system/local-mypython.service.
pi@piTL:~$ systemctl list-unit-files --type=service local-mypython.service
UNIT FILE STATE VENDOR PRESET
local-mypython.service enabled enabled
```

It's alive!
Well, enabled
at least.

path of the program Python 3 be provided. **Restart=** shows conditions in which the system will automatically attempt a restart of the service if it crashes.

The **[Install]** section is used to define whether the unit is to be enabled or disabled. **WantedBy=** is the most common way to enable or mark a service for a startup at boot.

Practice makes perfect

From the command line, apply a permission change to the new file to establish that it's ready for use:

```
$ sudo chmod 755 /etc/systemd/system/local-mypython.service
```

With the file configured, we need to inform systemd that the file is available to be processed for when the system starts at boot.

```
$ sudo systemctl daemon-reload
```

The file contents can now be edited without requiring a reload. If the filename is changed or deleted, the Systemd process needs to be reloaded with the **systemctl** command.

With a new unit file added, we need to tell systemd to enable the service on startup. The command **systemctl list-unit-files --type=service local-mypython.service!** provides the status of our new unit. The service needs to be moved from **disabled** to **enabled** using the following command: **sudo systemctl enable local-mypython.service**.

A reboot of the Pi will enable Systemd to establish the service. Recall that the **systemctl** tool provided the capabilities to both monitor and control the services under Systemd.

Now that we've established a unit **.service** let's get in some additional practice time to entrench the knowledge by using the Systemd tool used to administer the service.

From the command line on the Raspberry Pi enter **\$ systemctl status local-mypython.service**

See the screenshot of the command line output (above), annotated with letters to assist the reader in seeing the command output.

The command **A** checks the status of the service by outputting what the expectations of the service are when the system boots **B**, what the current status of the service is **C**, and the first few log file lines **D**. The output provides a status overview and tell you if there are problems that require action.

```
pi@pi:~$ systemctl status local-mypython.service
● local-mypython.service - user script to run on startup
   Loaded: loaded (/etc/systemd/system/local-mypython.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2022-05-20 17:16:48 CDT; 4min 40s ago
     Main PID: 562 (python3)
    Tasks: 3 (limit: 1597)
      CPU: 3.894s
   CGroup: /system.slice/local-mypython.service
           └─562 /usr/bin/python3 /home/pi/mypython.py

May 20 17:16:48 piTL systemd[1]: Started user script to run on startup.
pi@pi:~$ sudo systemctl stop local-mypython.service
pi@pi:~$ sudo systemctl disable local-mypython.service
Removed /etc/systemd/system/multi-user.target.wants/local-mypython.service.
pi@pi:~$ systemctl status local-mypython.service
● local-mypython.service - user script to run on startup
   Loaded: loaded (/etc/systemd/system/local-mypython.service; disabled; vendor preset: enabled)
   Active: inactive (dead)

May 20 17:16:48 piTL systemd[1]: Started user script to run on startup.
```

```
$ sudo systemctl stop local-mypython.service
```

To stop a currently running service, the command requires elevated privileges available through the superuser account:

```
$ sudo systemctl disable local-mypython.service
```

This elevated privileges command disables the service **B** from starting automatically on boot. Issuing a status check command of the service **F** will confirm what the expectation of the service is at boot time. **G** once again shows the status of the service as inactive (in other words, dead).

It's important to note, that enabling a service doesn't start the service in the current session. To accomplish that, you need to use the start directive. **Disable** and **Enable** are directives for Systemd during boot.

The command line support tool for Systemd, **systemctl**, requires some practice in order to become familiar with what the command does and the outputs it provides. Trying to stop or start a service using **systemctl** without elevated privileges can lead to frustration because it won't work. If you're not familiar with Systemd, now is the time to gain some knowledge so you have it available when needed.

This tutorial described two changes for the administration of Raspberry Pi OS. The improvements to support Advanced User Options means it's now less work to establish an OS image on an SD card that comes complete with custom configurations. With less work comes the need for less knowledge.

Systemd has been slowly replacing the old init system. The Unit File is the approach to managing resources. If you want to start a script at boot time, then it's time to gain the knowledge of how to do it with Systemd. **LXF**

Watch for the letter call in the tutorial to guide you.

QUICK TIP

The first three resource finishers for Systemd are: <https://systemd.io>, <http://0pointer.de/blog/projects/systemd.html> and www.freedesktop.org/wiki/Software/systemd.

» SMALL OFFERINGS OF SYSTEMD

Systemd running as PID 1 (Process Identifier) is the system and service manager for Linux operating systems. Since 2015, most Linux distributions have adopted Systemd, having replaced other init systems.

Here's a breakdown of Systemd's key features:

- > The design intent was to be backward compatible with SystemV init, handling the boot process for the Linux system.

- > It maintains a list of logged-in users, running containers, virtual machines, system accounts, runtime directories and settings.
- > It implements transactional dependency-based service control logic, keeping track of processes using Linux control groups.
- > It manages services.
- > It provides a logging daemon and a centralised location for all messages

logged by different components in a Systemd-enabled Linux system.

- > It supports utilities to control basic system configuration such as the hostname, date and locale,
- > It starts and manages daemons including simple network configuration, network time synchronisation, log forwarding and name resolution.
- > Finally, it maintains mounting and automount points.



World Web 3

Battle lines are being drawn up as technologies for a Web 3.0 are primed and loaded, **Mats Tage Axelsson** helps you prepare for a user-owned world.

Lately, there have been many news stories on how the web is evolving into something new. A lot of this news cycle is about Meta, but that's just one vision of how we can interact with each other online. The paradigm shift that's coming is irrelevant to the display, which is great for advertising but is no more than a glitzy distraction.

Instead, the most powerful change is decentralisation. This is the idea that your data can be spread on everybody's computers instead of hoarded inside corporate headquarters (*don't look at us!—Ed*). In the mind of the pioneers, users will have full control and everything will be equal. You can think of the third web as a mash-up of regular web pages, peer-to-peer networks and blockchain technologies.

To grasp the power of these

technologies, it's crucial to understand a few key concepts: the basics of blockchains; cryptography on a user level; and how you're part of the redistribution of all the data involved.

Initially, you'll get the feeling that all of this will be great and that the "hippie generation" has caught up with modern computing. But as we'll soon discover, not everything is rosy. People will still need to be vigilant for fairness and to avoid corporate rot. We'll explain how to take a cautious approach to this new world, avoid investing your money in dangerous schemes, and spot those trying to take advantage of new weaknesses that this new technology could be susceptible to.

Web 1.0

In the first version of the web, sites were just a collection of static documents. The author created them and they stayed the

same. For a single user this is a good start and great for presenting yourself to a wider audience. The people whose attention you want to capture don't require more than this if you have a clear message.

We've all seen the rise of social media, which is a significant part of what's become Web 2.0. The main aspect of the second web experience is that the initial code is just the foundation of the platform. Real value comes from the user content – the regular users! To get paid, we're being tracked through everything we do, making us the product – something that has been pointed out by many commentators. To counter this trend, centralised computing needs a serious overhaul.

For Web 3.0, this central nature of the digital world will be uprooted and your data will go back to being yours. The first step is to start addressing data by its content, using IPFS and Swarm (more on these

later). The next step is to create a system that can enable commerce with as few middle men as possible. This is where blockchain technology comes in.

The current state of affairs is that getting started requires capital from everyone – even users. In some cases, a small amount can get you started. One example is *Axie Infinity* (<https://axieinfinity.com>), where low-income people, many in the Philippines, have been making extra money playing a game. Initial investment is a bit high, but it can pay off as a second income. Unfortunately, as with many of these systems it was hacked with \$625 million being stolen in March 2022 (see <https://nbcnews.to/3LEhT2n>), leaving the victims out in the cold; something of a recurring theme.

Web 3.0

A current online search for Web 3.0 reveals that it's built on the Ethereum blockchain. This is untenable, of course, for many reasons. For example, not everyone can edit the chain; adding data to the chain is expensive; and other blockchain types must be possible. To mitigate such problems, several other technologies are in the pipeline. You can already use IPFS (short for InterPlanetary File System, see **LXF238**) and SWARM, a decentralised platform for storing your data. They use Distributed Hash Table technology. Many outfits are creating "side-chains" that don't need the whole network to verify every single transaction in real time.

The reality for 2022 is that the web pages run Web3.js (<https://web3js.readthedocs.io>). This library is very dominant. It has support for connecting to the Ethereum Virtual Machine (EVM), which runs the blockchain. Other parts of the Web3.js API are support for decentralised storage access primarily with IPFS and SWARM (using BZZ tokens). SWARM has been created

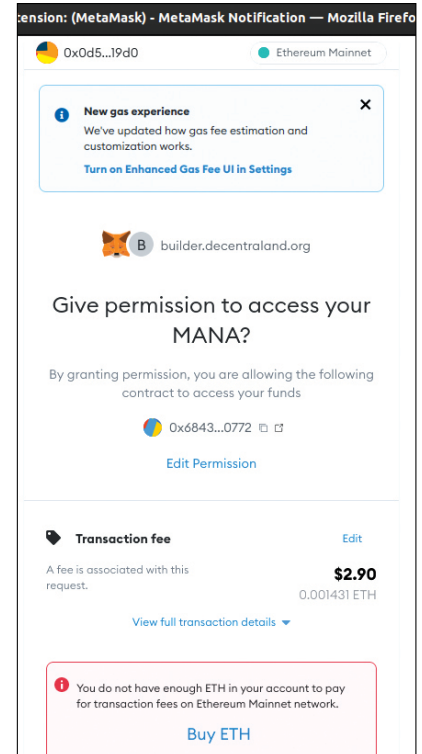
specifically by the Ethereum Foundation and was launched in 2021. Additionally, web3-shh is used for communications and is often called Whisper protocol.

Looking through some of the features you need, it's soon clear that Ethereum has taken a central part in this new system. This doesn't mean that the Ethereum blockchain is the only one that has to be used, but it does dominate at the moment. Many new coins come from "smart contracts" on the Ethereum blockchain using ERC-20 tokens, which has emerged as the technical standard for these smart contracts via token implementation. It is effectively providing a list of rules that all Ethereum-based tokens must follow.

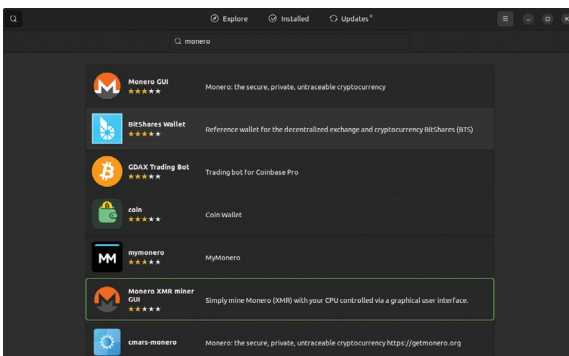
What you see is a distributed application and a web front-end to the blockchain. Web3.js is a common framework that talks to the chain on your behalf. You may also have static content on the IPFS or your SWARM instance. These two later ones use Distributed Hash Tables to run their pages. You can use these for a static web site. To find out more how this works check out the *Beaker Browser* (<https://beakerbrowser.com>), but keep in mind that this is still experimental.

Many providers are also setting up "side-chains" to handle scalability. These chains will execute on their own and integrate on the main chain on a regular basis. How does this all tie together? Some of the existing possibilities are referred to as Decentralized Finance (DeFi). This is a blanket term for financial services on public blockchains using smart contracts. The idea is that DeFi can supplant banking services handling interest payments, borrowing, lending, insurance, derivative and asset trading and more – but without that pesky regulation or oversight.

Distributed Autonomous Organisations (DAOs) run their business according to the rules of the original contract. Inside of these organisations you have voting rights according to your part in the DAO. Auctions are



When you want to both connect and pay, you'll see a notification similar to the one that's shown here.



In Ubuntu, you can easily find many wallets from the repositories. You also have many other wallets available as snaps or AppImages.

» SOLANA BLOCKCHAIN

As Ethereum continues to evolve and usage increases, limitations cause bottlenecks and higher costs for users. Designers devised Solana (<https://solana.com>) to counter this problem. Solana is its own blockchain with smart contract capabilities. Performance across several metrics is improved, and energy use and transaction times are considerably lower, as well.

The first difference to Ethereum is that Solana is Proof of Stake, which saves energy and makes the network faster. Solana also has a system where some users are validators and others are delegators. Validators will run nodes that secure the network and delegators are users who want to stake their coins. The validators receive voting rights for the transactions on the network, helping to

secure the network in the process. Validators also take a fee for their work.

Solana has a huge set of projects that do the same thing as the Ethereum projects. You can borrow, lend and play with NFTs. With faster transactions and lower costs, this version is gaining ground. Ethereum is also moving to Proof of Stake to improve its network, because they'll be left behind otherwise.

also set up with a contract. The seller sets what a final winner receives, what the reserve is and when the auction is over. NFTs are also contracts. Some of these state that the original creator receives payment at every sale, thus setting up a rolling income for the creator.

All contracts are on the EVM and can be viewed by anyone. Putting up a contract will cost you fees, which depend on many things. Some of it depends on the amount of bytes you put on the chain. Other factors depend on the contract itself. Many competing systems exist that makes fees lower. These include Polygon (<https://polygon.technology>) which uses 'side-chains' and even some completely separate chains. This system was hacked with a loss of 800,000 coins at the end of 2021 (<https://bit.ly/lxf291polygon>).

The Opera Crypto browser has all the features you need to trade and store your NFTs, along with other digital assets.

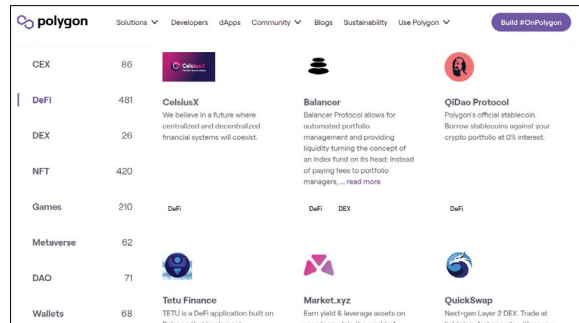
Wallets, wallets, wallets...

Your wallet is your first step to joining the Web 3.0 revolution. These wallets are used for all transactions and since that requires identification, you connect using the matching wallet. Note that you can get a wallet without identifying your physical self, which is the reason governments want to introduce regulations.

You can download your own wallets from several providers, although the simplest option is to add an extension

to your browser. If you're really eager choose *Brave* or *Opera*, which both have built-in wallets. *Opera*'s wallet supports many tokens, including Ethereum and their (ERC-20) compatible coins.

When you want to do anything blockchain-related, you need to be connected with a wallet. This is the closest thing to logging in that you'll encounter on the decentralised web. With the correct browser extension, you'll be able to connect to the sites that you want to use. Like other extensions, they can interact with your web page according to what happens on the page. When entering a Web 3.0 page, you need to click Connect. The page will call your wallet and usually show a message that you need to



Polygon works hard to improve transaction fees and performance. Many projects use its technology to enhance their offerings.

acknowledge. The most common at the moment is an exchange or a marketplace.

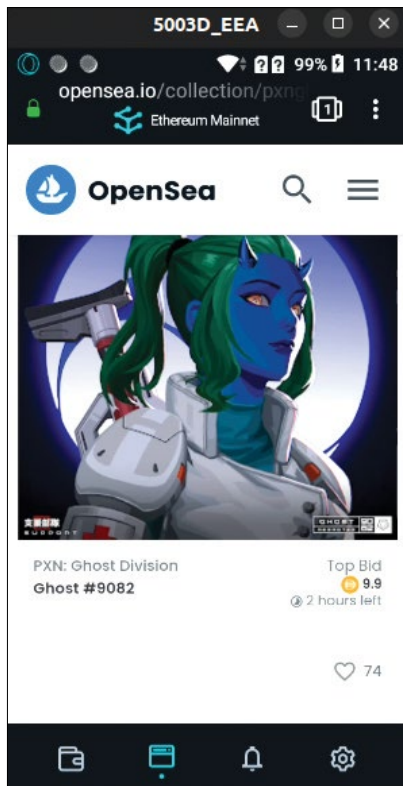
Metamask (<https://metamask.io/download>) is the most popular wallet extension, and it's available for *Chrome* and *Firefox*. It dominates in this space. As you get started though, you'll notice that your extensions tab fills up with wallets. This is because many wallets come directly from one platform with their own quirks and many times their own tokens.

Other wallets can handle many tokens. *Opera* has set its wallet to handle a big selection of coins including many blockchains. When you use an exchange, they usually have their own wallet. **coinbase.com** is a big one that provides trading and opportunities to buy and sell cryptocurrencies. A separate product named Coinbase Wallet connects to the most common services.

Who's in charge?

Proponents say that the users own and operate the network, a statement that needs a bit of analysis to wrap your head around. It's true that you can create your own node and run it on your own hardware. Yet, in the case of Ethereum 2.0, you need 32 ETH and pretty hefty hardware to run one. At the time of writing (shortly after the May 2022 crypto-crash) one ETH was £1,620, meaning you need around £52,000 to be a part of the new more efficient network. Ordinary users won't have that kind of cash, so they'll have to find other solutions. You can stake your Ether using services that pool your stakes. Exchanges like Coinbase will help you with this. There are a plethora of these services, though.

If you want to take the plunge and run your own node, then apart from the hardware requirements you also need a reliable connection. Staking means that you risk your funds, so make sure that you also have a reliable connection to the internet. You'll be penalised if



» OPERA'S WALLET

Current browsers aren't yet ready for Web 3.0. Most browsers need, at least, a wallet extension to handle the initial Web 3.0 infrastructure system. *Opera* includes a built-in wallet, as does *Brave*, but *Opera* is more active in this space with a separate mobile browser called *Opera Crypto*. The regular browser has added a wallet in the side panel of the interface, so is in line with *Brave*. Yet, the

Beta version of *Opera Crypto* shows a stronger commitment to the bright new future that *Opera* envisions.

Specifically designed for handling Crypto sites, *Opera Crypto* defaults to keeping track of new Air-drops and news. You have the same wallet as the regular *Opera*. That wallet supports ERC-20 compliant tokens and most other tokens. Having a single wallet for all your needs

is a good idea. The *Crypto Browser* makes interacting with Web 3.0 sites the default and makes it easy to get started.

The *Crypto Browser* supports direct access to decentralised exchanges, gaming sites and auction sites. Sadly, at the time of writing Linux users must turn to their Android phones since the Linux version is not yet available. Hopefully, this will be remedied with the full release.

your system is unavailable for long periods of time. The risks are considerable for a single user, and that's why you can use pooling services.

Most exchanges now offer staking services, where they take a percentage of the rewards. The rate you receive is usually displayed from the beginning, but their take is only explained in the small print.

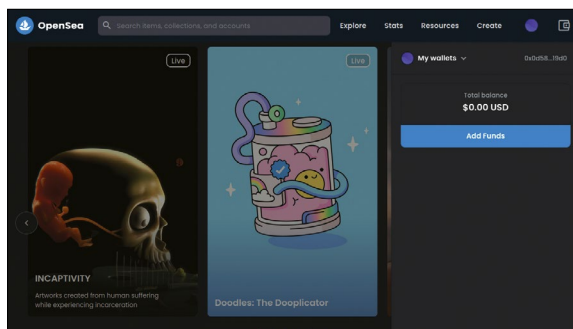
The services that make the headlines (after Meta) are Decentraland and Polygon. In the former you can buy virtual real estate. It works like *The Sims* but with Ethereum tokens called Mana. You can buy these coins on different exchanges. Exchanges also comes as both centralised and decentralised – currently most are regular web pages.

The other big pages deal in NFTs and are art auction sites. Most of these are as serious as Sotheby's store in Decentraland. The major one is Opensea (hacked in February 2022 via a phishing attack, see <https://bit.ly/lxf291sea>), where you can publish your artwork and sell it. The big exception is Axie Market, where you buy your characters to be able to play *Axie Infinity*. For the time being, most offers are either high risk and require trading expertise, or you can gamble. As mentioned earlier, staking and trading your crypto tokens prevail. The only way to minimise that risk is to use pooling (or not partake), which you can find on the big exchanges.

Smart contracts

Currently, most smart contracts create new tokens and NFTs. However, there are already some insurance companies covering investment risk in some tokens, loans and trading contracts that can balance your assets. Future ideas for smart contracts are spread all over industry and entertainment. Purchasing items in stores is available, but this has limited availability.

Smart contracts also help create and validate



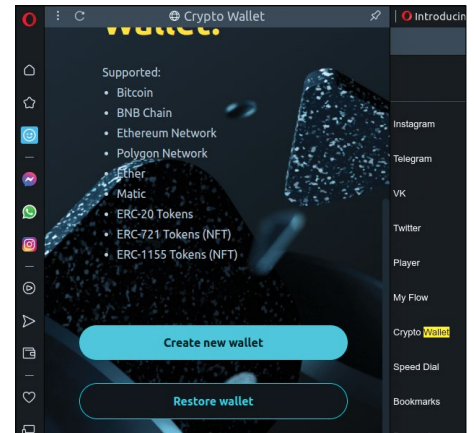
On Opensea, you can sell your artwork and get paid in Ethereum. This is the biggest art site in the space.

identity. In the future, you may also secure the ownership of your home using smart contracts. The blockchain will have a contract matching the property you want to buy. As soon as you buy, you put the sale on the blockchain. As long as that blockchain exists, nobody can change the transaction that's there.

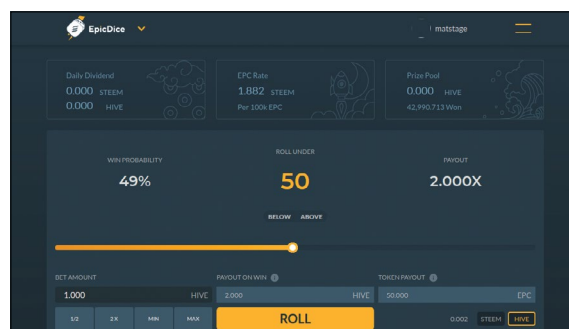
In healthcare, all your documents are confidential but need to be accessible to healthcare workers with your permission. On a blockchain, everything can be made available publicly but locked with your private key.

Many people see the Web 3.0 as a playground for scammers. This is a narrow way of looking at an expansive technology, though it certainly suffers issues. Many efforts are going into building systems that can make trades faster and more efficient. Huge issues for BitCoin are sloth-like trading speed (single-digit levels per second) and its obscene 1,000kWh per transaction energy use, Ethereum 1.0 uses around 240kWh per transaction. With Ethereum 2.0 this power use should drop 2,000-fold thanks to its proof of stake system, and enable huge increases in transaction speeds.

Problems will persist, including lack of regulation and trust in the system, the high investment for individuals, the potential security risks (see <https://web3isgoinggreat.com>) and the volatile nature of crypto markets – with swings of between 10 and 30 per cent in a few hours against "fiat" currencies. You can't rely on it for next month's mortgage payment or food purchases, but the genie is out of the bottle and the technology isn't going anywhere. **LXF**



When first opening Opera, you need to create a wallet that holds your assets and identifies you to Web3-distributed apps.



The fastest growing and first area to blossom in use of distributed apps is gambling. Epic dice is only one example.

» FINDING WEB 3.0 SERVICES

Data is stored on a blockchain using hash numbers. To find and remember these isn't humanly possible. Just as DNS helps you find web pages by a recognisable name, so the Ethereum Name Service (ENS) helps you find web pages for Web 3.0 pages. To get started, you can buy your own page.

Once you have an address, you can also store many other types of data. If

you want to point it to your current web page, the best way is to create links in IPFS (InterPlanetary File System) and link that to the ENS address.

The addresses in this solution has a few top domains. These are smart contracts on the chain that you access when you register. Inside the contract you receive a position and you can store the stuff you need to have your address.

Your records can also point to DNSSEC addresses so you can use regular web page addresses. Once done, all the ordinary records you need to be on the web can be set with Web 2.0 services.

Despite having Ethereum in the name, you can point to other blockchains and resources on other chains. You can even store your Twitter handle and emojis if you like.

Multi-boot USB drives

Shashank Sharma can't parallel park, but he knows how to just copy and paste live distributions into a USB drive to boot 'em up.



OUR EXPERT

Shashank Sharma

is a trial lawyer in Delhi. He's always on the hunt for affordable geeky memorabilia..

As far as we're concerned, the birth of live installable distributions remains the greatest watershed moment for the Linux ecosystem. You could quickly burn distributions on to a CD or DVD (*you're making us sad! -Ed*) and boot into a functional Linux environment, without having to suffer through an installation and all that it entails. The only downside to this approach was the single-use nature of the CD/DVDs. But even this problem was overcome with USB drives, which you could wipe them clean and put in a new distribution at your pleasure.

With *Ventoy*, you can easily create multi-boot USB drives in almost no time at all. You can also deploy *Ventoy* on to a SD card, or even SSD, NVMe drives. You can also configure persistent mode on your *Ventoy* device, meaning that any changes you make to a Live session of a distribution will be available on subsequent reboots. For users looking to run an unattended installation, *Ventoy* supports this feature on distros such as Debian, Fedora, RHEL and Ubuntu Server.

The tool also has support for ISO files larger than 4GB in size, a customisable theme and menu, non-destructive upgrades and more. In addition to hundreds of Linux distributions, *Ventoy* has also been tested to work with a host of Unix operating systems. You'll find a list of these on the project's GitHub page (<https://github.com/ventoy/Ventoy>).

```
1: linuxlala@playground: ~/Downloads/projects/ventoy-1.0.74
linuxlala@playground:~/Downloads/projects/ventoy-1.0.74$ sudo ./Ventoy2disk.sh -u /dev/sdb
*****
Ventoy: 1.0.74 x86_64
longpanda admin@ventoy.net
https://www.ventoy.net
*****
Upgrade operation is safe, all the data in the 1st partition (iso files and other) will be unchanged!
Update Ventoy 1.0.47 ==> 1.0.74 Continue? (y/n) y
esp partition processing ...
Open ventoy efi file 0x010ac0 ...
ventoy x64 efi file size 1810432 ...
Open bootx64 efi file 0x010ac0 ...
Open ventoy ia32 efi file 0x010f10 ...
ventoy efi file size 1216512 ...
Open bootia32 efi file 0x010ac0 ...
Update Ventoy on /dev/sdb successfully finished.
linuxlala@playground:~/Downloads/projects/ventoy-1.0.74$
```

You can safely upgrade Ventoy to a newer version with the `-u` command option, which preserves all data such as image and json files.

Released under the GPLv3 license, *Ventoy* isn't available in the software repositories of most popular desktop distributions, but that's quite alright as the tool doesn't require any installation. With the exception of *VentoyGUI*, all other utilities are shell scripts, which don't require installation.

Apart from a CLI and desktop utility, *Ventoy* also offers a desktop GUI application as well as a browser-based GUI if you're so inclined. To begin, download the latest tarball and extract the files:

```
$ cd Downloads/projects
```

» PERSISTENT USB

Ventoy currently supports persistent storage on only a handful of distributions. To begin, run the `sudo ./CreatePersistentImg.sh -s 512` command. This is yet another shell script included in the *Ventoy* tarball.

Here, the `-s` command option is used to define the size of the persistent storage in MB. This command creates a `persistence.dat` file in your current directory. Copy this file on to the mounted *Ventoy* partition on your USB drive. In our case, we copied this file on to the `/media/linuxlala/Ventoy1` partition.

You still need to inform *Ventoy* that you wish to use this `persistence.dat` file

as storage for distributions. This is done by way of a json file. Open your favourite text editor and add the following lines:

```
{
  "persistence":
  [
    {
      "image": "/distros/ubuntu-21.10-desktop-amd64.iso",
      "backend":
      [
        "/persistence.dat"
      ]
    }
  ]
}
```

Take careful note of the square ([]) and curly brackets ({}). Save the file as `ventoy.json` and place it in the mounted `ventoy` partition of your USB drive. The `image` line of the file refers to the distribution for which you wish to enable persistent storage. The `backend` line is used to point to the file that you wish to use.

You can use the same `persistence.dat` file for different distributions if you prefer. But add separate `image` and `backend` lines in their own curly braces.

Ventoy now enables you to choose whether you wish to boot the specified distro with persistent mode.


```
$ tar xvf ventoy-1.0.74-linux.tar.gz
$ cd ventoy-1.0.74/
$ ls
boot          README      VentoyGUI.aarch64
VentoyVlnk.sh
CreatePersistentImg.sh tool      VentoyGUI.i386
VentoyWeb.sh
ExtendPersistentImg.sh ventoy      VentoyGUI.
mips64el WebUI
log.txt       Ventoy2Disk.ini VentoyGUI.x86_64
plugin        Ventoy2Disk.sh  VentoyPlugson.sh
```

The *VentoyGUI* tool is the native graphical utility, and you'll find versions for different architectures in the tarball. The *VentoyWeb.sh* utility provides the same functionality, but runs entirely in a web browser. For this tutorial, we'll work with the **Ventoy2Disk.sh** script to turn our vanilla USB drive into a multi-boot behemoth. You only have to invoke the relevant shell script from the terminal to work with *Ventoy*.

Deploy Ventoy

After extracting the tarball, plug a USB drive into the system. You can run the `fdisk -l` command to identify the USB device you've just plugged in:

```
$ fdisk -l
Disk /dev/sdb: 7.23 GiB, 7759462400 bytes, 15155200
sectors
Disk model:
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: NTFS
Disk identifier: 0x7efb5116
```

We've clipped the output of the `fdisk -l` command due to space restraints, because the command outputs relevant details for all identified disks as well as each of their partitions. As you can see, our USB drive is identified as `/dev/sdb`. We'll be installing *Ventoy* on to this drive using the **Ventoy2Disk.sh** script extracted from the tarball. The process will destroy all data on the drive, so make sure to check and double-check the device name.

Once you've identified the disk which you wish to use with *Ventoy*, navigate into the directory where you extracted the **Ventoy2Disk.sh** file. We can now install *Ventoy* on to the USB drive:

```
$ cd ~/Downloads/projects/ventoy-1.0.47/
$ sudo sh Ventoy2Disk.sh -I /dev/sdb
[sudo] password for linuxlala:
<output snipped for space constraints.>
Install Ventoy to /dev/sdb successfully finished.
```

When installing *Ventoy* to a USB drive, you can use the `-I` command option to force installation. The `-i` command option will result in failure if the device already has a version of *Ventoy* installed. Alternatively, you can run the `Ventoy2Disk.sh -u /dev/path` command to upgrade *Ventoy*. The **Ventoy2Disk.sh** scripts creates two partitions in the specified device. You can run the `fdisk -l` command again to confirm. We'll only work with the larger of the two partitions.

To now create a multi-boot USB drive with *Ventoy*, you only have to copy the ISO files on to the newly

created *Ventoy* partition. When you mount the USB drive, this larger partition is the only one that becomes mounted. Depending on your distribution of choice, the partition would be mounted under `/media/linuxlala/Ventoy1`.

You can copy as many ISO files as you like on to this partition, depending on the size. When you unmount the device and boot into it, you'll be greeted with a GRUB menu that offers the choice of all the different distributions that you put into the *Ventoy* partition on the USB drive.

Ventoy will scan the entire USB drive and automatically identify all ISO files, irrespective of their location, and present a GRUB menu for you to choose which distribution to boot into.

By default, all changes you make to a Live environment are destroyed as soon as you restart or shutdown the distro. Refer to the Persistent USB box (*opposite page*) if you wish to enable persistent storage for Live distros on *Ventoy*-powered USB devices.

Ventoy is highly configurable and you can play around with themes and menu items. The project provides **VentoyPlugson.sh** script to help you create a **ventoy.json** file which controls the behaviour and appearance of *Ventoy*.

Run the script with the `sudo ./VentoyPlugson.sh /dev/sdX` command, replacing the device name at the end to reflect the device in which you've installed *Ventoy*. You'll be provided a URL, such as <http://127.0.0.1:24681>. Point your browser to it, and then use the sidebar to navigate the different configurable parameters and create your custom **ventoy.json** file, which will automatically be stored in the mounted USB drive.

Head over to the project's website (www.ventoy.net) for a list of all the supported distributions as well as the ones that support persistent storage. You'll also find useful information on all the different *Ventoy* utilities such as *VentoyGUI* and *VentoyPlugson*. It's also possible to replace the default *Ventoy* boot menu with one that features custom images.

The ability to work with different devices, and not just USB, makes *Ventoy* indispensable. Couple this with persistent storage, safe upgrade and the customisation options, and you have a tool that can help you effortlessly impress your peers. **LXF**

QUICK TIP

In addition to storing ISO files, the partition on the USB drive can also be used as regular storage. However, to maintain sanity within the USB drive, we recommend you create a directory such as 'distros', or 'ISOs' to store your ISO and IMG files.

Run the `sudo ./Ventoy2disk.sh --help` command to access the help section. All the *Ventoy* scripts included in the tarball provide basic assistance.

```
linuxlala@playground:~/Downloads/projects/ventoy-1.0.74$ sudo sh Ventoy2Disk.sh
--help
*****
Ventoy: 1.0.74 x86_64
longpanda admin@ventoy.net
https://www.ventoy.net
*****
Usage: Ventoy2Disk.sh CMD [ OPTION ] /dev/sdX
CMD:
-i install Ventoy to sdX (fails if disk already installed with Ventoy)
-I force install Ventoy to sdX (no matter if installed or not)
-u update Ventoy in sdX
-l list Ventoy information in sdX

OPTION: (optional)
-r SIZE_MB preserve some space at the bottom of the disk (only for install)
-s/-S enable/disable secure boot support (default is disabled)
-g use GPT partition style, default is MBR (only for install)
-L Label of the 1st exfat partition (default is Ventoy)
-n try non-destructive installation (only for install)

linuxlala@playground:~/Downloads/projects/ventoy-1.0.74$
```

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

How to take better screengrabs in Ubuntu

Nick Peers reveals how easy it is to take screengrabs and annotate them using a combination of Ubuntu's built-in tools and third-party utilities.



OUR EXPERT

Nick Peers has taken more screenshots than he cares to admit over the past <insert lengthy amount of time>.

Screen capture tools are a useful in anyone's armoury. You don't need to be a *Linux Format* writer to benefit from being able to show someone part or all of your screen. Whether trying to demonstrate a particular feature or get help from an expert by sharing a visual representation of your computer's ills (such as a specific error message), a screen capture program is the tool you need.

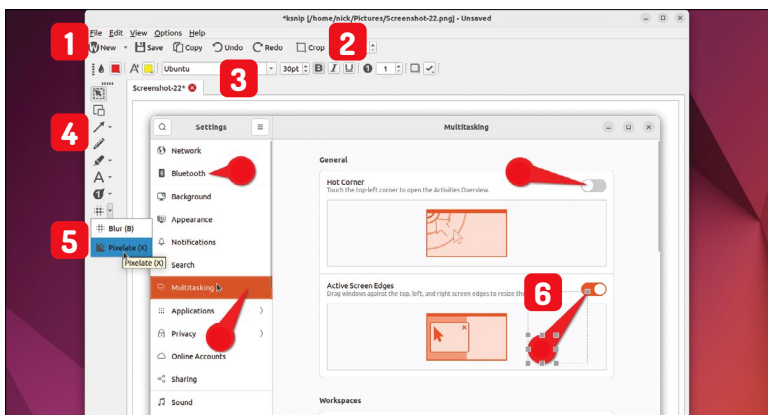
The ability to grab your screen is an integral part of the Linux desktop, whichever one you use. However, the built-in screenshot tool doesn't always have all the features you need, so in we're going to take a look at not just the default tools for GNOME and KDE, but also a selection of third-party programs that take screen captures to the next level.

The quickest way to capture your current screen is with a single keypress: the aptly titled Print Screen or PrtScr key. In older versions of Ubuntu – including 20.04 LTS – pressing this key results in the sound of a camera shutter, the screen will flash and an image of your entire desktop titled **Screenshot from YYYY-MM-DD HH-MM-SS.png** will be saved to your Pictures folder using the *GNOME Screenshot* tool. Double-click this to view a picture of your desktop sans mouse cursor.

It's quick, dirty and not particularly elegant. For starters, a 1:1 screen capture of a typical Ubuntu desktop produces a file around 2.1MB in size. That's fine if you need to capture the entire desktop, but in many cases you'll only want to capture a specific part of it, such as a program window. The good news is that *GNOME Screenshot* has anticipated this very need: press Alt+PrtScr and only the currently active window will be captured. But what if you want to select a specific part of the screen – say one with overlapping windows? You're covered here, too: press Shift+PrtScr, then click and drag to select the area you want.

If you need more control over your screen captures, there's a separate screenshot portal tool accessible via the Launcher. This displays a dialog enabling you to select what you want to capture – full screen, application window or rectangular selection – but it also makes it possible to add a delay to the capture, so you've time to set things up as you need to. It also gives you the option of including the mouse pointer and a border around your application window if required. After capture, you'll also be able to save the screenshot to a specific folder with your choice of filename.

ANNOTATE WITH KSNIP



1 Main toolbar

You can easily create, save and copy images to the clipboard from here, as well as reverse unwanted changes.

2 Crop images

If you need to crop into a specific part of your captured image, click this button to do so using a resizable bounding box.

3 Markup tools

Ksnp supports a decent range of annotation tools. These range from arrows and pens to text, blurred areas (for redactions) and even emojis.

4 Select object type

Click the down arrow next to a tool to choose what type of object you want – for example, a blur or pixellate tool for redactions.

5 Formatting toolbar

This toolbar displays options and tools based on the selected annotation object, such as font, size and effects for text-based objects.

6 Objects

Each annotation is treated as an independent object, enabling you to select them individually to move, resize, delete or make other changes.

New and shiny

The latest LTS build of Ubuntu unveils a new version of Gnome Desktop and with it a new screenshot tool. Now when you press PrtScr the tool pops up as a floating widget in Selection mode, with part of the screen pre-selected inside a resizable bounding box. Next to Selection are Screen (full desktop) and Window, which pulls out all open windows and invites you to select which one you want to grab.

Beneath the selection tools you'll see three further controls. On the left is an option to choose between simple screen capture or video capture. On the right

you'll see a mouse pointer – it's excluded from the capture by default, so select this if you need to include it (for example, you want to draw attention to a specific option within a program window).

The central button is the actual capture tool – it's a white shutter for screen captures, a red record button if you opt to record video. Press the white button to capture a static screenshot. You'll hear the familiar click, and then you'll be notified the image has been copied to the clipboard for pasting elsewhere. It's also saved into a **Screenshots** folder inside your Pictures folder using a similar naming convention as the older screenshot tool.

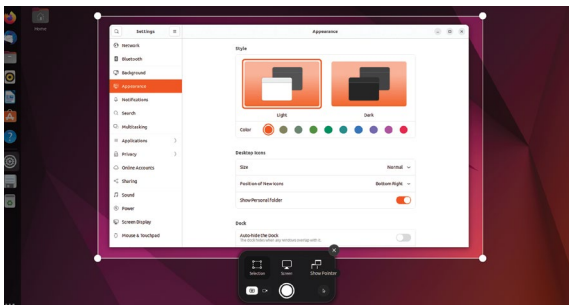
If you're capturing video, after clicking record, the screen remains dimmed, but you can perform whatever action you like – you'll see a record meter appear in the menu bar, so click this to end the recording when you're done. Video clips are stored in .webm format inside a special **Screencasts** folder inside Videos following a similar naming convention to screenshots: 'Screencast from DD-MM-YY HH:MM:SS'.

Those keen to capture the entire desktop or just the active window – as before – will be pleased to learn that the new tool offers two keyboard shortcuts that do just that: press Shift+PrtScr to capture the entire desktop, or Alt+PrtScr to capture the currently selected window.

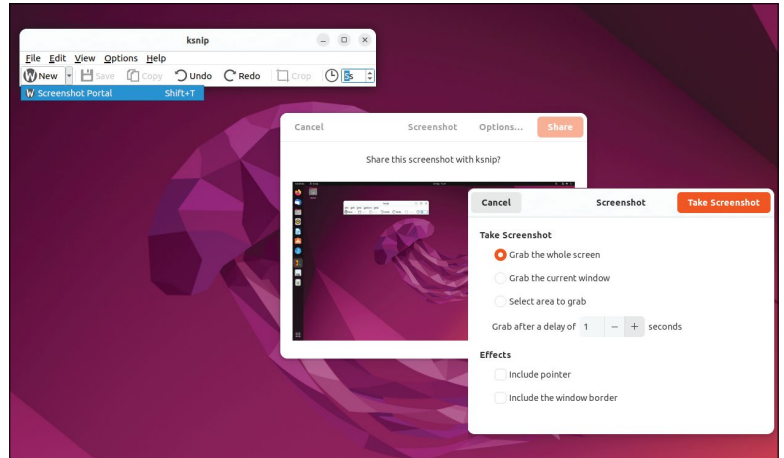
Beyond screenshot

While the new screenshot tool is easier to use than its older precursor, it lacks several key features. For starters you can't choose where to save your screen captures or even change their name, and you also lose the ability to delay screen captures to give you time to set up your screen before capture if required.

Thankfully, you'll find several third-party screen capture tools exist to fill in these – and other gaps. KDE



Ubuntu 22.04 LTS unveils a new desktop screenshot tool – simply press the PrtScr key to invoke it.



If you're running Ubuntu 22.04 on Wayland then your screenshot tool must be able to invoke the GNOME Screenshot Portal to capture grabs.

users should check out *Spectacle* (see the box, *overleaf*), which replaces the *Gnome Screenshot* tool. GNOME users can install *Spectacle* through the *Software Centre*, but it's not compatible with GNOME's implementation of the Wayland protocol, so doesn't work in Ubuntu 22.04 unless you switch back to Xorg – see the box for more details (*below*).

The same restriction applies to several other screen-capture tools we'd have otherwise recommended, including *Kazam* (<https://launchpad.net/kazam>). This offers all the features found in the original GNOME screenshot tool, including the ability to delay captures with optional countdown timer and save files manually, plus some additional features of its own. The most notable of these is the ability to configure *Kazam* to save screenshots automatically to a folder of your choosing using your own choice of filename prefix.

You can still use *Kazam* to capture the entire desktop screen in Wayland, but its screen-casting tools don't work and it'll crash if you attempt to capture a window, making it an impractical solution unless you switch back to Xorg.

Getting ksnip

Thankfully, you don't need to abandon Wayland to access all of *Spectacle*'s features in GNOME. There are two Wayland-friendly tools offering both screen capture and post-capture annotation tools to choose from, each with their own unique features, and both programs can

QUICK TIP

You can expand ksnip with plugins. Visit <https://github.com/ksnip/ksnip-plugin-ocr/releases> to download a package to add OCR capabilities to the tool. Use it on scanned documents to extract editable text via the Options>OCR menu.

» THE TROUBLE WITH WAYLAND

For those stepping into the world of Ubuntu 22.04 for the first time, a potentially nasty surprise awaits: Wayland, the new desktop protocol that replaces X11. For the most part it's a good switch – Wayland is faster and more secure, but it's that additional security which poses a problem for screen-capture tools.

On a desktop running the Wayland protocol, many screen-capture tools are unable to grab screens using their own

capture tools. That's because they're not engineered to work with Wayland's more robust permissions structure, and you can see this in action when you attempt to install and launch *Spectacle* through an Ubuntu 22.04 install running the GNOME desktop on Wayland – when the program opens, you'll be told it "could not take a screenshot".

The problem doesn't occur with *Spectacle* in its native KDE environment because it's been fixed to work with

Plasma Wayland since May 2021. Because GNOME is stricter about what programs can access screen-capture capabilities, other tools have been forced to adopt a different approach.

In the case of *ksnip* and *Shutter*, both authors have engineered a workaround whereby the tools use the GNOME Screenshot Portal in Wayland instead of their own more flexible screen-capture capabilities (including support for global hotkeys), which continue to work in X11.



QUICK TIP

Shutter may ship with more stickers than ksnip, but the latter supports scalable vector graphics (in SVG format). You can add your own via Tools>Settings>Annotator>Stickers. You'll find loads of free vector art online – start your search at <https://openmoji.org/library>.

be installed through *Ubuntu Software*. They are *Shutter* and *ksnip*.

Let's start with *ksnip*. In Wayland, it's unable to capture screenshots using its own tool; instead, it relies on the *GNOME Screenshot Portal*, which works in a similar way to the old *GNOME Screenshot* tool. It can be installed any number of ways – through *Ubuntu Software* (choose the *ubuntu-jammy-universe* version over the *snap* in *Ubuntu 22.04*), or you can visit <https://github.com/ksnip> to discover how to obtain it as a flatpak, DEB/RPM package or standalone *Applmage*.

Once installed, open *ksnip* through the *Launcher* and it'll place its icon on the menu bar. Click this to reveal a pop-up menu with various options: *Show Editor*, *Screenshot Portal* and *Open* (which works with any image file, not just screen captures).

To take a screenshot, choose *Screenshot Portal* and it'll immediately capture the entire screen and open it in the separate *Screenshot Portal* application. Ignore this initial capture and click the *Options...* button to configure the portal to grab the entire screen, the current window or a user-selected area. You can also set a delay timer in seconds, plus capture the cursor and add a border effect to any captured windows, just as with the old *Gnome Screenshot* tool.

Once you've configured your choices, click *Take Screenshot* and a new capture will be taken. If you're happy with your grab, click *Share* to transfer it to the *ksnip* editor; if not, click *Options...* to try again.

Once you've taken your first screengrab, the *ksnip* window expands to reveal the image and a selection of

tools for performing post-capture editing. From here, you'll see a selection of markup tools to the left of the image, which are summarised in the annotation opposite. Each added element is created as an independent floating object, and the *Select* tool at the top of the tools list enables you to click existing objects to move, resize, edit or delete them (press the *Del* key) using their own context-sensitive controls.

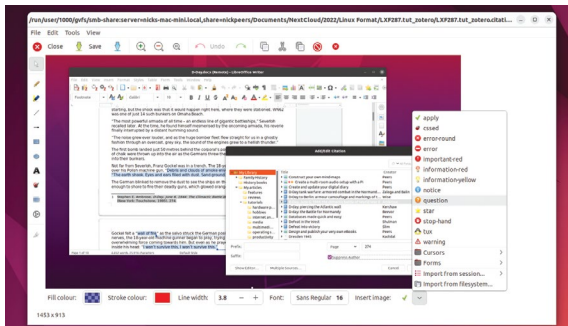
Beneath the *Select* tool is the *Duplicate* tool. After selecting this, click and drag on a portion of your image, which will be duplicated as a separate layer sitting on top of the main image. This can be moved or resized. For more advice on using *ksnip* as an editing and markup tool check out the step-by-step guide (*opposite*).

Snap with shutter

If you can't find the markup tool you need in *ksnip*, then take *Shutter* for a spin (install it through *Ubuntu Software*). The tools share many similarities – including a shared reliance on the *GNOME Screenshot Portal* in *Wayland* – and while we prefer *ksnip*'s workflow, there are a few reasons why you might want to try *Shutter*.

First up is its support for exporting your screengrabs directly to a series of supported online accounts: *Imgur*, *vgy.me*, *Gyazo*, *Dropbox* and *ToileLibre* are all supported along with *FTP*. Second, *Shutter* supports a wider range of graphics for 'stamping' on your images – whereas *ksnip* gravitates towards a selection of smileys, *Shutter* provides more practical icons such as errors, information, questions and even a selection of callout graphics, inside which you can combine with a text object to form a label. Note that they're bitmaps and don't scale up well – see the *Quick Tip* (*left*) for a way to add more scalable vector stickers to *ksnip*.

Third, *Shutter*'s plugin repository has seen it benefit from more than a dozen post-capture effects being added, including effects like turning your grab into a *Polaroid* photo complete with caption to a wider range of border effects. The plugins crashed when running in *Wayland*, but worked fine once we switched to *Xorg*. Apply them from the main *Shutter* window via the '*Screenshot>Run a plugin...*' before clicking *Edit* to annotate your grabs. **LXF**



Shutter provides a wider choice of icons for labelling your screengrabs with, but beware: they're bitmapped, so don't scale well.

» KDE'S SPECTACLE

If you're a *KDE* desktop user, then *KDE* ships with the default *Spectacle* screenshot tool, which goes a bit further than *Gnome*'s standard *Screenshot* tool. Press *PrtScr* to invoke it, at which point you'll see a preview of the entire desktop appear along with options to set up another screengrab.

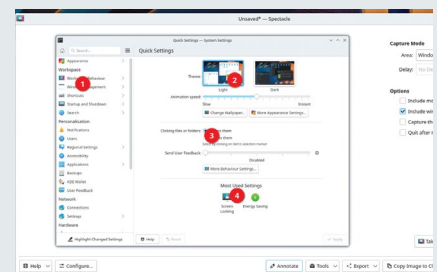
In addition to the usual *Full Screen* and *Rectangular Region* options, *Spectacle* offers two options for capturing windows or dialogs. *Active Window* works in the same way as other screenshot tools, while *Window Under Cursor* captures whichever window is beneath the cursor when it's taken. Look for the '*Capture the current pop-up only*' option, which works

with this to enable you to selectively grab items like pull-down menus.

Beneath this is a '*Delay*' option, for either setting a timer before the snap is taken or ticking the *On Click* box so that when you click '*Take a New Screenshot*', *Spectacle* waits for the next mouse click before capturing the screen. Other features include the ability to choose which file format to save your grabs in, along with an *Export* button for opening captures directly in your choice of image editor or to transfer elsewhere, with *Bluetooth*, *email*, *NextCloud* and other devices all available as destinations.

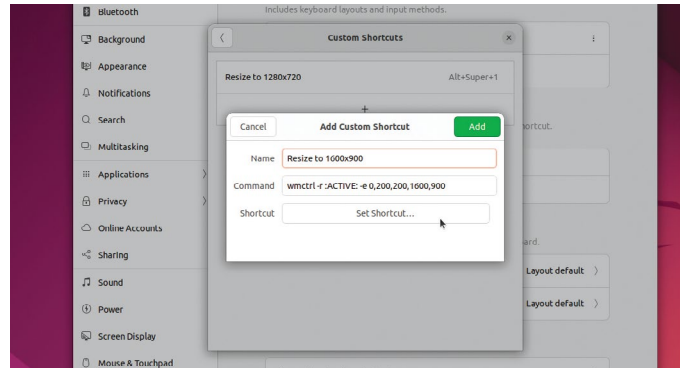
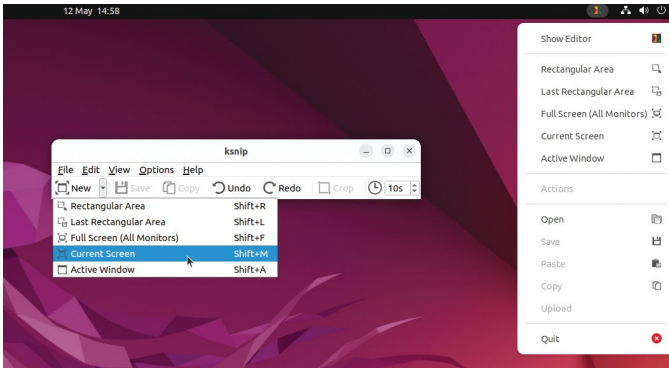
Spectacle also offers a built-in annotation tool. This enables you to

apply effects and mark up your captured images. You can configure its save and capture settings, as well as assign keyboard shortcuts for super-fast screen captures.



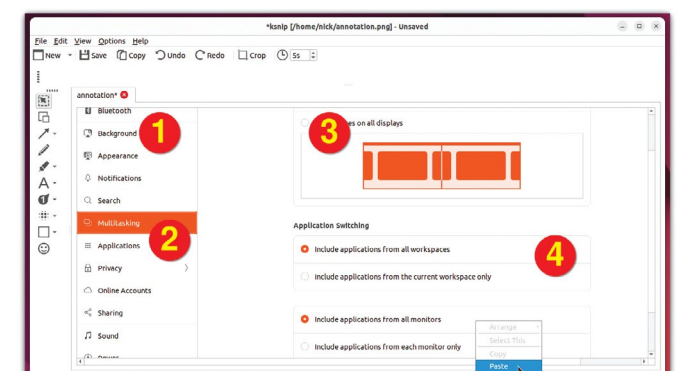
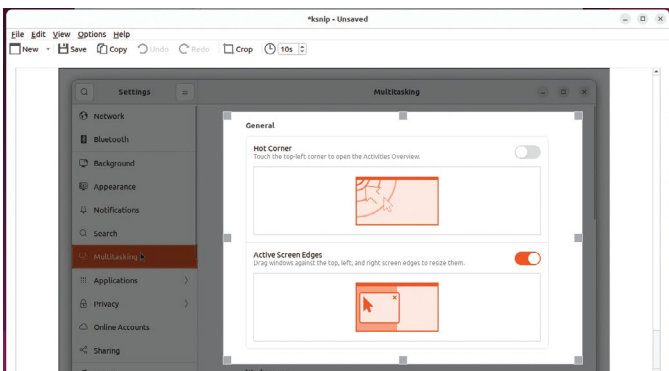
KDE's built-in screenshot tool is flexible and powerful, and ideal for KDE desktop users.

QUICK TIPS WITH KSNIP



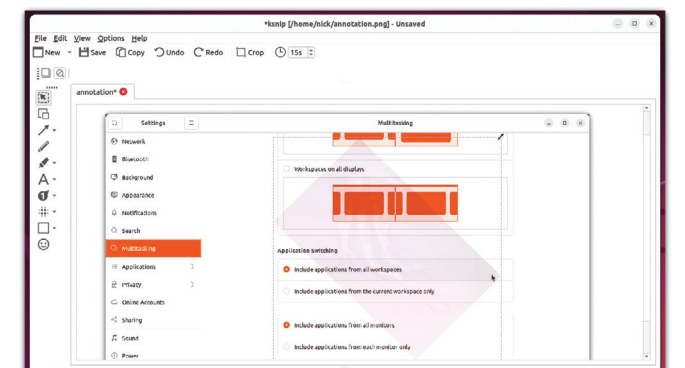
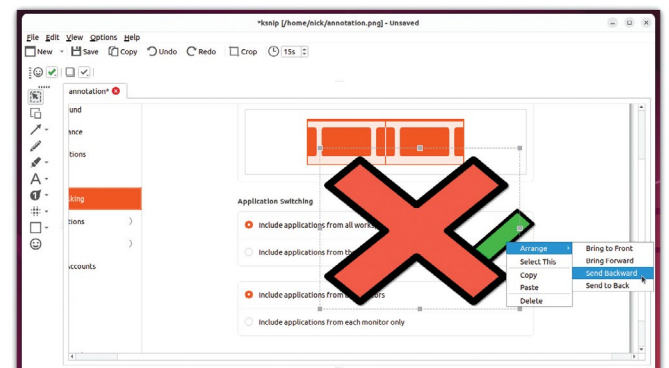
1 Go native
If you'd like to give *ksnip*'s own screen capture tool a go, log out of Ubuntu. Click your username followed by the cog in the bottom-right corner, then select Ubuntu on Xorg and log in. Launch *ksnip* to reveal a New option with five capture types as shown above. It also has its own global shortcuts: use Alt+Shift instead of just Shift.

2 Resize windows quickly
Want to resize a program window to a specific size before grabbing it? First, install *wmctrl* (`sudo apt -y install wmctrl`), then open Settings and navigate to Keyboard>View and Customise Shortcuts>Custom Shortcuts. Create your shortcuts using the following command, changing 1280 and 720 to your desired width and height: `wmctrl -r :ACTIVE:-e 0,200,200,1280,720`.



3 Crop to perfection
Sometimes it's easier to crop an image after capture. Thankfully, *ksnip* includes a dedicated Crop tool for this very purpose. After capturing your image, click the button and your image will be enclosed in a bounding box – you can resize this through click and drag, or use the X, Y, W and H fields at the bottom to set it manually. Click Apply when done.

4 Copy and paste
When you create a new object, *ksnip* always reverts to its default settings, including font and size. To get around this, select your existing object, right-click it and choose Copy, then right-click where you want the new object to go before choosing Paste. A duplicate will be created for you to modify – if it's a number, it'll automatically go up in sequence.



5 Rearrange items
Every annotation you place on your image is treated as a separate, movable object. Click it to select it, where you'll find you can resize and move it easily. By default, newer labels sit on top of older ones, but if you right-click the currently selected object and open the Arrange sub-menu you can choose how items overlap each other.

6 Add a watermark
Protect your images with an image-based watermark. Select Options>Settings>Annotator>Watermark and click Update to select your image file, such as a scan of your signature. Ignore the cramped look in the box – the image's original aspect ratio will be respected when you choose Edit>Add Watermark. A faded copy of the image will be shown, which can be resized and moved.

EMULATION

Credit: <https://norbertkehrer.github.io>

Emulate the first portable computer

Rounding off our emulation series, **Mike Bedfords** revives the first portable computer that ran the bizarre APL language.



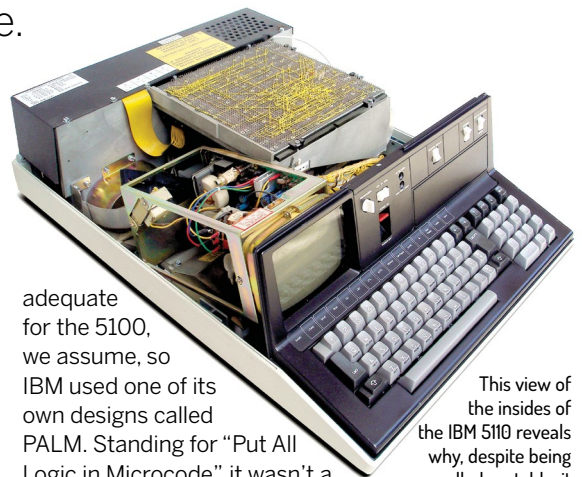
OUR EXPERT

Mike Bedford certainly has no nostalgia for an era when portable PCs cost upwards of \$50,000, but he admits to having a fascination with APL..

We'll warn you that identifying the first portable computer is a battlefield. Indeed the Osborne 1, which was launched in 1981, is often given that title. Some people refer to the Osborne 1 as the first commercially successful portable computer, but the term "commercially successful" is even more vague than "portable".

The Osborne 1 didn't predate the IBM 5100, though. That first saw the light of day in 1975, while its successor, the IBM 5110, launched in 1978. However, to be pedantic, neither of those was IBM's first portable computer, the true identity of which was the IBM SCAMP, which first appeared in 1973. What's more, given our interest in APL, the SCAMP ran APL, something that will be obvious when we point out that its name stood for Special Computer APL Machine Portable. The explanation for the apparent contradiction of two IBM machines both claiming to be the first portable computer, is that the SCAMP was only ever a prototype or, in other words, IBM never sold SCAMPs.

So, now we've made that as clear as mud, let's take a look at what you got when you paid between \$8,975 and \$19,975 (equivalent spending power of \$47,962 to \$106,745 today) for an IBM 5100. The state-of-the-art microprocessor back in 1975 was the 2MHz, eight-bit Intel 8080 that executed around 0.3MIPS. This wasn't



CREDIT: oldcomputers.net

adequate for the 5100, we assume, so IBM used one of its own designs called PALM. Standing for "Put All Logic in Microcode", it wasn't a microprocessor, but a circuit board containing 13 gate array devices and a handful of other components. It ran at 1.9MHz, but it's not clear how many MIPS it managed to clock up. Although PALM could directly address only 64KB of memory, this was increased in the IBM 5100 by bank switching. This way it boasted up to 64KB of ROM (Read Only Memory) which stored the APL and/or BASIC interpreters, and up to 64KB of RAM.

This view of the insides of the IBM 5110 reveals why, despite being called portable, it was barely so.

Most personal computers that predated the IBM 5100 – and most of those that followed – comprised two or more boxes. At the very least, there'd be a monitor and a case containing the processor board and keyboard. Alternatively, while many of the home computers of the early 80s came as a single box, you had to connect them to a TV and a cassette recorder. The IBM 5100 was different in being a single-box computer, which is why it was considered portable. That box contained the processor board, keyboard, tape drive, and a tiny five-inch monochrome screen that could display 16 lines of 64 characters. Whether you'd call it portable is debatable since it weighed 25kg and you had to plug it into a mains supply.

IBM 5110 emulator

The emulator we're going to be using is written in JavaScript and it's hosted at https://norbertkehrer.github.io/ibm_5110/emu5110.html. In addition to the screen and the keyboard, there's a representation of main front panel switches that appear at the top right. The switches are mostly toggle switches, and the half that's shown in grey is the side that you can click. IBM



CREDIT: Daderot, CCO 1.0, public domain, Wikimedia <https://bit.ly/1xf291ibm>

The IBM 5100 was, perhaps, the first portable computer, but the slightly later 5110, shown here, was very similar.

5110s were available with either BASIC or APL or both; the emulator features both. Initially, it starts in BASIC mode but, to change to APL, click the APL side of the BASIC/APL switch and then on RESTART. We're only going to see a small part of the IBM 5110's capabilities here so, if you want to learn more, you'll find all its user manuals at www.bitsavers.org/pdf/ibm/5110.

The keyboard will appear strange, mostly because many of the familiar symbols aren't in their normal positions, there are unfamiliar symbols in the shift position of many of the keys, and there are command shortcuts on the front of most keys. Some words of instruction appear at the bottom of the page, but a few other bits of guidance will help you get up to speed.

You can either use your actual keyboard or the on-screen one. However, because some of the strange mappings, we suggest you use your PC's keyboard only for the letter and figure keys. To get the alternative characters on an actual keyboard, use Shift to enter the characters shown above the letter or figure, and Ctrl for the commands on the fronts of the keys.

If you're using the on-screen keyboard, you need to use the on-screen SHIFT and CMD keys, but note that they each operate for just a single keystroke. So, for example, if you click SHIFT and then P you'll enter an asterisk, but if you then click P again, you'll get a P. We should also warn you that there's a bug which causes many of the BASIC commands to be in the wrong places, although that might have been corrected by the time you read this.

Although our emphasis is on APL – because you're probably already familiar with BASIC and might have even delved into its various historical dialects – since the emulator starts up in BASIC, let's look at that briefly. Perhaps one of the most unusual aspects is that, as well as programming, you can use it as a calculator, which was useful because the 5110 didn't have a calculator application. Admittedly, you can do that with most versions of BASIC, but only by entering PRINT first. So, for example, to calculate $5 * (13 + 24)$, on most machines you'd have to enter `PRINT 5 * (13 + 24)`. In 5110 BASIC, though, you only need to type `5 * (13 + 24)` as you'll find if you type it now.

It might not seem a huge advantage, but entering all those PRINT statements, even though that just needed a single key depression together with CMD, would soon have become tedious. We promised not to say much

about BASIC, though, so let's now move on to APL.

Again, APL could be used as a calculator, and a more powerful one than is provided by BASIC. We're going to enter `!5`, but we'll have to explain how to do that, because you won't find the ! symbol on any of the keys. The solution is to overstrike the single quotation mark and the full stop. To do that, enter the first of those characters, which is the Shifted K key, use Backspace (the one on the on-screen keyboard, not your real keyboard) and then enter the other. Now, when you press EXECUTE (or the Enter key on a real keyboard), APL will respond with `120`. You might recognise this as 5 factorial ($5 \times 4 \times 3 \times 2 \times 1$), and this hints at the power of APL. Many languages don't have factorial built in – it's often used as a programming exercise to teach the principles of recursion.

As an alternate example, which also uses a character that has to be entered by overstriking, specifically Shift O and Shift P, enter and execute `⌈100`. APL will show the answer as 4.6052, which you might not recognise, although if you try `10 ⌈100` you'll get an answer of `2`, which might shed more light on this function. It's actually the logarithmic function and, if you enter it before a single number, as in our first example, it evaluates the natural logarithm. If, on the other hand you add a number before it, as in our second example, it calculates the logarithm to the base of the first number. And in case you're wondering about that strange signal, it's supposed to look like the cut end of a wooden log.

Easy as APL

We've seen just a bit of APL, undoubtedly one of the IBM 5110's most innovative features, but there are easier ways to learn APL than by tangling with the idiosyncrasies that old machine. We can suggest two alternatives. First up is an online implementation that you can find at <https://tryapl.org>. Alternatively, if you

Processors are invariably single chips but not in the IBM 5110. As you can see, its processor was a complete board of components.

QUICK TIP

You might not have heard of IBM 5100 series computers, but you'll be familiar with the IBM 5150 if we use its alternative name. It might not have looked much like the 5100 or the 5110, but the 5150 was what we think of as the IBM PC, which was launched in 1980 and changed computing forever.

» UNLEARNING THE RULES

When you first started to read about programming, you'll have learned various rules that apply to most languages. For example, while you'll have used the \times and \div symbols in your early lessons in arithmetic, you'll have been told that in coding `*` and `/` are used instead for multiplication and division. With APL, all that and more changes.

To illustrate that, using either TryAPL or GNU APL, execute `10 * 5`. Bizarrely, APL will respond with `100000`, which you will probably recognise as 10 to the

power of five which, in other languages would be represented by `10 ^ 5` or `10 ** 5`. If you really do want to multiply you should use `10 × 5`, which uses one of the many APL symbols that aren't used in other languages.

Similarly, to divide 9 by 3, you should execute `9 ÷ 3`. If you were to accidentally revert to form and type `9 / 3`, APL would give the response of `333333333`, which is probably the last thing you were expecting. What it's done is to duplicate the value 3 nine times. That's not

especially useful, although if you execute `x ← 9 / 3` and then `x` to display the contents of `x`, you'll see that `x` contains 9 values of 3. What it's done, therefore, is to create a vector, i.e. a one-dimensional matrix, containing those nine threes.

And finally, try this: `10 × 5 + 5`. With your conventional programming hat on you'll have expected 55 but APL's answer was 100. In the absence of brackets, APL simply executes the expression from right to left with no precedence of operators.



QUICK TIP

The home computers of the 1980s are usually thought of as BASIC machines but there were a few exceptions. We've already seen one in our foray into emulating old computers - the Jupiter Ace which ran Forth - and that's not all. The Commodore SuperPET, introduced in 1981, although not commonly used by hobbyists, included an APL interpreter.

prefer something running locally, our recommendation is *GNU APL*. The TryAPL site is straightforward to use, thanks in part to the fact that all APL's strange characters can be entered by clicking the symbols on the top line. However, while *GNU APL* is a fuller implementation, it requires a few words of explanation.

If Ubuntu is representative, you won't find it in the repositories, although the installation instructions worked as expected. *GNU APL* has an Emacs mode, which should make it a bit more user-friendly and, while we didn't try it, you might like to do so. Natively, though, *GNU APL* is purely a command line utility. However, unless you suitably configure your keyboard, you won't easily be able to enter all those weird and wonderful characters. So, configure your keyboard, using `setxkbmap`, to have two layouts: your normal layout as one, and the APL layout as the other. You also specify a key (for example, Alt) that you'll press to select APL characters. Because you're unlikely to have a keyboard with the APL characters engraved on its keys, though, we should tell you that typing `]KEYB` in *GNU APL* will display the keyboard layout.

So you're not continually entering that command, though, it would be a good idea to print it out. You'll notice, for example, that the 8 key shows * followed by

⌘ on the top row, and 8 followed by ≠ on the bottom row. This means that 8 and * are entered by pressing the key without and with Shift, respectively, whereas ≠ and ⌘ require you to press the key with Alt (if that was the key you specified in `setxkbmap`) and with Shift+Alt, respectively.

APL's pros and cons

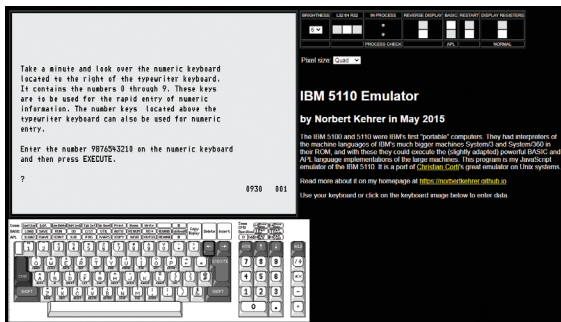
APL differs from nearly every other language by using symbols in place of the more familiar keywords. There are pros and cons. On the plus side, you'll end up typing far fewer characters to do a particular job, and your code will be much more compact. Conversely, code is much harder to read and understand. However, if you like the idea of being a respected as a programming guru, then getting up to speed with APL is surely the way to go. As a final word of introduction, we should point out that APL isn't just a historical curiosity. It continues to have a small but enthusiastic following, and at least a couple of commercial implementations for PCs are available today.

APL has been described as a glorified calculator, and a lot of the time you'll be just typing a command of a few characters and seeing the result, rather than writing programs in the normal sense of the word. So, give `5 + 3` a try and you'll see that APL gives the answer of `8`, just as you'd have expected. Okay, so that's what you'd have expected of a calculator, but why is it a glorified one? There's no single answer to that question, but after a few more example we trust you'll have started to see some of the power of this ultimate calculator.

For a start, you can assign values to variables on the fly, so you can use these for temporary storage. So, rather than typing `5 + 3` and seeing an immediate answer, try typing `N ← 5 + 3`. This time you'll see no immediate response, but APL will have assigned the value of `5 + 3` to the variable `N`, as you'll see if you now type and execute `N`.

One of APL's most powerful features is its support for matrix operations, which is uncommon, even today, except in specialist maths languages like MATLAB. Let's look at an example. The command `M ← 0 1 1 0` will assign those four values to the variable `M` which, because we've not specified otherwise, will be a one-

Here we're running an introductory BASIC program on the IBM 5110 emulator, but the APL language was that machine's most unusual feature.



» APL "PROGRAMS"

You might have concluded that APL is an environment for sophisticated mathematical manipulation but definitely not a programming language. Certainly, many people use it as the former, but it's a language and applications genuinely are written in APL.

As a first step, execute the following, which defines the function Minimum `Minimum ← { / ω }`. We'll leave you to discover what the `/` does (*annoys the art editor?—ED*), but the `ω` is the argument, which appears after a call to Minimum. Now try executing `Minimum 6 8 10 2 4` and you'll discover that APL gives `2` as the result. You might not think of a single line function definition as a program, but it's the closest we've come so far. So much can be done in a single line of APL, so function definitions will commonly be a single line. But if you want to make a function definition more readable, you can use several lines as the following example shows, and you'll discover that it does just what the name suggest:

```
MinTimesMax ← {
  Min ← / ω
  Max ← / ω
  Min × Max }

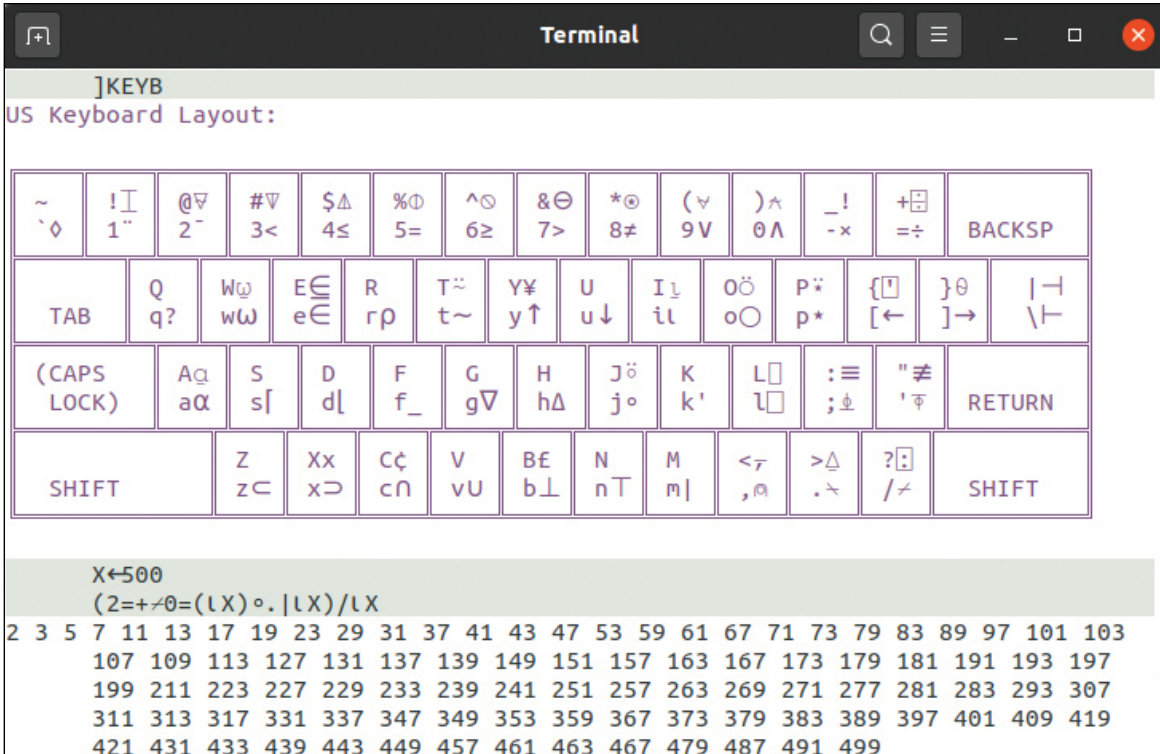
```

We found no way of editing multi-line function definitions in TryAPL, and we have to admit that, while it's great if you're just starting out, there are some drop-offs. In time, therefore, you'll probably want to migrate to a more fully featured APL such as GNU APL.



CREDIT: Rex Swain

As well as the IBM 5110, APL was used on larger IBM machines such as mainframes. This golf ball was used in the hard copy terminals often used with APL on these computers.



GNU APL can display the APL keyboard layout. Also of interest is the single line of code that's displayed all the prime numbers up to X.

QUICK TIP

The names of early languages often said something about them, so FORTRAN is Formula Translation. Not so today, with one popular language owing its name to Monty Python's Flying Circus. And C was so called because it was a successor to B. So what about APL? Well it genuinely stands for A Programming Language.

dimensional array, i.e. a vector, as you'll see if you execute that command. Type `M` to see what it contains.

However, if you now try `M ← 2 2 ρ 0 1 1 0` (note that `ρ` is a special APL character, not the letter p), you'll discover that the values are now in a 2×2 array, which APL will format as such when it's displayed.

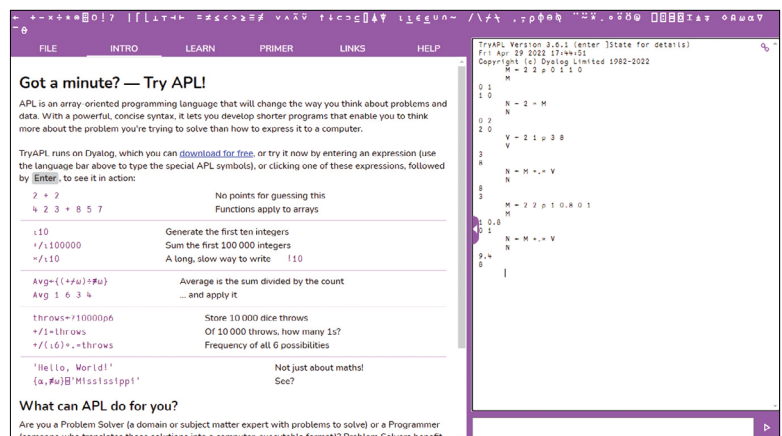
Now let's try a bit of matrix arithmetic via the command `N ← 2 × M`. Because `×` (not the letter x) is APL's symbol for multiply (see Unlearning the Rules box, page 59) this multiplies the vector M by two using scalar arithmetic. Or, in other words, it multiplies each of the values in the matrix M by two, as you'll discover if you give it a try. However, two matrices can also be multiplied together, as we're about to see.

Called the dot product, this way of multiplying two matrices together only works if the number of columns in the first matrix is equal the number of rows in the second. The implication of this is that the order of the two matrices is important, unlike the case with scalar multiplication. In addition, the result will have the same number of rows as the first matrix, and the same number of columns as the second. To see it in action, we'll create a vertical vector, which represents the X and Y coordinates of a point in space, using the command `V ← 2 1 ρ 3 8`, and check it by typing `V`.

Now, to calculate the dot product of the matrix M and the vector V, execute `N ← M +. × V`, and you'll discover that dot multiplying V by M has flipped its two values or, in other words the X- and Y-coordinates have been exchanged in the vector N. Admittedly, there are easier ways of swapping a pair of numbers. However, the two values in V are the X- and Y-coordinates of 2D space. In that case, therefore, the dot product has reflected the point across the diagonal. In fact, pretty much any type of transformation, or combination of

transformations, including reflections, shearing, scaling and more can be done using an appropriate transformation matrix M. It's commonly used in computer graphics. Finally, on this subject, you might like to know why dot multiplication gives the result that you've seen. We'll leave you to read up on the details, but the process involves both multiplications and additions which, presumably, is why it's denoted by `+.`. In particular, our dot product of a 2×2 array and a 1×2 vector involves four multiplications and two additions.

We haven't even see anything that could be thought of as an APL program, as opposed to a single line statement. That was deliberate because we wanted to underline its "glorified calculator" status, and we can't hope to cover everything in just four pages. However, if you've got the APL bug and want to learn more, be sure to look at the APL "Programs" box (opposite). **LXF**



TryAPL isn't a fully featured APL implementation, but it's easy to use so you might prefer it for your first forays into APL.

» GET MORE ODD LANGUAGE... Subscribe now at <http://bit.ly/LinuxFormat>

BACK ISSUES » MISSED ONE?

ISSUE 290

July 2022

Product code:
LXFDB0290



In the magazine

Expand your skill-set by exploring the formidable hacking tools within Parrot Security OS. Get the best fully-featured CMS for to host your web content on. We also explore sensors for the Pi, control Docker over the internet, emulate the Commodore 16, get to grips with FreeCAD Sketcher's Workbench, and learn the history of Pop!_OS from System76 developer Michael Murphy.

ISSUE 289

June 2022

Product code:
LXFDB0289



In the magazine

Discover what's new in the latest version of Ubuntu with our guide. We bring you a hands-on review of Valve's Steam Deck, rate five open source app stores, explain how to deep-clean your hard drives, get more from the Windows system layer Wine, run virtual machines with ease and emulate the Amstrad PCW. Plus we test your reflexes with a Python-based reaction game!

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.

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
p16

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!



» 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

SONIC PI

Credit: <https://sonic-pi.net>

Make beautiful algorithmic music

Create royalty-free music automatically, or try your hand at producing music by coding. **Mike Bedford** explores both composition options...



OUR EXPERT

Mike Bedford never progressed beyond piano lessons as a child. Despite that, music production holds a fascination with him, and all the more so if it's done digitally.

To say that digital technology has revolutionised the world of music is a massive understatement. Starting with the introduction of the audio CD in 1982, and continuing through to today's streaming services, for the vast majority of people, listening to music is a digital experience, despite the recent vinyl revival.

And the change has been no less dramatic in the realm of music composition, performance, recording and post-processing. Music notation software has largely displaced pen and paper and, let's face it, when did you last hear of music being recorded on a multi-channel analogue tape recorder? Our subject here, though, falls between these two extremes of music consumers and professional musicians. Here too, digital technology comes to the fore in the form of what we're calling algorithmic composition. And even if you don't need royalty-free music, you'll still get some hands-on experience of this intriguing type of software.

A history lesson

Let's go back to Austria in the 1790s. Attributed to Wolfgang Amadeus Mozart, *Musikalisches Würfelspiel*, which means *Musical Dice Game*, involves generating waltzes from the repeated rolls of two dice. Like most schemes for algorithmic composition that followed, despite the random element, the rules ensured that the laws of music were upheld. What's more, it's in keeping with our aim of enabling non-musicians to create music. As Mozart says in the introduction to his creation, it

enables someone, "to compose without the least knowledge of music so much German Walzer or Schleifer as one pleases, by throwing a certain number with two dice". And just in case you're wondering, there are 759,499,667,166,482 unique compositions.

Implementing *Musikalisches*

Würfelspiel in software has commonly been used as a programming exercise so you might like to pick up the gauntlet. If so, you can download Mozart's original rules at <https://bit.ly/lxf291mozart> and, fortunately, the text appears in English as well as German. Alternatively, if you don't want to reinvent the wheel, you can get a feel for this early form of algorithmic composition using the JavaScript implementation at <https://bit.ly/lxf291mozzier>.

Abundant music

If you want truly abundant music, not just the meagre 760 trillion tunes offered by Mozart's dice rolling approach, you should check out *Abundant Music* at <https://pernyblom.github.io/abundant-music>.

The algorithm is driven by a song seed, and you can either enter your own or ask it to generate one at random. And from repeated pressings, it appears that the seed is 32-bits long, so that gives us 4,294,967,296 different seeds. That's far fewer than Mozart's method offers, but we're not done yet.

You could just enter a seed and press Compose, but beyond this there's scope for customising the music to better match your requirements. In fact, there are plenty of supplementary seeds that between them provide 2,432-bits worth of randomness. Beyond that there are 80 user-definable parameters and so it goes on. Let's just say that takes us way beyond the number of atoms in the known universe – and they're not all waltzes!

Given those mind-blowing numbers, you might wonder how you can possibly get your head around *Abundant Music's* capabilities. Even though lots of the settings have meaningful names, such as "major scale probability" or "tempo range", to start we suggest that you try to get a feel to what's on offer by altering settings almost on a whim. After all, while you'll probably understand what "major scale probability" means, others such as "melody shape bias ranges" are far from obvious.

First, define your selection of seeds and parameters. Next, click the Compose button. Then click the Player button before selecting the play icon. Your music will appear on-screen, not as a musical score but as a representation that's reminiscent of a coloured version of the perforated paper rolls that were used to control



Long before digital technology gave us computer generated music, Mozart was creating random waltzes using a pair of dice.

CREDIT: <https://en.wikipedia.org/wiki/File:Croce-Mozart-Detail.jpg>



When they called it *Abundant Music* they weren't joking. Create any amount of music to your heart's content with this remarkable algorithmic composition tool.

old-fashioned pianolas, otherwise known as player pianos. The horizontal position represents time, while pitch is represented vertically. The representation scrolls as the tune is played, and the current notes are highlighted in white. You'll also hear the music, although it warns you that the player is a work in progress.

You'll probably go through this sequence several times until you're happy with your composition. Next, click Export to be taken to a screen where you can select yet more options before clicking Export Midi. You can now listen to this outside of *Abundant Music* and, if our experience is typical, it'll sound better. Using your preferred software, you can then use the Midi file generate an audio file, such as an mp3.

A noisy alternative

Our next topic is a bit of an aside because the algorithm doesn't exactly generate music. Instead, it creates so-called ambient noise. Algorithms for generating

ambient noise are so much simpler than for creating genuine music, so you're not going to be pondering how these utilities could possibly work, as you might with *Abundant Music*, but you might find it useful, nevertheless. After all, like more conventional music, ambient noise can be useful as a background soundtrack to games, videos or websites. You might also find it useful because many people listen to ambient noise as a relaxing or de-stressing experience, and even claim it makes them more productive at work by obliterating distracting sounds including traffic, phone calls or chattering colleagues.

myNoise, which you can find at <https://mynoise.net>, is one of the best respected ambient noise generators. There seems to be no way of creating ambient noise from scratch, but there are dozens of potted soundscapes, which are each composed from 10 different audio tracks. And while the tracks within a soundscape are fixed, you can alter the tempo and,

QUICK TIP

The concept of evolutionary music composition is a clever idea. It involves algorithmically creating tunes at random – they don't have to be good – and asking the user which they prefer. The best ones are then merged to create a brand new tune.

»» THE APPEAL OF LIVE CODING

We've looked at *Sonic Pi* as a tool for composition, but it can be used in a quite different way which its users call live coding. We can think of it as a bit like a jazz jam session – in which the artists improvise their performance – but in the digital domain. This is specifically supported in *Sonic Pi* by enabling users to edit the code without interrupting the performance. To learn more, we put some questions to *Sonic Pi* creator Sam Aaron.

So what skills were the most important for a live coder – musical or coding? Sam's response was, perhaps, surprising. "In order to get started having fun with live coding you need neither musical nor coding experience. Instead, it's much more important to have an open mind,

lots of patience and an experimental attitude. *Sonic Pi* is best learned through play and exploration – having fun trying



In this Summer School, live coding using *Sonic Pi* was used in conjunction with conventional musical instruments. Credit: Raspberry Pi Foundation.

new things out and surprising yourself with new sounds."

So what type of audience most appreciates a live coding event? According to Sam, "live coding events can be a lot of fun and can be inherently interesting to technically-minded people, especially if the code is projected so the audience can read it." That's not essential, though. "Just like you don't need to know how to make your own guitar to enjoy a rock band, live coding events can be enjoyed by everyone. I believe it's only a matter of time before we see live coding techniques in more mainstream contexts – there is just too much potential and expression in code as an instrument for it to be ignored."



QUICK TIP

Want access to random algorithmically generated music while you're not sitting at your PC? In that case, you need SoundHelix. Certainly you can run it as a Java app from a PC, but alternatively you can instruct a suitably equipped device with a verbal request such as "Alexa, open SoundHelix and play a song."



As an aside from genuine music, MyNoise enables you to create ambient noise to help you relax.

probably more importantly, you can use sliders to specify the volume of each tracks, potentially cutting some out entirely.

Sonic Pi

We're now shifting emphasis. So far, we've looked at utilities that do a useful job of creating music or ambient noise for various projects. Some of that software is remarkably sophisticated and, while a degree of musical knowledge is useful to get the best of it, we don't get to learn much about how it works. If you're equally interested in the technology, though, and if you enjoy coding, our next topic might be just what you're looking for. Specifically, we're now going to see how to generate music, algorithmically, by writing your own code, albeit potentially simple code.

The software we're using is called Sonic Pi which, as you can probably guess from its name, was originally designed for algorithmic music composition on the Raspberry Pi. However, it's also been ported to several other operating systems, including Linux, so the choice is yours. However, while it'll either be pre-installed on Raspberry Pi OS or is easy to install, if our experience is representative, the versions you'll find in some Linux repositories don't work. All that means, though, is that you'll need to compile it yourself using the simple

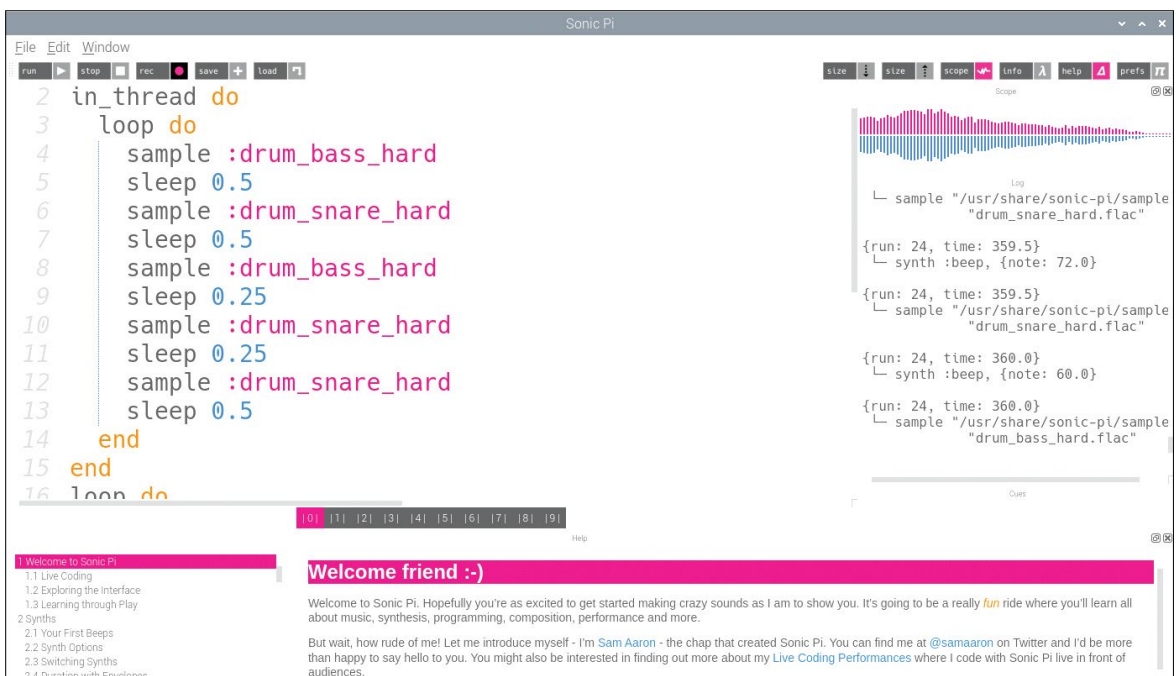
instructions provided.

Sonic Pi has been called a new form of musical instrument, and we can think of it as a combination of synthesiser and scripting language. It's often stated that Sonic Pi is a live coding environment, and we look at how it's used in that context (see boxout, page 65), but it can also be used for algorithmic composition.

When you fire up Sonic Pi, you'll see a large area in which you'll write your code, a smaller area where messages will appear, and several buttons. One of the best ways to get up to speed is to try it out by writing and executing some simple code. If you're a Sonic Pi newbie you won't have used its language before, although it's based on Ruby with extensions related to music. Initially, though, you'll be using just a handful of instructions which you'll soon learn.

Our recommended first step is to type `play 60` and click Run. You should hear a note; in fact it's middle C, because 60 is the MIDI number for that note. You can confirm that by using the alternative representation of a note which is its musical name and the octave number so, if you try `play :c4` instead of `play 60` you'll hear exactly the same note. To play a tune, you might think it's only necessary to write some code with multiply play statements one after the other. However, if you try playing `c4, e4, g4` and `c5` that way, instead of it playing an arpeggio (the musical term for a spaced-out chord), it plays them all together or, in other words, we hear what we normally think of as a chord. Of course, chords are useful, after all you'll probably aspire to play a tune that sounds rather better than what you might have played in your first piano lesson. Even so, we definitely need to be able to play sequences of notes, or sequences of chords for that matter. This is achieved by adding a `sleep 1` instruction between each play instruction, or between each group of plays for chords.

What we've seen so far isn't too different from sitting at a piano, or on-screen keyboard, and trying to knock out a tune if you don't know how to play a keyboard instrument. In other words, it's a trial-and-error exercise and, what's more, doing that on a piano keyboard is,



For true control of the algorithmic composition process you need a tool that's driven by a scripting language – enter Sonic Pi.

arguably, easier.

However, there's a big difference, which gives *Sonic Pi* a huge advantage, if you're just starting out in music composition. As soon as you've played a sequence of notes on a piano they're gone, so if you want to play them again, with a view to adding to your composition, you have to remember what you played previously. With *Sonic Pi*, though, the notes you've just played will be shown right there on-screen, so you can play them again, or add to your composition, and you can save it to use later of course. Alternatively, if one note in the sequence isn't quite right, it's easy to change that errant note, while keeping all the others in place.

Okay, so *Sonic Pi* offers the benefit of enabling you to repeat or edit your music, but it can do a lot more than requiring you to code every single note individually. As with pretty much any language, you can use loops in *Sonic Pi* and that's something you're going to be doing extensively. Unless you use a random number in the loop, it'll sound the same on every iteration, of course, but there are times when that's exactly what you need. A repeating melody might get a bit tedious if you overdo it, but a rhythmic accompaniment can certainly withstand quite a bit of repetition. To get a feel for that, try the following code:

```
loop do
  sample :drum_bass_hard
  sleep 0.5
  sample :drum_snare_hard
  sleep 0.5
  sample :drum_bass_hard
  sleep 0.25
  sample :drum_snare_hard
  sleep 0.25
  sample :drum_snare_hard
  sleep 0.5
end
```

Try it out and, hopefully, you'll be impressed that you've generated a never-ending rhythmic accompaniment. You're probably not going to want it to run forever, though, so limit the number of repetitions by replacing `loop` with `4.times` or similar. Alternatively, you could have several drum loops in an outer loop. You'll also have noticed that we're not using the `play`

instruction. Because percussion instruments can't usually play pitched notes, we used the `sample` instruction. This plays a pre-recorded sample of which *Sonic Pi* has about 130. They're not all drums, and you can also record and add your own.

Having a rhythmic accompaniment is great, but so far we've not seen any way to play the melody it'll accompany. After all, if you add code for the melody after the drum loop, it'll play after the percussion or, if the percussion is in an everlasting loop, the melody will never run. The solution is to use the `thread` feature. To turn our previous example into a thread you need to add an `in_thread do` before it, and an `end` after it. Now, whatever follows the loop (which can be a never-ending loop) will play at the same time.

The term "barely scratched the surface" is more than appropriate here, both to algorithmic composition in general, and to *Sonic Pi* in particular. However, if we've inspired you to learn more, and hone your skills, then we'll have succeeded in our aim. Your creations might not compare favourably to those of Mozart or Brian Eno (see *Generative Music boxout*, opposite), but we hope you'll have plenty of fun trying. **LXF**

QUICK TIP

The online **Muzoti utility** at www.muzoti.com is worth a look. You don't get nearly as much control as you do with **Abundant Music**, but you might consider that a good thing!



CREDIT: How We Get To Next, CC BY 3.0, www.youtube.com/watch?v=57tqnddgF2Q

Brian Eno's musical journey has taken him from glam-rock in the 1970s to digital techniques including creating generative and ambient music.

» GENERATIVE MUSIC

Recorded music sounds exactly the same each time we listen to it, while a live performance will never sound exactly the same twice, even though the changes will be subtle. The idea of music that's not set in concrete was taken forward by Brian Eno – former member of the 70s glam-rock band Roxy Music – and first brought to the public's attention with his album, *Generative Music 1*, in 1996.

Like any other album, Brian's creation contained several named tracks, each of which played for a specified time. But it

wasn't supplied as an audio CD, or a vinyl record, and download and streaming services had yet to see the light of day. Instead it was PC software that was distributed on a floppy disc.

The software implemented an algorithmic composition algorithm, driven by random numbers, but that didn't mean that a track would be totally unrecognisable each time it was played. For despite that randomness, Brian only gave each track so much leeway, having defined the basic musical style and

structure himself. Brian has been quoted as saying that this is the future of music, and that our grandchildren will be amazed that our generation could have been satisfied with recorded music that always sounded exactly the same.

So was generative music just a flash in the pan? Apparently not. More recently Brian has teamed up with musician and programmer Peter Chilvers on various projects. For example, you might like to take a look at their *Bloom* app, available for Android or iOS.

» GET US DANCING TO YOUR TUNE Subscribe now at <http://bit.ly/LinuxFormat>

CLONEZILLA

Credit: <https://clonezilla.org>



Back up and copy entire hard drives

Michael Reed shows you how to tame Clonezilla, a powerful piece of software that can help copy and restore hard disk partitions.



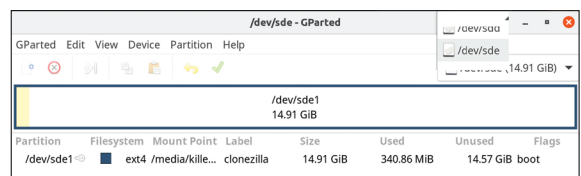
OUR EXPERT

Michael Reed wishes he could use Clonezilla to clone himself when he's got multiple deadlines to meet.

Clonezilla is a piece of software that can copy and restore (and destroy, if you're not careful) partitions and entire hard drives. Most backup programs copy files, but Clonezilla works by copying the actual disk blocks, making a perfect copy of the entire state of the system. The advantage of this approach is that, because the entire disk is copied, there's no need to reinstall the operating system when restoring from a backup. In fact, you can even backup a complicated setup such as one with multiple partitions with multiple operating systems (Linux, Windows etc) if you choose to back up the whole disk at once.

Clonezilla is itself a small Linux distribution that you boot into to use. Overall, we rate Clonezilla as a fairly easy-to-use tool (for Jonni - Ed), if you can make some notes to keep track of which drive is which, and are prepared to ride the cursor keys in a text mode interface. Clonezilla is efficient, and it runs close to the maximum speed of the hard drive while its working. It only copies the blocks that have data on them, and it compresses the resultant images in real time. If you were going to use it in a system backup role, you could possibly have a stricken system up and running again within, say, half an hour.

You can restore on to a system that's dissimilar to the one that the image was created from. For example, you could use Clonezilla if you wanted to upgrade to a larger hard drive without having to reinstall your system. To do this, you could create an image of the system (stored on an external hard drive), swap in the new hard drive and then restore the image. You can even restore an image on to a different computer. You can also store the images on a network resource such as a NAS



GParted uses a drop-down menu to choose between hard drives. Be extremely sure that you're operating on the right drive!

(Network Attached Storage) drive or even another computer if it can be accessed through standards such as SSH, NFS or Samba.

Clonezilla also has a role to play when setting up new systems. You could install a new system, clone it and then use that image to install on to various computers.

When we refer to Clonezilla, we're actually talking about Clonezilla Live, one of three versions of Clonezilla. Clonezilla SE is a dedicated server for deploying Clonezilla images over a network. The SE version can even support multiple computers booting over a network for mass deployment of images. Clonezilla Lite Server can also be used for less-elaborate mass deployment, and it's built into Clonezilla Live.

Let's look at a typical usage scenario, and see how you'd tackle it using Clonezilla Live.

The basic setup

Copying an entire disk to an image is a common job for Clonezilla, and it's the simplest example of what Clonezilla can do. Here, we'll assume that you have a blank USB flash drive on to which you can install Clonezilla and an external hard drive on which you can store backup images. We'll also look at what it takes to restore the image to a physical hard disk afterwards.

If you wanted to, using the method we're going to outline, there's no reason why you couldn't use a single USB storage device as both the Clonezilla boot medium and to store the hard disk backup images. If so, skip the partitioning advice in this next section, but carry out the file copying. When considering how much space you'll need to store the image of the drive, reckon on needing around 50 to 100 per cent of the used space on the source drive, because the data will be compressed.

We backed up a fairly recently installed Ubuntu installation to test Clonezilla. Like most modern Linux

This is Clonezilla in remote mode. We've selected the drive, and now Clonezilla is waiting for the other machine on the network to connect.

```
feeding partition /dev/sda1 in listen mode with port 9015...
Using partclone to clone...
Running: partclone.ext4 -z 10485760 -L /var/log/partclone.log -c -s /dev/sda1 -o -2 /dev/null
zstdmt -c -3 | nc -l -q 0 -p 9015 &
*****
Now you can boot the destination machine via Clonezilla live, then enter clonezilla mode, and c
"remote-dest" then follow the wizard, or you can run the following command(s) on the destinat
chine to start the remote cloning:
sudo su -
ocs-live-netcfg (Configure network first if necessary)
ocs-onthefly -s 10.0.2.15 -d [TARGET_DEV]
TARGET_DEV example: sda, sdb or...
For example, if you want to clone the disk to hda on the target machine, you can run it on the
t machine like:
ocs-onthefly -s 10.0.2.15 -d sdc
*****
Checking if udevd rules have to be restored...
*****
Waiting for the target machine to connect... _
```

installations, this one consisted of more than one partition, the largest of which occupied 11GB of space on the disk. The resultant image files are stored in a directory and consisted of a collection of 28 files that took up 6GB of space – a 45 per cent reduction.

Prepare to flash

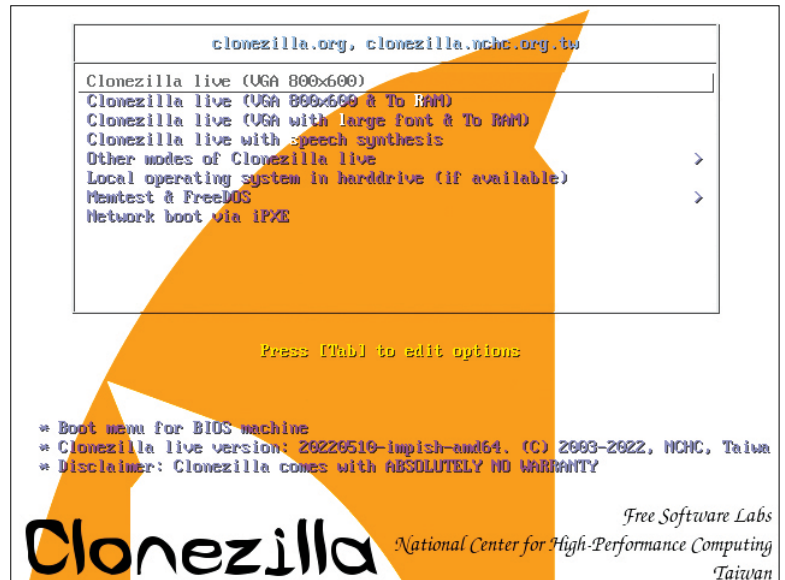
The first stage in this example is to prepare a USB flash drive to boot Clonezilla from. Obtain the latest Clonezilla build by downloading it from the official site (<https://clonezilla.org>). Next, you need a medium such as a USB flash drive to copy the files to. Because Clonezilla is pretty small at less than 1GB in size, any old flash drive you've got lying around should do the job.

To prepare it, remove the existing data and partitions and create a FAT32 formatted partition. In this example, we'll use *Gparted* (<https://gparted.org>), a GUI partition editing tool that's in the repositories of most Linux distributions. Start by installing it using your package manager front end. On most distributions, you'll need to supply your root password before carrying out operations using a partition editor.

We can't emphasise enough the importance of being absolutely sure that you're operating on the correct partitions. Measure twice, cut once. If you make a mistake, it's easy to destroy your entire system with a few clicks of the mouse! You have been warned. Make some notes on a piece of paper if you need to.

There's a drop-down menu in the top-right hand side of *Gparted*. Use this to select the flash drive. Typically, the flash drive will be the last drive on the list because it should be the most recently installed drive. Beyond that, you should be able to identify it by the size of the drive. For example, if you're using a 16GB drive, the total size of the drive should be around the 16,000MB mark. Typically, you'll have to unmount a partition (select partition and then select Partition>Unmount in the main *GParted* menu) before you can begin changing the partition setup on the drive.

Once you're sure that you've selected the right drive, click each existing partition, and then click the delete icon in the toolbar. This should leave you with a completely blank drive. Click the Create Partition icon, and create a single partition that fills the entire drive. Select FAT32 as the format. Give the partition a name, such as Clonezilla by editing the Label: (not Partition name:) field.



Click OK followed by clicking the tick icon to carry out the operation. Click the newly created partition and select (Partition>Manage Flags). In this menu, check the Boot flag and click OK to make the partition bootable. The easiest way of mounting the newly created partition from this point is to quit *GParted* and remove and reinsert the flash drive.

Use a file manager to open the archive that you've downloaded from the Clonezilla website and expand the files on to the flash drive. This is everything you have to do to prepare the Clonezilla distribution.

Clonezilla's startup screen. Apart from the first option, there are some other useful utilities that are handy to have on a keyring flash drive.

External drives

Once the flash drive is prepared and ready, it's worth readying your external drive that you'll use to store Clonezilla images. The drive that you use doesn't have to be blank, and you should be able to use any drive that's readable using a normal Linux distribution. For convenience, you should create a directory for image storage on the external drive (remember to avoid having spaces in the filename).

Later on, you might appreciate it if you take the time to make some notes at this stage about the sizes of each drive that you're going to be working with. This helps to identify which drive is which when you're using Clonezilla.

QUICK TIP

Because Clonezilla is open source it will, hopefully, continue to be maintained and older versions will always be available so that you'll be able to access your older backups.

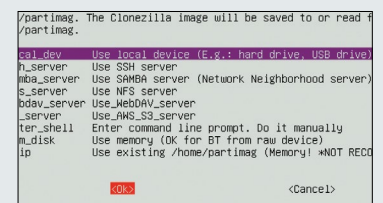
»» THAT USER INTERFACE

Clonezilla uses a user interface provided by a library called *curses* (the later, *ncurses*, in this case), and the overall look of the interface should be a familiar sight to veterans of an older generation of Linux tools. The implementation here is a fairly typical one. Use the cursor keys to move between options. When you have selected the intended option, press Return.

There's another way of enacting a selected feature that sometimes has to be used. Select an option using up and down cursor keys, as before, and then press Tab to move

the selection cursor on to the bottom part of the interface. You can then use the left and right cursors to move between the two buttons, usually Done or Cancel.

Do note that what Cancel does can vary according to the operation being carried out, sometimes skipping a stage, sometimes moving back a stage. It would be better if it behaved consistently or a Back button was always present. As it stands, unless a restore or copy operation is in progress, you can always power down and reset the computer to get back to the beginning again.



Sometimes, you have to press Tab to get the cursor into the bottom of the screen before pressing Return.

QUICK TIP

Clonezilla has some advantages over file-based backup programs, but it's not right for every situation. For example, it's not currently capable of incremental backups.

On some occasions, you might need to transfer a system from a larger drive to a smaller drive. For example, you might want to go from a mechanical drive on to an SSD drive with less capacity. Clonezilla can't automate every part of this procedure, but you can do it by freeing up an appropriate amount of space on the original hard drive and then using *GParted* to reduce the size of the partitions so that they'll fit on to the smaller drive.

Boot into Clonezilla

Now we need to boot into Clonezilla. Reset the computer with the flash drive inserted into it. How you select a temporary boot device varies from machine to machine, but the instructions should be presented during startup on the boot screen. Finally, this is your chance to press the mystical Pause button (located above the better-known Page Up) to have time to read those instructions! It's usually something like pressing F10 or F12 and then selecting the boot device using a menu.

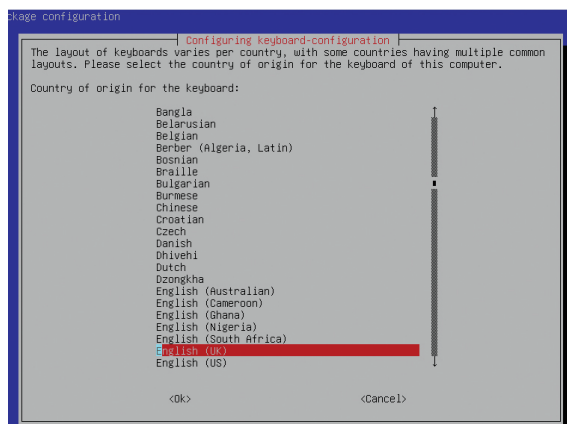
Hopefully, this places us on the startup menu screen of Clonezilla. Apart from starting Clonezilla, there are also some useful tools that are worth looking at, such as being able to boot into a command line and run *memtest*. In a pinch, you could use Clonezilla as a recovery boot disk to carry out simple maintenance and repair tasks from the command line. In most cases, the first option, to launch Clonezilla with 800x600 screen resolution, is the right one. So, press Return to select it.

See the box (page 69) for some tips on how to operate the text mode user interface of Clonezilla. The next section is a series of text menus to select the keyboard type, and it's usually worth getting this right so that all of the symbol keys are the correct ones.

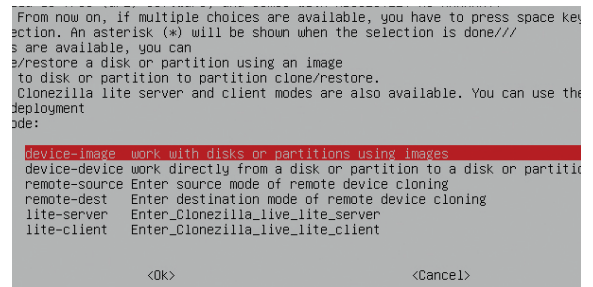
For a typical UK keyboard layout, select Change Keyboard Layout when prompted, and then accept the default options for everything in this section other than the keyboard language (select English (UK)). It's a shame that this has to be done every time, and perhaps a future version of Clonezilla could somehow deduce the keyboard language or store the setting somewhere? Finally, select the Start Clonezilla option when prompted.

Select Clonezilla Mode

Now we've arrived at the meat of what Clonezilla can achieve. There are six different modes of Clonezilla operation that can be selected from the following menu.



Selecting the correct keyboard type is usually worth it as the position of certain symbol keys is different on the default US keyboard.



Selecting the Clonezilla mode. There are six options to determine whether you're copying from partition to image or vice versa, and whether you're using either local or remote storage.

Notice that most of the Clonezilla menu options have a short name followed by a longer description of what that option does. Our stated purpose is to clone a hard disk to an image file and so we'll select the first option: Device-image.

Some of the other options here are worth examining. It's possible to have the source and destination on different machines that are connected by a network. This works by booting a copy of Clonezilla on both machines. On one machine, you select the Remote-source option and on the other machine you select Remote-destination. In both cases, you have to answer some questions about the network. Apart from that initial setup difference, most of the remaining process is the same as the one that we're describing here. However, we'll assume that you're working with images and partitions on a single computer for the rest of this example.

On the next page, select the option Local_dev. Again, the other options on this page relate to using non-local, network resources. For example, the image directory could be on an SSH or NFS accessible resource on the network such as another computer or a NAS (Network Attached Storage) drive. This holds true if you're either intending to copy a disk to an image or to go the other way around by restoring an image to a disk.

Now you have to press Return. This puts Clonezilla into a mode where it waits for you to insert any devices that you want to use for image storage. There's a text display at the bottom of the screen showing you what valid storage devices have been detected. When you've verified that your external storage is present and correct, press Ctrl+C to progress to the next stage.

Select the storage partition

Next, select the partition that you'd like to use to store Clonezilla images. This is possibly an area of Clonezilla that could be improved because it could display the label of the partition to make identification simpler. Refer back to your notes about the exact size of your external storage if you have any doubts. If you inserted the external drive most recently, it should be the last drive in the list. However, it's not a disaster if you were to make a mistake at this stage because you're simply storing files on the target drive.

You can run a check on the filesystem using the *fsck* tool, but it's not usually necessary. On the screen following this, you can select the directory to store the images. If you didn't create a directory when preparing the external storage, you can select Done to accept the default choice of the root (/) of the drive.

passwords scrawled on an A4 pad and hidden away in a locked drawer, along with kompromat pictures of the boss at last year's Christmas party.

This is pretty secure, but it's also time-consuming to dig out every time you need to access another resource with a new password. It's time-consuming to type in, and when you're using a randomly generated string as a password, it's error prone, too. If you lose the pad or Post-it Note then, well, that's it.

But this is 2022. Most modern browsers have built-in password managers. Every time you fill in a password field online, they'll helpfully ask if you want them to remember the password. Some will even back up your passwords to an online vault. But what if you lose your laptop? You can still restore your passwords from the online backup, but in the meantime an attacker is reading your emails, spending your hard-earned cash, and looking at swimsuit photos from your ill-advised Ibiza vacation.

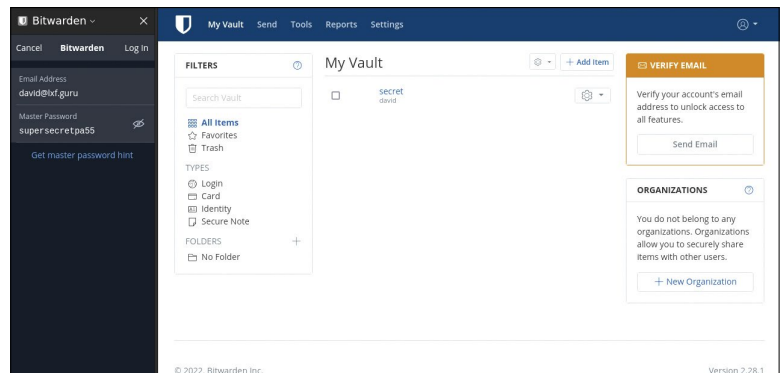
Leaving your unlocked machine unattended for even a few seconds can see a ne'er-do-well slip behind the keyboard and dump a zipped CSV file of your usernames, passwords and the corresponding websites on to a USB stick. In the interest of thorough research, this writer practised on his wife's laptop while she went to answer the door – it took about 20 seconds.

The other issue with browser-based password managers is that people use different browsers for different things. The laptop on which this article is being written has two separate *Firefox* browsers, *Brave*, *Chromium*, and *Surf*. These don't play nicely with each other. *Chromium* can't read the passwords saved on *Firefox*, for example. To quote the immortal and oft-repeated words of Frank Skinner, there must be something better than this.

Pass masters

Bitwarden is a free, open source self-hosted password manager that puts password security back into your hands. It's one of several similar FOSS offerings out there, including *KeePass*, *PassWall*, *PassIt*, and *Passky*. But *Bitwarden* is our favourite. It's easy to deploy and comes with killer features, including email alias generation, which we'll get to further down the page.

The advantage of your password manager being self-hosted is that it's accessible from anywhere, regardless



This writer's Bitwarden vault currently contains details for only one login. It's for a super-secret site and you're not allowed in.

the browser you're using and on which computer. Would you trust a company to keep all of your passwords secure and safe from prying eyes? We certainly wouldn't, so we'll do it ourselves.

As always, the first step on deploying any new software to your Virtual Private Server is to log in:

```
$ ssh user@your.vps.ip.address
```

Then install fresh updates to the system with:

```
$ sudo apt update && sudo apt upgrade
```

If you've been following this tutorial series from the beginning, you should already have installed a recent version of Ubuntu, along with Apache, Docker, Docker-Compose, *curl*, and a few other essentials. If you're joining us for the first time, grab the free introductory PDF from **LXF282**, and bring yourself up to speed (see *Quick Tip, facing page*).

Now go to your registrar's DNS management page to create a new subdomain (or entirely new domain if you're feeling flush), through which you intend to access *Bitwarden*. Select A record as the type, your VPS IP address as the value, and your chosen subdomain name as the host. For our *Bitwarden* instance, we chose warden as the subdomain, meaning that we can access the *Bitwarden* web interface at **warden.lxf.guru**.

Set the Time to Lease (TTL) value as low as possible, then go and make a cup of tea while you wait for the record to propagate. Visiting **https://dnschecker.org/#A/warden.lxf.guru** should result in an entire world of green ticks when the process has completed.

To quickly grab an SSL certificate, you should quickly

» MOSTLY 'ARMLESS

Throughout this VPS tutorial series, (which is regrettably but inevitably drawing towards a close, unless there's a huge outcry...), we've endeavoured to deploy software that's hardware agnostic. Yes, we're writing about deploying software on a 64-bit x86 server, but most of it will run perfectly well on the ARM-based Raspberry Pi 3 that you have gathering dust in the hidden bottom drawer of your bedroom armoire. In fact, this author has practically all of the published projects served for his own

pleasure from a Raspberry Pi 4 B that's nicely balanced on the (currently off) radiator behind his couch: *PhotoPrism*, a Fediverse server, *Mealie*, *Nextcloud*, *Jellyfin*, and a whole lot more.

Unfortunately *Bitwarden* doesn't play nicely with ARM64. This seems odd because the Raspberry Pi is one of the most affordable, competent, and widespread home servers in existence.

But just because *Bitwarden* doesn't love Raspberry Pi doesn't mean that dedicated ARM self-hosters are left

entirely out in the cold if they need to keep their passwords safe and secure.

KeePass supports ARM64 and is almost as fully featured as *Bitwarden*, lacking only the email aliasing option and coming without 'premium features' or any other limitations. *KeePass* server software can be installed with docker, and like *Bitwarden*, is a doddle to use. Other worthy contenders include *TeamPass* and *VaultWarden*, which is in fact based on Rust implementation of the *Bitwarden API*.

QUICK TIP

Wondering where the rest of the LXF VPS sites have disappeared to? We ran out of VPS space and nuked them to oblivion.

create an Apache conf file:

```
$ sudo nano /etc/apache2/sites-available/bitwarden.conf
and enter:
<VirtualHost *:80>
ServerName warden.lxf.guru
</VirtualHost>
```

Save and exit *nano* with Ctl+o and then Ctl+x.

Next, enable the conf with:

```
$ sudo a2ensite bitwarden.conf
```

and restart Apache with:

```
$ sudo service apache2 restart
```

Now use *certbot* to obtain an SSL certificate from Let's Encrypt:

```
$ sudo certbot
```

Certbot will ask for your email (you need to enter something in this field), confirm that you've read the terms and conditions (you have to agree), request that you sign up to the EFF newsletter (it's up to you), then ask which names would you like to activate HTTPS for.

Select your chosen domain name from a list and then hit Enter.

Once the process has completed, *certbot* will have

created a new conf file: **bitwarden-le-ssl.conf**. You'll be returning to this later. In the meantime, return to your home directory with:

```
$ cd ~
```

Bitwarden requires an ID and a key in order to install on your VPS and will send them to you via email. We're not 100 per cent keen on this feature, but it enables the devs to contact you about updates, and send push notifications to mobile and desktop clients. Oh, and it validates "licensing of paid features". Hmmmm...

Visit <https://bitwarden.com/host> in your browser and enter your email address. As soon as you hit Submit, the key and ID are displayed onscreen. They're not sent to you via email, so grab a pen and jot them down on a Post-it Note. Paste it to the side of your monitor – you'll need these details in a moment.

Download the *Bitwarden* installation script to your home directory with:

```
$ curl -Lso bitwarden.sh https://go.btwrn.co/bw-sh
```

and then modify the permissions with:

```
$ sudo chmod 700 bitwarden.sh
```

This means that the file owner (you) can read, write and execute the file, but no one else can.

Run the install script with:

```
$ ./bitwarden.sh install
```

Like Sir Lancelot approaching the Bridge of Death in *Monty Python and the Holy Grail*, you'll be asked three questions (none involving the air-speed velocity of an unladen swallow): the domain name of your instance, whether you want to use Let's Encrypt to generate a free SSL certificate, and the database name for your *Bitwarden* instance.

For our demo instance the domain name is **warden.lxf.guru**. We're using *certbot* and Apache, and we don't need to have *Bitwarden* interact directly with Let's Encrypt, so that's a no to the second question.

For the database name, choose a name and hit Enter. The script suggests 'vault', so we went with that.

The install script will now pull the relevant images from docker hub, then ask you for the unique installation ID and key. Paste them into the terminal.

Choose no when the script asks you if you have a SSL certificate to use, and no when asked if you want to generate a self-signed certificate.

Bitwarden will have created a new directory called **bwdata** in your home directory containing everything it needs to run. This includes a config file that we're going to tweak now in *nano* with:

```
$ nano ~/bwdata/config.yml
```

Because you'll be using Apache as a reverse proxy to access *Bitwarden*, change the **http_port** value to something other than 80. We went with **8888** because it's easy to remember. Remove the value for **http_port**. Again, Apache will be handling SSL, so *Bitwarden* doesn't need to worry about it.

The **ssl_certificate_path** needs to contain the path to your SSL cert. This was set by *certbot* earlier on, and should be:

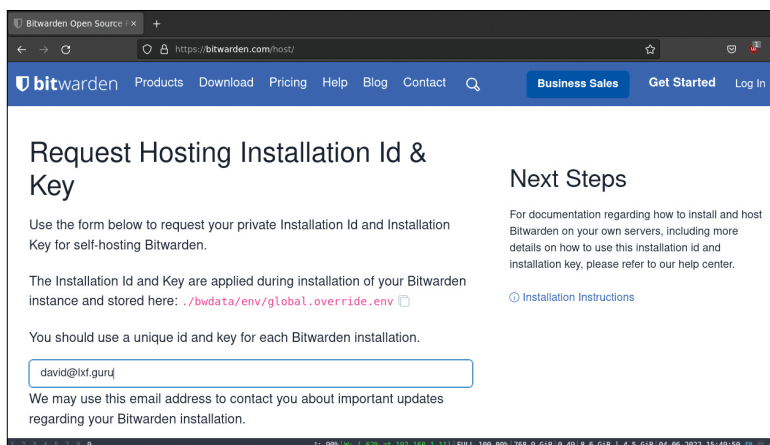
```
/etc/letsencrypt/live/your-domain-name/fullchain.pem
```

Meanwhile, **ssl_key_path** should be set to

```
/etc/letsencrypt/live/your-domain-name/privkey.pem
```

Save and exit with Ctl+o then Ctl+x, and then enter:

```
$ ./bitwarden.sh update
```



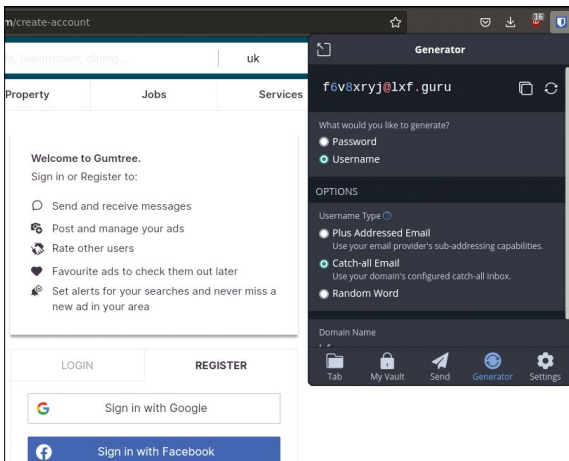
Having to provide your email address in exchange for an identifiable key somewhat undermines the point of independent self-hosting on a VPS in our opinion.

» AN EMAIL FOR ALL SEASONS

This is where our favourite *Bitwarden* feature comes in. Unlike many other self-hosted password managers, if you own your own domain (as you obviously do if you're using it to deploy *Bitwarden*), then the browser extension can create email aliases on the fly. You can then use catch-all email forwarding to ensure that emails are forwarded to your main account.

The first advantage of this is that trackers attempting to build a profile on you based on your email address will be thwarted. There's nothing to suggest that Presoak5036@lxf.guru is the same person as Shiny8246@lxf.guru or even Unbounded3922@lxf.guru.

Aside from thwarting evil capitalists who want to create a dossier of everything you do online, using a randomly generated email address for each site you visit helps thwart evil capitalists who sell your email address to spammers. With *Bitwarden* you can easily keep track of which email address you've giving to which site, and if it leaks then you can report it to the data commissioners' office and land the offending company with a huge fine. It's such a great feature that we've bludgeoned the old lxf.guru email server to death, just so we can use *Bitwarden*'s catch-all email feature.



BitWarden's aliasing can generate a completely random email address in two clicks and ensure that mail is delivered to your actual inbox.

Docker will then pull yet more images from the Docker hub. Once these have all been downloaded and extracted, to check if everything is running smoothly enter:

```
$ docker ps
```

One important thing to note is that *Bitwarden* requires 2GB RAM to run, so either make sure your VPS has that much available, or adjust your swap size.

At this point you should be able to type `<your-vps-ip-address>:8888` into the URL bar of your favourite browser and see the *Bitwarden* login page in all of its glory. Hooray!

The last thing you need to do is set up Apache to forward all incoming requests to **your-domain-name.tld** to port 8888.

```
$ sudo nano /etc/apache2/sites-available/bitwarden-le-ssl.conf
```

Erase everything and enter:

```
<VirtualHost *:443>
```

```
ProxyPreserveHost On
```

```
ProxyPass / http://localhost:8888/
```

```
ProxyPassReverse / http://localhost:8888/
```

```
ServerName your-vps-ip-address
```

```
SSLCertificateFile /etc/letsencrypt/live/your-domain-name/fullchain.pem
```

```
SSLCertificateKeyFile /etc/letsencrypt/live/your-domain-name/privkey.pem
```

```
Include /etc/letsencrypt/options-ssl-apache.conf
```

```
</VirtualHost>
```

Restart apache with:

```
$ sudo service apache2 restart
```

And that's it. Your *Bitwarden* vault is active and accessible through <https://your-domain-name>.

The keys to your kingdom

That wasn't too hard, was it? The first thing you need to do is create a master account with a master password which will give you access to your other passwords.

Click Create Account on the login screen and fill in the necessary details. You can choose whatever password you like, but be aware that this single password will grant access to everything everywhere,

so choose carefully – and make sure that you're capable of remembering it.

Bitwarden will warn you if your password is too weak, but it's not going to stop you. Once that's done, you can log into your account.

Somewhat predictably, you'll notice that there are no passwords saved in your vault. On the left of the screen, there's a menu giving the various types of secure credentials you can create and store on your *Bitwarden* instance. Passwords are great, but what about credit cards, secure notes, or even complete identities?

Sticking with passwords for now, click Add Item under My Vault, and choose Login from the drop-down. Pick a name for the password, and fill in your the username for the service you want to use, your current password, and the URL of the service. Clicking the recycling icon will create a password for you. Click Save and your password will be saved. You can access it by clicking the name in the My Vault section.

Browsing your passwords

That's all very well, but opening a new window and navigating to your *Bitwarden* instance every time you want to login to a website is a chore. Luckily for the security conscious web user, extensions are available for most popular browsers.

Open up the extensions or add-ons menu from within your browser, search for *Bitwarden* and install. An intrusive bar will appear on the left of your screen, inviting you to log in. Don't log in yet. Instead, click the cog icon to access the self-hosted environment fields. The only one that you really need to fill out is the server URL, and you should leave the custom environment fields blank.

You may have noticed a new icon on the top right of your browser – a tiny blue and white shield on a blue background. That's right! It's the *Bitwarden* logo. Clicking it will reveal your password vault, while other tabs will give you the option of generating user names and passwords for the page you are on.

BitWarden also integrates seamlessly with browsers and *Bitwarden* on both Android and iOS – even to the extent of autofilling passwords in your favourite social media tools. As this writer doesn't use social media, Android, or iOS, he can't comment on this, but the reviews are great. **LXF**

QUICK TIP

If you want to set up a catch-all email address so you can receive notifications from websites and reset your account details, you'll need to visit the Advanced tab of your domain registrar.

Create Account

Email Address

You'll use your email address to log in.

Your Name

What should we call you?

Master Password

Strong

The master password is the password you use to access your vault. It is very important that you do not forget your master password. There is no way to recover the password in the event that you forget it.

Re-type Master Password

Master Password Hint (optional)

A master password hint can help you remember your

The password for your master account is the key to every other password you've ever used. Ensure it's memorable and not guessable.

» **♥ PASSWORDS? MAKE ANOTHER...** Subscribe now at <http://bit.ly/LinuxFormat>

BARRIER

Credit: <https://github.com/debauchee/barrier>

Configure a cross-platform KVM system

Use Barrier to replace a physical KVM switch and enable a keyboard and mouse to be shared between multiple devices. **Matt Holder** explains all.



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.

QUICK TIP

Documentation can be seen on the project's website: <https://github.com/debauchee/barrier>.

How many times have you had multiple PCs, laptops and Raspberry Pis on your desk that all need to be used at the same time, but not the desk space to have a keyboard and mouse for each? And if you do start typing on a keyboard, it's connected to the wrong device – frustrating, right?

Barrier aims to ease these issues by enabling one keyboard and mouse to be shared between multiple devices. On some operating systems files can be shared between devices by dragging and dropping from one screen to another, as well as being able to share the clipboard. While writing this tutorial it was possible to copy and paste text between a Windows and Linux device. However, copying files between devices didn't work, either using drag and drop or copy and paste.

The software aims to replicate a hardware device called a KVM (Keyboard, Video and Mouse) switch. This device can accomplish the same task, but would need cables from each device to the switch unit as well as for a keyboard button to be pressed to switch the keyboard and mouse to another device.

Barrier is an open source project and was forked from an older version of the *Synergy* project, which is open source, but copies of ready-built executables are charged for by the team behind the project. Executables of *Barrier* for MacOS, various Linux distributions and Windows are all available from the project's GitHub pages as well as on Flathub, Snap store and the repositories of various Linux distros (see boxout, right).

Unfortunately, support isn't yet available for the Wayland (*noooooo!!!!–Ed*) display system, so Linux distros need to be configured to use an X session prior to logging in and starting *Barrier*.

Let's get started with the installation of *Barrier*. In this article, we'll discuss a Windows device and a Fedora Linux device sharing the keyboard and mouse, which is connected to the Linux device.

Barrier operates using a server and client model. The device connected to the keyboard and mouse is configured as the server. Devices sharing this keyboard and mouse will be configured as clients.

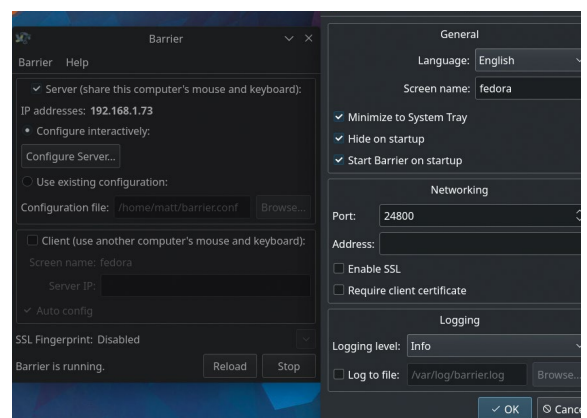
First, on Fedora, open a terminal and install *Barrier* using `sudo dnf install barrier`. Next, open *Barrier* from the application launcher and follow the wizard to configure it as the server side of the connection. Once

the wizard has completed, a small window will be displayed. This shows the *Barrier* panel and the fact that the server has been selected. Other options, such as keyboard hotkeys and screen locations, can be viewed under the Configure Server option. This will be visited later on in the tutorial. In addition, auto start can be enabled from the *Barrier*>Settings option. Once everything has been configured, click Start.

Barrier, meet Bonjour

Next, on the Windows device, download the latest version of *Barrier* from GitHub. At time of writing, this was 2.4.0. Install *Barrier* and ensure that the client option is selected. Open the software from the Start Menu and then follow the wizard to configure it as a client device. During the installation C++ redistributable libraries will be installed if required. When the wizard has completed, enable auto config and install *Bonjour* when prompted. From the *Barrier*>Settings menu make sure that the option is selected to elevate permissions when required. When completed, return to the server configuration panel on the server device.

On the Linux device, select the Configure Server button and on the Screens and Links tab drag a new screen to the grid, to correspond to the physical location of the screens on the desk. Double-click the newly created screen and enter the device's hostname in the Screen name box. This field is case-sensitive, so



Barrier configured as a server, with the software settings on show.

ensure it matches exactly. Change any options as required and click OK.

Within the Hotkey tab, define a shortcut key of Ctrl+Alt+1, which defines an Action to switch to the newly created screen. Press OK to close the dialog window and then Reload the settings. Now move the mouse between screens and experience the magic of your mouse controlling a different device that's running a different OS! The configured hotkey, when selected, will move the mouse directly to the chosen device. This saves a few seconds of finding where the mouse currently is and then moving it to the desired location.

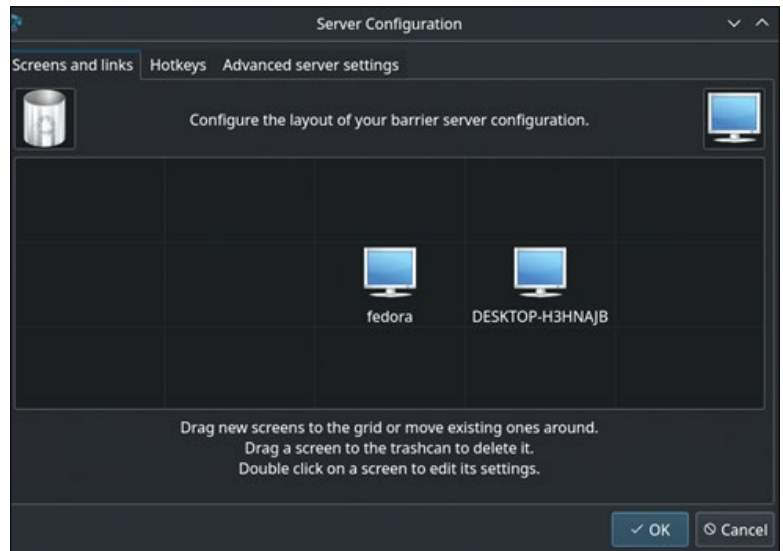
Solving any installation hiccups

Some troubleshooting was required during the installation process. When the defaults had been selected, the client and server weren't able to communicate with each other. Our first thought was to switch off auto-config. When this is deselected, the IP address of the server needs to be added. However, when the server was restarted the connection still couldn't be established.

Next, we switched off the firewall on the server side of the connection, but this didn't alleviate the problem either. Finally, by switching off the SSL option on both sides of the connection, the client was able to communicate with the server successfully. By troubleshooting in a logical manner and by changing one thing at a time (and undoing the changes that were unsuccessful), the underlying cause was able to be found. It's unlikely that the client side of the connection will require any firewall changes, but more likely that a firewall rule will need to be added to the server side. Hopefully, there aren't any issues when following along with this tutorial! Other troubleshooting tips worth knowing about are making sure that the same version of the software is installed on each device.

Another useful feature to talk about is called dead corners. This enables areas of all, or selected screens to be set to ignore the request to change screens. This is a useful feature because sometimes it's easy to mis-trigger the keyboard and mouse moving to a different device by accidentally moving the mouse to particular corners.

The scroll lock key is also useful. When this is activated it'll lock the keyboard and mouse to the



current screen. Deactivate the scroll lock key and the keyboard and mouse can span to other devices.

When configuring keyboard shortcuts, they can be set up in a number of ways. As previously configured, shortcuts can be created to switch the keyboard and mouse to a particular screen. A second option is to use a shortcut to move the keyboard and mouse to the screen on the left or the screen on the right. For every press of the hotkey, control moves to the next device along. Another powerful feature, which is included, is the ability to use a keypress to send a hotkey to all devices. For example, pressing Ctrl+Alt+2 could open the Launcher/Start menu on every device concurrently.

As has been shown in this article, *Barrier* is an excellent piece of software that solves a specific problem. This is a tool that will remain in the proverbial toolbox the next time this author's desk is covered in laptops, monitors and Raspberry Pis.

To further future-proof the software, enabling better support to copy files between OSes would be great, as would support for Wayland, which is becoming ever more popular. The project would appreciate any help in being able to implement some of the functionality that it's currently lacking, such as Wayland support and drag and drop file and clipboard sharing between all supported operating systems. **LXF**

Adding screens to the grid configures which direction the mouse needs to be moved to be able to control another device.

»» PACKAGING FORMATS

Over many years, different packaging formats have been developed for Linux distributions, and two formats lead the field. Debian-based operating systems (Debian, Ubuntu and Mint to name a few) use the .deb format. These packages are installed using the *apt* packaging system (for example, `sudo apt install barrier`). Operating systems from the Red Hat family (Cent OS, Fedora, Red Hat Enterprise Linux) as well as Suse-based distributions will use the RPM packaging

format and will use the *dnf* or *yum* software to install package (for example, `sudo dnf install barrier`).

Other distributions may use deb or rpm, or may have their own format (such as Arch). Historically, Windows hasn't had access to a package manager, which led to the creation of *Chocolatey* (<https://chocolatey.org>). Once *Chocolatey* is installed, *Barrier* can be installed using `choco install barrier` from a terminal running at an administrator level.

Homebrew is a package manager tool for Mac OS and a version of *Barrier* can be found here. Various releases can also be downloaded from the *Barrier* github page (see the Quick Tip, *opposite*). More modern package systems also exist that "containerise" the program to give a greater level of security. *Barrier* is also available using Canonical's Snap format as well as from the Flatpak store. Both containerised formats are supported by a large number of distros.



SEEING THE LIGHT

Three humble developers of Bodhi Linux enlighten **Jonni Bidwell** on the ways of Zen and the art of distro maintenance.



We're almost as guilty as some of our readers when it comes to hoarding old hardware. If it ain't broke don't throw it away, we always say.

However, tracking down a distro that runs on older hardware can be a challenge, and finding one that holds its own against mainstream desktops seems very tricky indeed. But it needn't be this way. Bodhi Linux will run on anything with 512MB of RAM and will occupy less than a gigabyte of its precious storage. There's even a 32-bit version that will probably run on

your dusty old Pentium III. Not only that, but Bodhi is pretty much unique in that it offers a fully configured *Enlightenment*-based desktop out of the box. This means you get speed, usability and a unique style.

For those unfamiliar with the lore, the *Enlightenment* (*E* for short) toolkit was for a long time a rival to GTK and Qt, and in many ways was ahead of both of them. It's still around today, now at version 25, but is used more by embedded offerings (for example, Samsung's Tizen) rather than Linux distros. Bodhi's Moksha desktop was forked from an older (and lighter) *E*

release, but a special *E25* BodhiDev release is available.

We were honoured to share some words with Bodhi lead dev Robert Wiley (aka ylee), Moksha guru Štefan Uram (aka the_waiter) and community and theming don Gareth Williams (aka hippytaff). These legends, and a handful of other community members, have been running things since original Bodhi founder Jeff Hoogland took a step back from the project in 2018, after eight years at the helm. So put some Tibetan bowl music on, take a deep breath and perhaps you'll even glimpse satori.

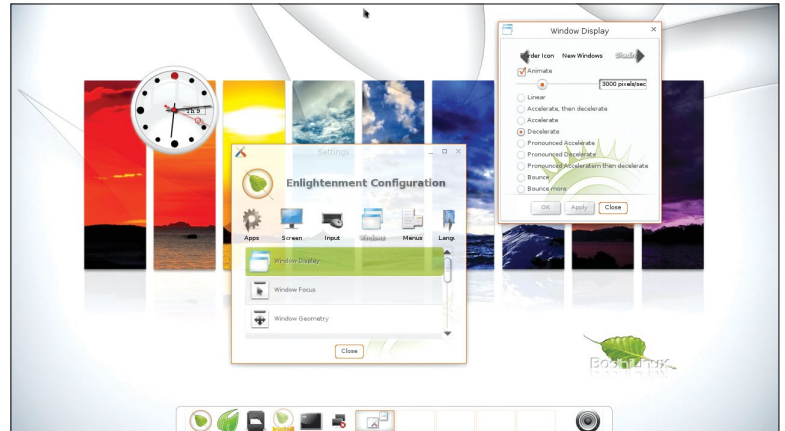
Back in 2011 mathematician and coder Jeff Hoogland had a vision. Not necessarily a Saul on the road to Damascus vision (*wasn't he blinded? – Ed*) but nonetheless one that encouraged him to share Bodhi Linux 0.1.6, the first official version, with the rest of the world. Jeff's vision was for an 'enlightened', minimalistic distribution free of bloat, but without sacrificing elegance or usability. Very quickly the distro gained traction, and soon Jeff had a veritable band of coders, artists and translators helping him on the project. Against a backdrop of criticism for 'modern' desktops such as Gnome 3 and Unity, Bodhi's early popularity was surely in large part due to its unique use of an *Enlightenment*-based desktop.

Beating the bloat

Enlightenment (a toolkit and window manager) had been around in some form since 1996, and while it was popular with people who liked to configure everything for themselves, no one had yet packaged it into a fully configured desktop. By doing exactly this, Bodhi tapped into the growing demand for bloat-free distros with sane defaults that didn't hide configurability from users. If you've never seen *Enlightenment*, a first glance will show you why people are so fond of it. Fans of old-style desktops will enjoy windows can be 'rolled up' into their titlebars, effects that are subtle but unique, and diverse tray and screen widgets. Oh, and it did what newcomers Gnome 3 and Unity definitely did not: adhere to the traditional desktop metaphor. It's commonly referred to as "the original eye-candy window manager".

As the *Enlightenment* project developed into a more complicated beast, Jeff realised that its fitness for Bodhi was at stake. Bodhi 3.0 shipped in 2015 with *Enlightenment 19 (E19)*, but devs and users weren't happy with it. Even its predecessor *E18* was criticised for having unnecessary bloat and breaking compatibility. So Jeff decided to fork the 'last good' *E17* release, backporting some of the new features but none of the cruft. That fork became known as Moksha (a Sanskrit word for enlightenment, liberation and such), and it's been at the heart of Bodhi ever since.

Gareth Williams points out that Moksha provides "superb performance on older hardware without compromising on delivering a modern look, fancy bling, animation and compositing, for those that like that sort of thing". He's happy to give up his free time (he's a



mental health professional by day) to Bodhi because it "allows schools, organisations and individuals in poorer countries to use and own computers that nothing else will run on".

You'll find Gareth (hippytaff) on Bodhi's (ever-buzzing) Discord server. In his words, "I believe that a welcoming and friendly community is just as important as the stability and usability of the distro, and we pride ourself on both. We don't abide meanness or any kind of RTFM attitude and welcome anyone who wants to be part of Bodhi. Regardless of the amount of Linux knowledge or experience they have. Anyone reading this that's looking for a friendly corner of the internet to enjoy FOSS and Bodhi particularly, don't hesitate to join our forums and community. You'll be made to feel very welcome. Every single user and community member is valued and there's always someone on hand to provide technical support. EveryBodhi is welcome (grimaces)." We agree. Linux needs a more welcoming attitude.

Besides helping on the chat and forums, Gareth has been learning the sinister ways of EFL theming. "We use *edje* for theming, lots of .edc files compiled into .edj binaries which makes them fast and light, but a huge PITA to develop and maintain. You can decompile one by doing `decc theme.edj`, which is the name of the file and you'll see how cumbersome and difficult they are to make head nor tail of. I developed the *vice versa* theme and it was a month of headaches, and that was just converting a different themes colours, essentially."

Stefan Uram (the_waiter) has been around since the early days. Believe it or not, before he was a Bodhi

Bodhi 1.4.0 from 2011 looks rather svelte, even by today's standards. We can even configure the fade acceleration.

» WHAT'S IN BODHI?

Bodhi has always been based on the LTS Ubuntu releases, but while those ISOs have grown from 1.5GB to nearly 3GB, Bodhi's have always been in the hundreds of megabytes. We've said that there isn't much in the standard Bodhi release. But that's by design, and what there is, is awesome. The default browser switched from *Midori* (the lightweight effort that used to be part of Raspbian) to *Chromium* a couple of versions back, though the team recommend *Pale Moon* for those seeking something lighter.

What we're perhaps a little too enamoured by is *Terminology*. This was created by the *Enlightenment* team "purely because they could", and makes it possible for everything you can think of and more to be themed, customised or otherwise tweaked. We don't usually like an audible terminal bell, but we spent much more time than we should preparing this screenshot so we could show you Bodhi's awesome visual terminal alert. Bodhi also contains an online app center where curated

applications from the Ubuntu repos are catalogued. They can be installed from here with a single click.



Does your terminal have bells and whistles like this?

developer he was a proud Windows fan. “I was on the forums in 2011, having just got Bodhi installed on a Lenovo S12 notebook. I was asking a lot of questions. I think the forum populous were a little nervous at first because I kept praising Windows for its comparative simplicity. Of course, after some time I realised the truth, lol!”

It didn't take long before Štefan started contributing to Bodhi. “I was just a common user and probably around 2014-2015 I asked Jeff Hoogland to give me some little tasks to contribute to Bodhi. It's very important to say I had no dev experience. I wanted to know how to make .deb packages, so I started to participate.

“I made some but after a while I became more curious and I tried looking at the *E17* modules' code. Many of the modules were broken or built in a way I didn't like. So I made some first attempts to edit the code, compile and understand how things work together, then I became a module maintainer.”

Štefan continues: “After gaining some C skills, I was able to understand Moksha code as well as the edj layer [which is used in themes]. My real progress started after contacting ylee. He guided me a lot and we got started on some small projects (for example, the



Community fella and theming padawan Gareth “hippytaff” Williams works his magic in sunny Bristol, just downriver from Future Towers.

clipboard module). Later we would work on many more projects.” Indeed, and all while Štefan earns a living as a computer technician.

Enamoured by Enlightenment

Robert Wiley (ylee) is Bodhi's lead developer (Jeff Hoogland's successor) and a respected stone mason. He cut his teeth using Knoppix in 2004, had a brief dalliance with Ubuntu, but now all of his systems run Bodhi. Asked how he discovered it, Robert reveals it all began with resurrecting an Acer netbook that was suffering from the advanced effects of Windows XP.

“I ended up installing Debian on it and compiling EFL and *E17* from Git. I went with *E17* because at that point I was using *DWM* (the very minimal window manager) on Ubuntu. However, I had kinda grown tired of *DWM* and wanted something easier to configure and customise for the netbook, but still lightweight enough to run on the limited hardware and support my workflow.

“Soon after Bodhi was first released I heard about it and ended up installing it on the netbook instead. It saved me from having to recompile EFL and *E17* all the time, a rather time-consuming task on that machine. So I've been using Bodhi since 2011. I became involved in the community almost immediately, first answering



Štefan “the_waiter” Uram learned C on the job and tirelessly uses his skills to make Bodhi better.

» SOLID FOUNDATIONS

Today, most *Enlightenment* development is aimed at embedded applications and takes place under the Enlightenment Foundation Libraries umbrella. This covers all the toolkit widgets, threading and libraries (for everything from scene graphs to serialisation).

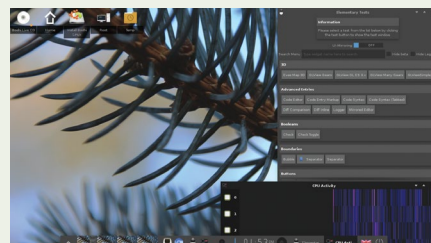
Mostly EFL is everything except the window manager. Work continues on the window manager, which is the bit that's actually called *Enlightenment* (or *E*). The combination of *E* and EFL thus make a more or less complete desktop environment.

Robert explains that EFL “was first developed as a collection of libraries that made the development of

Enlightenment easier. For example, this includes Eina, which includes among other things C implementations of many common data structures a C developer might need. Evas is the rather unique EFL canvas library – unique in the sense that Evas is fully state-aware. Efreet is the EFL library that handles **Freedesktop.org** standards.”

If you want to see what the latest release, *E25*, looks like, then good news – there's a special edition of the BodhiDev (a preview of the next release) ISO available, which uses *E25* instead of *Moksha*. Give it a whirl and you'll see that everything is smooth and slick, but just not quite as harmonious as *Moksha*. If

you look closely you'll see hints of other places where EFL is used. It's found its way into in-vehicle infotainment systems, smart TVs, fridges, watches and anything else that's powered by Samsung's operating system Tizen.



Enlightenment these days is not quite as light as *Moksha*, but its slickness cannot be denied.

questions on the forums or IRC channel and then later doing almost anything Jeff Hoogland asked of me.”

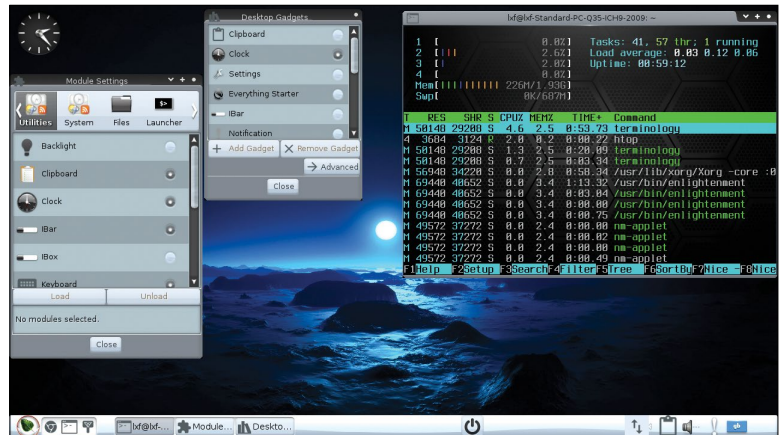
If you look on the website you’ll see that Bodhi, unlike so many Ubuntu-based distros, still maintains a 32-bit edition. Since there was no 32-bit Ubuntu 20.04, this is based on the previous LTS (supported until at least 2023). However, there’s a beta of Bodhi 6.0, based on Debian, that you can try now. Robert assures us that it’s “perfectly usable” and that there will be an official 32-bit, Debian-powered release of Bodhi 7.0.

A forward-looking distro

We asked Robert if Bodhi was always meant to work on older hardware. He replied that’s never been a design goal: “I don’t see Bodhi as being particularly geared towards older hardware and I want to emphasise it does fantastically on new hardware. It’s a great OS for gaming and other high CPU usage cases. Since the window manager has relatively low resource needs the OS can provide more to the applications running.

“But Bodhi has gained a reputation for doing great on older hardware. And better than most, if not all, the other distros commonly recommended for old machines. As a result, Bodhi has become a base for another distro, Escuelas Linux, used in school systems primarily across South America. Most of the machines Escuelas Linux ends up installed on are very old and most are 32-bit. Because of this I feel somewhat obligated to ensure Bodhi works on these machines. So we work with Escuelas Linux to ensure that it does and we address whatever issues they find.

“The reason and full credit for the reason why Bodhi does well on limited resources and older hardware goes to the *Enlightenment* developers themselves. We inherited this from them. The only thing we do that perhaps helps is to cut back some on the background processes and unnecessary stuff included in a vanilla Ubuntu install. I don’t take this to heroic efforts because



if I did users would complain. Our users have certain expectations and one is that stuff just works when installed from the repos. They also have expectations out of *Moksha*, our *E17* fork. But with effort and skill one can take Bodhi’s memory usage much lower than it is on a default install, or further trim its package list.”

One thing that’s been removed from the Ubuntu base is Snap support. “It’s disabled in Bodhi 6.0 and I plan on keeping it disabled,” says Robert. “I don’t think a .deb file should install a Snap package as Ubuntu has been doing lately [with *Firefox*]. Users are, of course, free

If for some reason you don’t like the default ArcGreen theme, plenty of others are but a click away.

ROBERT WILEY TALKS UP BODHI

“I don’t see Bodhi as being particularly geared towards older hardware and I want to emphasise it does fantastically on new hardware.”



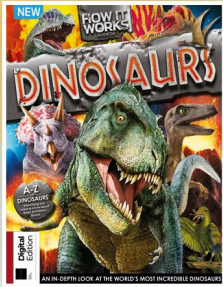
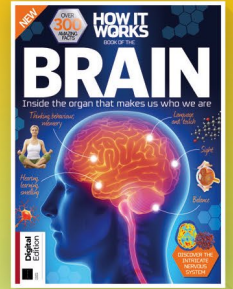
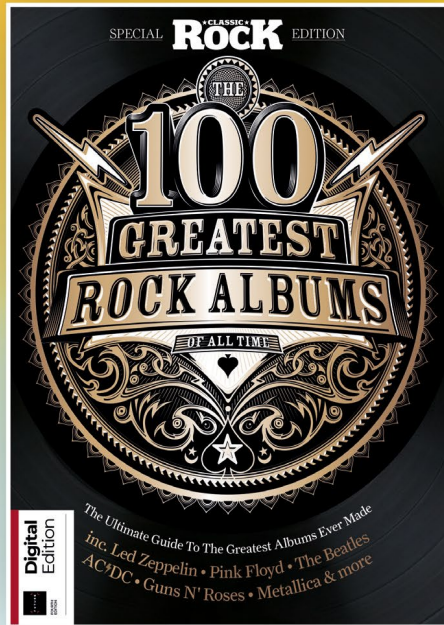
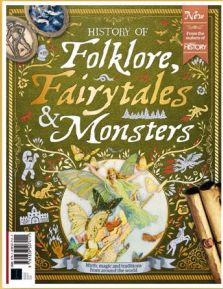
Lead developer and master craftsman Robert “ylee” Wiley runs a tight, but ever-so-welcoming ship.

to re-enable it and install Snap packages. Likewise, I have no plans on Flatpack support out of the box.” Robert also disagrees with Ubuntu’s switch to Wayland: “Wayland isn’t ready for prime time in my view. And besides, compositing is problematic on many older machines. We do release a pure *Enlightenment* ISO [see box, page 91] and I plan to continue doing so. *Enlightenment* provides almost full support for Wayland.”

So we hope you’re inspired, if you haven’t already, to check out Bodhi. It’s always good to see a community develop from people’s willingness to help with a great project. If you want to get involved, Gareth and the team will make you feel most welcome: “We always need help with translations, and C programmers who know EFL are like gold dust. Original art for backgrounds is another area we could do with some help with. And financial contributions for running costs are always welcome. It would be great if we could somehow get enough regular income to let ylee retire or at least work less to focus on Bodhi, but my merch attempts and other ‘benefit’ schemes never really take off and sometimes feel bad, insincere or even anti FOSS.”

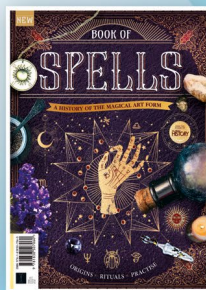
Nah, don’t worry Gareth – FOSS people love T-shirt, stickers and mug purchases. **LXF**





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

FUTURE

www.magazinesdirect.com

Magazines, back issues & bookazines.



Hot Picks



Mayank Sharma

After slaving... er... scribbling for *Linux Format* for over a decade, Mayank Sharma likes to think of himself as the magazine's Man Friday.

Kooha » ApplImage Pool » Blanket » Universal Media Server » Beaker Browser » superProductivity » Safe Eyes » Mudlet » StackAndConquer » Bottles » Filmulator

SCREEN RECORDER

Kooha

Version: 2.0.1 Web: <https://github.com/seadve/kooha>

Gnome's built-in screen-recorder is one of the few that can record even when running on top of the Wayland compositor. However, not only is it hidden from the menus, even when activated, it offers no interface to give you any options for customising the recording.

Kooha is a relatively new screen recorder for Gnome, which doesn't only work on top of Wayland, but also offers the usual screen recording conveniences, and despite its minimal interface is chock-full of features.

Kooha provides you with two screen recording modes. You can either record action on the entire screen, or restrict the recording to a secreted area. Additionally, you can toggle buttons on its main interface to record audio from either the microphone, the system, or both, together with video. Similarly, you can also choose to disable recording the mouse pointer.

The tool can be controlled from the keyboard and also gives you control over the saved location of the recording. In terms of video formats, you can choose between WebM, MKV, mp4 and GIF.

The project's release page offers multiple options to install the tool, although the easiest way to install *Kooha* on your Linux system is via Flathub.

Type in the following command to first add the Flathub repository:

```
$ sudo flatpak remote-add --if-not-exists flathub https://flathub.org/repo/flathub.flatpakrepo
```

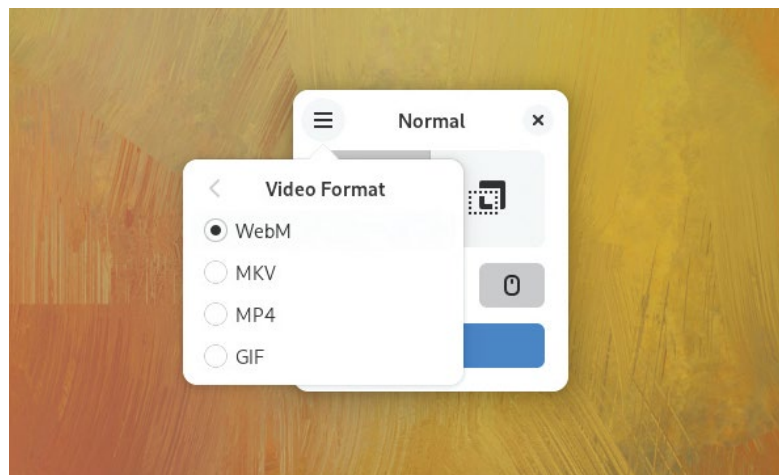
Once added, you can pull the *Kooha* Flatpak with:

```
$ sudo flatpak install flathub io.github.seadve.Kooha
```

And that's all there's to it. You can now run *Kooha* either from the Applications menu or from the command line with:

```
$ flatpak run io.github.seadve.Kooha
```

If you're using a modern computer then you can enable the hardware-accelerated encoding option to enable the encoder to utilise the GPU for faster encoding. For this you'll have to jump on the command line interface to enable all the supported drivers and

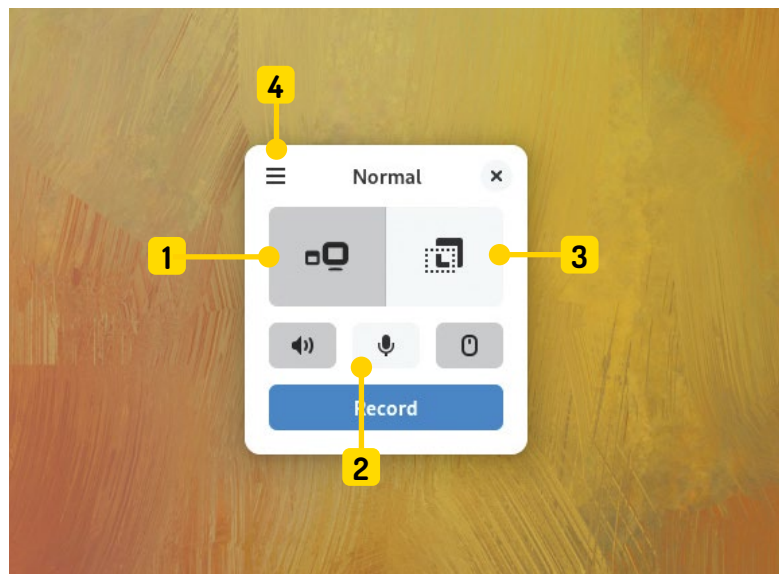


Kooha's minimal interface offers more options than Gnome built-in screencast, but lacks some of the controls that are provided with other established screen recorders.

then toggle video acceleration API (VAAPI) with the following command:

```
GST_VAAPI_ALL_DRIVERS=1 KOOHA_VAAPI=1 flatpak run io.github.seadve.Kooha
```

A GUIDE TO KOOHA'S INTERFACE



1 Fullscreen
This option will offer you the option to either capture the entire screen, or a particular open window.

2 Audio source
The first icon records audio from the speakers, and the second from the microphone.

3 Screen selection
Selecting this option enables you to select the portion of the screen you wish to record.

4 Other options
Head to the menu to add a delay before the start of the recording, and to select a video format for the recording.

APP STORE

AppImage Pool

Version: 5.0.0 Web: <https://github.com/prateekmedia/appimagepool>

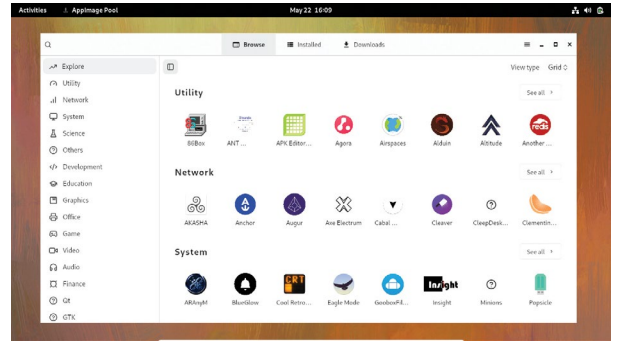
Universal file formats such as Flatpaks, AppImages, and Snaps have significantly reduced the amount of effort that's required to install new applications in Linux. By decoupling it from the underlying package management scheme, such formats have helped to make the process consistent across distros.

However, while these new packaging formats have their respective online application stores (Flathub for Flatpaks, AppImageHub for AppImages and Snapcraft for Snaps), what if you want to browse the available programs and then install them from outside the browser environment?

AppImage Pool fills exactly that void. As its name suggests, it's a desktop application store that enables you to search, download, install, and manage AppImages in any Linux distro.

The application store is available both as an AppImage, as well as a Flatpak. To install the AppImage, type the following:

```
$ sudo chmod a+x appimagepool-x86_64.AppImage.AppImage
```



Similarly, install the Flatpak with:

```
$ sudo flatpak install flathub io.github.prateekmedia.appimagepool
```

```
$ flatpak run io.github.prateekmedia.appimagepool
```

AppImage Pool is written in Flutter and has a clean, intuitive interface, reminiscent of any modern Linux application store. You get a categorised list of programs, and instead of fetching the latest version of the application the Download button brings up a list of all available versions, giving you the ability to install a particular version, which is a pretty nifty feature.

Once you've selected a program, you can track its progress. After downloading, it'll be listed under the Installed tab, from where you can launch it with a single click. However, the application doesn't integrate the installed AppImage with the distro's application menu, so you'll have to fire it from under the Installed tab in *AppImage Pool*. You can also remove all installed AppImages from under this tab by pressing the Trash icon next to the program you want to remove.

Surprisingly, while some of the programs in the AppImageHub offer multiple versions, some of the listed tools don't list any.

AMBIENT SOUND TOOL

Blanket

Version: 0.6.0 Web: <https://github.com/rafaelmardojai/blanket>

While some of us prefer to work (or relax) in a silent environment, for many particular sounds can help improve focus, and can also help calm their nerves (see [page 65](#) for more).

Blanket is a simple and straightforward tool that helps generate different kinds of ambient sounds. You can use it to improve focus, but blocking other distracting noises, or to help relax as you take a power nap, or even doze off for the night.

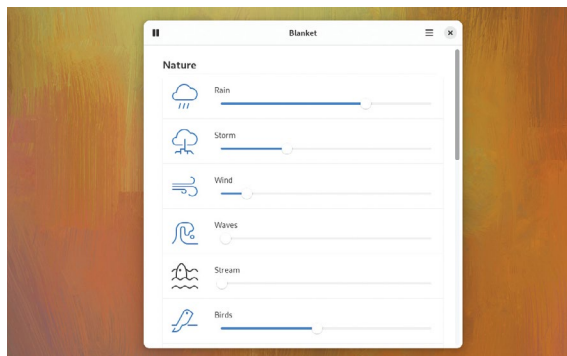
The program produces third-party packages for most distros. For instance you can install it on Ubuntu through a PPA with:

```
$ sudo add-apt-repository ppa:apandada1/blanket
$ sudo apt-get update
$ sudo apt install blanket
```

Similarly, Fedora users can fetch it via the COPR repository with the following:

```
$ sudo dnf copr enable tuxino/Blob
$ sudo dnf install blanket
```

The most convenient option, of course, is using the Flatpak, with:



```
$ flatpak install flathub com.rafaelmardojai.Blanket
```

```
$ flatpak run com.rafaelmardojai.Blanket
```

Blanket has a minimalistic interface, which has a categorised list of various sounds such as rainfall, chirping birds, waves crashing and such. You can use the slider corresponding to each sound to set the precise volume for each sound to create your ideal atmosphere for working/relaxing/sleeping.

To create the perfect atmosphere every time you boot into the distro, you can ask *Blanket* to launch automatically on startup, and even play your perfect sounds in the background even when the tool is closed.

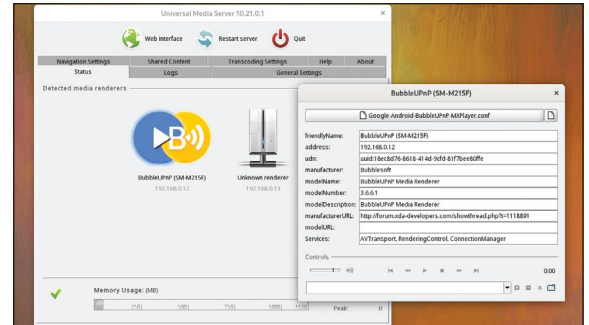
The only real downside to *Blanket* is that it ships with a limited set of preset sounds. However, it does offer the dexterity to add custom sounds, and you can find plenty of them online. The application can recognise any ogg, mp3, wav, and FLAC audio files.

The default selection of sounds in the program are quite impressive, but we were a little disappointed at the limited selection.

STREAMING MEDIA SERVER

Universal Media Server

Version: 10.21.0.1 Web: www.universalmediaserver.com



Universal Media Server (UMS) is a cross-platform DLNA-compliant UPnP media server, which is written in Java. It can convert any media file format and stream it to a wide range of hardware devices, including smart TVs, smartphones, media centres and even gaming consoles.

UMS streams media via the Universal Plug and Play (UPnP) protocol to any DLNA-compliant device. You'll have to manually roll-out UMS, though the procedure is fairly simple. It ships with Java, so you only need to fetch a handful of dependencies, all of which are available in the official repos of most desktop distros. Fedora is an exception, in that you'll have to enable the RPM Fusion repositories (https://docs.fedoraproject.org/en-US/quick-docs/setup_rpmfusion) before you can install the VLC player in the distro.

Once you have the dependencies, download the tarball of the latest version, and extract with `tar xzvf UMS-10.21.0.1-x86_64.tgz`. Fire up the server with:

```
$ cd ums-*;  
$ ./UMS.sh.
```

The first time you launch it, UMS takes you through a configuration wizard. When it's complete, the wizard asks you to specify the shared folders that contain the media, under the Navigation/Share Settings tab.

Although you can start streaming without further configuration, UMS does include an administration panel that offers several customisable options and helpful tool tips to guide new users. It also includes a minimal web interface for streaming content that lists shared directories as well as recently played files.

After configuration, and on subsequent startups, you'll land on the Status tab that lists all the detected compatible sources that the server can stream to. If your compatible devices are powered on and connected to the same local network as the server's, these should be listed here. You can now view the server from any of these devices and browse and stream content.

You can convert your Android handset or tablet into a DLNA-compatible device by installing the BubbleUPnP tool from the Google Play Store.

P2P WEB BROWSER

Beaker Browser

Version: 1.1.0 Web: <https://beakerbrowser.com>

Beaker is an experimental peer-to-peer browser. It's built using the Electron framework and uses the open source Chromium browser as its rendering engine.

Just like other web browsers, you can use it to browse `http://` and `https://` websites. What sets the program apart, however, is its peer-to-peer technology, using which Beaker makes it possible for users to publish their own websites directly from the browser itself, eliminating the need for setting up a web server, or using a web host for hosting the website on a remote third-party server.

Linux users can install Beaker browser through either its AppImage or Snap packages. To use snap on Fedora, first ensure that it's activated with `sudo dnf install snapd`, before installing the browser with `sudo snap install beaker-browser`.

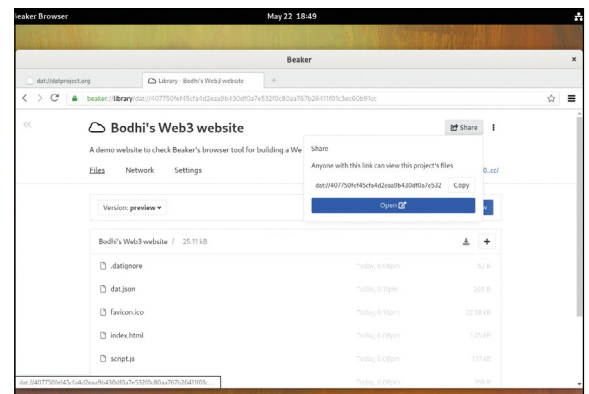
The Beaker browser is part of the emerging paradigm known as decentralised web (dweb) or the slightly more memorable Web 3.0 (see page 48 to find out more). The whole idea behind dweb is to wrest back control from centralised corporations, without adversely affecting

the user experience in the process.

The secret sauce behind Beaker's web-hosting ability is the support for the peer-to-peer Dat protocol, which does the grunt work. Known as Hyperdrives in Beaker's parlance, these websites are accessible over the `dat://` protocol, such as `dat://beakerbrowser.com`.

Websites served over the `dat://` protocol work just like any other webpage. They're a collection of HTML, CSS, and JavaScript files that come together to form a webpage. Just like with `http://` websites, you can click links, download images and use developer tools to interact with the page.

While we wouldn't advise you to switch completely to Beaker for your everyday web browsing, we do encourage you to use it to take a peek into this new emerging paradigm, and perhaps even use the opportunity to plant your flag by creating your very own Web 3 website.



Remember to ask other peers to share a copy of your website, since there must be an instance always running for the website to be accessible.

TIME TRACKER

superProductivity

Version: 7.11.5 Web: <https://super-productivity.com>

Working as a freelancer, keeping track of the time spent on a task or project is especially essential, though it can be of use to everyone. *superProductivity* is a to-do utility that includes the ability to track and manage time spent on each task or project. While freelancers can use this for billing (*now steady on there—Ed*) purposes, others can use it for efficiency purposes.

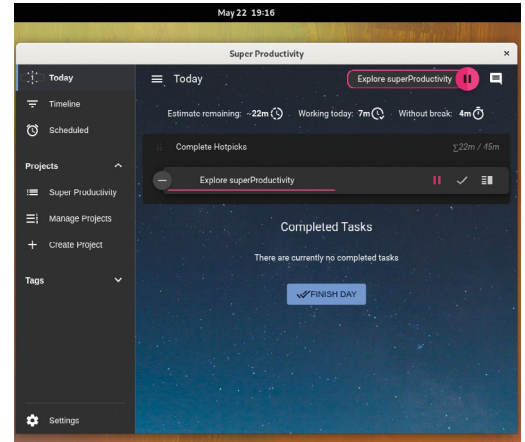
The program is available as pre-compiled RPM and DEB binaries, as well as ApptImages and Snaps for added convenience. There's also a web version of *superProductivity*, although it has to remain open for time tracking to work properly in the web version. Furthermore, you must also install its *Chrome* browser extension, which helps track idle time accurately.

The application has an intuitive interface for anyone used to the ways of time-tracking programs. However, if you've only dealt with simple to-do task managers then you could save yourself some time by going through some of the articles and videos the developer points to

(<https://github.com/johannesjo/super-productivity>).

You can create timesheets and work summaries to better track and document your work. These can be nested to accurately break down and provide detailed tracking details about complex projects, made up of several smaller tasks. The program has time-boxing abilities that you can use to assign time estimates to tasks, which isn't required, but can help increase focus and avoid procrastination.

Besides its impressive time tracking abilities, *superProductivity* has all the usual features you'd expect from a to-do task management tool. It can also directly import tasks from Jira, GitHub, GitLab and Open Project. The Jira integration will also keep you abreast of any ticket changes, without having to check with the project management tool.



superProductivity makes it very easy to help create and share work and time summaries.

RSI PREVENTION TOOL

Safe Eyes

Version: 2.1.3 Web: <https://slgobinath.github.io/SafeEyes>

You know what makes meeting deadlines and time targets worthwhile? To do so without injury (*Linux Format... life on the edge – Ed*). *Safe Eyes* is a simple tool that's designed to ensure that long hours on the computer don't result in repetitive strain injuries (RSI).

The aim of *Safe Eyes* is to ensure you don't strain your eyes by forcing you to take regular breaks when working for long hours on a computer. The project can be installed on all major Linux distros. If you use Fedora:

```
$ sudo dnf install libappindicator-gtk3 python3-psutil cairo-devel python3-devel gobject-introspection-devel cairo-gobject-devel
```

```
$ sudo pip3 install safeeyes
```

```
$ sudo gtk-update-icon-cache /usr/share/icons/hicolor
```

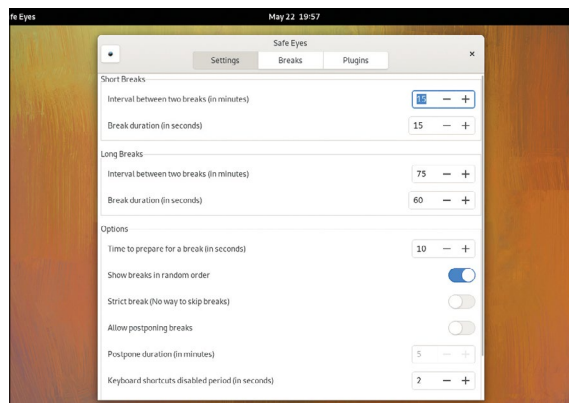
Ubuntu users can install *Safe Eyes* through its PPA:

```
$ sudo apt install gnupg2 software-properties-common apt-transport-https lsb-release ca-certificates
```

```
$ sudo add-apt-repository ppa:slgobinath/safeeyes
```

```
$ sudo apt update
```

```
$ sudo apt install safeeyes
```



Safe Eyes has a detailed but straightforward configuration interface, which asks you to define various parameters to control its short and long breaks. One of the best things about *Safe Eyes* is that its break screen asks you to do some exercises to further prevent the chances of RSI.

The program displays notifications about upcoming short breaks, which can be skipped if you're in the middle of something important (*like Jonni getting his cover feature in on time – Ed*). That said, you can enable Strict breaks that can't be skipped. In addition, Long breaks ask you to either lean back on your chair and relax, or even get up and take a walk. In these cases, it'll lock the display to prevent unauthorised access.

Better still, *Safe Eyes* is intelligent enough not to bug you when it sees that you're working with a full-screen program. It'll also track any screen idle time, which signals that you aren't working, and will adjust the break periods accordingly.

Safe Eyes ships with reasonable defaults, but you can tweak any of its behaviour to better suit your working style.

TEXT GAME

Mudlet

Version: 4.16.0

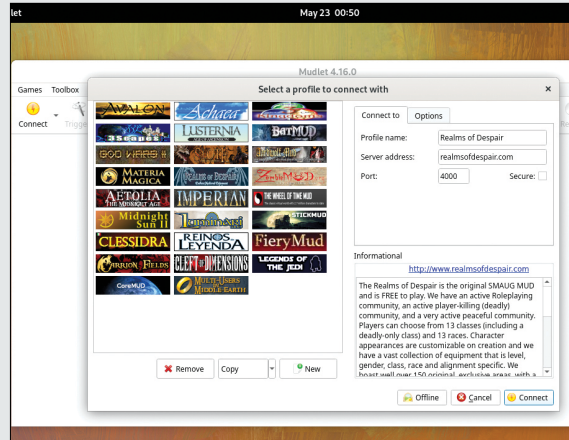
Web: www.mudlet.org

You can't really call yourself a gaming connoisseur if you haven't played a multiplayer text-based game. Fondly referred to as a MUD, standing for multi-user dungeons, these games had elements of role-playing and interactive fiction, all delivered through a text-based interface.

Mudlet is a gaming platform focused mainly on enhancing the whole MUD gameplay experience. It supports a wide variety of protocols and sports an intuitive interface. The cross-platform program is available as an AppImage for Linux.

Unpack it with `tar xvf Mudlet-4.16.0-linux-x64.AppImage.tar`, give it executable permissions with `chmod +x Mudlet*.AppImage`, and launch it with `./Mudlet*.AppImage`. From the launch screen, pick from one of the listed games and use the Connect button to connect to its server. Click New to enter the connection details of your favourite MUD game.

Mudlet enables players to read through descriptions of rooms, objects, other players, non-player characters, as well as about the actions performed in the virtual



Mudlet ships with over 20 pre-configured MUDs, and can connect with other MUDs as well, as long as you have their connection details.

world. Players typically interact with each other and the world by typing commands.

Mudlet gives a modern spin to the classic MUDs by enabling users to use its specially designed scripting framework, based on Lua, to automate many tasks through a combination of aliases, triggers, keybindings, and other *Mudlet* components.

Mudlet supports simultaneous gameplay, and the project has an extensive wiki for gamers, which talks about everything from basic gameplay to *Mudlet*'s automation features, which can significantly improve the gaming experience. *Mudlet*, and its active community of developers and users, are proof that text-based gaming isn't ready to scroll out of existence just yet.

BOARD GAME

StackAndConquer

Version: 0.10.0 Web: <https://github.com/EITh0r0/stackandconquer>

S*ackAndConquer* is a challenging tower conquest game that's inspired by the board game *Mixtou* (<https://spielstein.com/games/mixtour>). Your objective is to build a stack of stones with at least five stones and a stone with the players colour on top.

Ubuntu users can grab the game from its PPA with:

```
$ sudo add-apt-repository ppa:elthoro/stackandconquer
$ sudo apt update
$ sudo apt install stackandconquer
```

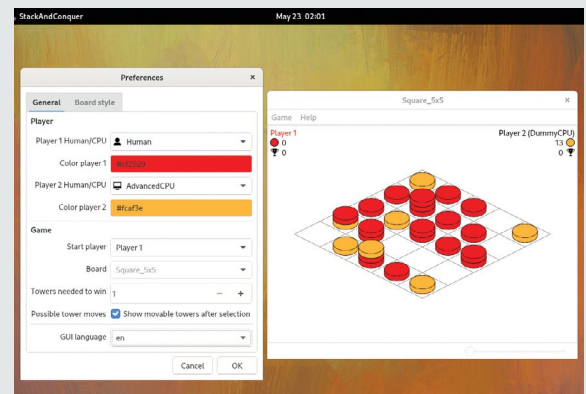
Similarly, the project uses the OpenSUSE Build Service to produce builds for distros such as Debian, Fedora and OpenSUSE. You can follow the simple instruction of its website to add the repository for your favourite distro, before pulling in the game. For a quicker and easier installation, you can grab the game's AppImage file instead.

The game is a standard two-player affair, one of which can be the computer. The game supports two levels of computer opponents. The DummyCPU player

is the novice AI player, while the AdvancedCPU is a very tough opponent.

By default the game allows Player 1, which is you, to make the first move. However, you can spice things up by selecting the option to randomise the start player. Also, by default you only need one tower to win the game, though this again can be increased from within the settings menu.

StackAndConquer is a pretty simple logic-based game, and it doesn't take long to comprehend the logic behind it. You click the board to place a piece with the objective to build a stack at least five pieces high, with your piece on the top. Players can also choose to move one or more pieces from a stack, either orthogonally or diagonally. Pieces are taken from the top of a stack, but can be the player's own or the opponents. You can follow the built-in tutorial to better comprehend the rules of the game.



StackAndConquer has a multilingual interface, which besides English can also be made to render in German, Italian, and Dutch.

WINDOWS PROGRAM EMULATOR

Bottles

Version: 2022.5.14-trento-3

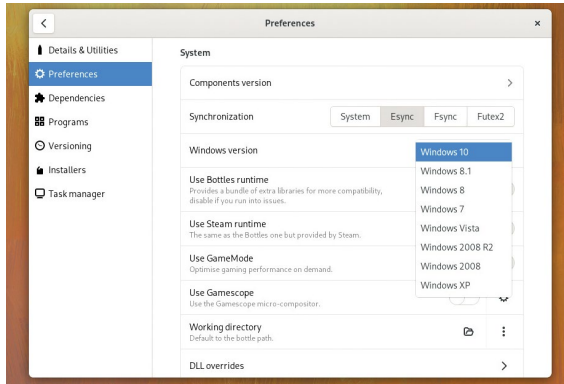
Web: <https://usebottles.com>

We might not be big fans of proprietary software here at *LXF Towers*, but we're also pragmatic, and know that there are times there's no escaping them. However, while there's no dearth of popular proprietary software that have native Linux versions, there's an equally sizable number of programs that are only available as Windows executables.

You can either fire up a virtual machine (VM) to run them, or let *Wine* take them on via user-friendly interfaces such as *PlayOnLinux*.

Bottles is another option that exposes the dexterity of *Wine* through an approachable user-interface. *Bottles* helps create a Windows-like environment to run Windows software. *Bottles* gets its name from *Wine* prefixes, also known as a *Wine* bottle, which is a folder where *Wine* keeps everything it needs to help run a Windows program. You can create multiple *Wine* prefixes manually, or use *Bottles* to do the heavy lifting for you.

The project recommends installing *Bottles* via its official Flatpak with the following:



```
$ sudo flatpak install flathub com.usebottles.
```

```
bottles
```

```
$ flatpak run com.usebottles.bottles
```

On first launch it'll download some extra software, which it couldn't distribute with the installer due to licensing constraints. The app comes pre-configured to run a range of popular Windows games and tools. Each of these environments is made up of ready-to-use settings, dependencies and the appropriate libraries. You can choose between Gaming and Software environments, but you can also create ones from scratch by choosing the Custom option.

You can think of these bottles as self-contained containers, which are also isolated from other bottles, and you can have multiple programs in one bottle. To fully appreciate *Bottles'* features you have to spend some time exploring its configuration options.

Bottles isn't unique in what it does, but it makes the process very approachable even for users who aren't used to the ways of *Wine*.

RAW IMAGE EDITOR

Filmulator

Version: 0.11.1

Web: <https://filmulator.org>

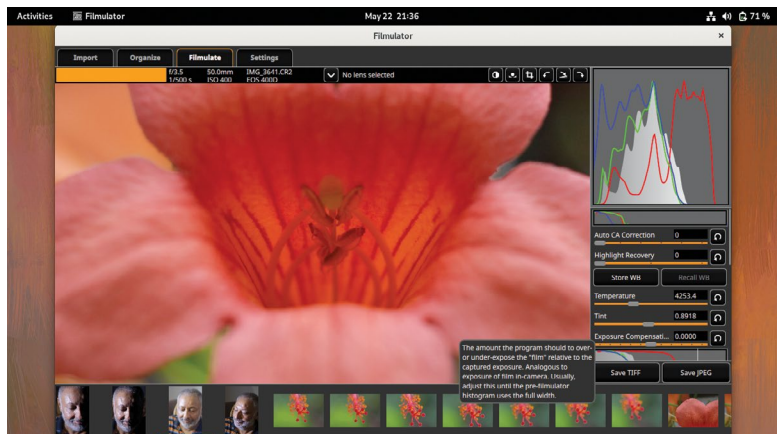
Once the exclusive domain of professional photographers, RAW images are more democratised now with most cameras, including several point-and-shoot ones, now offering RAW images to their users.

RAW files contain uncompressed and unprocessed image data, which enables photographers much more post-processing flexibility. While most desktop distros come equipped with an image viewer, it's unlikely they'll be able to handle RAW images, let alone edit them.

That said there's no shortage of open source RAW image editors, and you'll be able to find one in your distro's official repository as well.

Filmulator is part of the growing tribe of RAW image editors, and elegantly manages to combine the conveniences and usability of a normal image editor with the workflow of a professional RAW image editor.

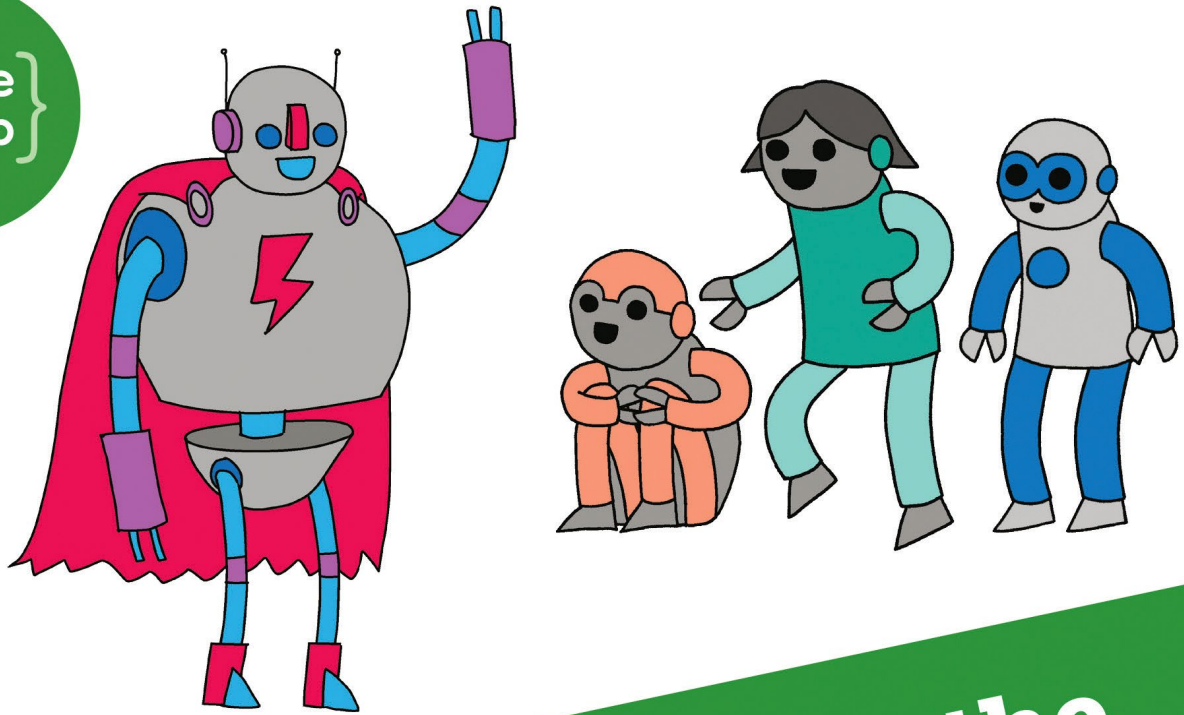
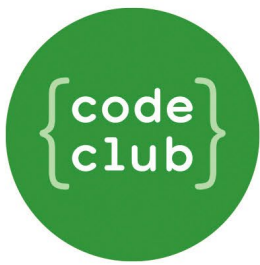
It's available as an AppImage so you can download it from the website, give it executable permissions with `chmod +x Filmulator_v0.11.1.AppImage`, and then launch it with `./Filmulator_v0.11.1.AppImage`.



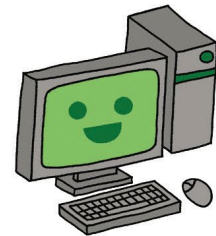
Filmulator has library management options as well, and you'll be asked to point it to your repository of RAW images on first launch. The various editing parameters in the program all have descriptive tool tips that help explain their purpose, and are worded to make sense to a novice image editor. The application also includes a helpful guide to photo editing, again written for newbies, over on its wiki.

If you're a beginner photographer and don't have much experience working with RAW files then *Filmulator* is a great way to learn the ropes and develop an image-editing workflow, while familiarising yourself with the process employed by professionals. It doesn't include all the features that you get with some of its peers, but it has the most useful ones, and is easy to approach. **LXF**

Filmulator is so named because it tries to replicate the old school process of developing a film to editing digital images.



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

Controlling processes and using Unix signals

Join **Mihalis Tsoukalos** as he demonstrates how to use Rust to work and manage child processes, and handle Unix signals.



OUR EXPERT

Mihalis Tsoukalos is a systems engineer and a technical writer. You can find him at [@mactsouk](https://mactsouk.com).

The subject of this tutorial is signal handling and working with processes in Rust. We'll begin by showing how to spawn a new process in Rust – the relevant source file is named `spawnNew.rs` – without doing anything useful with it, just to show you the simplest form of process spawning. So, the logic of `spawnNew.rs` is found in the following code excerpt:

```
Command::new("ls").spawn().expect("Error in command");
```

All of the work is done by the call to `Command::new()` and the use of `spawn()`. As you might recall from the previous Rust tutorials, the `expect()` part is executed in case of error. As a reminder, you can replace `expect()` with `unwrap()`, but in that case you're not going to be able to print your own error message.

The screenshot (right) shows part of the code of `spawnNew.rs` that contains multiple variations of `Command::new()` that show how to pass arguments to `Command::new()` and how to set the current directory. You can learn more details about `std::process::Command` at <https://doc.rust-lang.org/std/process/struct.Command.html>.

Running `spawnNew.rs` generates the output shown in the screenshot (opposite page). What's important in the output is the fact that the error message from the last `Command::new()` call is printed before the `Command::new()` call with the `.arg("/doesNotExist")` parameter. This means that currently, we can't be sure about the order of execution. Put simply, our code doesn't wait for a command to finish before executing the next `Command::new()` call – this happens due to the use of `spawn()`.

This time we're going to use `output()` instead of `spawn()` when calling `Command::new()` because `output()` tells Rust to wait for the child process to finish before continuing. So, the technique is illustrated in the following code from `output.rs`:

```
let o1 = Command::new("ls")
    .output()
    .expect("Error in command");
```

Apart from the use of `output()` in the place of `spawn()`, we need a variable to keep the output from the `Command::new()` call. Running `output.rs` prints the

```
fn main() {
    Command::new("ls")
        .spawn()
        .expect("Error in command");

    // -X sorts the output alphabetically
    Command::new("ls")
        .arg("-X")
        .spawn()
        .expect("Error in command");

    // Use a different current directory
    Command::new("ls")
        .current_dir("/tmp")
        .arg("-X")
        .spawn();
}
```

Here's the Rust code of `spawnNew.rs`. There are many ways to invoke `Command::new()` based on the external command we want to execute.

expected output sequentially. However, spawning a process without any control over it isn't usually enough. We need to be able to check the status of that child process, which is the subject of the next section.

Exit status

We'll need to know how to check the status of a child process. In this case, we have two options. Use `output()` as we did in `output.rs` and obtain the return code from there, or use `.status()` instead of `output()` in case we're only interested in the exit status code of a command. `childStatus.rs` illustrates both techniques.

In the first case, we use the status field to obtain the status code whereas in the second case, we use the return value of `Command::new()`. Both methods are illustrated in the following code excerpt:

```
println!("Status: {}", o1.status);
println!("Status: {}", status);
```

What isn't shown here is that you can use `o1.status.success()` to check whether the command was successful or not, which saves you from having to look at the return code value on your own. Additionally, you can use a call to `o1.stdout` to obtain the output from a

QUICK TIP

Get the code for this tutorial from the Linux Format archive: www.linuxformat.com/archives?issue=291.

You can learn more about Rust at www.rust-lang.org.

command. The output of `childStatus.rs` is going to be similar to the following:

```

$ ./childStatus
printing: abort.rs
...
Status (o1.status): exit status: 0
ls: cannot access 'doesNotExist': No such file or directory
Status (status): exit status: 2
    
```

So, because `o1` was successful, its exit status code was 0. In the second case, the exit status code is 2 because `doesNotExist` could not be found – see the Rust code in `childStatus.rs` for more details.

Environmental variables

Let's now take a look at the use of `env()` and `env_clear()` for setting and clearing the environment of a child process. The use of `env()` is illustrated in the following code excerpt:

```

let o1 = Command::new("sh").args(&["-c", "echo $MAGAZINE"]);
.env("MAGAZINE", "Linux Format")
.output().expect("Failed to execute process");
    
```

The previous code declares a new environment variable named `MAGAZINE` with the value of `Linux Format`. The use of `env_clear()` is shown below:

```

let o2 = Command::new("sh").args(&["-c", "echo $LSCOLORS"]);
.env_clear().output().expect("echo command failed!");
    
```

In the previous code, the `LSCOLORS` environment variable is unset – if it was originally set – due to the use of `env_clear()`.

Using environment variables such as `$LSCOLORS`, `$PATH` and `$MAGAZINE` inside `Command::new()` is a special case because spawning a child process doesn't include the entire shell environment. This means that the characteristics of a UNIX shell and its environment variables aren't included automatically. Therefore, we should specifically run the desired UNIX shell and then execute the command that uses an environment variable, as occurred in the previous code excerpt. Finally, remember that we're free to include both `env_clear()` and `env()` in a single `Command::new()` statement. Running `envChild.rs` generates the following type of output:

```

$ ./envChild
printing o1: Linux Format
printing o2:
printing o3: /usr/bin:/bin
    
```

The first line prints the value of the `MAGAZINE` environment variable that's created inside Rust whereas the second line proves that `env_clear()` is going to unset all environment variables including `LSCOLORS`. The last line of output shows that the value of `PATH` is now what we've set in `envChild.rs` even if we've also used `env_clear()` in `Command::new()`. Finally, `env_remove()`, which isn't shown in `envChild.rs`, should be used for removing a specific environment variable from a child process.

Piping output

To continue these essential basics we show how to get the output from a `Command::new()` statement and use it as input in another command, which in UNIX

This shows the output of `spawnNew.rs`. Due to the use of `spawn()` in `Command::new()`, commands are executed as child processes without the program waiting for them to finish first.

terminology is called piping. The logic of `pipe.rs` is found in the following code excerpt:

```

let wc = match Command::new("wc")
.stdout(Stdio::piped())
.stdout(Stdio::piped())
.spawn() {
Err(error) => panic!("Error in wc: {}", error),
Ok(wc) => wc,
};
    
```

Here the `wc` command, which is defined as `Command::new("wc")`, expects its input from a pipe and is going to send its output to another pipe – both of these are implemented with the use of `Stdio::piped()`. In our case, the output that's generated dynamically will be displayed as soon as the relevant pipe is closed.

The screenshot (*overleaf*) shows part of the Rust code of `pipe.rs`. As the `Command::new("env")` finishes, `stdin` closes as well, which means that the pipe that we've opened also closes. After the pipe closes `wc`, as executed using `Command::new("wc")`, starts processing its input and produces its output, which is printed on screen using `wc.stdout`.

Running `pipe.rs` produces the following output:

```

$ ./pipe
Data sent to wc!
wc returned:
29 34 2456
    
```

Although it isn't desirable, errors happen. Let's see how to forcefully terminate processes in two ways.

There are times that we need to terminate a process. This occurs mainly either because there's an

QUICK TIP

Having an `expect()` or `unwrap()` part on the Rust statements that you execute is considered good practice, so do not forget to add one of them in your statements. Always be prepared for the unexpected!

» ABOUT UNIX SIGNALS

UNIX Signals are software interrupts that offer a method of handling asynchronous events on a UNIX system. Every application, apart from the trivial ones, must be able to deal with signals. Each signal can be identified either by its name or a numeric value; the former approach is easier to remember and therefore is the recommended way. To send a signal to a running application, you should have the right privileges or just be root. The most critical use of signals is for avoiding blocking situations. Blocking can happen when waiting for user input, reading a file or reading from a device.

The `kill -l` command shows the list of available signals. All signal names begin with `SIG`, which isn't always shown in the `kill -l` output (it depends on your Linux variant) and have a default action. Most of them enable you to bypass their default action by writing your own handling code. As mentioned elsewhere in this tutorial, not all signals can be handled, blocked or ignored.



QUICK TIP

If you put an underscore at the beginning of a variable name, the Rust compiler isn't going to complain if that variable isn't being used. Put simply, it tells the Rust compiler that this is intentional.

unrecoverable error condition or because the process is misbehaving. We can end a process using `exit()` and `abort()` – the use of these two calls is illustrated in `exit.rs` and `abort.rs`. The code of both `exit.rs` and `abort.rs` can be found in the screenshot (*top of opposite page*).

The output of `exit.rs` is similar to the following:

```
$ ./exit
Ping!
$ $?
100
```

The good thing with `exit()` is that it returns the desired exit code to the UNIX shell.

The output of `abort.rs` is similar to the following:

```
$ ./abort
Ping!
[1] 30615 abort (core dumped) ./abort
```

The main issue with `exit()` and `abort()` is that neither of them cleans up or calls any destructors.

Don't panic(), actually do

Terminating a process is so important that we're going to see an additional technique for performing process termination. This technique uses `panic!()` and requires the use of a match block. The advantage of `panic!()` is that it displays a backtrace of the stack of the current thread and calls all destructors. The use of `panic!()` is illustrated in the following code excerpt:

```
let _o = match Command::new("DoesNotExist")
    .stdin(Stdio::piped())
    .stdout(Stdio::piped())
    .spawn()
{
    Err(err) => panic!("Error in spawning child process:
    {", err),
    Ok(_process) => {
        println!("Success!");
    }
};
```

The match block determines the success or the failure of the `Command::new()` statement in order to act accordingly. Once again, bear in mind that `panic!()` does some housekeeping before exiting the program, which isn't the case with `exit()` and `abort()`. The output of `panic.rs` is similar to the following:

```
$ ./panic
thread 'main' panicked at 'Error in spawning child process: No such file or directory (os error 2)', panic.rs:10:21
```

```
let wc = match Command::new("wc")
    .stdin(Stdio::piped())
    .stdout(Stdio::piped())
    .spawn() {
    Err(error) => panic!("Error in wc: {}", error),
    Ok(wc) => wc,
};

// Get the input
let data = Command::new("env")
    .output()
    .expect("Error in env");

// Send the input
match wc.stdin.unwrap().write_all(&data.stdout) {
    Err(error) => panic!("Failed writing to wc: {}", error),
    Ok(_) => println!("Data sent to wc!");
}
```

This shows the Rust code of `pipe.rs` that demonstrates how to pass the output of a `Command::new()` statement as input to another `Command::new()` statement.

note: run with `RUST_BACKTRACE=1` environment variable to display a backtrace

Signal handling

The subject of the rest of this tutorial is signal handling in Rust. Have you ever pressed Ctrl+C to stop a program from running? Ctrl+C sends the SIGINT signal to the program – SIGINT is the most widely used signal. Another important signal is SIGKILL that occurs when you kill a process using the `kill -9` command. Another useful signal is called SIGHUP that's often used to notify server processes to reread their configuration files.

SIGKILL is a special kind of signal, in the sense that it can't be caught, blocked or ignored. The same applies to SIGSTOP. SIGKILL is usually called in extreme conditions where you need to act fast, so it's the only signal that's usually called by number because it's quicker to do so. The effect of the KILL signal is similar to unplugging your computer from the mains instead of powering it down normally. It can cause various problems, especially when used to stop server processes such as database, HTTP and email servers.

Unlike most programming languages, including Python, C and Go, Rust doesn't come with built-in signal handling capabilities. This means that we have to use external crates for signal handling. We're going to start by illustrating how to handle a single signal (SIGINT) with the help of the `ctrlc` crate.

To have a play with this we're going to create a utility that handles a single signal, which in this case is going to be SIGINT. For that purpose, we need to use Cargo and create a new Cargo project. Feel free to review the use of cargo in the previous Rust tutorials.

The name of the Cargo project is going to be `single_signal`. Please add the next line in the dependencies section of `Cargo.toml` to add support for the `ctrlc` crate:

```
ctrlc = { version = "3.0" }
```

The `ctrlc` crate handles the SIGINT signal only. The screenshot (*opposite page, bottom*) shows the source code of `./src/main.rs`, which is simple and straightforward, as well as a small interaction with the program. In its current version, the program sleeps for 20 seconds before terminating.

More details about the `ctrlc` crate are at <https://crates.io/crates/ctrlc> and at <https://github.com/Detegr/rust-ctrlc>. Among other things, the documentation states if you want to handle all SIGINT, SIGTERM and SIGHUP signals, you should change the dependency line in `Cargo.toml` to the following:

```
ctrlc = { version = "3.0", features = ["termination"] }
```

So, although `ctrlc` is simple to use, it supports a limited number of predefined signals (SIGINT and SIGTERM).

Multiple signals

Let's learn to handle multiple signals in a Rust program. Once again, we're going to create a Cargo project, which in this case is going to be called `multiple_signals`. In this case, we're going to use the `signal-hook` crate, and in order to enable the more advanced features of `signal-hook`, we need to add the following line to the dependencies section of `Cargo.toml`:

```
signal-hook = { version = "0.3.10", features = ["extended-siginfo"] }
```

Have in mind that when a newer version of `signal-`

```

→ code cat abort.rs
use std::process;

fn main() {
    println!("Ping!");
    process::abort();

    println!("Pong!");
}

→ code cat exit.rs
use std::process;

fn main() {
    println!("Ping!");
    process::exit(100);

    // Not going to get executed
    println!("Pong!");
}
    
```

This screenshot displays the source code of `exit.rs` and `abort.rs` that illustrate the use of `exit()` and `abort()` for terminating a program, respectively.

hook, or any other crate you're using in a cargo project, becomes available, you're going to need to change the version number in the dependencies section to match the newer crate version. This isn't obligatory because you can continue using your current version.

You declare the signals that you want your application to handle by using a variable to hold them – in our case the name of the variable is `SIGNALS` and is defined as follows:

```

const SIGNALS: &[c_int] = &[
    SIGTERM, SIGQUIT, SIGINT, SIGTSTP,
    SIGWINCH, SIGHUP, SIGCHLD, SIGCONT, SIGUSR1
];
    
```

So, our utility is going to be able to handle the `SIGTERM`, `SIGQUIT`, `SIGINT`, `SIGTSTP`, `SIGWINCH`, `SIGHUP`, `SIGCHLD`, `SIGCONT` and `SIGUSR1` signals. You don't have to write code for each one of them because the match block that's going to handle them can have "a catch all remaining signals" case. The thing that you should remember is that you can define as many signals as you want. However, usually, we don't need to be able to handle so many signals.

The logic of `./src/main.rs` is found in the following code excerpt:

```

match signal {
    SIGINT => {
        println!("Got SIGINT");
    },
    ...
    other_signal => {
        println!("Got {:?}", other_signal);
    },
}
    
```

What the previous Rust code shows is that you should define the way each signal is going to be handled using pattern matching and the help of a match block.

Among the things that the utility does is to print its process ID using `process::id()` because it's required by the `kill(1)` command. Additionally, there's an endless loop that keeps running in order for the utility to keep handling signals. The only way to normally exit that loop is by sending the `SIGTERM` signal.

Interacting with the `multiple_signals` Cargo project generates the following output:

```

Process ID is 39908
Got 10
    
```

» TYPES OF LINUX PROCESSES

A process is an execution environment that contains instructions, user data and system data parts, and other types of resources that are obtained during runtime. On the other hand, a program is a binary file that contains instructions and data that are used for initialising the instruction and user data parts of a process. Put simply, a process is a program that runs. Each running UNIX process is uniquely identified by an unsigned integer, which is called the process ID of the process.

There are three categories of processes: user processes, server processes and kernel processes. User processes run in user space and usually have no special access rights whereas kernel processes are executed in kernel space only and can access all kernel data structures. Server (Daemon) processes are programs that can be found in the user space and run in the background without the need for a terminal. A dedicated user other than root owns them (for security reasons) and usually they have more rights than an ordinary user process but fewer privileges than a kernel process. A web server is a server process, the `cp` utility is a user process and `kworker` is a kernel process.

```

Got 1
Got SIGINT
Got SIGTERM
Good bye cruel world!
    
```

The commands that created the previous output were the following:

```

$ kill -SIGUSR1 39908
$ kill -SIGHUP 39908
$ kill -SIGINT 39908
$ kill -SIGTERM 39908
    
```

Because both `SIGUSR1` and `SIGHUP` are handled by the default case, they are printed by number, which isn't the case with `SIGINT` and `SIGTERM`.

What you should remember from this section is that signal handling in Rust isn't as straightforward as in other systems programming languages, mainly because Rust doesn't have built-in support for signal handling. However, with a little help from external crates, it can be done. You just need a little perseverance! **LXF**

```

let mut signals = Signals::new(SIGNALS)?;

'outer: loop {
    for signal in signals.pending() {
        match signal {
            SIGINT => {
                println!("Got SIGINT");
            },
            SIGTERM => {
                println!("Got SIGTERM");
                break 'outer;
            },
            SIGCONT => {
                println!("Got SIGCONT");
            },
            other_signal => {
                println!("Got {:?}", other_signal);
            },
        }
    }
}

println!("Good bye cruel world!");
    
```

Figure 6: Here's the main loop found in the `multiple_signals` Cargo project that showcases how to write a utility that handles multiple UNIX signals.

PYTHON

Updating the LXF GIT monitoring tool

John Schwartzman adds new features to an existing PyQt5 program that finds all of your git repositories and displays their status.



OUR EXPERT

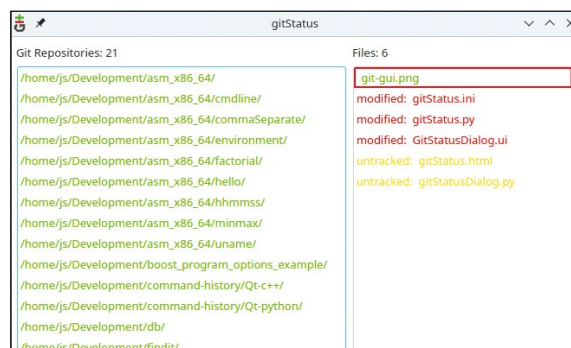
John Schwartzman has enjoyed an active career as an engineer, college professor, and consultant to business and government. He can be reached at john@forte.system.com.

The program that's the subject of this tutorial first appeared in **LFX260**. It found and displayed your git repositories with a colour scheme indicating their status, but we wanted it to be a one-stop shop for monitoring git repositories, viewing or editing files and using git to modify repositories. We achieved this by adding two context-sensitive menus. If you right-click a selection in the file widget (the right pane), an editor will appear that enables you to view or change a component file of a repository. Select which editor you want to use in the configuration file, **gitStatus.ini**.

If you right-click a selection in the repository widget (the left pane), a terminal will be opened in the repository directory and you can use git to add, remove or commit files. When you exit the terminal, the **gitStatus** tool will display any changes you may have made. The **gitStatus** program is shown in the screenshot (below).

The colour (priority) that's assigned to each file and to each repository is arbitrary. We decided that a checked-in file should be green, a modified file that hasn't been checked-in should be red, and that any other variety, for example, added, removed, renamed or untracked files, should be orange. The colour of a repository entry is the colour of worst-case file in the repository. If you want a different colour (priority) scheme, modify **quickCheckPriority(repoPath)**

Qt5 Designer shows the edit tab order screen. All of the widgets have been added to the project.



The **gitStatus** tool has found the git repositories. A repository has been selected in the left pane. The right pane shows the status of the files.

and **repoSelectionChanged()**. Each element of a **listWidget** contains colour data.

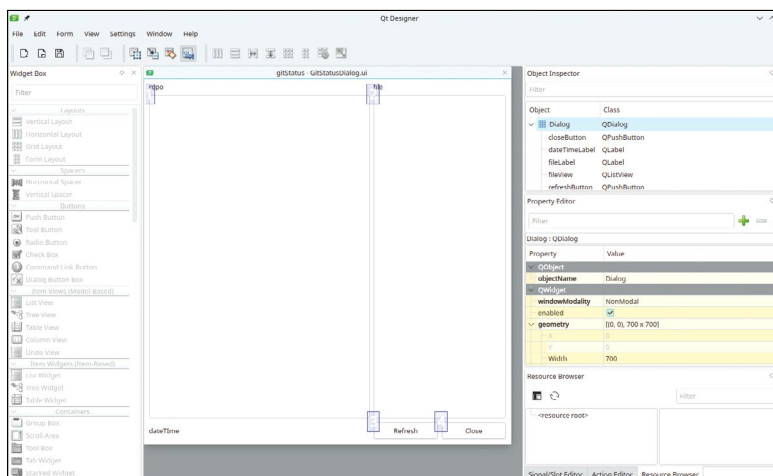
We'll create **gitStatus.py** using **Qt 5 Designer**. This enables you to create your tool's main dialog visually. **Qt 5 Designer** saves your work in an XML file. It gives your file a **.ui** extension. We'll use the Qt 5 utility program, **pyuic5**, to convert the XML file into a python file.

Start **Qt 5 Designer** from the system menu or by typing **designer-qt5** from the command line. Select 'Dialog without Buttons' from the File>New menu. (**Qt 5 Designer** should automatically open the New Form Dialog when you start the program without a file name.)

Drag three labels, two list widgets and two buttons on to the dialog from the Widget Box list on the left-hand side of the page. Give all of the dialog's components meaningful names using the right-hand side Property Editor. Select the dialog in the Object Inspector and select Form>Lay Out in a Grid. This will cause the dialog components to resize when you resize the dialog. You can select Form>Preview to make sure the dialog and all of its components resize properly when you drag the dialog's right or bottom edge.

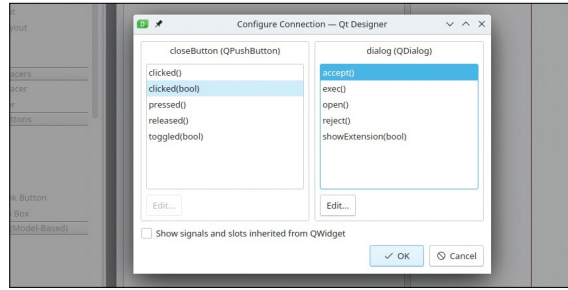
Select Edit>Edit Tab Order and ensure the tab order matches the screenshot (see left). Edit the tab order on this screen if it doesn't look like the screenshot.

Highlight Dialog in Object Inspector, scroll down to windowTitle in the Property Editor and type **gitStatus**. Scroll down to windowIcon in the Property Editor, click



the drop-down, select Choose File... and highlight *git-gui.png* in the program directory. (This icon is from the *Git Gui* application.)

Select Edit>Edit Signals/Slots and drag the closeButton on to the dialog's built-in close button. Release left mouse button and complete the Configure Connection dialog as shown in the screenshot (right). Try Form>Preview again and make sure that when you click closeButton, the preview ends. That tells you that you connected closeButton properly. Save your work as **GitStatusDialog.ui** and close Qt 5 Designer.



Connecting the buttonClose(bool) clicked SIGNAL to the Dialog accept() slot in Qt5 Designer. Notice the line dragged between buttonClose and the dialog's X button.

Convert **GitStatusDialog.ui** into **GitStatusDialog.py** by entering the following on the command line:

```
$ pyuic5 -x -o gitStatusDialog.py GitStatusDialog.ui
```

This creates **gitStatusDialog.py** from **GitStatusDialog.ui**. Because **gitStatusDialog.py** is a standalone program, it needs a main method. The `-x` argument tells *pyuic5* to add a main method. Note that if you run *pyuic5* more than once with the same arguments, *pyuic5* will recreate your target python file. Be careful not to lose your work in this way. We make our changes to a temporary file like **gitStatusDialog.py** and merge any changes into **gitStatus.py**. Now type:

```
$ python3 ./gitStatusDialog.py
```

You should see the dialog you created. It has all of its components in all of the right places, but it doesn't really do anything.

Take a look at **gitStatus.html** in the section marked 'Methods defined here:.'

```
$ firefox gitStatus.html
```

You can make **gitStatus.html** yourself by typing the following at the command line:

```
$ pydoc3 -w ./gitStatus.py
```

Now edit **gitStatusDialog.py**. Add `#!/usr/bin/env python3` as the first line of the file so that you can run **gitStatusDialog.py** from the command line.

```
$ ./gitStatusDialog.py
```

To use context-sensitive menus we need to change the inheritance of `Ui_Dialog` so that its parent is `QDialog` instead of `object`. Once you've changed `Ui_Dialog`'s inheritance, you also need to add the following method:

```
def __init__(self):
    """
    This method initializes Ui_dialog's base class QDialog.
    """
    super().__init__()
```

At this point you can add all of the methods from **gitStatus.py** that are noted in **gitStatus.html**'s 'Methods defined here:' to **gitStatusDialog.py**, or you can use

gitStatus.py from now on. See the partial list of **gitStatus.py**'s methods in the screenshot (overleaf) or open **gitStatus.html** in your browser and view the full list of method's there.

Now let's look at a few of the most important methods in **gitStatus.py**. The main method instantiates a `QCommandLineParser` object to read or create **gitStatus.ini**. It needs two initialisation variables, `$HOME` and `$EDITOR`, to get started. These are typically defined in `~/.bashrc`, but if you start **gitStatus.py** from a menu or from an icon, it won't be able to find these variables. To get around this problem, edit `/etc/environment` as root and add the following two lines:

```
HOME=/home/<your user name>
EDITOR=/usr/bin/<the editor gitStatus.py should use>
```

On our distro `/etc/environment` looks like this:

```
HOME=/home/js
EDITOR=/usr/bin/kate
```

After running **gitStatus.py** for the first time without **gitStatus.ini** present, our ini file looks like this:

```
[startDirs]
/home/js
[exceptDirs]
[terminal]
name = konsole
[viewers]
standard = kate
image = okular
browser = firefox
```

We have a kde distro, so we found *konsole* for the terminal and *okular* for the image browser. On our Ubuntu distro that uses GNOME our ini file looks like

```
[startDirs]
/home/js
[exceptDirs]
[terminal]
name = gnome-terminal
[viewers]
```

QUICK TIP

To get the latest version of python3 and PyQt5, use `sudo apt-get install python3.10` then run `pip install PyQt5`.

QUICK TIP

Get the code for this tutorial from the Linux Format archive: www.linuxformat.com/archives/?issue=291

» **FETCHING THE STATUS**

In order to determine the status of the files in a git working directory, we interrogate the repository using the `git status --porcelain` command. The `--porcelain` option makes it a bit easier for a computer to read the output. It simply proceeds the file path with an M for modified, D for deleted, R for renamed, A for added and ?? for untracked. A modified file has a different version in the repository from the version

in the working directory, so your repository isn't up to date: you haven't committed your latest changes.

An untracked file is present in the working directory, but doesn't exist in the repository. Perhaps it's a file that you meant to commit, but haven't got around to it yet.

A deleted file exists in the repository but isn't present in the working directory. It can be recovered from git, but perhaps

it should no longer be in the repository and should be removed.

Note that there may be files present in the working directory that aren't meant to be in the repository. For example, the build process may produce artefacts. Any file like this should be listed in a **.gitignore** file in your working directory or in **.gitignore_global**. Everything in either your local **.gitignore** file, or in your global **.gitignore** file is ignored by *gitStatus*.



QUICK TIP

Running `pydoc3 ./gitStatus.py` provides a list of the methods and data used in `gitStatus.py` and enables you to browse it using `less`. Running `pydoc3 -w ./gitStatus.py` will produce `gitStatus.html`, which you can view in a browser.

```
standard = gedit
image = eog
browser = firefox
```

Note that you can use *Firefox*, *Chrome* or another browser as your image viewer.

On both distros this tells the `populateDialog()` method to look for git repositories in the `/home/js` directory and that there are no directories that should be excluded. If you put `gitStatus.py` in a directory like `/usr/local/bin`, which isn't writable to everyone, you can run `gitStatus.py` as root the first time, or create and edit `gitStatus.ini` as root and fill in the values you want. We place our own dev work in the `/home/js/Development` directory. So we change the `gitStatus.ini`'s `[startDir]` to read `/home/js/Development`. That saves time searching each time `gitStatus.py` is run, because it doesn't have to walk through our home directory looking for directories named `.git`.

If you run `gitStatus.py` from the system menu or from an icon, it will die silently if it can't create and write its `.ini` file. If you run it from the console in verbose mode:

```
$ ./gitStatus.py --verbose
```

then `parseConfigFile()` and `populateDialog()` will tell you what's happening. Incidentally, when you press the Refresh button while running `gitStatus.py`, both `parseConfigFile()` and `populateDialog()` will be called. In addition to the `--verbose` option, `gitStatus.py` also has a help option (`-h` or `--help`) and a `--version` option. These options are only useful when you run `gitStatus.py` in a terminal. Because the Refresh button causes `parseConfigFile()` to be invoked, you can edit `gitStatus.ini` while `gitStatus.py` is running and then press the Refresh button to see your changes.

`parseConfigFile()` also checks whether the standard editor that you pick needs to be run in a terminal and sets the `useTerminal` to `True` or `False` accordingly. Editors like *vim* or *nano* need to run in a terminal, while GUI editors like *kate* or *kwrite* don't.

`populateDialog()` walks all the `[startDirs]` directories depth first and looks for all git repositories that don't exist in any of the `[exceptDirs]` directories. It uses the python list comprehension at line 438,

```
dirs[:] = [d for d in dirs if not in self.exceptDirs]
```

after the `dirs` list has been populated. For each path that's left in `dirs[]` we check to see if it contains the hidden directory `.git`. If it does, we've found a git repository working directory.

`populateDialog()` places each git repository that it

```
class Ui_dialog(PyQt5.QtWidgets.QDialog)
    @dialog(parent: QWidget = None, Flags: Union[Qt.WindowFlags, Qt.WindowType] = Qt.WindowFlags)

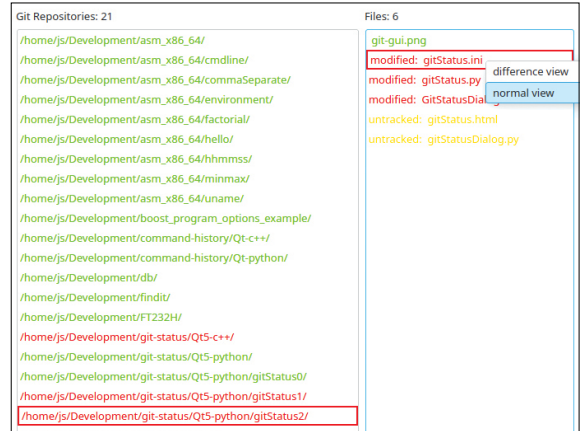
    Method resolution order:
    Ui_dialog
    PyQt5.QtWidgets.QDialog
    PyQt5.QtWidgets.QWidget
    PyQt5.QtCore.QObject
    sip.wrapper
    PyQt5.QtGui.QPaintDevice
    sip.simplewrapper
    builtins.object

    Methods defined here:
    __init__(self)
        This method initializes Ui_dialog's base class QDialog.

    appendSlash(self, str)
        This function returns 'str/'
        whether or not str ends with '/'

    eventFilter(self, source, event)
        handle context menu clicks (right click)
        on 'items' in repoView and FileView
        - create and display appropriate menu based on file type and status
        - create command
        - call runProcess(command, path)
        Note: path may already be in the command. Use path = None (default)
```

`gitStatus.html` is being viewed in a browser. Notice the Methods defined here: section.



`gitStatus.py` shows the context sensitive menu for a selection right-clicked in the fileView pane.

finds in the `QListWidget repoView`. It calls `quickCheckPriority(repoPath)` to assign a colour (priority) to each repository that it finds. It then places the focus in `repoView` and sets the current row to 0 (lines 474 and 475). It then writes the number of git repositories that it found in `repoLabel` (line 472). Setting the `repoView` selection to the first item in the list causes the `repoSelectionChanged()` method to be invoked. This method is also invoked when you manually navigate to another row in `repoView`.

`repoSelectionChanged()` changes `repoView`'s stylesheet to place a border around the selected item. It then clears the `fileView` `QListWidget` and repopulates it. `repoSelectionChanged()` is a complicated method that first invokes `git ls-files` to ask git what files it thinks belong to the selected repository. `git ls-files` doesn't give us any status information about the files. It places that list in `strArray` (line 325). It then places the list with three spaces in front of each item into the list `strList`.

`repoSelectionChanged()` then calls `git status --porcelain` to find out the status of all of the files in the repository (line 336). Now we have two lists: one with status information, one without. It then reconciles the two lists into `strList`. `repoSelection()` then calculates the colour for each item and adds each item to `fileList`.

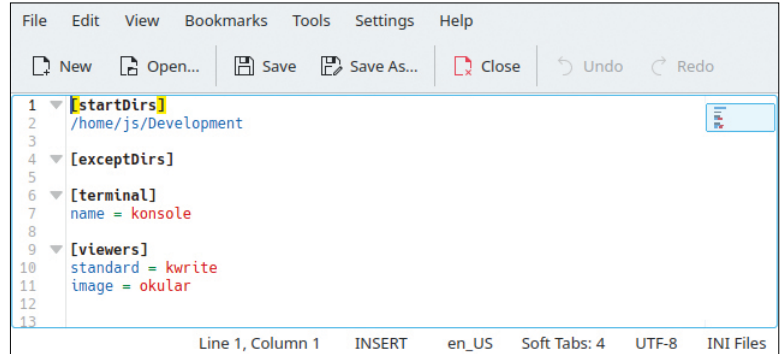
The older version of `gitStatus.py` didn't show any file status information on a git repository that has all files committed and no added, deleted, moved, renamed or untracked files. This version does, so we have to reconcile the two lists using `repoSelectionChanged()` for creating and reconciling the two lists:

```
# get list of files in repository
dir = self.repoView.currentItem().text()
p = Popen(["git", "ls-files"], cwd = dir, stdout = PIPE)
p.wait()
out = p.communicate()[0]
strArray = out.splitlines()
i = 0
sizeOfList = len(strArray)
strList = []
while i < sizeOfList:
    # leave space for status
    s = strArray[i].rjust(3 + len(strArray[i]))
    strList.append(s)
    i += 1
# check for status of files
p = Popen(["git", "status", "--porcelain"],
```

```

cwd = dir, stdout = PIPE)
p.wait
out = p.communicate()[0]
strArray = out.splitlines()
# reconcile the 2 file lists
i = j = 0
itemsResolved = []
while i < len(strArray):
    found = False
    if itemsResolved.__contains__(i):
        i += 1
    while j < len(strList):
        a = strList[j][3:]
        b = strArray[i][3:]
        if a == b: # file names match
            strList[j] = strArray[i]
            itemsResolved.append(i)
            found = True
            break
        j += 1
    if not found:
        # might be untracked file
        strList.append(strArray[i])
        i += 1
# color code the file list
sizeOfList = len(strList)
i = 0
while i < sizeOfList:
    line = strList[i].decode()
    i += 1
    x = line[0] # char x indicates changes for commit
    y = line[1] # char y indicates changes not yet staged
    if x == 'M' or y == 'M': # modified file
        s = "modified: "
        color = COLOR_RED
    elif x == 'D' or y == 'D': # deleted file
        s = "deleted: "
        color = COLOR_ORANGE
    elif x == 'A' or y == 'A': # added file
        s = "added: "
        color = COLOR_ORANGE
    elif x == 'R' or y == 'R': # renamed file
        s = "renamed: "
        color = COLOR_ORANGE
    elif x == 'C' or y == 'C': # copied file
        s = "copied: "
        color = COLOR_ORANGE
    elif x == '?' or y == '?': # untracked file
        s = "untracked: "

```



```

color = COLOR_ORANGE
elif x == '' and y == '': # normal file
    s = ""
    color = COLOR_GREEN
# prepend status string
item = QListWidgetItem(s + line[2:])
item.setData(Qt.UserRole, color)
item.setForeground(QColor(color))
self.fileView.addItem(item)

```

The file viewer/
editor presented
when normal view
is chosen.

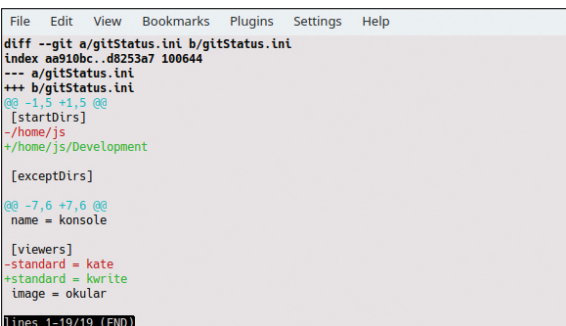
`repoSelectionChanged()` then selects `fileView`'s selection to row 0. That invokes `fileSelectionChanged()` that changes `fileView`'s stylesheet to place a border around its selected row. This also happens when you scroll through `fileView`.

The last method to examine is `eventFilter()` (line 108). This is invoked when the user right-clicks a file in `fileView` or a repository in `repoView`. This is the context-sensitive menu event – see screenshot (*facing page, top*).

If a file was right-clicked in `fileView`, `eventFilter()` sees if the file is an image or an HTML file. If the file is an image file, the context menu offers you an image viewer to view the image. If the file in a HTML file, the context menu offers you a browser to view the file. If `git` reports the file as modified, the context menu offers you two ways to view it: in an editor, or as `git diff` in a terminal. The screenshot (*below*) shows `gitStatus.ini` being opened in difference view, where the old values are in red and the new values in green. The screenshot (*above*) shows `gitStatus.ini` edited in normal view. Normal view shows one version of a file, which can be edited, while difference view compares two versions. The difference view can't be edited. Added, moved, renamed or untracked files can be viewed with your default editor. Deleted files can't be viewed. `eventFilter()` uses the `useTerminal` member variable to alter the command variable based on whether `useTerminal` is True or False. The `git diff` viewer must be run in a terminal.

If a repository was right-clicked in `repoView`, `eventFilter()` offers you a terminal that you can use to talk directly to `git`. After you close the terminal, `gitStatus.py` reappears and refreshes itself so that you see any changes that you might have made.

We were unable to make `gitStatus.py`'s dialog work as a modal dialog and we couldn't disable the dialog when we launched a process. We chose to hide the dialog when launching a process and show it again when the process finished. This is shown in the function at line 193: `def runProcess(self, command, path = None)`. We hope this gives you ideas for your own projects! **LXF**



The file viewer presented when difference view is chosen.

» GET UPDATES AUTOMATICALLY Subscribe now at <http://bit.ly/LinuxFormat>

NEXT MONTH

LINUX
FORMAT
The #1 open source mag

Future Publishing Limited,
Quay House, The Ambury, Bath, BA1 1UA
Email linuxformat@futurenet.com

EDITORIAL

Editor Neil Mohr
Virtual editor Jonni Bidwell
Art editor Efrain Hernandez-Mendoza
Operations editor Cliff Hope
Group editor-in-chief Graham Barlow
Group art director Jo Gulliver

Editorial contributors

Michael Bedford, Neil Bothwick, Sean Conway, Dave James, Matthew Hanson, Matthew Holder, Jon Masters, Nick Peers, Aaron Peters, Les Pounder, Michael Reed, David Rutland, John Schwartzman, Mayank Sharma, Shashank Sharma, Mihalis Tsoukalos

Cover illustration magictorch.com

Raspberry Pi is a trademark of the Raspberry Pi Foundation.
Tux credit: Larry Ewing (lewing@isc.tamu.edu) and *GIMP*.

ADVERTISING

Commercial sales director Clare Dove
clare.dove@futurenet.com

Senior advertising manager Lara Jaggon
lara.jaggon@futurenet.com

Head of commercial – Technology Dave Randall
dave.randall@futurenet.com

Account director Andrew Tilbury
andrew.tilbury@futurenet.com

INTERNATIONAL LICENSING

Head of Print Licensing Rachel Shaw
Linux Format is available for licensing and syndication.
To find our more contact us at licensing@futurenet.com or view our content at www.futurecontenthub.com.

NEW SUBSCRIPTIONS & BACK ISSUES

Web www.magazinesdirect.com
UK 0330 333 1113 **World** +44 (0) 330 333 1113

EXISTING SUBSCRIPTIONS

Web www.mymagazine.co.uk
UK 0330 333 4333 **World** +44 (0) 330 333 4333

Subscription delays: Disruption remains within UK and International delivery networks. Please allow up to seven days before contacting us about a late delivery to help@magazinesdirect.com

CIRCULATION

Head of newstrade Tim Mathers

PRODUCTION AND DISTRIBUTION

Head of production UK & US Mark Constance

Production project manager Clare Scott

Senior ad production manager Jo Crosby

Digital editions controller Jason Hudson

THE MANAGEMENT

MD, tech specialist Keith Walker

Head of art and design Rodney Dive

Design director Brett Lewis

Commercial finance director Dan Jotcham

Printed by Wyndeham Peterborough, Storey's Bar Road, Peterborough, Cambridgeshire, PE1 5YS

Distributed by Marketforce, 5 Churchill Place, Canary Wharf, London, E14 5HU www.marketforce.co.uk

Tel: 0203 787 9001

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries. GNU/Linux is abbreviated to Linux throughout for brevity. Where applicable code printed in this magazine is licensed under the GNU GPL v2 or later. See www.gnu.org/copyleft/gpl.html. All copyrights and trademarks are recognised and respected.

Disclaimer All contents © 2022 Future Publishing Limited or published under licence. All rights reserved. No part of this magazine may be used, stored, transmitted or reproduced in any way without the prior written permission of the publisher. Future Publishing Limited (company number 2008885) is registered in England and Wales. Registered office: Quay House, The Ambury, Bath BA1 1UA. All information contained in this publication is for information only and is, as far as we are aware, correct at the time of going to press. Future cannot accept any responsibility for errors or inaccuracies in such information. You are advised to contact manufacturers and retailers directly with regard to the price of products/ services referred to in this publication. Apps and websites mentioned in this publication are not under our control. We are not responsible for their contents or any other changes or updates to them. This magazine is fully independent and not affiliated in any way with the companies mentioned herein. If you submit material to us, you warrant that you own the material and/or have the necessary rights/permissions to supply the material and you automatically grant Future and its licensee a licence to publish your submission in whole or in part in any/all issues and/or editions of publications, in any format published worldwide and on associated websites, social media channels and associated products. Any material you submit is sent at your own risk and although every care is taken, neither Future nor its employees, agents, subcontractors or licensees shall be liable for loss or damage. We assume all unsolicited material is for publication unless otherwise stated, and reserve the right to edit, amend, adapt all submissions. All contents in this magazine are used at your own risk. We accept no liability for any loss of data or damage to your systems, peripherals or software through the use of any guide. Notes: Cliff has been dreaming of nuclear war breaking out.

We are committed to only using magazine paper derived from responsibly managed, certified forestry and chlorine free manufacture. The paper in this magazine was sourced and produced from sustainable managed forests, conforming to strict environmental and socioeconomic standards. The manufacturing paper mill and printer hold full FSC and PEFC certification and accreditation.



Future is an award-winning international media group and leading digital business. We reach more than 57 million international consumers a month and create world-class content and advertising solutions for passionate consumers online, on tablet & smartphone and in print.



Future plc is a public company quoted on the London Stock Exchange (symbol: FUTR)
Chief executive **Zillah Byng-Thorne**
Non-executive chairman **Richard Huntingford**
Chief financial officer **Penny Ladin-Brand**
www.futureplc.com Tel +44 (0)1225 442244



LXF292
will be on sale
Tuesday
26 July
2022

OPEN SOURCE GRAPHICS

As Nvidia moves to open source its driver, we take a deep-dive into the kernel graphics stack.

Get some photo flare

Discover the photo-editing software that's easy to use, super speedy and won't melt your brain like *GIMP*.

RISC OS relived

We kick off a tour of open source operating systems to learn how they work, what they can do and where they came from.

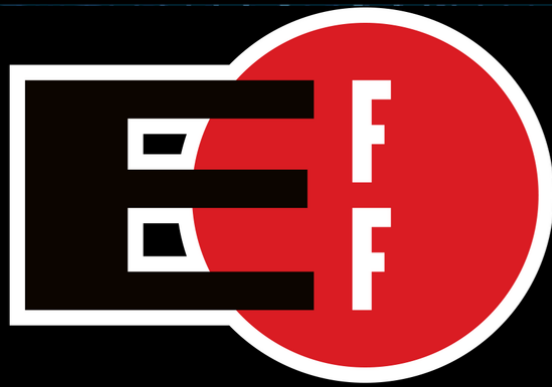
Let's play Tetris!

Pick up the skills to code a non-trademark infringing version of *Tetris* that you can control with an Xbox controller.

Digital whiteboards

Need to present your ideas live? Want to brainstorm a project online with your team? We test out the best tools to use.

Contents of future issues subject to change – we might be too transfixed by the eye candy!



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

WIN £50
OF NATIONAL BOOK TOKENS

We're giving away £10,000 to spend on books!
Scan the **QR code** or visit **lovepaper.org** to enter



PAPER POWER

62% of the energy used to produce paper and paper-based packaging in Europe comes from renewable sources.

Source: Confederation of European Paper Industries (CEPI), 2020.
CEPI represents 92% of European pulp and paper production



Discover the story of paper
www.lovepaper.org
Scan for paper facts, activities,
blogs and much more!



www.lovepaper.org

With thanks to

