NEOPIXEL LED MASTERCLASS! | THE ALL-NEW $6 Pi PICO W

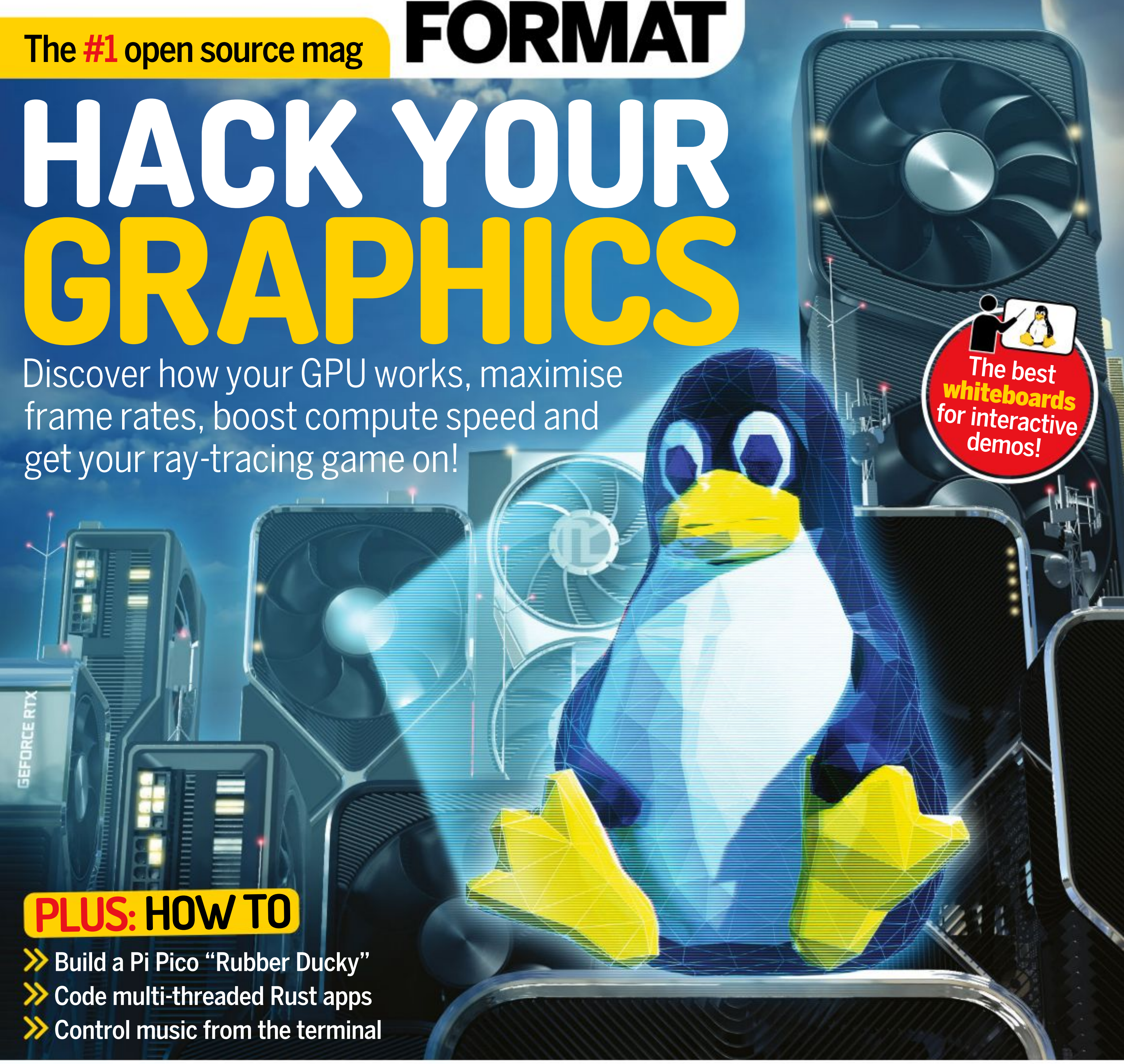
LINUX FORMAT

The #1 open source mag

Dive inside the original Arm-based RISC OS

HACK YOUR GRAPHICS

Discover how your GPU works, maximise frame rates, boost compute speed and get your ray-tracing game on!



The best **whiteboards** for interactive demos!

PLUS: HOW TO

- Build a Pi Pico "Rubber Ducky"
- Code multi-threaded Rust apps
- Control music from the terminal

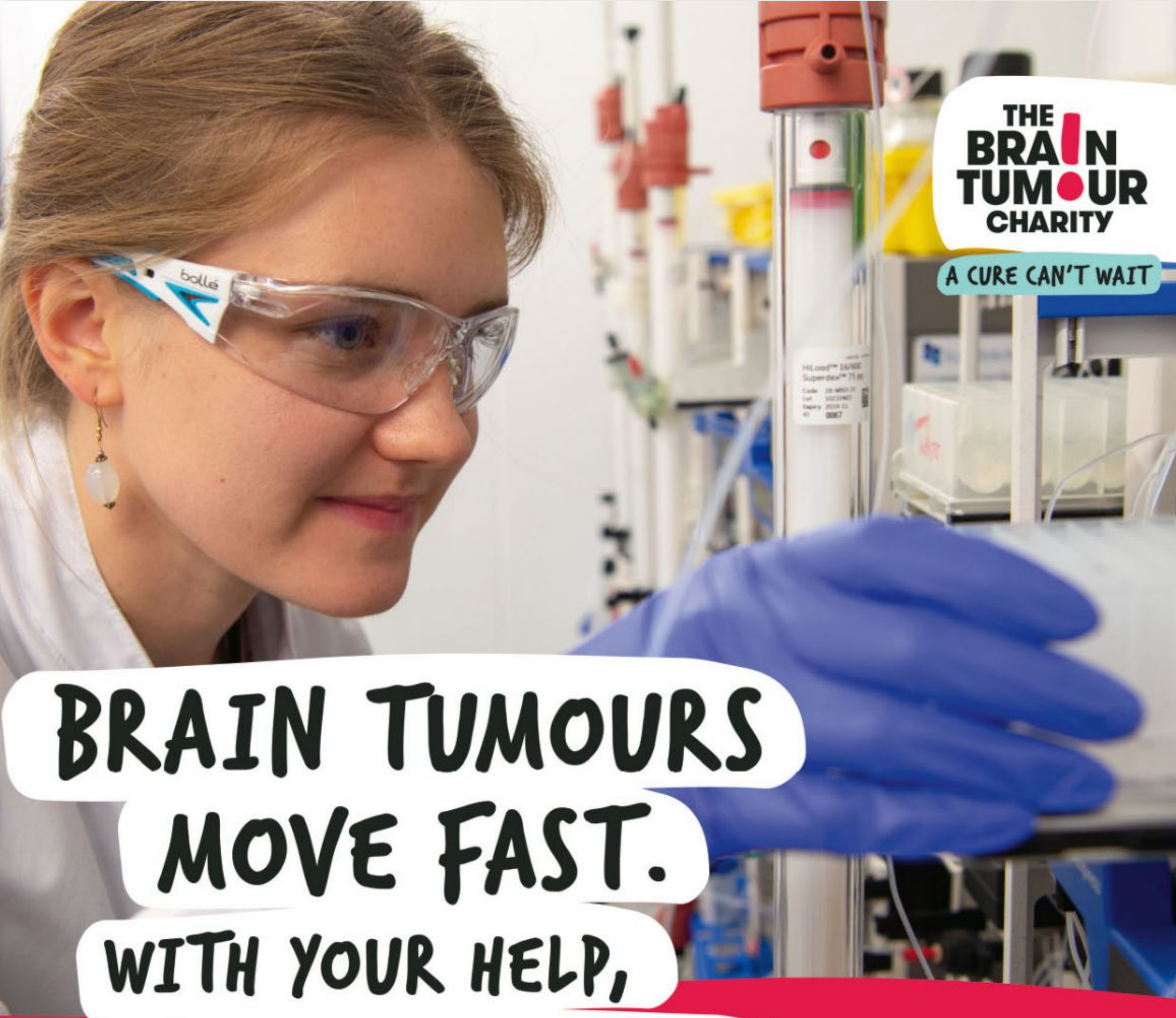
PHOTO EDITING Use Photoflare's easy yet powerful tools

GET INTO GITPOD Taking Git further with dev environments

CODING TETRIS! Adding gamepad support to a classic

LXF Summer 2022

FUTURE



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

MEET THE TEAM

This issue we're diving into the kernel's graphic stack. We asked our writers if there's an area of kernel (or general open source) technology that they've fallen down a rabbit hole while investigating?

Jonni Bidwell
I was overwhelmed when I first booted Slackware in 1996. I knew **/dev/fd0** was a disk, but had no idea what 'mounting a filesystem' meant. I don't think I bothered with Linux again until five years later. Since then there have been more rabbit holes than there are rabbits in the South West…

Matthew Holder
Ahh, isn't the stench of nostalgia wonderful? The biggest rabbit hole I've fallen down was configuring my **XF86Config/xorg.conf** file to support dual monitors. This was back in the day when the fear of entering incorrect settings could damage your monitor. I do not miss those days!

Les Pounder
The Pi's GPIO hides many things, including alternative GPIO options. Underneath the standard digital I/O is a myriad of protocols, and I spent too long investigating I2C and SPI. These protocols offer less wires, multiple devices chained together and offer a range of components.

Michael Reed
I only tweak my kernel these days for audio work. Unfortunately, this can still be a dark art. The general advice is to use the kernel version with RT (real-time) patches, but I've sometimes found this approach causes problems, and I've gone back to the stock kernel. Maybe Pipewire will improve matters?

Mayank Sharma
Disagreements in open source tend to all become rabbit holes thanks to the potpourri of personalities and cultural sensibilities, but the deepest one has to be the adoption of *Systemd*. You can spend aeons reading through threads, but won't be able to understand why it was (is?) so heavily lambasted.

New world order!

It can feel like desktop Linux users are second-class computer citizens, even if in reality this isn't true. Perhaps it's because so much marketing goes towards Windows and iPhones, and every system you see for sale is Windows or MacOS. And while gaming gets a lot of attention, until recently it always felt like an afterthought for Linux.

While its motivations aren't for gaming, after talking about Nvidia intending to open source its driver stack last month, we're diving in head first to look at how the Linux kernel, your graphics hardware and drivers all work in harmony to produce delightful displays, fast fancy 3D and wonderful windows.

It's certainly an area that Linux has struggled with, but is a clear sign of how the world has changed. It does remain a complicated area and Jonni's drawn a huge diagram to help, but it also gives you plenty of fun stuff to play with along the way and you should come out the other side with a solid understanding of how the many elements all fit together to displays work.

Speaking of fitting, we've certainly crammed a lot into this month's issue of *Linux Format*. We're covering more coding with Rust, playing music from the terminal, exploring Gitpod development environments, reviewing open source whiteboards, controlling LEDs, assessing the new Pi Pico W and more! Open source and the Linux kernel are so expansive in topics and technologies, there will always be something for everyone to enjoy.

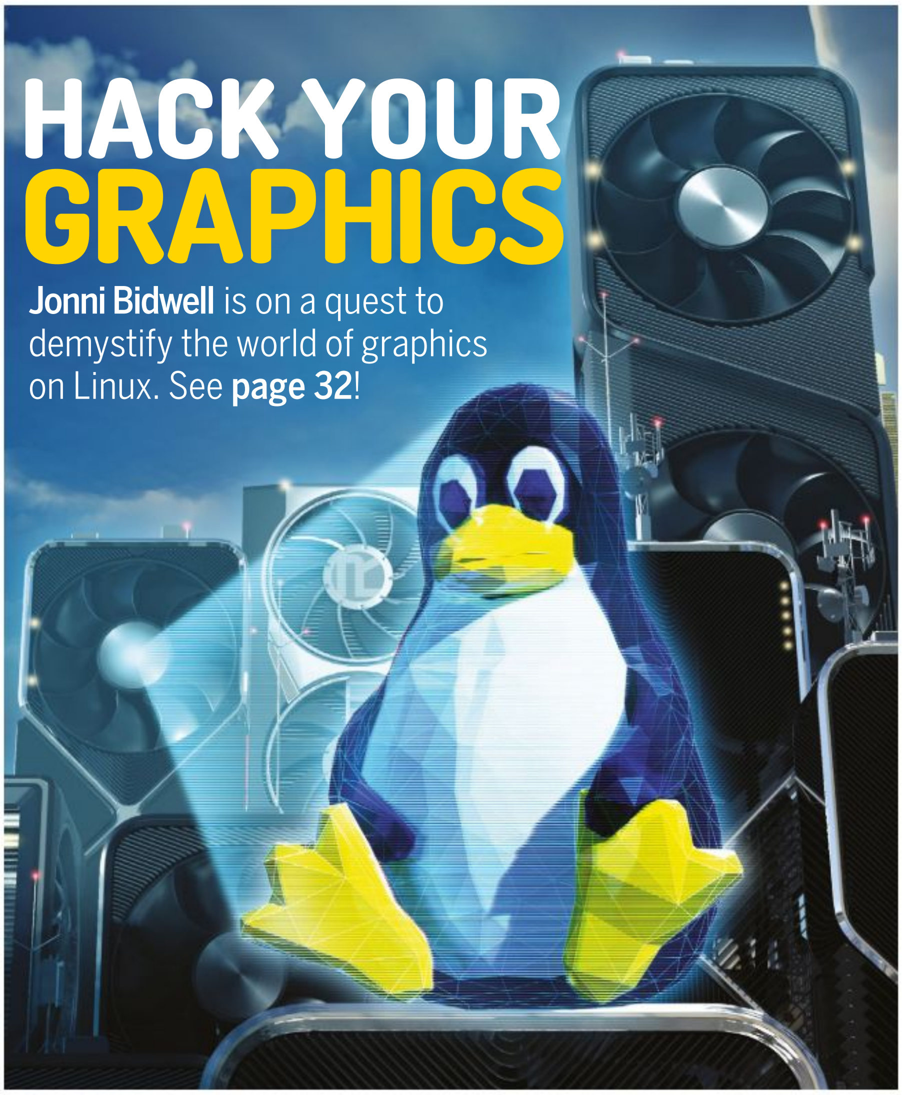
Neil

Neil Mohr Editor
neil.mohr@futurenet.com

Subscribe & save!
On digital and print – see p16

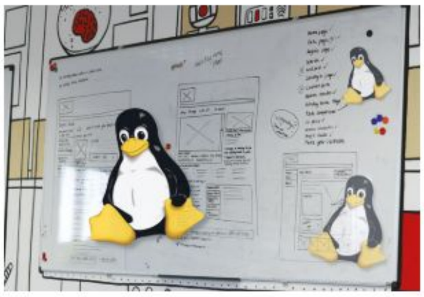
Contents

REVIEWS

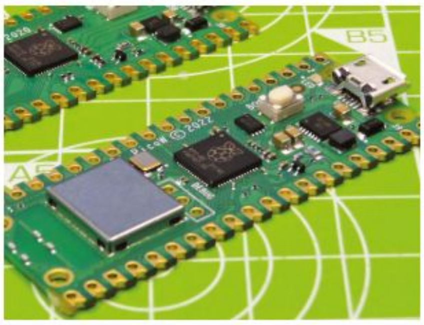
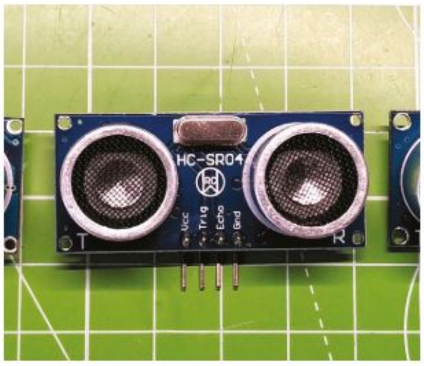


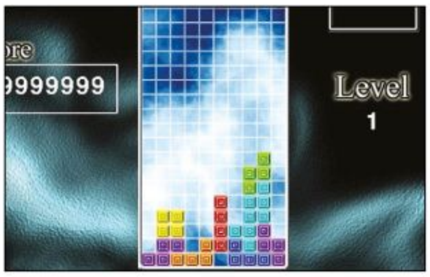
HACK YOUR GRAPHICS

ROUNDUP



IN DEPTH



HC-SR04
Vcc
Trig
Echo
Gnd
T
R


re
999999
Level
1

$.DrawFile Lock
Linux Format
Linux Format

Newsdesk

THIS ISSUE: Unix commits archived » AMD takes upscaling tech open source » Linux on HP laptops » Microsoft supports Gnome

HISTORY

The complete history of Unix commits

Want to feel old? A project on GitHub gathers together over 50 years of Unix commits for your delectation.

Unix is the operating system standard that Linux is based on, and it's been around for a long time. If you want a glimpse of computing history, or you're feeling nostalgic (and want to feel old) then a project on GitHub has collected every Unix commit since 1970 until today – and you can view it at **https://bit.ly/lxf292unixhistory**.

As you can imagine, it's a fascinating look at how the operating system evolved, "from its inception in 1970 as a 2.5 thousand-line kernel and 26 commands, to 2018 as a widely used, 30-million line system," as the project's GitHub page explains. The repository weighs in at 1.5GB, and contains half a million (amusingly displayed as infinity) commits and over 2,000 merges. The project is made up of two repositories: unix-history-repo, which offers a reconstructed version of Unix history, and users are encouraged to fork or archive it, rather than add contributions; and unix-history-make (**https://bit.ly/lxf292unixhistorymake**), which contains code and metadata for building the repository – and contributions are encouraged. Using a local copy of the repository, you can view and run Git commands, and explore over 50 years of code.

In other let's-feel-old news, graphical desktop system X Window has recently turned 38, having first been released way back on 19 June, 1984. You can read the original release announcement by Robert W. Scheifler at **https://bit.ly/lxf292xwindow**, and it's another fascinating glimpse back in time. As Scheifler writes, "I stole a fair amount of code from W, surrounded it with an asynchronous rather than a synchronous interface, and called it X. Overall performance appears to be about twice that of W." The lighthearted tone of the release announcement suggests Scheifler would never imagine that X Window would be so influential, even after almost four decades. He finishes off the announcement with, "Anyone who wants the code can come by with a tape. Anyone interested in hacking deficiencies, feel free to get in touch."

While we're on a bit of a history kick, a video has been uncovered of a certain Linus Torvalds from 1994, where he announces the new Linux kernel 1.0 (visit **https://bit.ly/lxf292linus94** to view the footage). In it, he explains that " Why do we make this kind of Unix? Especially in this kind of university environment, [because] there are these kind of Unixes available, even for PCs but their price level is very high." Offering a Unix-like operating system for free, and which people could freely contribute to, fork and clone, was revolutionary at the time, but it's a testament to Torvald's vision that Linux, too, has stood the test of time.

EXPLORE THE MANY UNIX COMMITS

"The repository weighs in at 1.5GB, and contains half a million commits and over 2,000 merges."

GitHub is host to over 50 years of Unix updates. Make yourself comfortable...

GAMING

AMD FidelityFX Super Resolution goes open source

AMD's performance-enhancing upscaling technology is now open source with version 2.0.

AMD FidelityFX Super Resolution 2.0 has been released. Not only is it an updated version of AMD's upscaling technology that can help improve game performance even on older graphics cards, but the company has also published the source code on GitHub as open source (**https://bit.ly/lxf292amdfsr2**).

While we're always pleased to see companies share their source code, this is a canny move by AMD. It puts a dividing line between FidelityFX Super Resolution and the rival DLSS (Deep Learning Super Sampling) from Nvidia, with AMD getting kudos from the open source community and gamers alike, who may be annoyed at Nvidia's proprietary technology. The latter also promises to improve game performance by upscaling a lower resolution image to make it appear to run at higher resolutions, without the high hardware demands.

AMD's FidelityFX Super Resolution is available for a range of GPUs, including much older graphics cards and even products from Nvidia. In contrast, DLSS is limited to recent Nvidia RTX GPUs. Making FidelityFX Super Resolution 2.0 should also hopefully make it easier for game developers to implement the technology in their titles.

AMD claims that the first version of FidelityFX Super Resolution was its "fastest adopted software gaming technology to date" with over 110 games supporting the tech. Yet this number could soon grow rapidly, and could put pressure on Nvidia. As AMD explains, "AMD FidelityFX Super Resolution 2 (FSR 2) technology is our brand-new open source temporal upscaling solution. FSR 2 uses cutting-edge temporal algorithms to reconstruct fine geometric and texture detail, producing anti-aliased output from aliased input."

AMD FidelityFX Super Resolution has been a big hit already, and by open sourcing version two we should see even more games support it.

LAPTOPS

HP launches Linux laptop

HP Dev One comes with Linux Pop!_OS preinstalled.

HP is the latest big laptop manufacturer to embrace Linux – to a degree – with the release of the HP Dev One. Rather than running Windows, this laptop, which HP is aiming at developers, comes with Linux Pop!_OS preinstalled, as well as a custom keyboard for using with Linux.

The specs are pretty impressive, with an eight-core AMD Ryzen 7 Pro CPU, 16GB of DDR4 RAM and 1TB of SSD storage, and it has a 14-inch display with 1080p resolution and 1,000 nit brightness, so it should be pretty comfortable to work on. At the moment it can only be pre-ordered in the US, and costs $1,099. Pop!_OS comes with the Cosmic desktop environment, enabling users to quickly tile windows and use multiple virtual desktops to help with coding, and it has access to Flatpak applications via the Pop Shop, along with frequent kernel and driver updates. If you're concerned about the amount of RAM, the good news is that it can be user-upgradable, so you can bump it up to 32GB or higher if your workload demands it.

It's great to see another big laptop maker provide a Linux alternative to its Windows laptops, which HP's main rival Dell has been doing for a while now. HP's recent laptops have been stylishly designed and well-built devices, and the HP Dev One looks set to follow suit. Hopefully, it'll come to more countries soon, and you can pre-order at **https://hpdevone.com**.

OPINION

SEE YOU IN VALHALL!

Alyssa Rosenzweig
is a software engineer at Collabora.

"The open source Panfrost driver for Mali GPUs now supports the new Valhall architecture with fully conformant OpenGL ES 3.1 on Mali-G57. The Valhall GPU is featured in the new MediaTek Chromebooks. With Mesa 22.2 and an appropriate kernel, accelerated graphics will work out-of-the-box on Linux on these laptops.

Valhall is based on the older Bifrost architecture that we already support. The driver only changes where the hardware changes. For example, Valhall has a streamlined instruction set, requiring a new back-end in our existing Bifrost compiler.

Valhall adapts its fixed-function hardware for Vulkan, removing legacy features like transform feedback and speeding up modern features like indirect draws. Fortunately, we can emulate the legacy features at a performance penalty. Valhall trades performance of the old for the new. And that's okay. Process node improvements mean hardware is getting faster, yet software is written for contemporary hardware, so old software remains fast on new hardware. No matter how much legacy cruft is removed to improve Vulkan, old OpenGL games will remain playable. The future is Vulkan – and Valhall embraces it."

OPINION

CRITICAL THINKING

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

"You back up all your company data, and hopefully you test those backups occasionally. But when it comes to backups do you treat all data equally? There'll be some data that's critical to your business. Customer lists, employee details, next year's budget: they're important, but they're not critical. If you lost them, there would be embarrassment, maybe loss of revenue, but the business would survive.

When business critical data is lost, that spells the end of the business. It's often the business's Intellectual Property (IP), which represents the value of the business to the shareholders and investors. For many research and high-tech businesses, it's a git repository. For my business, Tiger Computing, it's a git repo and a wiki. What's yours?

The critical data must be backed up, automatically, off-site to multiple independent locations as frequently as makes sense. The restore from backup needs to be documented and tested regularly.

You can rebuild your prospect list. You can ask your employees for their home addresses again. But if you have nothing to sell, you're sunk."

FUNDING

Microsoft gives Gnome a boost

Thanks to Microsoft staff and its FOSS Fund.

Gnome is a desktop environment (and much more) found in many distros, and has won a $10,000 grant from Microsoft.

The Gnome project has just received a helping hand thanks to funding from what some people may consider a rather surprising source – Microsoft. The company's Free and Open Source Software Fund (FOSS Fund), which on its GitHub page (**https://bit.ly/lxf292msfossfund**) says "provides a direct way for Microsoft engineers to participate in the nomination and selection process to help communities and projects they are passionate about."

A prize of $10,000 in sponsorship is awarded to projects that are selected by Microsoft, and the 20th fund, for May 2022, has been given to Gnome. Microsoft states that "rom low-level libs over the window manager to end user applications the GNOME project is an essential part of many graphical Linux devices. It also powers Ubuntu which is the first Linux work environment supported at Microsoft." At the time of writing, nominations for FOSS Fun #21 are open.

It's another great example of the once FOSS-hostile Microsoft now embracing and actively supporting open source projects, and while some of the more paranoid corners of the internet are worried that this could be a precursor to Microsoft acquiring Gnome, this is clearly a great thing to see, and will hopefully help Gnome reach greater heights.

MOBILE

Ubuntu Touch OTA-23 released

Supported devices getting the Ubuntu release for mobiles.

After being abandoned by Canonical, Ubuntu Touch was taken on by UBports, and the latest version, Ubuntu Touch OTA-23, has been released. Supported handsets, such as the Google Pixel 3a and OnePlus 6, will be able to install the mobile OS, which offers privacy and freedom to its users.

Ubuntu Touch, like the previous version, is based on Ubuntu 16.04, but there's some big updates. FM radios in supported handsets is now included, though it's in its early stages; the Messaging App is now improved; and support for wireless displays has also been fixed. See **https://bit.ly/lxf292ubuntutouch**.

Ubuntu Touch lives on as a truly free alternative to Android and iOS.

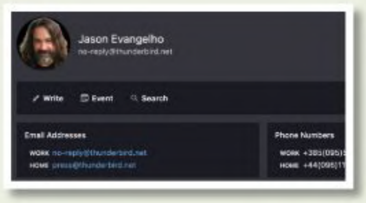
SOFTWARE

Thunderbird 102 now available

Big update for the open source email client.

Mozilla has announced a new version of its open source desktop email client, *Thunderbird* (see **https://bit.ly/lxf292thunderbird**). As the release announcement from Jason Evangelho states, "*Thunderbird 102* is loaded with highly-requested features, and we think you'll be delighted by them." The user interface has had a big refresh with new icons, colour folders and improved headers, along with a totally new Address Book.

According to Evangelho, this is the first step to modernising the entirety of *Thunderbird's* UI, with a clean design and compatibility with vCards, which is the industry standard for contacts. This should make importing contacts from other programs and services a lot easier.



This is the new-look Contact layout for Thunderbird 102. Snazzy!

Distro watch

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

PORTEUS 5.0

A new version of Porteus is available to download. Available in eight desktop variants, this Slackware distro now comes with the Linux kernel 5.18.8, as well as updates to Perl and various Slackware package managers. The latest version is also based on Slackware 15.0. A variety of bug fixes have been included, as well as optimisations to boost performance. You can find out more in the release announcement by visiting **https://bit.ly/lxf292porteus**.

Porteus 5.0 comes with eight desktop environments to choose from.

CONDRES OS 1.0

The first version of this Debian-based distro is now available. Based on Debian 11, and with a customised Plasma 5.20.5 desktop, the release announcement (found at **https://bit.ly/lxf292condres**) notes that "this first release will be followed by a full online calamares-based installer, we will integrate the Control Center that was previously available in the old releases on the new Debian-based version." Expect an update around September/October 2022 for these additional features.

Condres OS 1.0 is a new Debian-based distro, if you fancy giving it a whirl.

ENDEAVOUROS 22.6

This Arch-based distro, which has been termed "terminal-centric", has a new version out, code-named "Artemis", after the "upcoming NASA mission to the moon," according to the release announcement at **https://bit.ly/lxf292endeavouros**. It's certainly an ambitious release, mainly focusing on ARM support, and you can now choose an ARM installation from the main ISO, and while the current implementation only supports Odroid N2/N2+ and Raspberry Pi, it's a clear indication of where the team feel EndeavourOS' future lies.

EndeavourOS 22.6 continues the distro's embrace of ARM architecture.

TRUENAS 22.02.2

TrueNAS SCALE 22.02.2 is a new update for the distro aimed at network-attached storage (NAS) devices. According to the release notes (which you can read at **https://bit.ly/lxf292truenas**), this update brings over 160 bug fixes and improvements, including Dedup performance with SHA-512 checksum, and multiple containers can now use the same Intel GPU. For NAS devices with multiple drives, performance has also been improved when using middleware, and cluster deployment is also more simple thanks to the *TrueCommand* tool.

TrueNAS 22.02.2 is a great distro if you want custom software running on your network storage devices.

OPINION

THANKS ADMINS!

Matt Yonkovit
is the head of open source strategy at Percona

"Every year, on the first Friday in July, Database Administrator Appreciation Day takes place, which is about saying thank you to the people who are responsible for organising data. They are the ones who set up and run databases so they can manage and store information. They're responsible for keeping those databases secure, so your data doesn't leak. They want to keep things running as smoothly and efficiently as possible. And they make sure that all that data is backed up and protected, in case anything goes wrong.

All of those tasks listed are only noticeable when something goes wrong. It's all too easy to overlook that good work. We don't learn about areas like query design or efficient data management as much as we used to, as cloud services make it easier to just throw more resources and more money at problems. Our software creates data, needs data to function, and uses data to provide value to whoever uses the service.

Instead, this year, we should all decide to pay more attention to how we organise our data, how we ask the right questions of it, and how we make our investment around data go further. This takes skill to get right. So, DBAs of the world, thanks for your hard work."

OPINION

GO RUST!

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

"Linus Torvalds is a household name in Linux circles. Someone readers may not know as well is Dirk Hohndel, who's been a part of the open source scene since the 1990s, and is good friends with Linus. The two often conduct "fireside chats" at popular Linux Foundation events, and the latest Open Source Summit was no exception.

Among the topics discussed at the latest event was Rust support. As is typical of anything involving Rust, the justification is improved memory safety, and by extension fewer bugs or nasty security vulnerabilities. As Linus noted to Dirk, he's keen to "trial" adding Rust to Linux, and as soon as the next merge window perhaps, brushing aside concern that maintainers might not know the language.

So it seems that this might happen. It will be interesting because Rust still feels like it has a long way to go before it becomes truly mainstream, both in terms of features, and in terms of stability (language, not safety). But we do have an almost never-ending stream of nasty security bugs making headlines these days, so anything that can improve the status quo is worth considering. I'll use the opportunity to finally play with Rust, and write about it soon."

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 5.19-rc6, which he hopes to be one of the final "Release Candidate" (RC) kernels prior to the final 5.19 release. His eagerness showed, as his announcement mail noted, "Perhaps somewhat unusually, I picked up a few fixes that were pending in trees that haven't actually hit upstream yet." If things go according to plan, we'll see the opening of the "merge window" (period of time during which disruptive changes are accepted into the kernel) for what will likely be the last 5.x release (5.20) and have a subsequent summary of new features next month.

An unfortunate kernel bug

Recently, there came news of a nasty kernel bug that was affecting users of docker containers on Ubuntu systems, and in particular those running on cloud instances deployed at scale. The symptom was that upon starting a container, the virtual machine would "oops" (the term used for a fault occurring within the kernel itself), and ultimately panic (crash). Unfortunately, docker is started early in the boot process for many affected systems, meaning a simple *apt update* applied by an admin to a fleet of machines could lead to large numbers of them crashing on reboot and needing manual intervention to undo the damage.

What actually happened was that Ubuntu rebased (upgraded) kernels from 5.13 to 5.15 in updates to the 20.04 release since the 5.13 kernel had gone EOL (End Of Life). In the process of preparing the update, an Ubuntu-specific patch was included that contained a bug causing the oops. The same patch had previously been included in multiple other Ubuntu kernels, but it had bit-rotted over time and was no longer correct for recent kernels.

At the same time, the patch depended on a filesystem (AUFS) that's no longer enabled on 21.10+. Thus a broken patch lingered around but didn't cause problems until it was backported to a distro (20.04) that did enable AUFS. Such bugs caused by backports gone awry aren't rare across the Linux distros, but in this case a bunch of folks got unlucky by the fact that it would be automatically tickled on next boot and bring down machines that tried to spawn containers.

Various debates have occurred as a result of this bug, both about whether kernels should be rolled out differently (and removed from repos more swiftly in the face of mass crash reports), and about how better to fix large numbers of VMs hosted in the cloud. Fortunately, a growing number of clouds are now enabling serial console support for those who pine for the good old days of booting single user mode and fixing a machine from the console. Obviously, that does not scale beyond a handful of machines, but it is worth knowing about.

» ONGOING DEVELOPMENT

The latest series of CPU microarchitectural hardware security vulnerabilities include attacks against MMIO (memory mapped IO) peripherals in which a skilled attacker might be able to reconstruct certain stale data that is temporarily stored in internal buffers and not cleared between different operations. The amount of data that can be derived is limited, and an attacker needs to already have a certain amount of privilege. Patches are available in updated kernels.

Aneesh Kumar (IBM) posted a patch series titled "Memory tiers and demotion", which aims to overhaul the kernel's handling of tiered memory expressed as NUMA (non-uniform memory access) nodes within a system. Traditionally, "memory tiers are defined implicitly via a demotion path relationship between NUMA nodes, which is created during the kernel initialization". The patches change this to provide explicitly controls that benefit future system topologies, such as those using CXL (Compute Express Link) attached CXL.mem.

Ted Ts'o posted a note about the upcoming Kernel Summit, which will be co-located with the Linux Plumbers Conference in Dublin (September). The kernel developers will also have an invite-only "Maintainer Summit" following the main events. More details about the conference are at **https://lpc.events**. LXF

Answers

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

Neil Bothwick hammers open your kernel and walks away.

Q Pop!_OS woes

I recently refreshed my Pop!_OS install. Afterwards my account was no longer available on the lock screen or user settings screen, but I could log in using the User Not Listed button on the login screen. I did some duckduckgoing and decided to rename my home folder, then delete my old account and create a new account with the same username as before. I then deleted the generated home folder and replaced it with my old home folder. Is there a better way to do this, and is there a program I could use to reinstall some applications that I always install after an OS reinstall? I realise I could just make a bash script, but was wondering if there was an easier way.
Kieran Klukas

A While deleting and recreating your user like this seems to have worked, there is a risk. Linux uses numeric user IDs (UIDs) to determine who owns what. **/etc/passwd** is used to map user names to UIDs. The first user usually has a UID of 1000, but if your replacement user had a different UID, say 1001 because the system didn't want to reuse an ID, you wouldn't have write access to your home directory or any of its contents, which can cause the desktop to fail to load.

The fix for the problem is obscure, but simple to implement. For some reason the account for your user has been incorrectly flagged as a system account, and those users aren't included on the login screen. Run this commend in a terminal:

```
$ sudo nano /var/lib/AccountsService/
users/USER
```

where `USER` is your user name. Change the setting for `SystemAccount` from true to false and press Ctrl+X to save and exit. Your user should appear in the login screen.

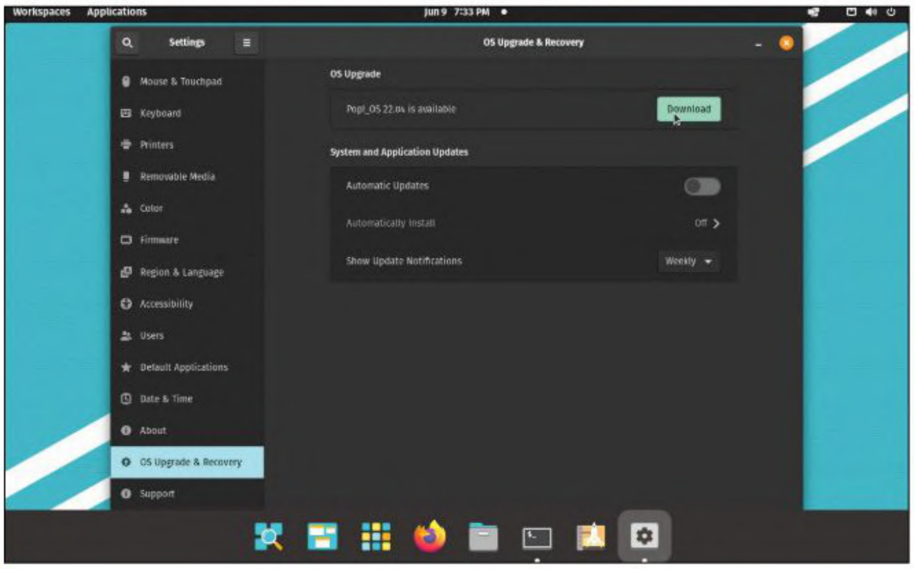
You can generate a list of all installed packages using:

```
$ dpkg --get-selections | cut -f1 >packages.
txt
```

The `dpkg` command lists information on all installed packages; `cut` extracts just the first item from each line, the package name. But this includes everything installed, including dependencies, so installing from this list would muck up the package manager's dependency tracking. Keeping a list that you could pass to `apt install` is one way to do it:

```
$ sudo apt install $(cat mypackages.txt)
```

A better approach is not to reinstall in the first place. Rather, you should use the option to update your distro to the latest version. In Pop!_OS, this can be found in the *Settings* program, under OS Upgrade and Recovery. Hit the Download button and leave it for a while. You should come back to the latest version of Pop!_OS with all your favourite applications still present.



OS Upgrade & Recovery

OS Upgrade

Pop!_OS 22.04 is available

System and Application Updates

Automatic Updates

Automatically Install: Off

Show Update Notifications: Weekly

Where a distro offers a path to upgrade to a new version, it's usually less work than reinstalling.

Q Poor network flow

I recently downloaded Jammy Jellyfish and am having trouble setting it up, particularly the Wi-Fi, I'm using an iOTA FLO laptop with 4GB RAM and no Ethernet port. The Wi-Fi network I'm trying to connect to is a Plusnet router with a dual-band Wi-Fi and four Ethernet ports. I've attempted to install drivers via USB and a secondary laptop, but to no avail, am I doomed to the financial woes of buying a USB to Ethernet cable, or is there a more cost-effective option?
Nicholas Hannah

A This author also bought an iOTA FLO recently. It's excellent value as a low-cost, portable machine in the tradition of netbooks. There was also an issue with wireless and there are two solutions. The Realtek wireless adaptor in this computer isn't supported by the Linux kernel, so you either need to install a driver or use a different adaptor. Installing a driver avoids the need for extra hardware, except that you need an internet connection to install it. If you want the easiest method, add a small USB wireless adaptor, which are relatively cheap. Our choice of an Edimax device worked out of the box. Once you have a working connection, you could install the driver with these terminal commands. You can't simply copy a driver to a USB stick on another computer as the driver has to be built to match your kernel.

```
$ sudo apt install build-essential
$ git clone https://github.com/lwfinger/
rtl8723du.git
$ cd rtl8723du
$ make
$ sudo modprobe 8723du
```

The first command installs the software required to compile source into programs and is needed only once. The second downloads the driver code, then the two `make` commands build and install the driver. Finally, you load the driver. Now your wireless connection should come up automatically whenever you boot the

»

computer. If it doesn't, you may need to tell the system to load the driver when it boots by creating the file **/etc/modules-load.d/rtl8723.conf** containing `8723du`.

Because the driver is built and installed for the current kernel, if you upgrade to a newer kernel you'll need to run the sequence of commands again to install the driver for the new kernel.

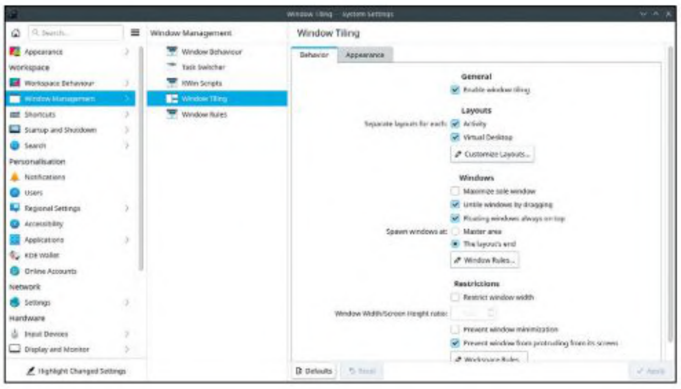
Q Type and browse together

I'm trying to take notes in the text editor *Kate* while reading something in the browser, but when I click *Kate* the browser disappears. I want them both active side by side at all times without either going out of view. I don't mean the browser is closed or exited. It's active, just not in view when I'm working in *Kate*.
Sean Hope

A This is down to how you manage window placement and focus on your desktop. Because you're using *Kate* as your editor, we'll assume you're using the KDE Plasma desktop. The default for *KWin* is to raise a window when you click it. This can be changed in the Focus tab of Window Management>Window Behaviour in Plasma's system Settings. If you set this appropriately and resize and place your browser and editor windows so that they are side by side, you should be able to do exactly as you ask.

It's possible to have a desktop where windows always appear alongside others rather than overlapping. This is called tiling and there are a number of window managers that work like this, such as *i3*. You can also do it with *KWin*, which gives the advantages of keeping the familiar software and being able to switch between tiling and conventional window management at the touch of a hotkey. One package that will do this is *Bismuth* (**https://bismuth-forge.github.io/bismuth**). You can install *Bismuth* from the package managers of most distros. The home page contains help if your distro doesn't offer it by default. Once you have installed it, you will see an extra option in the Window Management>Window Behaviour panel of Plasma's system Settings. Here you can enable tiling (it's turned off when you first install it), and set some of the parameters.

Tiling window managers always make window management from the keyboard a priority. While you have System Settings open, look in the hotkeys section where you'll find a host of shortcuts have been added for *Bismuth*. You can enable and disable tiling globally, which is useful if you only want it on for this one situation. You can also set keys to enable or disable it for individual windows as well as the keys to organise, move and resize windows, and those around them, while you're using the desktop. You may find yourself using tiling windows more and more and come to prefer it over the floating window approach.



General
[✓] Enable window tiling
Layouts
Separate layouts for each: [✓] Activity [✓] Virtual Desktop
Windows
[ ] Maximize sole window
[✓] Untile windows by dragging
[✓] Floating windows always on top
Spawn windows at: ○ Master area ● The layout's end
Restrictions
[ ] Restrict window width
Window Width/Screen Height ratio:
[ ] Prevent window minimization
[✓] Prevent window from protruding from its screen

The Bismuth extension brings tiling window management, either full or partial, to the plasma desktop.

Q Dolphin-friendly files

I can't get *Dolphin* to give previews of TIFF files, although I can see previews of almost every other graphic file that was ever invented. I've scoured the net – everyone has a fix, but none work. Surely, a sophisticated file manager like *Dolphin* is capable of previewing TIFF files without having to jump through hoops? *Nemo* has no such problems, but I would prefer to stick with *Dolphin*.
Sean Millar

A There seems to be two main causes of this issue with *Dolphin*: configuration or missing dependencies. Even compressed TIFF files are larger than other formats; the uncompressed ones even more so. *Dolphin* has a size limit for files from which it generates previews, so it may be that your TIFF files are too large for *Dolphin* to preview with its default settings. Being KDE, there's a setting for everything and you can change this limit in *Dolphin* by going to the Previews tab of Configure> Configure Dolphin and altering the setting for "Skip previews for local files above" to a suitably large setting, or to No Limit. You may also want to change the corresponding setting for remote files if you view TIFFs on network shares.

The other possibility is missing software. *Dolphin* uses a couple of packages to handle some file types, including TIFF. These are **kio-extras** and **qtimageformats**. The names may be slightly different, depending on your distro, but you should be able to find them in your package manager. You probably need to restart *Dolphin* after installing this, but a full KDE restart shouldn't be necessary.

Q ISO removal

I created an ISO file on a USB stick using *balenaEtcher*. I used it to boot from USB to install my Linux distro. I would now like to delete the ISO file. When I insert the USB stick, the OS doesn't see

» A QUICK REFERENCE TO: DOAS

Sudo is the way most distros enable you to run a program as the root, or any other, user. It works well and contains features that aren't needed by the home or small-scale user but are important for enterprise use. The downside of all this is that the configuration file is complex, with the syntax almost being a language in itself. If you just want to use the default setup of your distro, there's no major reason to change, but what if you want to alter some permissions without delving into the complexities of *sudo* and risking locking yourself out with a typo?

Doas is an alternative to *sudo*, developed on OpenBSD but available on Linux. It does everything most users of *sudo* want, but will far less complexity. The default Ubuntu *sudo* setup can be recreated with a one-line configuration file at **/etc/doas.conf**:

```
permit :admin
```

The colon indicates that this is a group name, otherwise this would be seen as a user. If you have a user you want to be able to use *doas* without giving their password, the setting is:

```
permit nopass fred
```

That's a little risky – what if you just want them to be able to reboot without a password? Use:

```
permit nopass fred /sbin/reboot
```

Unlike *sudo*, *doas* doesn't require full paths, but it's best to use them. You can also deny a user access to a command:

```
deny fred reboot
```

it. That is, it doesn't appear in the output of `df -k` or `mount`. How can I access and delete the ISO file?
Jasmine Patel

A An ISO file is an image of a filesystem for an optical disc (CD or DVD), and is effectively a whole disc in a file. When you write it to a USB stick with *Etcher* (or the old-fashioned way with *dd*) you overwrite the original partitions and partition table, and the stick now effectively boots like the original optical disc. The reason you can't mount it is that there's no disk filesystem to mount. If you now want to use the stick for something else, you need to create a new partition table. If you want to write another ISO file to it, just do it. That will overwrite the previous ISO image and you can now boot with the new distro or whatever was on the ISO.

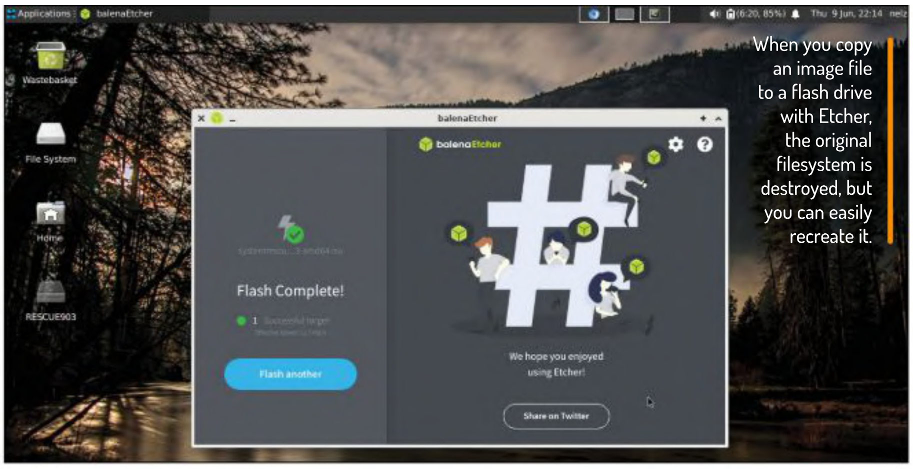
If you want to start using the disk for storage again, you need to create a partition table and partition, and then put a filesystem on it. You can do this from the command line or using a graphical program such as *GParted*. The command line approach uses *fdisk*. Run these commands, replacing `/dev/sdX` with whatever your USB stick appears as.

```
$ sudo fdisk /dev/sdX
```

Press o to create a new partition table followed by n to create a partition, accept all the defaults to make a single partition filling the drive. This creates a Linux partition. If you want a Windows type partition for cross-platform use, press t to change the type and c to set it to Windows. Finally press w to write the partition. Now you can create a filesystem on that partition with

```
$ sudo mkfs.vfat /dev/sdX1
```

Alternatively, run *GParted*, Select your device, go to Device>Create partition table to blank it. Now click New, change the File system to FAT32 but leave everything as is and press the Apply button. Now, whichever method you use, you'll have a USB stick that mounts and takes files just as it used to.



When you copy an image file to a flash drive with Etcher, the original filesystem is destroyed, but you can easily recreate it.

Q A sign of the times

I've used *cron* to run scheduled jobs for years, but now most distros use *Systemd* and that has its own timers. However, the man pages make it look complicated and far more technical that simply editing a crontab file. Can you give me some help here, or is there a way to keep using *cron*?
Oliver Dawson

A *Systemd* often has its own way of doing things, and its implementation of timers is no exception. However, the various *Systemd* components are consistent between themselves, so each time you get the hang of one component, the next one becomes easier to learn. *Systemd* timers are more complex that the old **crontab**, but that's partly because they have more capabilities and partly because they're different so you have to learn again.

If you don't want to go through a new learning curve, there is a shortcut. A package called **systemd-cron** (**https://github.com/systemd-cron/systemd-cron**) enables you to run timers based on your existing **crontab** files. This is more than a simple converter – it's a service that scans your *cron* files at startup to create timers, but also watches for *cron* changes to create new timers as needed. This is useful for the lazy among us, but also handy when installing software that comes with its own *cron* scripts.

If you want to manage your own timers – and you can do this alongside *systemd-cron* if you wish – the basics are that each job requires two *Systemd* units: a timer to kick off the event and a service file that specifies the action to take. Normally, there have the same name but with a different suffix, so you could have **/etc/systemd/system/foo.timer** containing:

```
[Unit]
Description=what it does and when
[Timer]
OnCalendar=10:30
```

to run the timer at 10:30 every day. Times are specified as HH:MM or HH:MM:SS, days and dates can be added, too. See the **systemd.time** man page for full details of the *OnCalendar* time specifications. There are also shortcuts of hourly, daily, weekly and so on. As well as setting absolute times with *OnCalendar*, you can also set timers to run on boot, or an interval afterwards, and other options. With all of these options, how can you be sure that your timer will run when you want it to? The `systemd-analyze` command can be used, among other things, to evaluate timer expressions.

So typing the following

```
$ systemd-analyze calendar --iterations=6 '00:00/10'
```

will show you the next six occurrences for the given expression, which in this case triggers every 10 minutes.

Each timer needs an associated service to run when it triggers. This is normally named the same as the timer, just a different extension, but you can run a different service with the `Unit=` option. A service file can be as simple as:

```
[Unit]
Description=service for foo.timer
[Service]
ExecStart=/usr/bin/foo
```

So that's four lines in two files (excluding the descriptions) to do the same as

```
30 10 * * * root /usr/bin/foo
```

We never said *Systemd* timers were more compact! 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
```

Subscriptions: for magazine issues email **help@mymagazine.co.uk**

Mailserver

WRITE TO US

Do you have a burning Linux-related issue that you want to discuss? Write to us at *Linux Format*, Future Publishing, Quay House, The Ambury, Bath, BA1 1UA or email **lxf.letters@futurenet.com**.

Low profile latency

I recently noticed that, when upgrading my Ubuntu machine with the latest Linux kernel, you no longer have the low-latency option. I believe kernel 5.16.11 is the last kernel where you got to choose between the generic and low-latency kernel (see https://kernel.ubuntu.com/~kernel-ppa/mainline/v5.16.11). I can't find anything about this – do you know why this option disappeared?
Koen Rousseau

Neil says...
Oddly, back when you emailed, this blog on industrial use for low-latency kernel does suggest Jammy 22.04 support: **https://ubuntu.com/blog/industrial-embedded-systems-iii**.

Then at the start of June this blog appeared saying the RT kernel is in Beta, so perhaps this is the issue? See **https://canonical.com/blog/real-time-linux-qa**.

Software discovery

I love Linux. It's open, free and fast. I've converted several systems (at home and at our football club) to Linux. But my biggest gripe with Linux is software. There's so much available, so how do you decide what to choose? And this is even with just the software in the repositories. How about all the software that's available outside the repositories? What software is available, how reliable or functional it is? Not to mention how many users are out there? Is there a solution? Do you have tips?

Some time ago I stumbled over a web application to share passwords. It works like WeTransfer instead of a file where you share a password, all encrypted of course. I did write it down, but now I can't find it. That open source project is doomed to stay a hidden gem, unless someone can point me in the right direction?
Stephan Kamphuis

Neil says...
Let's be honest – finding good software issues aren't just restricted to Linux, I think any platform has this problem. In a way Android and iOS are in an even worse position because there are extra issues of freemium, ad-filled and malware options littering both ecosystems. User reviews are supposed to help, but they can be subverted by people hell-bent on leaving protest votes, with not enough genuine users leaving feedback.

That's partly what the *App Store* concept was supposed to fix: offer curated, peer-reviewed software. The addition of Snap and Flatpak is even supposed to ensure easier instals, too. One of our favourite tools is **https://alternativeto.net**. It's not perfect because many suggestions can be unsupported, but it's another method for discovering suitable software.

As for a password transfer service, how are you supposed to trust any of these online services? We did spot **https://bitwarden.com/products/send** and at least Bitwarden is a known service.

Is there a low-latency kernel for the Jellyfish?

Retro Deck

Since the Steam Deck runs Linux, how would one add a retro gaming experience akin to Retro Pi?
Chris Williams

Neil says...
This is a great idea and the short answer is it all just works out of the box. The slightly longer answer is if you go to the Steam Deck desktop and its software store *Discover* it has all the retro emulators you could want including *RetroArch*. You then add any ROMs to an SD card and to make things slick add the new emulators to the interface via the desktop *Steam* client and its 'Add a non Steam game' option, and away you go. There's the Steam Deck ROM manager, which enables you to better integrate these "non Steam" emulators and even individual ROMs.

Helpdex

LES POUNDER: AS A REWARD FOR YOUR GREAT WORK IN YESTERDAY'S HACKATHON YOU'RE DIRECTED TO PLAY THE FULL VERSION OF INDIANAPOLIS 500
DRIVING AN OLD EA VIDEO GAME FOR FIVE HOURS IS A REWARD ?!?
IT'S A REWARD, WHEN THE PUNISHMENT FOR LAST PLACE IS TO BE DIRECTED TO SPEND OVER EIGHT HOURS PLAYING "DESERT BUS"
Shane

It turns out that the Steam Deck is an excellent retro platform.

Docked incantations

I've been a subscriber of *Linux Format* for at least 10 years and love the magazine. In all that time one thing really stands out: the simplicity and speed with which a docker container can be set up.

With the passage of time I thought I really should update my containers. This is where I discovered updating has far more pitfalls than the initial setup and surprisingly my faithful TimeShift didn't save my first attempt to update *Mealie*. Long story short I've subsequently learnt that there's usually a precise process to follow to back up data and databases along with various `docker exec...`, `php artisan...`, `mysqldump` and `mysql` incantations at the command line. Who knew you can execute bash commands inside a container? I certainly didn't.

I've now updated *Pihole*, *Mealie* and *BookStack*. Each one was different so I recommend researching the update process first before diving in. I think an article about updating docker containers would be very interesting. While I've learnt quite a bit about my containers, I don't know what else I don't know!
John Parr

Neil says...

Well effectively each container is a specialised Linux distro without the kernel. It might only need the minimum of software for the target to run but if it's based on an existing distro that'll still (likely) include *bash*, although a Docker container doesn't have to, it's just an example. For updates, the usual approach is stop the container, pull the update and restore your data if needed, though each can be different so read the docs! But certainly it's something of a game changer.

LETTER OF THE MONTH

Amiberry corrections

As the author of *Amiberry*, I noticed a few inaccuracies in your LXF290 review, that I'd love to help you address, if you're interested.

For example, *Amiberry* doesn't require the FKMS driver – it works just fine with the default KMS one as well. For the Raspberry Pi platform specifically, *Amiberry* comes in separate versions: pure SDL2 (using whatever back-end is available, like X11 or KMSDRM) or Dispmanx (which is considered obsolete, requires FKMS, lacks some features but still offers decent speed). It's only if you want to use the Dispmanx version, that you'll need the FKMS driver enabled – and that's a limitation of Dispmanx itself, not *Amiberry* (it applies to all applications using that framework).

Another detail you may not know about, is that the A500 Mini also uses a (modified, older) version of *Amiberry* as well.

And there are lots of features that are unique to *Amiberry*, such as the special WHDBooter – designed to make it possible for you boot WHDLoad games directly from the .lha compressed archive and choosing the best-known settings for the emulator automatically for you.

There's a lot to cover, and I'm sure you don't have room for all of it, but if you are interested, let me know.
Dimitris "MiDWaN" Panokostas

Neil says...

We've forced Les Pounder to replay a full race of *Indianapolis 500* until he's learnt his lesson. Also, let's feature your project in this very issue!

We're not sure this is a punishment or a pleasure.
CREDIT: Electronic Arts

ON A SCALE OF 00 TO 10, HOW GOOD IS YOUR KNOWLEDGE OF BINARY?
01
Shane
shane_collinge@yahoo.com

SUBSCRIBE

to *Linux Format* and get your

STM MYTH Laptop Bag

Protect your tech while on the move with the extraordinarily spacious STM Myth Laptop Bag!

Product features

- Durable and water repellent
- CableReady design enables charging on the go with hidden cables
- Removable, adjustable shoulder straps with non-slip pads for extra comfort
- Impact protection isolates your laptop from the 'bump zone'
- Smart pockets, with a portable power pocket and a zippered section for your laptop and tablet for easy access
- Large 28-litre capacity – fits most 15-inch laptops

SUBSCRIBE NOW!

www.magazinesdirect.com/lin/A68F

Call **0330 333 1113** and quote **A68F**

» **PLUS:** Exclusive access[1] 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**

1) Only available to www.magazinesdirect.com subscribers

OUTSIDE THE UK? Turn to page 65 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 30 September 2022. Offer open to new UK subscribers only. Pricing is guaranteed for the first 12 months and we will notify you in advance of any price changes. Please allow up to six weeks for delivery of your first subscription issue (up to eight weeks overseas). Your gift will be delivered separately within 60 days after your first payment has cleared. Gifts only available to subscribers on the UK mainland. Gift not available with a digital subscription. The full subscription rate is for 12 months (13 issues) and includes postage and packaging. If the magazine ordered changes frequency per annum, we will honour the number of issues paid for, not the term of the subscription. For full terms and conditions, visit www.magazinesdirect.com/terms. For enquiries please call **+44 (0) 330 333 1113**. Lines are open Monday-Friday 9am-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.

Backblaze B2

A simple pricing model makes it easy to migrate to, says **Jonas DeMuro**.

IN BRIEF

A well-polished product in Amazon's S3 (Simple Storage Service) space. We appreciate that the first 10GB of storage is free, and that the storage costs undercut many competitors. The good documentation is a plus; however, we have concerns that there's no direct phone support, and the hours are limited for a online chat interaction.

Backblaze, which started in a small Palo Alto apartment, has seven co-founders that have worked together since 2007, first founding their cloud backup offering, and then in 2015 its B2 Cloud Storage offering. Because it's an Infrastructure as a Service (IaaS), this latter product targets software integration for the business market.

Backblaze B2 has a simple pricing model. While there's no free trial period, the offer of 10GB of free storage without a time limit is arguably better. This enables users with more basic needs to access to the service and try it out before deciding to fully commit.

Beyond that, the cost is a simple $0.005 GB/month (Amazon S3 is $0.021) for data stored, which makes us wonder why others in the space don't go with such a simple route for pricing. There's no cost for uploads, while the download cost of $0.01 GB/month is significantly less than other major services (Amazon S3's cost is $0.05+). As an inducement to go with them, Backblaze B2 covers the transfer and any legacy egress fees for migrations that are over 10TB.

The Backblaze team states its service can handle "millions of objects and petabytes of data." It's designed for ease of setup and to be up and running in minutes. It's also well suited for media because the files are always accessible, and not stored on tape or another offline solution, with the inherent delays of accessing the data. Backblaze B2 is also designed to be "workflow friendly" because it can easily connect to NAS, SAN and MAM configurations.

Choice of migration routes

Migrations to Backblaze B2 can be performed through a variety of different methods. It can be done via cloud to cloud, with movement of the data from your existing cloud service – for example, Google Drive or Amazon S3, and then written to the Backblaze B2 server. Two further options are the on-premises to cloud solution, in which the data is moved from a server, NAS or SAN; and data migration through either "optimised cloud pipes" or "Backblaze's 96TB Fireball rapid ingest device." The latter option is from tape media, either from a reel cassette or a cartridge, with the data going through a high-speed direct connection.

At the heart of this service is the IaaS which is S3-compatible and native APIs, SDKs, and CLI, and is also designed to be able to handle developer workloads. It also makes use of multiple security features including SSE and CORS rules.

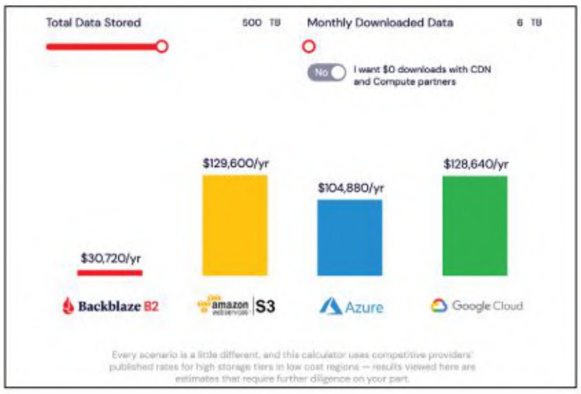
Support for Backblaze B2 is available via a number of methods, but there's no phone support. We consider this a shortcoming: there's a phone number on the website but it specifically designates as "Not for customer support." Rather, the support is available via email, or via a support portal that enables the user to attach a file such as an annotated screenshot. Elsewhere it states that users will receive a response in a business day through the portal. There's also a help page, with a long list of articles grouped by category.

Online chat is also a contact option, but available hours aren't obvious. It does state 9am to 5pm PST Monday to Friday, but it's closed from 12pm to 1.30pm on Mondays, and from 12pm to 1pm on other days. There are no hours listed for the weekends and chat is completely closed on alternate Wednesdays! LXF

Simple, Scalable Cloud Storage Services

Grow on object storage made for building applications, safeguarding data, and doing more with ease.

This is how we think the interwebs stores it.



Every scenario is a little different, and this calculator uses competitive providers' published rates for high storage tiers in low cost regions – results viewed here are estimates that require further diligence on your part.

Backblaze has a fun storage cost calculator that neatly highlights competitor prices.

VERDICT

DEVELOPER: Backblaze
WEB: www.backblaze.com/b2/cloud-storage.html
PRICE: $0.005 per GB (first 10GB free)

FEATURES	7/10	EASE OF USE	7/10
PERFORMANCE	9/10	VALUE	9/10

Backblaze B2 is an attractive option for scalable cloud storage services with a focus on the business market.

» **Rating 7/10**

Intel Core i3-12100

A lowly quad-core serves up big performance gains, finds **Paul Alcorn**.

SPECS

Socket: Socket V LGA1700
Process: Intel 7 10nm
Cores (P+E): 4+0
Threads: Eight
P-cache: 5MB L2, 12MB L3
P-core: 3.3GHz (4.4GHz boost)
Unlocked: No
GPU: UHD Graphics 730, 24 EU, 300MHz, OpenGL 4.5, OpenCL 2.1
Mem max: 128GB, No ECC, two-channel
Mem speed: DDR5 4800MT/s, DDR4 3200MT/s
PCIe: v5 16-lanes, v4 four-lanes
Peak Base Power: 60W
Max. Turbo Power: 89W

Intel's four-core eight-thread Core i3-12100 comes with a highly competitive £129 price tag that easily helps us brand this as the best budget CPU of 2022 so far. Not to mention that the chip also comes in the option of a £99 F-series with deactivated graphics. Ever since AMD's Ryzen 3300X arrived in 2020 the sub-£200 processor market has been left lacklustre at best. AMD's cheapest Zen-3 CPU is the £230 Ryzen 5600G and at almost twice the price certainly struggles to fend off this Intel pocket rocket.

Alder Lake's performance advantages come even ignoring its support for DDR5 memory and PCIe 5.0 interfaces. As such, you can use standard DDR4 memory and PCIe 4.0 devices and still have superior performance and connectivity options.

This Core i3-12100 doesn't have the Alder Lake hybrid architecture; it only has four Golden Cove P-Cores active. This means you don't have to worry if your kernel has the latest Intel Thread Director support in place for optimal performance.

All Alder Lake chips support DDR4-3200 or up to DDR5-4800 memory, but caveats apply. PCIe support will vary by motherboard, but Alder Lake chips expose up to 16 lanes of PCIe 5.0 (technically for storage and graphics only, no networking devices) and an additional four lanes of PCIe 4.0 from the chip for M.2 storage. Intel's Alder Lake drops into Socket 1700 boards from the 600-series, including Z690, H670, B660, and H610.

Power and cooling

The Core i3-12100 comes with a 60W PBP (base) and 89W MTP (peak) power rating. The chip clocks in with a 3.3GHz base and boosts up to 4.3GHz. It also comes with 12MB of L3 cache. The Core i3-12100 isn't overclockable, though. Intel has revamped its stock air coolers with Alder Lake. These coolers are designed to address two major deficiencies with Intel's stock coolers: thermal dissipation limitations and aesthetics.

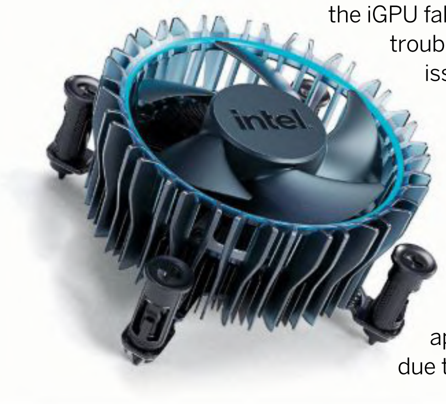
The standard Core i3-12100 comes with the UHD Graphics 730 engine with 24 EUs. The engine runs at 300/1,400MHz base/boost frequencies. Other than a lower price, going with the 12100F means you'll lose Quick Sync (hardware video encoding) capabilities and the iGPU fallback that you can use for troubleshooting in the event of an issue with a discrete GPU.

The Core i3-12100 is surprisingly agile in our cumulative measure of single-threaded performance. The 12100 is 27 per cent faster than the Core i5-10100. Compared to Ryzen, the 12100 dominates in single-threaded applications, which was expected due to the higher performance of the Alder Lake P-core. This saw its lead stretching between 25 per cent over the Ryzen 5 3600 to 11 per cent over the Ryzen 5 5600G.

Moving to multithreaded performance, the Core i3-12100 continues to assert its dominance over comparably priced chips with an 18 per cent lead over the Ryzen 3 3300X and a 30 per cent lead over the Core i3-10100. However, the 12100 isn't as impressive in multi-threaded work against the six-core chips. The Ryzen 5 3600 and 3600X lead by around 11 per cent, while the 5600G leads by 19 per cent.

Overall, the Core i3-12100 offers a solid blend of performance in both single- and multi-threaded applications given its price point, but its single-threaded performance stands out as exceptional. The 12100 also dispatches the Ryzen 3 3300X and Core i3-10100 easily in multi-threaded work. With better priced compatible boards now available (especially those supporting cheaper DDR4 for the Intel Alder Lake platform), coupled with this Core i3 12100, Intel has a budget-end winning gaming and application option on offer. LXF

Alder Lake offer amazing performance, but scary power draw.



The funky new Intel fans will blow you away <zing>!

VERDICT

DEVELOPER: Intel
WEB: www.intel.com
PRICE: £129

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

The quad-core Intel Core i3-12100 is the fastest budget gaming CPU on the market. Its price, performance and improved stock cooler dominates its price range.

» **Rating 9/10**

GeckoLinux Static 153

Michael Reed takes a close look at GeckoLinux, a distro with a wide variety of editions that puts a friendly face on openSUSE.

IN BRIEF

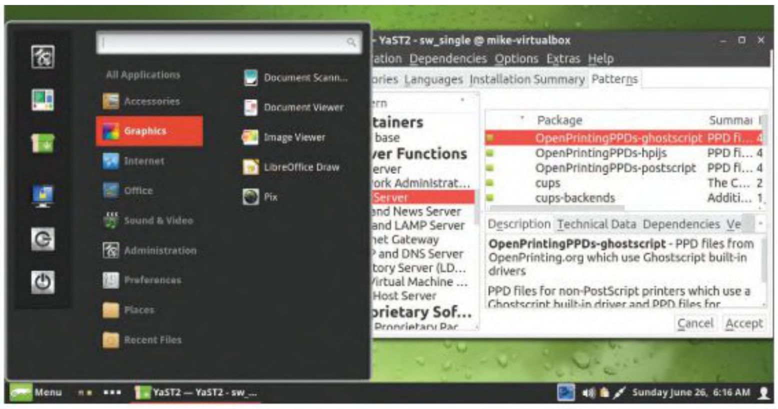
GeckoLinux sits on top of openSUSE Linux. This gives it a strong underpinning in terms of administrative tools and reliability along with extensive documentation. The variety of editions is a boon. It's preconfigured and set up to be easy to install and use. Maybe it can tempt some of the Ubuntu crowd over to the world of SUSE?

SPECS

CPU: 64-bit
Memory: 2GB
Builds: 64-bit only
Static: yearly cycle (six desktops)
Rolling: rolling release (nine desktops)

GeckoLinux is based on openSUSE Linux, and it offers a good selection of editions. There are six 'Static' editions with different desktop environments based on the current 'Leap' edition of openSUSE, which has a roughly yearly update cycle. There are also nine different editions that are tied to the Tumbleweed openSUSE rolling release. We're putting the Cinnamon desktop equipped Static (153.220104) edition through its paces.

The installation procedure is fairly standard, but Btrfs is the default choice of filing system. Btrfs is a good choice for a modern desktop, because it adds facilities such as snapshotting. On GeckoLinux, snapshots are created on each boot and whenever the package management facilities are employed. In use, we found that the snapshotting was well-integrated into the system as a whole, with the *Snapper* GUI for browsing and reverting snapshots. The only downsides to Btrfs on the desktop are that it can use extra CPU resources and it gobbles up extra disk space for the snapshots. From the outset, it's obvious that this distro is optimised for use on a reasonably powerful, newer machine.



GeckoLinux has impeccable and consistent looks, and features the impressive administration tools of openSUSE. It's not suitable for those new to Linux, though.

Well-balanced visuals

First-boot impressions were good. The font set has a grey tinge to it that we thought sat very well. Efforts has been expended on making sure that the entire system is well-balanced visually, with above average font continuity between all of the different areas and graphical toolkits that pervade a Linux desktop.

In the Cinnamon edition, the applications straddle the space between Gnome stalwarts such as *Firefox* and *LibreOffice* and slightly off-brand (and better, in our opinion) choices such as the *Nemo* fork of the Gnome file manager and the wonderful *Clementine* music manager and player. These applications are launchable from the bottom bar launcher, which is categorised, searchable and launchable via the super key for hands-free launching. It's unoriginal, but it works perfectly.

Once initial exploration of the desktop is over, the next stage is to add applications. As you might expect, the tools for doing this are appropriate to the world of openSUSE. Overall, if you're used to Debian-based distros such as Ubuntu, you'll have to learn some new conventions and commands, but having done this, you're likely to come away impressed. *Zypper* is the command line tool, and it has the advantage over *apt* that all of the functions, such as installing packages, updating the system and searching for packages, are contained in one tool. Flathub and Snap installation channels are absent by default, but they're fairly easy to add to the system.

YaST2 is the GUI package management front end, and we liked what we saw here too. 'Patterns' are a bit like the metapackages of Debian, but you can browse them, and pick and choose from the suggested packages. The downside is that there doesn't seem to be a shop application at all.

GeckoLinux is openSUSE that's been configured and polished to a tee. However, it's for people who know their way around Linux rather than being a beginner's distro – power users if you will. It strikes a balance between offering a preconfigured experience and not breaking compatibility with the parent distro. We solved any problems that cropped up by searching for openSUSE solutions, which worked as expected. The variety of editions are the icing on the cake for a distro aimed at existing Linux users who want a slick experience from the start while gaining access to SUSE's awesome admin tools and facilities. LXF

VERDICT

DEVELOPER: n/a
WEB: https://github.com/geckolinux
LICENCE: Various

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

GeckoLinux harnesses the power of openSUSE with some of the rough edges removed. It has a useful set of desktop environment and release cycle variants.

» **Rating 8/10**

MakuluLinux Shift

Michael Reed is wowed by the visual bling of MakuluLinux, a distro that claims to be 'game ready' thanks to its many pre-installed components.

IN BRIEF

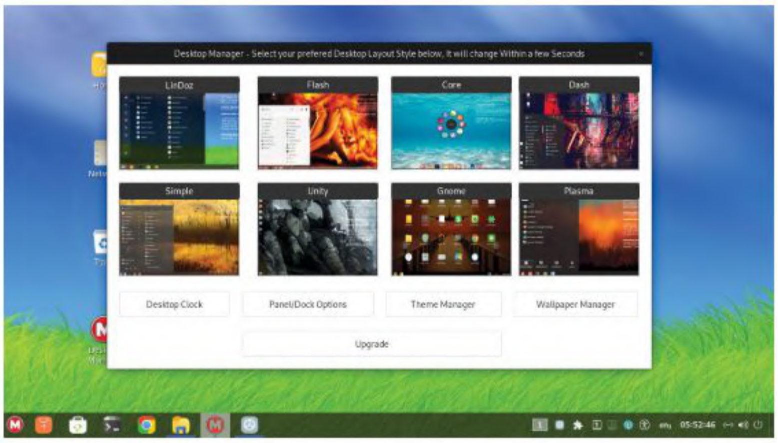
MakuluLinux has a bold and brassy appearance. It has a fast-change theme selection tool that can greatly alter the layout, all the way from a modern icon bar dock to something more conventional. Gamers will appreciate the built-in media and gaming support. Underneath it all, it's based on Ubuntu and Debian, giving it a good foundation.

SPECS

CPU: 64 bit only
Memory: 2GB RAM (minimum)
Builds:
Shift – the standard build
Shift Pro – a premium (paid for) version
Droid – with Android app compatibility

MakuluLinux Shift edition uses parts of Ubuntu and Debian Testing as its base, and it's a rolling release. Expect a bit of extra visual glamour at every stage of its use, and this includes booting into the live ISO, where you're greeted with an animation sequence (complete with musical accompaniment) that sums up the main features of the distribution.

Following this, we completed the fairly standard installation and rebooted. On first boot, we turned our attention to one of the major features of MakuluLinux: *Desktop Manager*. This tool has the ability to make radical changes to the desktop layout in semi-real-time, even though the desktop environment is actually Gnome underneath. These layouts range from conventional Gnome to 'Core', a desktop that has an icon dock at the bottom of the screen. 'LinDoz' has a layout that looks and functions suspiciously like Windows 10. Fortunately, the themes that feature the dreaded animated mouse cursor can be tamed with the Desktop Manager's sub-options.



It's actually Gnome underpinning everything, but the Desktop Manager can quickly swap desktop layouts, from the exotic to the conventional.

Behind the scenes, *Desktop Manager* is installing and removing themes, then reconfiguring the layout when you make a selection. This takes several seconds without having to reboot or reload running applications. As impressive as this feature is, we'd only expect it to be used around the time that the distro is first installed. However, it would be nice if this commitment-free flexibility was a more common feature in other distros.

Paying for extras?

It's at the desktop customisation stage that we first began to notice the reminders that a paid-for version exists, and these are sprinkled around fairly liberally. There's nothing wrong with this model – offering extra features and layouts in a premium version. However, users should be aware that features have occasionally been moved from the free edition to the paid-for edition. Beyond this, we'd rate the free edition of MakuLinux as fully featured, and some users will appreciate the chance to contribute financially to the project.

MakuluLinux calls itself 'game ready'. To this end, *Play On Linux* and *WINE* are preinstalled, meaning that a fair proportion of Microsoft software, including some games, can be run from the beginning. It's a simple matter to install *Steam*. Speaking of installation matters, as well as the Ubuntu repository, Snap and Flathub channels are present and configured from the get-go.

The basic application load-out is on the light end of what we'd expect from a general-purpose desktop distribution, but it's peppered with extras that would appeal to users with an entertainment and gaming bent. *DroidCam* handles webcam usage and *Discord* provides gaming-orientated text chat, for example. *Google Chrome*, rather than the open source *Chromium* browser, is installed, probably to give the best compatibility with DRM-infested streaming content. As well as the distribution being game ready, it's also media ready with most drivers and codecs that you're likely to need coming preinstalled.

Makulu Linux is a visually ballsy, fully featured distro. In the first week or so of using a new distribution, most of us experiment with different layouts, and it's nice that MakuluLinux gives you the freedom to experiment without committing to any particular style of desktop. Users coming from non-Linux systems for the first time will appreciate that nearly all of the fiddly proprietary components are preinstalled and preconfigured from the outset. This might be a good example of a distribution to hand to such a switcher, but it's not aimed at the Linux expert who likes to configure everything themselves. LXF

VERDICT

DEVELOPER: n/a
WEB: www.makululinux.com
LICENCE: Various

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

MakuluLinux is ready to go and comes with impressive visuals. Maybe one to pass on to a Windows-using friend?

» **Rating 8/10**

EasyOS 4.2

Michael Reed tries a Linux distribution that's suitable for extremely limited hardware, and finds that it breaks with conventions in some areas.

IN BRIEF

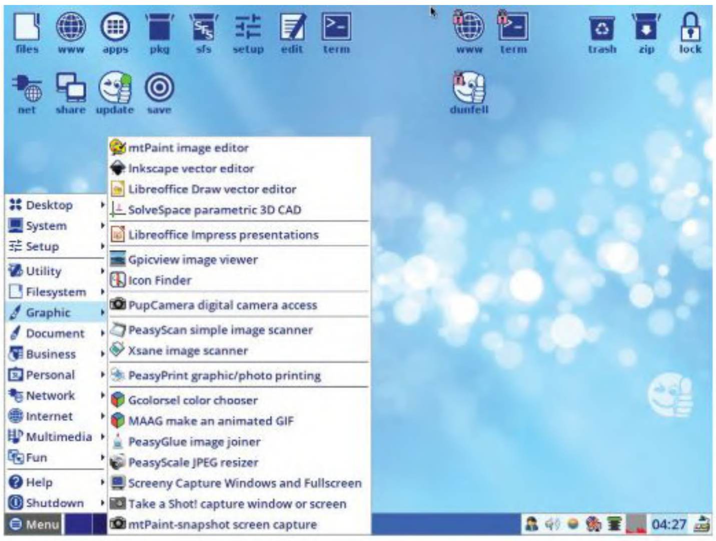
EasyOS is heavily influenced by PuppyOS, another lightweight distribution with which it shares much technology. It's intended to provide full desktop computer facilities on limited hardware. It's certainly an eccentric distribution in terms of how it works. And yet, if you can get past these eccentricities, it's a highly viable option for reviving older systems or as an emergency portable setup on a flash drive.

SPECS

CPU: 64-bit processor of around 1GHz
Memory: 1GB of RAM

EasyOS is a lightweight distribution that can provide a useful Linux desktop on PCs with limited resources. As well as being useful in that regard, its inner workings are, in many areas, different to the majority of Linux distributions.

This bucking of convention begins with the install procedure, in that there isn't one. EasyOS follows the trend of many Raspberry Pi operating systems: the user copies a complete disk image directly to the boot medium, and it uses about 1GB of space. For example, the official documentation encourages an approach in which EasyOS is copied to a flash drive and used from there rather than the hard disk of the target machine. Multiboot installation to a hard drive is possible, but it's a little bit more complicated to arrange – a downside of having no installer.



The EasyOS interface is clean and fast. Once installed, it should be approachable by nearly all computer users.

On first boot, the user is asked a series of questions to set up things like the keyboard language, boot password and the screen mode. We'd estimate that someone unfamiliar with Linux might need some help during the setup procedure because it's a bit rough and ready, cosmetically speaking. By the same token, the documentation, via the website lacks a professional look, but you can usually dig down to what you need

Whereas the bare-bones nature of the setup procedure might appear intimidating compared to the polish of something like Linux Mint or Ubuntu, we'd say that anyone who can use a computer would be able to figure out how to use EasyOS once it's installed and running. Applications are loaded with a single click on a backdrop icon or from the application launcher, which is located in the usual place. The theme throughout is a flat, monochrome one, and what it lacks in beauty it makes up for with clarity. In addition, to its credit the *JWM-ROX* window manager is lightning fast.

Surprisingly, standard applications such as *Firefox* and *LibreOffice* are installed by default. These relative heavyweights sit alongside a wide selection of lighter tools including a text editor (*Geany*) and file manager (*ROX Filer*) and others that cover nearly every area of standard computer use.

At the root of it all

What's going on under the hood is, as with everything else on this operating system, a bit unusual. There's a package manager, but it doesn't install the SFS format packages; they are downloaded and then mounted within the file system. The user is never prompted for passwords thanks to a fairly unique philosophy for a Linux distro: the user is root at all times, and some applications (such as web browsers) are instead run with limited privileges.

It's certainly an efficient OS. To put things into context, the lightweight Raspberry Pi OS on a Raspberry Pi 400 consumes only 226MB of RAM, once freshly booted. EasyOS beats that on a standard PC and consumes only 118MB of RAM! Things were a bit closer to being equal once we loaded *Firefox* on each machine as memory use leapt up to 457MB compared to 485MB for Raspberry Pi OS. You could easily take an old PC or laptop with only 1GB of RAM and turn it into a usable web browsing or word-processing station. Sure enough, when we booted it on an old notebook PC, launching and running programs worked at an impressive speed, and all hardware was picked up without any problems.

EasyOS is an unusual distribution, but it could be used to turn a pile of abandoned old PCs into a room full of light-duty office workstations. LXF

VERDICT

DEVELOPER: n/a
WEB: https://easyos.org
LICENCE: Various

FEATURES	10/10	EASE OF USE	7/10
PERFORMANCE	10/10	DOCUMENTATION	6/10

EasyOS is, ahem, easy to use once it's installed and set up. A powerful light system that can be booted from a flash drive.

» **Rating 8/10**

Inscryption

We're not trapped in here with The Management, they're trapped in here with **Jody Macgregor**. And that's just the way we like it.

SPECS

Minimum
OS: Ubuntu 18.04
CPU: Intel Core i5 760, AMD Athlon II X4 645
Memory: 4GB
HDD: 3GB
GPU: GeForceGTX 550 Ti, 1GB Vram, Radeon HD 6850 1GB Vram

Recommended
CPU: Intel Core i5 3470, AMD FX 4350
Memory: 8GB
GPU: GeForce GTX 1050 2GB Vram, Radeon RX 460 4GB Vram

Inscryption turns board game night into a nightmare, trapping you in a spooky cabin where you're forced to play a deckbuilder. Its villain is a shadowy figure who patiently waits at the table for the game to start. Though you have to play, under pain of death, you can also get up and stretch your legs. There are shelves of trinkets on the walls, a skull, a safe, a cuckoo clock. They're puzzles to solve as part of the larger puzzle: How do you get out of this cabin?

At *Inscryption's* best, the card game and environment work in harmony. The solution to a puzzle in the cabin is hidden in the smudgy grimoire that explains the card game's rules, and the reward for solving it is a card for your deck. The two progress in parallel. When we were blocked in one because we needed to explore the other for a while, though, that wasn't so fun.

The game within the game may be a deckbuilder, but it's less *Slay the Spire* and more like the single-player mode of a collectible card game. You collect a deck of animal friends, bullfrogs and wolf cubs and stoats with familiar stats and thematic abilities. The skunk smells so bad enemy cards attacking it have their power reduced; the beaver builds defensive dams.

The mechanics of this card game are off-kilter and grotesque. Most cards have a blood cost, which has to be paid by sacrificing other cards. Rather than being measured in hit points, damage against you and your opponent is counted in teeth, which fall into a pair of scales. You need to get ahead by five teeth to win a round. There's an item sitting on the table that promises to help you tip the scales. It's a pair of pliers.

Staring down your opponent

Your opponent isn't simulating another player and doesn't play by the same rules. He's more like a dungeon master, narrating encounters and putting on voices and masks to portray NPCs as you cross a map. Then he takes off the mask and sets up some minis around a campfire to play out a scene with suspicious, starving travellers who offer to warm one of your beasts by their fire. He may be a murderous kidnapper, but he puts in so much effort you have to respect him for that.

Why, if it isn't our friendly glowing eyed game master.

Like most deckbuilders, *Inscryption* is run-based. Losing means starting over, though you get to design one new card to be found in future runs, and they can be overpowered if you do it right. Plus, any solved puzzles in the cabin remain solved. Initial defeats still felt like progress, but after the first successful run you still have to solve another puzzle and then repeat the victory, and that was when it became a drag. The multi-stage boss battles are initially full of surprises, but become more chore-like each time they're repeated. The lovely creepiness of a game in which cards beg not to be sacrificed and fungal doctors offer to saw them apart and stitch them together was eroded by repetition.

Those layers can't be discussed without spoiling them, but if you don't mind that, here you go. *Inscryption's* second act is a 2D pixel-art RPG in the style of the Pokémon trading card game, but trading horror for whimsy. Although there's an automate option it doesn't create competitive decks, and we found manual deck construction a chore. Get through this and there's a third and final act, which returns to something more like the first only with a sci-fi theme, but *Inscryption* lost us before that. We preferred being trapped in a spooky cabin to being trapped in a succession of less-interesting video games. LXF

It's not just a deck building game – you'll find a game within this game.

VERDICT

DEVELOPER: Daniel Mullins Games
WEB: www.inscryption.com
PRICE: £16.79

GAMEPLAY	8/10	LONGEVITY	6/10
GRAPHICS	8/10	VALUE	7/10

It's at its best when it's at its strangest, but Inscryption doesn't know when to hold 'em and when to fold 'em.

» **Rating 7/10**

WE COMPARE TONS OF STUFF SO YOU DON'T HAVE TO.

Roundup

OpenBoard 1.6.1 » Xournal++ 1.1.1 » Lorien 0.4 » Rainbow Board 0.8.1 » Rnote 0.5.3

Mats Tage Axelsson is constantly looking for better ways to do things, including all things open source.

Software whiteboards

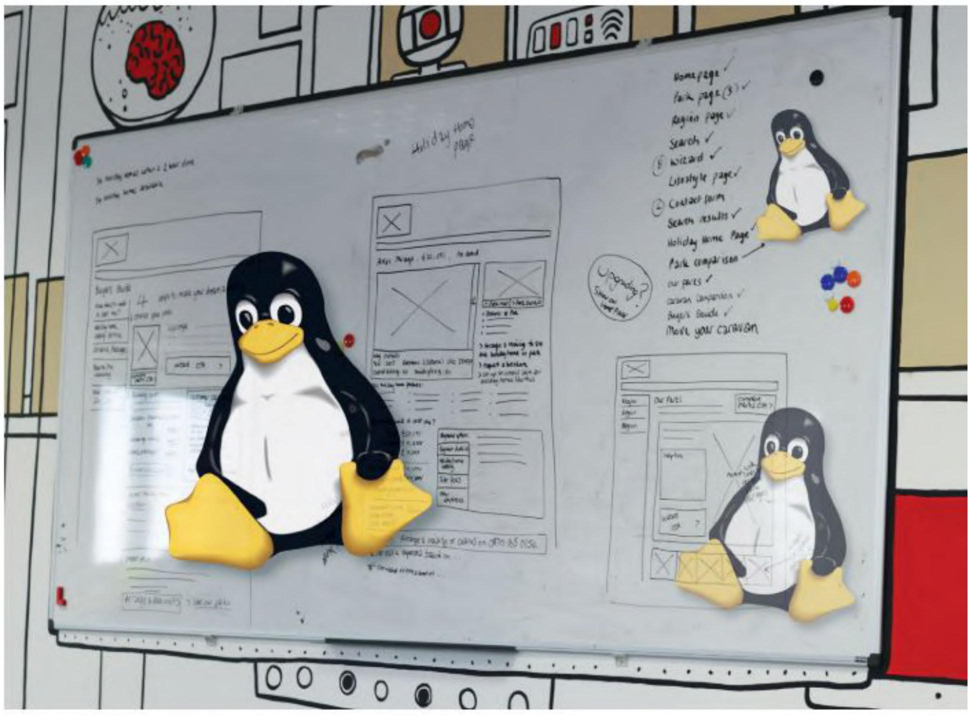
If you're starting a project you'll need to take notes, possibly for presentation down the line. **Mats Tage Axelsson** helps you to organise your thoughts.

HOW WE TESTED...

In this *Roundup* we used the programs on a Linux laptop in conjunction with a mouse pointer. We also used an external pen device to draw figures. We only used modules that were supplied with the programs by default, we didn't try to create any modules – this is something that's better suited for a future JavaScript programming tutorial.

We installed the programs using the most popular packaging method and also examined the repositories to see how difficult it is to compile them yourself. There's the option to install from source in all cases, but the most stable method will be to use whatever the developers release. These programs all come as Flatpaks bar *Lorien*, and we installed them using Flatpak, AppImage or the Debian installation process, according to the developer's recommendation.

Considering the small size and complexity of these programs, you don't need to choose one according to their performance. It's all down to personal preference.



Software whiteboards can be used for many tasks. At their most basic they can record your notes, which you create using a pen or simple pointer. As you become more confident in the software you can use them as presentation tools, enabling colleagues to understand your ideas, provide an overview of a project's options and helping with the decision-making process.

Yet whiteboards can be used for more than just project planning and presentation. If you're sketching visual ideas these can be expanded upon and built up within the whiteboard using feedback from others and any accompanying notes you make. More complex visual concepts can also be demonstrated using such applications, with physics and electronics being practically demonstrated through expansion modules. Meanwhile, the ability to import and export files in a variety of formats enables you to explain any number of visual media to a group either in the same room or remotely.

The live presentation aspect of whiteboards means they're widely used in education for teaching and as we'll soon discover, one application in this *Roundup* was created by educational establishments to meet their specific requirements. This *Roundup* has the usual range of FOSS options, with newer programs containing fewer tools compared to more established applications.

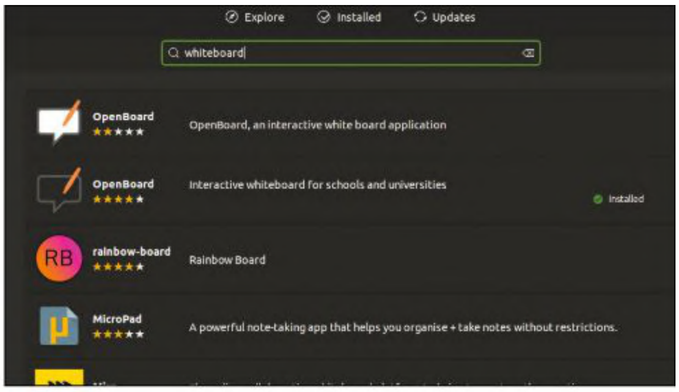
Installation options

How you choose to install will make a difference when you want to upgrade.

The manner in how your software integrates with your system depends on your package manager. Yet in the case of applications such as your whiteboard software you may want it packaged differently. Most of the programs in this *Roundup* are cross-platform, which gives you the opportunity to run your favourite on all platforms. For this reason, you may appreciate Appimage, Flatpak or even a binary based on something else. You have options for all of this, though it shouldn't be the main reason for choosing a particular application.

In an ordinary installation of Linux, there's support for the three main packaging systems. Choosing any of these packages will be enough to install. Only *Lorien* doesn't come with any of the known packages. Instead they've created a binary containing the *Godot3* engine – no other installation needed. The three best applications in this regard are *Xournal++*, *OpenBoard* and *Rainbow Board* as they come as all three. Support for other OSes are also well supported in this group, making it easier to use on several devices.

Rnote's developers have decided to focus on just a Flatpak version. If you don't want to support Flatpak, you have the source code of course, which is the case for all the packages. From a developer's perspective, choosing a single package is a good thing because it frees up time to develop features and correct bugs. If users want another package format, there needs to be someone who can contribute some time to set it up and then test it. Being free and open source, you can always compile it yourself.



OpenBoard ★★★★ OpenBoard, an interactive white board application
OpenBoard ★★★★ Interactive whiteboard for schools and universities Installed
rainbow-board ★★★★ Rainbow Board
MicroPad ★★★★ A powerful note-taking app that helps you organise + take notes without restrictions.

The Ubuntu Software tool lists a number of whiteboard programs, with the obvious one, OpenBoard, shown at the top. Not all programs will be listed, though.

If you want to install from source, you have a few choices to make again, because different languages are involved. *Xournal++* and *OpenBoard* are C++ and so are compiled using the *gcc*. *Rnote* uses Rust, which makes compiling quite similar. *Rainbow Board* is written in JavaScript and is an *Electron* application. The interesting outlier here is *Lorien* that requires *Godot3* to compile. Once you have that, you can compile for all the OSes.

None of the packages have huge dependencies to worry about, so your system can stay slim when using them. *Lorien* only needs *Godot3* installed if you want to compile it.

VERDICT

XOURNAL++	8/10	RAINBOW BOARD	6/10
OPENBOARD	7/10	RNOTE	6/10
LORIEN	5/10		

All the applications in this Roundup have well-known packages, so installing is available and simple for most distributions.

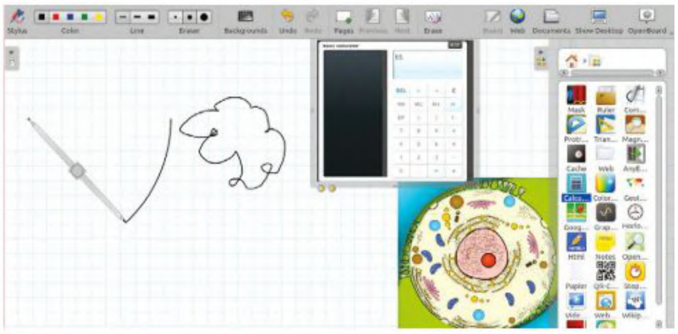
Whiteboard or notepad?

Do you need to take notes or capture your audience's attention?

The main input tool of all these packages is the pen. Drawing freehand lines is the foundation of artistic freedom and expression of thought. The actual shape of the line comes second. Using black, solid lines is enough when all you want to do is to write or mark sections. Simple graphs – including charts – don't need anything else, either. With that being said, lines that look more like charcoal marks or the ability to create circles and squares are welcome features, but when you present ideas, they need to be simple and clear.

OpenBoard provides a range of tools that you can use for visualising your ideas. You can even create your own if you have some JavaScript experience. The application also sports a web browser and the ability to have the board duplicated to another screen, which is a feature that's unique among the programs on test here. When it comes to features, *Xournal++* comes second to *OpenBoard*. The others are still limited in features because they're either new or have very few developers behind them.

Although *Rnote* is one of the newer projects, it actually fairs well in this test category. It sports ellipse and Bézier tools, along with a structured brush that enables you to create strokes, which look artistic and might help in capturing your audience's attention or making a certain point extra clear. *Lorien* and *Rainbow* have the fewest features but are stable and may improve in time as they build on their base. With both audience and code being small these two programs may find a niche to thrive in.



OpenBoard provides many features to teach mathematics, physics and even spelling. These are based on JavaScript code and make use of a library that supports interactive tasks, including quizzes.

VERDICT

OPENBOARD	9/10	RAINBOW BOARD	4/10
XOURNAL++	6/10	RNOTE	8/10
LORIEN	4/10		

Choosing an application to match your own uses isn't as easy as simply picking the one with the most features.

Day-to-day usability

Choose a program to meet your needs.

The programs in this *Roundup* have one thing in common: they all support freehand drawing. How advanced they are depends mostly on maturity, but also on the goals of the project. *OpenBoard's* developers clearly aim to support the teaching community, because its features include a host of educational tools. Other programs are small and nimble – useful for taking notes and getting started on projects.

Other features you might be looking for include the *LaTeX* editor that's built into *Xournal++*. For the more artistic, you can start with the simpler options and then export your work in your favourite format to continue in a more advanced drawing program. Make sure the whiteboard program's export options meet your expectations if you choose to go down this route.

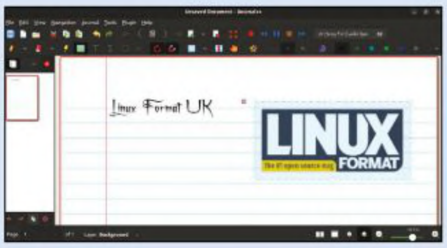
If you're teaching then there's an obvious choice in this month's *Roundup*: *OpenBoard*. Mathematics, spelling and physics are just a few of the modules on offer in this program.

Xournal++ 7/10

Using *Xournal*++ looks simple – even a little primitive at first – but don't be deceived. Once you start using the features you'll discover more tasks that you can do. The reason it looks so simple is that *Xournal*++ is designed for taking notes. It still works well for that particular task, but you can also share the results with an audience.

When you launch the program you're given lined paper and a pen to work with, but keep trying the tools and you'll find so much more. You can work with multiple pages and add images. Once you have got used to these extras, you can also use the PDF annotator to make notes, highlight parts of a PDF and save the result.

If you still lack some feature, you can also build plugins. This requires that you know the Lua programming language, though. This nimble package is a light version of what we're looking for here.

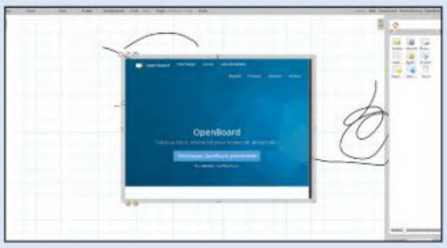


OpenBoard 9/10

OpenBoard has clearly evolved from its predecessors: *Uniboard* and *Open-Sankoré*. This project is aimed at teachers, and apart from using it on a tablet with a pen, it's also possible to display your work on a separate screen.

This teaching tool has a similar interface to the other programs on test, but when you look closer you'll find many modules to help you. A standard install includes modules to show mathematics and physics. Many of the included plugins are interactive – they contain exercises to learn different subjects and demonstrator gadgets to show scales and other physics-related subjects. While you're at it, check out the model of a human cell.

The *OpenBoard* tool truly shines when it comes to showing how things work and demonstrating concepts. If you want to teach, this is the tool for you. This makes it the most comprehensive of the bunch.



File export and import options

Combining files for pleasing visuals requires more than one file format.

The primary purpose for the majority of the applications in this month's *Roundup* is to take notes, either for teaching purposes or to help develop new projects. As such, it's crucial that you can move your work files over to other applications where necessary.

For example, if you have the basic idea for a logo after working up the basic shapes in the whiteboard software, you could then refine it in *Inkscape*. The best file format for this is, of course, SVG while PDF is a better format for sending documents for review and comment. If you want to add your results to other graphical projects in *GIMP* or *Krita*, then PNG and JPG export options are useful.

Exporting to PNG is universally supported; however, SVG is lacking in the tested version of *Lorien* (although version 0.5.0 has this export option). *Rnote* stands out by being able to export to SVG, PDF and *Xournal*++. It can also export a selection as well as the entire document. Naturally, you can use these formats in other documents such as ODT or a web page project.

Importing can also be useful when you need to refer to external information, real events or you want to go through a PDF's content. *Rnote*, *Xournal*++ and *OpenBoard* have this capability, initially designed to annotate the PDF you're reading. After importing a document you can draw lines and circles, and highlight what you want to talk about. With a little more artistic effort you can make compelling presentations commenting on other people's content. The amount of import options in *OpenBoard* is impressive. For example, the program makes use of the common video file formats for playing a video.

VERDICT

XOURNAL++	7/10	RAINBOW BOARD	5/10
OPENBOARD	9/10	RNOTE	6/10
LORIEN	3/10		

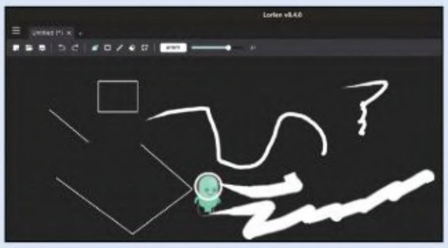
Even though importing and exporting files should be paramount for showing notes and demonstrating concepts, few applications support it very well.

Lorien 4/10

Lorien is in very early development, so doesn't have many features. Having said that, the interface is clean and simple to use. With more features this can be a powerful application, especially for artistic people. The features you do have are well implemented and there's a nice feel to it.

With just a few types of lines and a rectangle feature, you can make a few drawings and export them as a PNG. Version 0.5.0 provides an SVG export option, but you must run it yourself using the *Godot3* engine.

If you're interested in programming games using *Godot*, download the code from **https://github.com/mbrlabs/Lorien** and add a feature or two. This is a great way to start coding. You can then move on to your own project. The software is a *Godot* application, and it even includes a gaming Easter egg. Try pressing F12 while you're running *Lorien*.

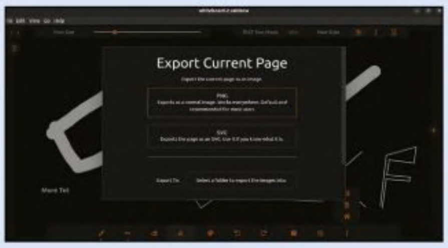


Rainbow Board 5/10

Rainbow Board is a good start for a whiteboard application. This one is also limited to just a few features, which include pens, lines and a text module. Export options includes PNG and SVG.

You can also create several pages in this application – useful when you want to show concepts or draw a book. There are no advanced features, but it works well and is stable. Because it's an *Electron* application it's written in JavaScript, and is easy to get started and is simple to use.

This application has started out with simplicity and grace which gives it a great future if the features match the needs of users. In general *Rainbow Board* seems more polished than *Lorien* but a touch ahead in features. These two applications seem to be heading for a race into this space. While *Lorien* is more focused on the artistic side, *Rainbow Board* is better suited at technical presentations.

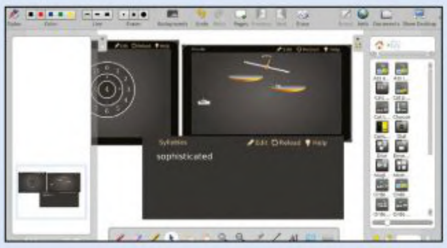


Rnote 7/10

Rnote has taken the route of supporting sketching and note-taking, with a focus on being able to handle many document formats. You can easily annotate PDFs, draw SVG shapes and export your work for developing in other programs. You have more options for how the pen works and you can use tablets for drawing.

As with other programs in this *Roundup*, there's no way to type in text. The addition of text entry using a keyboard would increase this program's usefulness. There's steady development taking place according to GitHub entries, but for now the program is good at note-taking tasks, and making them look appealing.

Written in Rust and using the Gtk project, *Rnote* has the potential to become a tool for Linux users who want a degree of flair when taking notes. Given the program's short history, its stability and competency is a remarkable achievement.



Variety of modules

Tools that can visualise objects are invaluable for demonstrating concepts.

Your own scribbles are a good way to start organising your thoughts, but they're not always helpful when trying to teach others. In many subjects, using tools to show what happens in reality works best. You may want to show how a scale works, and so an animation that shows a scale working would be useful when it comes to visualising objects.

OpenBoard excels at such tasks because it comes with a large pre-installed selection of modules with its regular install. You can also add pictures, video and, if you have the skills, your own "interactives". To make your own there's a JavaScript library you can use to create both graphical tools and quizzes. An entire test can be made using these elements, and there are many included examples to inspire you to create your own.

Rnote and *Xournal++* contain imports that you can use to place images in your presentations. You can also create neat documents using the built-in *LaTeX* editor in *Xournal++*. Other than that there's not much support for creating anything advanced on top of what's bundled with the software.

Rainbow Board and *Lorien* only support pens and in *Rainbow Board*, a rudimentary text tool. These projects are, of course, very new so just delivering the simple interface handling the pen is the current focus. The future will tell where they go on this subject; it would be interesting to see *Lorien* add animation tools considering that the code is based on *Godot3*.

The type of modules that you want will depend on your goals for using these tools in the first place. At the end of the day, if all you want to do is have a place to scribble notes, then you don't need modules.

VERDICT

XOURNAL++	4/10	RAINBOW BOARD	3/10
OPENBOARD	10/10	RNOTE	5/10
LORIEN	3/10		

Modules for whiteboard software is a detail for when you insist on using it for presentations and to educate people.

Development maturity

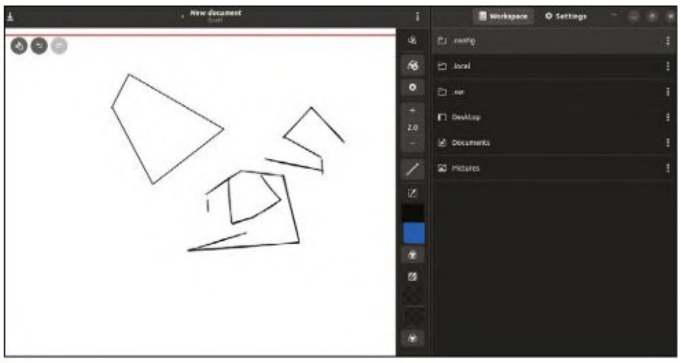
To evaluate these packages we need to know how far along the project is.

One thing we advise you do when choosing your whiteboard software is consider the level of development that's taken place and how much is going on right now. In this group, some applications are early in development, while others are old and forks of forks, which is all part of the beauty of free and open source software.

Usually, you can still use the software that's no longer actively maintained, but you may run into problems and have no one to turn to for help. That's why it's much better to check your package before you get used to it.

To keep up to date with dev activity, you can go to their repository. In this case, they're all on GitHub. The Insights pages are a good indicator of how many are working on the project and how often they contribute. Looking through the projects, you can see how they're doing. Beware of projects with low activity. Many projects are on the site without any activity for years. Avoid them!

OpenBoard is very active and has many contributors, which bodes well for the future. *Xournal* is in the same position but *Rnote,* while active, has only a few new.


Rnote is polished but lacks many features. As this project matures, it can become a contender for your toolkit as a computer user and visionary.

Lorien is also a new program, but has a large base of contributors, while *Rainbow Board* has few contributors. Fortunately, neither of these projects appear abandoned, although *OpenBoard* has the longest track record and high activity on the repository. For a special experience, consider supporting *Lorien*. Because it was started using *Godot3*, it can be a great tool for planning animation projects.

VERDICT

XOURNAL++	8/10	RAINBOW BOARD	5/10
OPENBOARD	8/10	RNOTE	4/10
LORIEN	3/10		

Don't use development maturity as your main factor for choosing which one to use. Instead, just keep it in mind.

Making live presentations

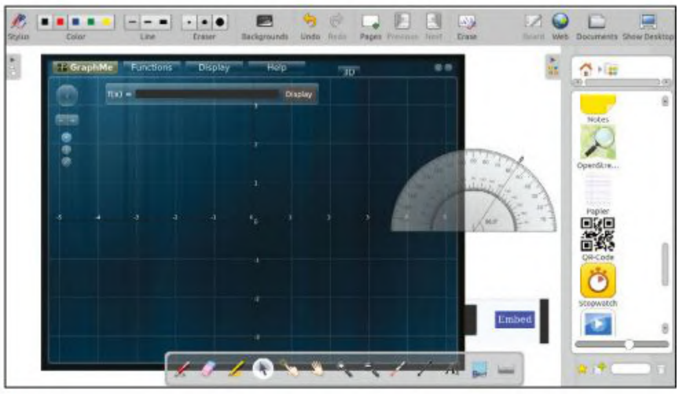
Use these programs to create a presentation or interactive discussion.

When giving a presentation, *LibreOffice Impress* would likely be your first choice. However, the fact that pre-prepared slides are necessary makes it unsuitable for certain settings. You'll have difficulties if you want to change slides during a presentation. In contrast, these whiteboard programs enable you to draw objects as you go through your presentation. You can also use them to lead the discussion or add ideas on the fly. None of these solutions have multi-user capability, though.

With all that in mind, you should know that for education, *OpenBoard* outshines them all. This comes as no surprise when you realise that the project's origin is from educational institutions in France. With the earlier mentioned "interactives", you can show all the objects you need in *OpenBoard*. If you want to show a video, just dump it on the board and click Play. There's even a built-in browser to show relevant web pages during the seminar.

Nothing compares to *OpenBoard* when it comes to presentations, but with *Xournal++* you can get fairly close using images and external documents. Surprisingly, *Rnote* has great annotation capabilities for PDFs, which could be used for highlighting within the presentation document.

Rainbow Board and *Lorien* lack such capabilities. Given that they're young projects this isn't surprising. Time will tell if they're interested in going down that route. Before that happens, you'll have to combine them with other tools or use the more advanced programs covered in this *RoundUp*.


You may want to show the effects of functions, how angles work or show maps. All these are available in OpenBoard. You can even create a QR-code for any string.

As mentioned earlier, multi-user capabilities would be a great addition. These functions exist on many related online systems. This all makes sense since you'd only be able to share when you have a network to share over. If you start with sharing everything on a web page it's logical to have everyone join in the fun.

VERDICT

XOURNAL++	7/10	RAINBOW BOARD	4/10
OPENBOARD	10/10	RNOTE	5/10
LORIEN	3/10		

Using these whiteboard packages for multi-user discussions is the next grand leap for the developers to achieve.

The Verdict
Software whiteboards

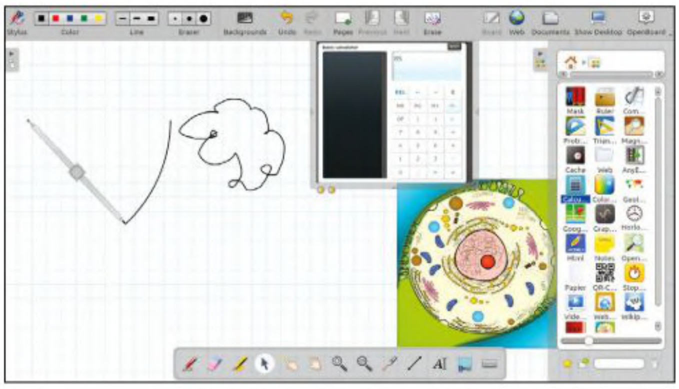
When starting using whiteboard software, you'll probably begin by using it for taking notes. This means you'll find even the simplest choice here adequate. Once you want to start annotating documents, you run into limitations using the simplest of tools. Not all support importing PDFs, which you'll need when going through reports. Mind you, with *Rnote*, you already have that and it's still in early development. *Rainbow* and *Lorien* lack this support, so will only be of interest to people who also want to contribute and have their own way.

Because of its many features and strength at teaching, we have a clear winner: *OpenBoard*. Developed by a university for the teaching staff, it has many contributors who teach and learn. They also use it in their own teaching, and it even contains quiz modules, and custom modules that you can add yourself using JavaScript. If you want a powerful tool to teach and explain things, this should be your first port of call.

As you start using whiteboard software, you'll soon discover if you need to export to other formats. For this, you have universal support for images, but PDF use is limited. You can choose to have an infinite scroll, or better still, use pages that split your efforts up into pieces. Like a slideshow, you can use this for presentations. Oddly enough, there's no export option to formats such as *Impress* or other slide-show solutions.

To share your creations, *OpenBoard* has a presentation mode that shows the result on a separate screen, presumably the bigger screen in the room. None of the packages can share straight to a video conference, though. You must do this by sharing your screen in some other solution. The other programs don't have multi-screen presentation modes, and neither do they have multi-user modes that would enable several contributors to scribble ideas simultaneously.

As you explore this class of program you'll soon notice whether you're making it easier to take notes or you want to show your work to an audience. For notes only, you can pick the simplest and leave it at that but when you need to present to others, you'll need more advanced solutions. Consider what you want to do. Perhaps *Lorien* as the smallest of the five on test here is the most interesting because it uses *Godot3*. If you want to know more about animating and building games, you would be wise to invest your time in this small project.



1st OpenBoard 9/10

Web: **www.openboard.ch**
Licence: **GPL-3.0** **Version:** **1.6.1**
Comes packed with features, while plugins give you even more to work with.

2nd Xournal++ 7/10

Web: **https://xournalpp.github.io**
Licence: **GPL v2** **Version:** **1.1.1**
Considering *Xournal++* is designed for taking notes, it's surprisingly capable.

3rd Rnote 7/10

Web: **https://github.com/flxzt/rnote**
Licence: **GPL v3.0** **Version:** **0.5.3**
Rnotes' short time in development explains its lack of features.

4th Rainbow Board 5/10

Web: **https://harshkhandeparkar.github.io/rainbow-board**
Licence: **MIT license** **Version:** **0.8.1**
Looks great, but needs more features to be classed as useful.

5th Lorien 4/10

Web: **https://github.com/mbrlabs/Lorien**
Licence: **MIT license** **Version:** **0.4.0**
With roots in *Godot 3.0* graphic artists should consider this application.

» ALSO CONSIDER

An alternative to whiteboard tools are sites that enable you to draw shapes and create graphics. One example is **https://vectr.com**. On this site, you can draw freehand, create shapes and play with Bézier curves. It also contains many arrows, symbols and cartoon-like shapes to add to your creation. If you're not on your own computer, this is a great option to get some ideas "out of your head" on a portable smart device. You can download your files for working up later. Other options are available. A simpler version is the **https://excalidraw.com**, which has fewer features and only a few shapes to add. You can only save your files as either PNG or SVG, though.

These two sites charge for more features, but even the free versions enable you to quickly visualise your ideas while out and about.

HACK YOUR GRAPHICS

Join pixel-eyed **Jonni Bidwell** on a quest to demystify the world of graphics on Linux.

Graphics on Linux is complicated. On Windows, you have one driver model and one graphics API (DirectX... okay there are multiple versions, and subsets such as Direct2D). On Linux we've lost count. And it's always been tricky, as anyone who's tried to run X11 in 2000 will testify.

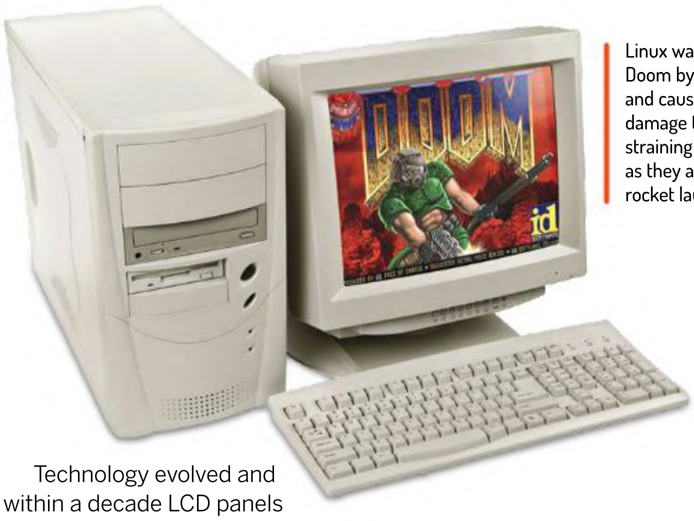
Efforts have been made to simplify things, and in a way they are. AMD and Intel cards, thanks to open source drivers, will generally work out of the box on most desktop distributions. Even hardware requiring Nvidia's proprietary drivers will work automagically in Ubuntu and Fedora. In future this driver will be free, too. Until then the open source Nouveau driver continues to improve, and (thanks to Nvidia finally releasing signed firmware) should provide much-improved performance for newer cards.

However, there's more to it than just kernel drivers. You might be lucky and have never had to edit an **xorg.conf** file in your life. Back in the day this was essential, and even today it's the primary way to set parameters for the drivers within the 38-years-old display server. Wayland is set to replace this, and is a much leaner, simpler construction than X.org. And then there's 3D accelerated applications, which sidestep X and Wayland altogether. In order to do their thing, they leverage Direct Rendering Infrastructure (DRI) and Mesa (or a proprietary middle layer) to draw spinning cubes or penguin-bearing racing karts with OpenGL.

OpenGL is old (30 years) too, so we have a new protocol for close-to-the-metal communication with GPUs. Through Vulkan we can translate from DirectX 12 in the blink of an eye, enabling us to play Windows *Steam* games at very close to native speed. Graphics cards are useful for things other than graphics, too. So protocols like OpenCL and CUDA for general-purpose GPU compute all need their own layers. As you may have fathomed, there's a lot to cover, so let's tarry not one second longer.

In the beginning…

Linux still talks to LCD monitors via a CRT controller interface. Find out where that came from!



Linux was running Doom by 1995 and causing retina damage to those straining at CRTs as they aimed rocket launchers.

Porting the X Windows System from (various flavours of) Unix to Linux was in some ways key to its success. A GUI, no matter how primitive, enables richer applications to be developed and richer content to be produced. In the mid-90s the only displays users needed to worry about were CRT monitors. Y'know, those heavy, retina-burning monsters whose depth was at least that of their screen's diagonal. These work by sweeping a beam (okay, three beams if you want colour) of electrons back and forth with a given frequency, and then up and down with a slightly lesser frequency. These are the horizontal and vertical scan rates

Even in this lawless era (which tolerated such atrocities as Chris Evans on *The Big Breakfast*) there were standards, one such being the VESA modes. These specified some basic modes that monitors had to support in order to be fit to bear the VESA stamp. Monitors and graphics card could often do better, but there wasn't a standard way for higher resolutions and refresh rates to be codified. So one would deftly open up **xorg.conf** in *Emacs* (or whatever was one's text editor of choice) and add a custom modeline. This specified the various frequencies and resolutions required to unlock the new mode, but in some cases required careful calculations to get right.

Behold, magic smoke!

It used to be said that if you miscalculated these values then the magic smoke might escape from your monitor. As monitors became more advanced, they began to support all kinds of different modes and frequencies, making it even more tempting for users to try their hand at enabling them. And probably much magic smoke did indeed find its way into the upper atmosphere as a result of over-zealous modeline misconfigurations.

Technology evolved and within a decade LCD panels had become the norm. These all have a native resolution (where they look best), and being digital in nature were capable of providing this (and other supported resolutions) to the software over EDID (and later DDC). In theory it was possible to turn a panel into a conflagration by ignoring this information and telling it to enter an unsupported mode, but X11 evolved to make this difficult. Still, the X server had to have all the power here, which was at odds with traditional ideas about privilege separation.

As GPUs became more complex, X11 found itself dealing with things like interrupts, memory access and other things that no sane developer would allow anywhere but the kernel. Apart from that, under this usermode setting (UMS) system, if there was a kernel panic while X was running, then the kernel wouldn't be able to switch out of the graphical mode to display debug information, and it didn't know anything about the graphical modes so nor could it overlay text there. A conundrum, to be sure. Enter our hero, Kernel ModeSetting, who you can read about in the box (*below*).

THE GOOD SORT OF DRM AND OTHER TLAS…

Kernel modesetting (KMS) took all those privileges away from the X server, allowed high-resolution consoles at early boot time, and obviated all the flickering as one's machine switched from 80x25 text mode to those higher resolutions.

KMS was, at least in part, enabled by AMD (which acquired ATI in 2007) and its open source driver strategy. With free drivers in the kernel for two out of three major graphics players, it meant KMS had majority support from quite early on. Nvidia users would have to wait until support was added to the proprietary driver, which happened in 2016. Yet, of course, this relies on external modules, which must be kept in sync with the kernel, so is still potentially fragile.

KMS is part of the wider DRM (Direct Rendering Manager) kernel subsystem, which was introduced to enable multiple applications to access the GPU concurrently so they could all avail themselves of 3D-acceleration. DRM has its counterpart inside the X server, the direct rendering infrastructure (DRI) interface. This enables userspace programs to access and allocate GPU resources, via DRM in the kernel, rather than indirectly via the X server (which, for example, becomes very slow in the case of OpenGL command queues which generate lots of data).

Before DRI, only one application could access graphics hardware at anytime. Out of necessity, this application was de facto the X server itself. If anything else tried, a resource conflict and likely a hard lock-up would ensue.

Into a third dimension

We add some depth to the story and learn about the first, bona fide, GPU.

The first consumer 3D graphics cards appeared under 3DFX's "Voodoo" branding in late 1996. These were add-in cards that still required a regular video card (or chip) to display 2D scenes. But they enabled games like *Tomb Raider*, *Descent*, *Grand Theft Auto* and *Unreal* to use 3DFX's proprietary Glide API and render things (well, some things) much faster than software rendering. A number of other proprietary 3D graphics APIs appeared around this time, including the S3 3D Engine (used in Virge cards), the Nvidia Multimedia Library, the Creative 3D Library (used in Blaster 3D devices) and the even more imaginatively titled ATI 3D C Interface (for 3D Rage hardware).

Silicon Graphics, makers of expensive workstations, also had a proprietary API, in the form of IRIS GL. This could do 3D things, but was also much more general purpose (it was also a windowing system). There was industry demand for parts of this API to be licenced (and competition from the likes of Sun and IBM), and in 1992 SGI tailored a version of IRIS GL into a new open standard called OpenGL. They also organised the OpenGL Architecture Review Board (OpenGL ARB), a group of companies that would maintain the specification in future. Today, OpenGL is steered in much the same way by the Khronos consortium.

We'll never forget PC Format's cover feature from 1996: "Quake Survival Kit".

Linux, of course, was still very young at this stage, but early efforts to support OpenGL appeared in the form of Utah GLX (circa 1995, see **http://utah-glx.sourceforge.net**). Complete OpenGL support for consumer cards wouldn't appear until the GeForce era, but when id Software's *GLQuake* was released in 1997, so too was a 'MiniGL' driver for the 3DFX Voodoo card, which implemented just enough OpenGL to run *Quake*. John Carmack had originally written *GLQuake* for SGI machines, which were powerful enough to do all this in software, and intended to port this only to the Rendition Vérité chips found in 3D Blaster cards. Home PCs couldn't hope to run *GLQuake* on their own, but manufacturers of other graphics cards (keen to quash 3DFX's monopoly) were able to release their own MiniGL drivers. And release they did.

It's worth reiterating that id Software was probably the earliest example of a big game studio that actively supported Linux. *Doom* was ported to Linux in 1995 by id's Dave Taylor, and *Quake* (including a port of *GLQuake*) made it across in 1998. If you look at the setup instructions for *Quake* (**www.linuxjournal.com/article/3180**) you'll see they reference Mesa 2.5, a project that hoped to provide a complete OpenGL implementation on Linux. Mesa is still around today – in fact it's one of the biggest FOSS projects out there.

» MESA TODAY

OpenGL remains the biggest API supported by Mesa today, but it's far from the only one. There's its embedded equivalent, OpenGL ES, OpenVG (for 2D vector graphics), EGL (used by Wayland to get its buffers on to a screen) as well as GLX (which isn't needed on Wayland). You may be surprised to hear that Mesa has supported Direct3D 9 since 2015. Strange but true – it's what provided *Wine's* original DirectX 9 support, before Vulkan came along.

Vulkan, also under the auspices of Khronos, is a new 'closer to the metal' graphics API designed to provide programmers with ultimate flexibility. DirectX 12 also followed similar, low-level design ideas, and as such it's much easier to translate between the two. Unlike the situation with OpenGL and DirectX-es 10 and 11. Vulkan is what enables Valve's Proton to play Windows titles at 'near native speed' and its support is handled through Mesa, too. More on Vulkan on the next page.

It's more fun to test your 3D capabilities with Supertux Kart than spinning the cogs of Glxgears.

In addition to 3D support for AMD, Nvidia and Intel GPUs using FOSS drivers, Mesa also includes drivers for embedded graphics devices, such as the Lima and Panfrost drivers for Mali hardware from ARM. The free vc4 driver for the Raspberry Pi is in there, too (as is the new v3d driver for the Pi 4). Oh, and we haven't even mentioned OpenCL for general-purpose GPU calculations, but that too is supported by Mesa. And so are the video acceleration APIs VA-API and VDPAU, and OpenMAX for XR (virtual and augmented reality).

Without it Linux would be entirely two-dimensional. But to understand the wherefore of Mesa and OpenGL, we must first understand Redmond and DirectX.

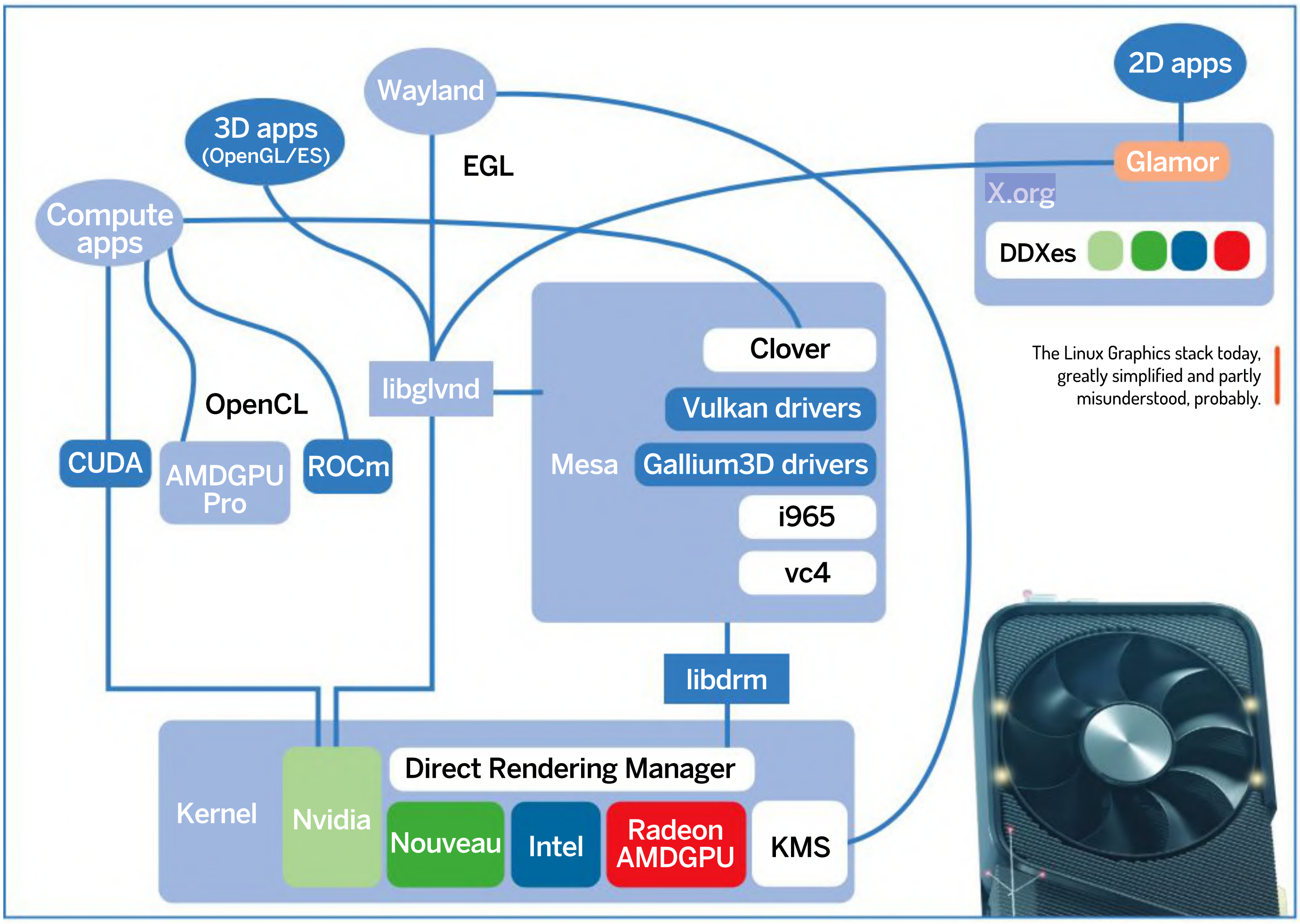
X marks the spot

Microsoft released the first version of DirectX in 1996, and by its third outing it included OpenGL's main competitor, Direct3D. Even in the late 90s this was still considered unstable, so developers focused on the more proven API. And some of those miniGL implementations became fully OpenGL compliant. 3DFX was toppled from its throne when Nvidia released the GeForce 256 (the world's first actual GPU). Vertex textures and lighting were still CPU-rendered with 3DFX hardware, but Nvidia's offering could accelerate them too. So *GLQuake* enjoyed even bigger benchmarks, and 3DFX's assets were eventually acquired by Nvidia. So how did DirectX come to dominate OpenGL?

It's a bit baffling to be honest. For years and seven versions DirectX failed to match parity with OpenGL's increasing featureset, and Microsoft, sick of playing catchup, entered into negotiations with Nvidia. Nvidia was planning the GeForce 3 and Microsoft was dreaming up DirectX 8. By more or less building a shader model to fit that hardware, and releasing the DX8 at the same time as the Geforce, Microsoft managed to do something right. When ATI released its competitor to the GeForce 3, Microsoft released D3D's Shader Model 1.1 to cater to that hardware, too.

Nvidia, meanwhile, wanted to have its textured and lit cake and eat it, so it got to work adding new OpenGL shader extensions. As usual, these were proprietary extensions. Instead of saying here's a feature that can be abstracted to any (capable) hardware, Nvidia's contributions were for Nvidia-cards only. So when ATI wanted in on the OpenGL action, it too contributed proprietary extensions. So a developer writing shaders for Nvidia hardware via DirectX could – without too much effort – refactor those shaders for ATI devices, while one developing in OpenGL would have to rewrite more or less everything. Hardly the utopia the ARB had envisioned. At GDC 1997 3D Labs proposed GLSL, the OpenGL Shading Language. This was a good idea, but poorly executed. GLSL was inflexible, inefficient and not as hardware agnostic as its newly hatched DirectX counterpart, HLSL (High Level Shading Language). Still, the ARB voted to include GLSL in OpenGL 2.0 – see page 40 for a complete timeline of GPU events!

And where was Linux in all of this? Well, Dan Stone put it quite nicely in our **LXF243** interview: "we were 15 years late to the party". Indeed, Linux's OpenGL stack of the early-to-mid 2000s was somewhat lacking. Mesa was still a small project, albeit one that had garnered contributions from all over the field, and OpenGL itself wasn't maturing in an ideal way. In fact, the ARB embarked on an effort to modernise the OpenGL API, but this didn't really achieve anything. So blame for this lies anywhere but with Mesa creator Brian Paul. 3DFX had open sourced its Glide API in 1999 and this was now in Mesa, but no one was really using Glide anymore. The X server now had its own GLX API, so that content rendered on the GPU could be displayed in a window.



The Linux Graphics stack today, greatly simplified and partly misunderstood, probably.

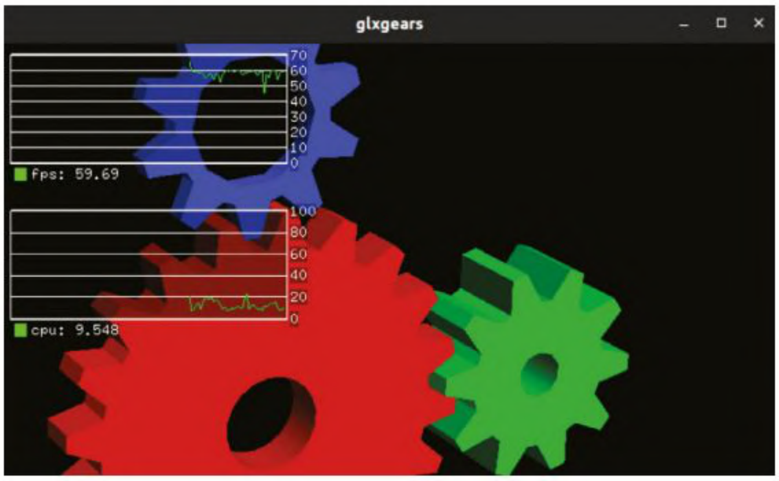
Querying your 3D stack

Intel's numbering systems make no sense, but whatever your card there's plenty of demos to play with…

For a long time, users of middle-aged Intel hardware were stuck using the venerable i965 Mesa driver. This caters to a range of hardware, from Broadwater (2006) to the HD and Iris graphics found in some Skylake chips (2015). For older hardware there's the i915 Mesa driver, not that such hardware had much in the way of 3D capability. Newer, so-called Gen8 graphics and above (from Broadwell through to the latest Xe graphics) is supported by the Iris Gallium3D driver.

The i965 (and i915) driver dates back to before the Gallium framework was developed, and throughout last year there were calls to retire it. However, Mesa aren't in the habit of mass-scale hardware abandonment, so until a replacement was finalised, i965, and other 'classic' Mesa drivers were granted a reprieve.

Glxgears looks much better overlaid with graphs from the Gallium HUD.



That replacement driver goes by the name of Crocus and it's ready for prime time. So the classic Mesa drivers were deprecated (more precisely demoted to 'amber' status, like the preserved bugs in *Jurassic Park*) in December 2021. If you're running the latest Fedora, Ubuntu or a rolling release distro (on appropriate hardware) then you will already be using Crocus.

What's in a name?

First though, let's clear up Intel's slightly awkward generational naming. Intel's seventh-generation processors (codename Kaby Lake, part of the Sky Lake microarchitecture) include the four-core 7700HQ desktop CPU and the mobile 7500U. That all seems consistent, everything begins with 7. However, some Kaby Lake processors sport Iris Plus 630 or 640 graphics, and some feature the lesser HD 6x0 technology. So not only is there a numerical discrepancy, but at first glance it would seem that the Mesa Iris driver (explicitly described as for Gen8 and later) shouldn't be used in (some) Kaby Lake systems. Except that's not right – the Iris driver caters to this generation of Intel graphics, too. In fact, it caters to one generation before that. So the *iris* driver actually caters to some non-Iris chips, and Gen9 graphics are actually found in processors marketed as 7th Gen. Even Intel aren't immune from off-by-one errors, it would seem.

We can check which kernel driver and Mesa driver is in use by running

```
$ LIBGL_DEBUG=verbose glxgears
```

Almost all Intel graphics use the i915 kernel driver (named for the first Pentium 4 chipset), and the Mesa driver will be shown immediately below, for example:

» AN ACCUMULATION OF ACRONYMS

Modern graphics cards are fitted with thousands of tiny processors, which is what enables them to compute so many shaders at once. These processors aren't much use for some compute jobs but for others, where the work can be broken down into lots of units, they're much more efficient than traditional CPUs.

Before Bitcoin ASICs became popular (and drove up the mining difficulty) GPUs used to be the most efficient way to mine them. Password cracking (or recovery) is another application where GPUs come into their own. *John the Ripper* and *Hashcat* both have options to use OpenCL to speed things up.

In addition to its open source driver stack, AMD has been working on a new open standard for GPU computing. Dubbed ROCm (Radeon Open Compute platform – apparently acronyms are allowed to do this now), which is aimed at high-performance computing (HPC), in particular machine learning (ML) applications. ROCm only supports newer AMD hardware (its Polaris graphics architecture and newer), with first-class status granted to its professional Instinct line of "GPU accelerators" (see **https://bit.ly/lxf292-amd-instinct**).

ROCm backends already exist for popular machine learning applications, including *TensorFlow* and *PyTorch*. *Blender* can also make use of ROCm kernels in its Cycles rendering engine (see **https://gpuopen.com/blender-cycles-amd-gpu**). This is good as the OpenCL backend was removed from Cycles last year.

Part of the ROCm initiative is AMD's Heterogeneous-computing Intrastructure for Portability (HIP, this is allowed too) which hopes to aggregate CPU- and GPU-compute, as well as GPU-compute across different hardware. For example, it makes it possible to target either OpenCL or CUDA from a single codebase.

```
libGL: using driver i915 for 4
libGL: pci id for fd 4: 8086:5916, driver iris
libGL: MESA-LOADER: dlopen(/usr/lib/x86_64-linux-
gnu/dri/iris_dri.so)
```

If you want something a bit more interesting than spinning cubes to test your OpenGL stack then the *GLMark2* benchmark is a fun diversion. Instead of spinning cubes, this will display spinning horses, boxes, rabbits and cats in a variety of filters and shadings. And it'll give you a score at the end (which is a little useless without some comparison). You can get *GLMark2* with

```
$ sudo apt install glmark2
```

In order to get information from it, or indeed any OpenGL application, you can use the **GALLIUM_HUD** environment variable, which will overlay a neat heads-up display. This has a huge number of options, but the popular ones include frames per second (FPS), load and temperature (GPU and CPU), VRAM usage as well as memory and shader frequencies. Here's a example:

```
GALLIUM_HUD=".c70fps,cpu" glxgears
```

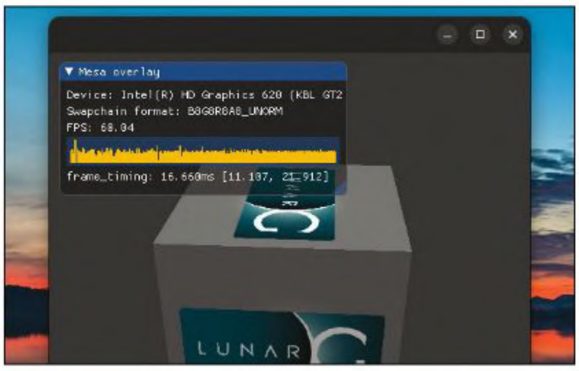
You may need to resize the window to see the effect here. The `.c70` part 'clamps' the graph to 70fps here. The comma separates the two panes vertically. Use a semicolon to do so horizontally, or use a plus sign to have two items share a pane. It's possible to specify co-ordinates for each entry, which can makes the syntax a little hard to parse.

Different Gallium drivers have different options. For example, the *iris* driver has no VRAM options. You can see a list with `GALLIUM_HUD="help"` .

Enter the Vulkan

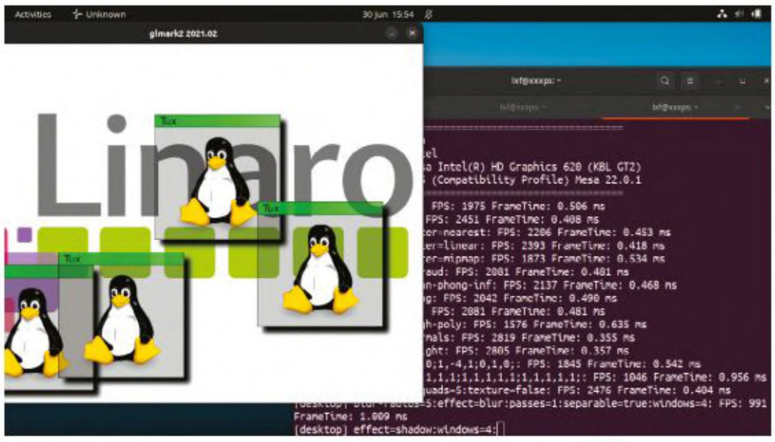
Vulkan is the latest, greatest graphics API on the scene. And it's very unlikely there'll be a Vulkan-DirectX 12 conflict in the same way that OpenGL had to face down DirectX in the past. Sure, Microsoft might rather developers use Direct3D 12, but even if they don't Vulkan is equally well supported on Windows, Mac, mobile and embedded. Basically the whole industry is behind it. Officially launched in 2016, it's designed to be low-overhead and to better allow developers to take advantage of multicore CPUs (as with OpenGL, it doesn't make sense for all types of work to get done on the GPU). It's predecessors all appeared when CPUs had but one core, and later versions of OpenGL and DX11 could only accommodate the most basic of support for more.

Vulkan is also unifying in the sense that there's no need for separate APIs, for example, for mobile and desktop, or for compute and graphics. Being a shiny,



```
▼ Mesa overlay
Device: Intel(R) HD Graphics 620 (KBL GT2
Swapchain format: B8G8R8A8_UNORM
FPS: 68.04
frame_timing: 16.660ms [11.107, 21.912]
```

Vulkan's cube drawing prowess is probably not that much different from OpenGL's, but it is the future.



```
el
a Intel(R) HD Graphics 620 (KBL GT2)
 (Compatibility Profile) Mesa 22.0.1
 FPS: 1975 FrameTime: 0.506 ms
 FPS: 2451 FrameTime: 0.408 ms
er=nearest: FPS: 2206 FrameTime: 0.453 ms
er=linear: FPS: 2393 FrameTime: 0.418 ms
er=mipmap: FPS: 1873 FrameTime: 0.534 ms
aud: FPS: 2081 FrameTime: 0.481 ms
n-phong-inf: FPS: 2137 FrameTime: 0.468 ms
g: FPS: 2042 FrameTime: 0.490 ms
 FPS: 2081 FrameTime: 0.481 ms
h-poly: FPS: 1576 FrameTime: 0.635 ms
nals: FPS: 2819 FrameTime: 0.355 ms
ght: FPS: 2805 FrameTime: 0.357 ms
0;1,-4,1;0,1,0;: FPS: 1845 FrameTime: 0.542 ms
1,1,1;1,1,1,1,1;1,1,1,1,1;: FPS: 1046 FrameTime: 0.956 ms
uads=5:texture=false: FPS: 2476 FrameTime: 0.404 ms
[desktop] blur-radius=5:effect=blur:passes=1:separable=true:windows=4: FPS: 991
FrameTime: 1.009 ms
[desktop] effect=shadow:windows=4:
```

Tuxes, everywhere Tuxes. GLMark also renders other animals and inanimate objects.

modern invention, it supports the latest GPU features such as ray tracing. Support is provided through Mesa, except for proprietary drivers. The Nvidia driver has had Vulkan support for some time.

Analogous to *glxgears* Vulkan has its own *vkcube* test application. Perhaps even more plain than the former, *vkcube* displays a only a single spinning cube with the LunarG logo. Don't worry, there's plenty of more interesting demonstrations, including *vkmark* which you might be interested in building from source yourself (see **https://github.com/vkmark/vkmark** for details). Many games now include a Vulkan backend, one of the first to be supported on Linux was Croteam's *The Talos Principle*.

We can get some measurements out of a Vulkan application with by prefixing a delightfully awkward environment variable:

```
VK_INSTANCE_LAYERS=VK_LAYER_MESA_overlay
vkcube
```

This does nothing on non-Vulkan applications. Like the OpenGL overlay, the Vulkan one can be configured with lengthy syntax too. See **https://gitlab.freedesktop.org/mesa/mesa/-/tree/main/src/vulkan/overlay-layer** for details. The Vulkan overlay has been forked to create MangoHUD, a more fully featured dashboard, which you should certainly check out.

One of the most exciting applications of Vulkan appeared in 2018, when lone developer Philip Rebohle started the DXVK project. Initially, that project had one mission: leverage Vulkan to translate the DirectX 11 calls of the popular game *NieR:Automata*, so that it could be played at near-native speed via *Wine* on Linux. It sounded crazy, but soon the mission was accomplished. Soon DXVK's remit expanded, and it became a general layer for translating DirectX 9, 10 and 11 calls into Vulkan ones, with the side-effect that suddenly a huge number of Windows titles could suddenly be played. The reason for this rapid development soon became clear: Rebohle had been sponsored by Valve, which was about to announce its DXVK-powered fork of *Wine*, now called *Proton*. Thanks to *Proton*, Linux (and in particular Valve's Steam Deck) can play over 5,000 titles that used to be strictly Windows-only.

But the adventures don't stop there. A second project, VKD3D appeared soon after DXVK with the goal of targeting DirectX 12 titles. This was started by *Wine* developer Józef Kucia who sadly passed away before being able to complete it. VKD3D was then picked up by Valve and integrated into *Proton*. So there's less reason than ever to be running Windows.

»

Hack your graphics

Switch drivers, interrogate GPU memory, try and keep the magic smoke from escaping into the atmosphere…

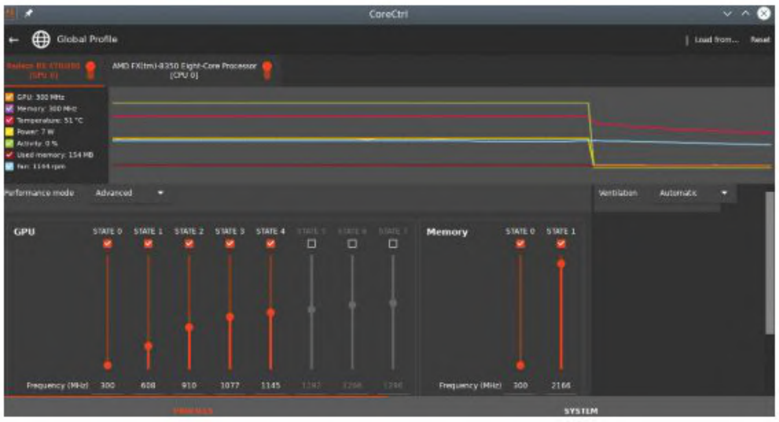
There are some great tools for probing, tweaking and otherwise tinkering with GPU on Linux. Most of these are vendor-specific. Intel provide a number of utilities via the *intel-gpu-tools* package. Once installed you can run `$ sudo intel_gpu_top` to see how busy your GPU is, how much power it's using, whether hardware video codecs are in use and more. It's also handy if you get confused about which 'Gen' your graphics are. If only we'd known about it two pages back…

AMD's *Catalyst Control Centre* on Windows enables you to overclock your GPU and its memory, as well as tweak fan profiles. *Corectrl* on Linux makes it possible to do exactly the same thing, but probably with much less bloat. *Corectrl* is available on Fedora, Arch Linux's AUR or you can build it yourself from source see (**https://gitlab.com/corectrl/corectrl**). For Ubuntu there's a handy PPA (**https://launchpad.net/~ernstp/+archive/ubuntu/mesarc**), where you'll also find newer builds of Mesa, libdrm and the *amdgpu* driver for X.org.

INTEL MIXES THINGS UP

"The release of the more powerful Alchemist cards might end two decades of the AMD/Nvidia duopoly."

Another AMD tool popular with developers and tweakers on Windows is its *Developer Tool Suite*. This has been open sourced (as part of its longstanding GPUopen initiative) and can be obtained from **https://gpuopen.com/tools**. The suite is aimed more at developers than anyone else, but anyone who's even the slightest bit curious as to what their GPU is up to should find at least some of it interesting. The suite includes *GPU Profiler*, *Memory Visualizer*, *GPU Analyzer* and *Developer Portal*. That last one sounds a little scary, tbh.



CoreCtrl enables you to tweak all kinds of settings, which some would say you have no business tweaking.

Canonical's hardware enablement stack (HWE) was intended to offer new kernels and drivers for users of the Ubuntu LTS. Otherwise they'd be missing features for two years until the next LTS appears. The HWE stack was previously an opt-in affair, with the recommendation being for users only to opt-in if they needed support for new hardware, or if something was broken. Since Ubuntu 20.04 though, everybody using desktop Ubuntu is automatically treated to the HWE stack. You can check if you have it by running:

```
$ hwe-support-status
```

Users running Ubuntu 22.04 will track the HWE stack without intervention, too. But if you want the very latest Mesa, Wayland, Vulkan and the like you might want to activate the Oibaf PPA (which has been generously providing fresh Mesa packages since 2011). This can be done with:

```
$ sudo add-apt-repository ppa:oibaf/graphics-drivers
$ sudo apt update
$ sudo apt upgrade
```

The new packages are unlikely to break things (they do get tested), but it's easy to remove them if you run into difficulties:

```
$ sudo apt install ppa-purge
$ sudo ppa-purge ppa:oibaf/graphics-drivers
```

Hardware ray tracing

New graphics cards make much fanfare of their ray-tracing abilities. This certainly makes for some pretty-looking games and renders already, and the future will be even more exciting. In May 2022 Mesa announced a big change to the free RADV driver for Vulkan on Radeon hardware. With Mesa 22, support for Vulkan ray tracing has been extended as far back as GCN 1.0 cards. Previously, this support only extended back as far as Polaris and Vega cards.

It should be stressed that only the newest RDNA2 (RX 6000) cards have bespoke ray-tracing hardware; support on older cards is provided through Linear Bounding Volume Heirarchy (LBVH) calculations. Essentially, it's soft-ray tracing, but it still provides a welcome boost for middle-aged cards. Nvidia actually did a similar thing for its non-RTX cards in 2019, albeit for Microsoft's DXR API.

Speaking of Nvidia, it would be remiss of us not to mention (apart from its new driver strategy, of which you can read in the box, *right*) that in 2019 it released *Quake 2 RTX*. This is a ray-traced remaster of the original, created to show off their RTX cards. You can get it for free on Steam or directly from Nvidia at **www.nvidia.com/en-us/geforce/campaigns/quake-II-rtx**. It's not just for Nvidia (or Windows) users, either. Platform-agnostic Vulkan ray tracing was added in 2020 so this should look quite glorious with the new RADV driver.

If you'd like to test out your GPU's potency without exploding monsters and gibs, there is no shortage of synthetic benchmarking tools available. One of the most popular is *GravityMark*, which you can get from **https://gravitymark.tellusim.com**. Fans of astronomy will find this especially pleasing because it renders all sorts of lovely sights from our universe. And also what our planet would look like surrounded by a ring of as many asteroids as your GPU can draw. There's a scoreboard too, so if you happen to have two RTX 3090s you might just be able to win the top spot.

Another popular purveyor of pretty benchmarking scenes is *Unigine* (see **https://benchmark.unigine.com**). They haven't put out a new offering in a while, but their *Superposition* (2017), *Valley* (2013) and *Heaven* (2009) releases still look lovely and all run beautifully on Linux. *Superposition* now includes a VR option for use with Oculus and SteamVR devices.

Smoother browser behaviour

We've mention VA-API a fair bit in previous cover features, mostly as a way to get smoother playback in your web browser. We're a little miffed to report that to get this working on *Firefox* (at least with Intel hardware on Ubuntu 22.04) still seems to require starting the browser with a rather ugly (and potentially insecure):

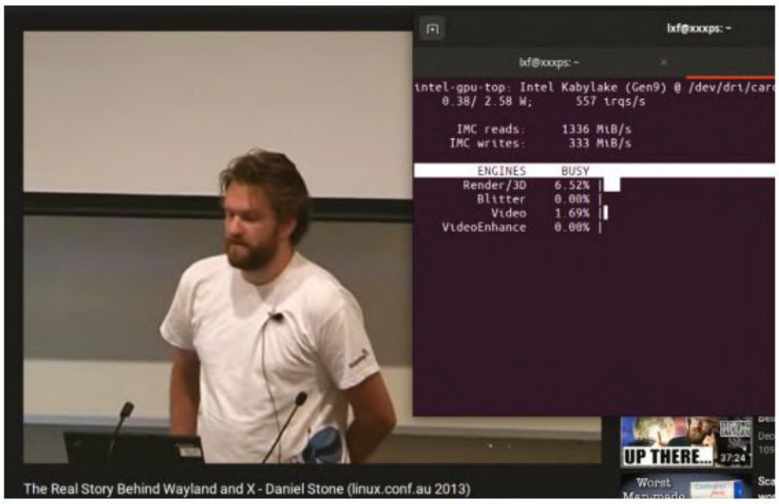
```
$ MOZ_DISABLE_RDD_SANDBOX=1 firefox
```

Be that as it may, Intel is working on a few new things related to this. Its new, open source media API – OneAPI – has been enjoying a lot of attention lately, and part of that initiative is the OneAPI Video Processing Library (OneVPL, **www.intel.com/content/www/us/en/developer/tools/oneapi/onevpl.html**). OneVPL is only for the very latest, Gen12 or later, hardware. In other Intel news, the first of its Arc line of discrete graphics cards, the A380 has launched in China. Initial benchmarks are rather underwhelming (behind that of Nvidia's 1650 of three years ago), but this is an entry-level unit. We're excited for the release of the more powerful Alchemist cards which maybe, just maybe, will end two decades of the AMD/Nvidia duopoly.

It's worth mentioning that the AMDGPU Pro driver stack (which still uses the open source AMDGPU kernel module) has its own Vulkan driver. It also has its own proprietary OpenGL, Vulkan as well OpenCL stacks. To use the Vulkan driver, you'll need to disable Mesa's RADV and also grab the Vulkan SDK from **https://vulkan.lunarg.com/sdk/home**. If you want to start exploring OpenCL (or proprietary Vulkan implementations), then you can install the whole stack directly from the package at **www.amd.com/en/support/linux-drivers**:

```
$ amdgpu-install --usecase=workstation -y
--vulkan=pro --opencl=rocr,legacy
```

There's also AMDVLK, AMD's open alternative to the RADV Vulkan driver. But all accounts suggest that RADV is, well, pretty rad in comparison. And with that nod to late 80s colloquialisms we end our deep dive into the weird world of graphics on Linux. We'd love to hear your stories of pixelated woes or vector victories. Meanwhile, we'll be busy playing the original (and now open source for 12 years) *Quake*. See over the page for our timeline.



```
intel-gpu-top: Intel Kabylake (Gen9) @ /dev/dri/card
    0.38/ 2.58 W;      557 irqs/s

     IMC reads:     1336 MiB/s
    IMC writes:      333 MiB/s

         ENGINES      BUSY
       Render/3D     6.52% |
         Blitter     0.00% |
           Video     1.69% |
    VideoEnhance     0.00% |
```

lxf@xxxps: ~
lxf@xxxps: ~
The Real Story Behind Wayland and X - Daniel Stone (linux.conf.au 2013)
UP THERE...
37:24

The intel_gpu_top program shows that Collabora graphics guru Dan Stone was hardware accelerated back in 2013.

NVIDIA'S CHANGE OF HEART

This feature was in part motivated by Nvidia's release of an open source driver, which is production ready for its datacentre GPUs, and alpha-quality for its consumer cards. Exactly when we'll see distributions start including this driver is a mystery, but the change of strategy is welcome. You can check out this driver right now at **https://github.com/NVIDIA/open-gpu-kernel-modules**. Only newer cards (post-Turing, where a r515 or later driver is available) are supported, and you still need firmware from the binary driver. It's possible to install that using the .run file from **nvidia.com** without the closed modules using, for example:

```
$ sh NVIDIA-Linux-x86_64-515.57.run --no-kernel-modules
```

Or if you don't mind a pre-compiled version of the open modules you can get that from the **.run** file using the `-m=kernel-open`. Finally, in order to use the module on consumer cards you need to flip a switch to let it know you don't mind 'alpha-quality'. This requires creating a file, **/etc/modprobe.d/nvidia.conf** containing the text:

```
options nvidia NVreg_OpenRmEnableUnsupportedGpus=1
```

Older cards, including Pascal cards like the GTX1080 aren't supported because the open driver relies on newer architecture. An open source solution exists for these cards in the form of Nouveau, although performance won't be good because there's no reclocking support. And Nouveau doesn't include a Vulkan driver yet.

There are signs of hope, though. Nvidia has released the signed firmware files for Ampere (RTX30 series) GPUs that enables reclocking. Old school Nvidia GTX 600/700 cards don't require this signed firmware, and Nouveau support on this hardware has long been in rude shape.

Of GPUs and APIs...

And leave us nought but grief and pain... isn't thankfully something Linux users have to experience these days!

We're lucky to live in a time when the Linux Kernel gets, on the whole, first-class hardware support, but it hasn't always been this way. This is especially so when it came to 3D graphics support – or frankly graphic support of any kind. After Jonni's in-depth look into the Kernel's graphics stack and how you can tinker with it, we thought it would be useful to outline how graphics hardware has evolved alongside all those mentioned APIs – and their growing feature sets – used to access all the fancy hardware features.

As you've already read, OpenGL sprang out of Silicon Graphics' proprietary IRIS GL, developed over the 1980s for its own hardware and eventually running on flavours of Unix. This gave OpenGL rich hardware rendering alongside processor-based software rendering options.

Microsoft created DirectX (initially DirectDraw and then Direct 3D) to ensure Windows could compete both with OpenGL for professional applications but more importantly for Microsoft against the emerging gaming consoles. The original DirectX was released at the end of 1995 with Direct3D appearing mid-1996, this version 3.0 was widely panned as being awful and led to version 4.0 being entirely skipped.

We're mentioning these early dates because it's useful to understand how increasing hardware functionality of GPUs (a term coined by Nvidia) went hand-in-hand with new flashy software features, largely used to sell new versions of DirectX/Windows, graphics cards and later consoles. This technological/marketing drive going from the basic "moah resolution!" (read FullHD 1080p) and "moah colours!" (the whole 32-bit colour depth) fanfare to offering greater effects and computational enhancements of GPGPU shaders.

Colourful language

Going back to the early days of the PC (1982) you'd expect a framebuffer card to offer a basic CGA (Color Graphic Array) display of 640x200 in two colours. Yet in 1984 EGA could still only muster 640x200 in 16 colours from a fixed palette of 64. It took until 1987 for VGA to appear with 640x480 in 16 colours or 320x240 in 256,

Timeline

1981

"GPU": IBM PC CGA
Features: 640x200 two colours, 16KB memory, ISA interface

1992

OpenGL 1.0

1995

DirectX 1
GPU: S3 VIRGE
Features: 800x600, 24-bit, 2MB VRAM, PCI interface, uses the S3D library

1996

DirectX 3
GPU: 3dfx Voodoo
Features: 640x480, 16-bit, 50MHz, 4MB VRAM, PCI, uses a cut-down version of OpenGL called Glide

2006

DirectX 10
GPU: Radeon X1950 XT
Features: Unified Shader Model 4, 1,625MHz, 256MB GDDR3, PCIe 1 x16

2008

OpenGL 3.0 (GLShaderL 1.3)
OpenCL 1.0

2009

DirectX 11
GPU: GeForce 285
Features: Unified Shader Model 5, Compute Shaders, 1,648MHz, 1GB GDDR3, PCIe 2 x16

2010

OpenGL 4.0 (with GLShaderL 4.0), OpenCL 1.1

but still just from a customisable 64-colour palette. Moving into the late 80s, SVGA and XGA appeared. These extended resolutions to 800x600 and 1,024x768 in 256 (eight-bit) colour, with 640x480 offering 65,536 (16-bit) colour.

During this time systems such as the Amiga had appeared. These offered accelerated 2D graphics and hardware sprites that not only improved the speed of new GUI windowing systems, but powered a new generation of flashy colourful 16-bit games, too.

The PC, with its ever-improving x86 CPUs, marched into the 486 era. It would eventually overcome any shortcoming in power, though at a much higher price point. In 1995 the S3 Virge 2D/3D PCI card pointed the way the world would go. It offered 3D acceleration, a custom DOS s3d library and support for around 20 games at the time. Despite good sales, bugs and poor performance left the market wide open for a rival.

A door that 3dfx and its ex-Silicon Graphics employees were happy to step through in late 1996 with its Voodoo Graphic 3D accelerator card. Using an optimised OpenGL 1.1 library called Glide it provided a simple-to-use gaming API and hardware that delivered flashy accelerated 3D at 640x480 in full 16-bit colour at (for the time) high frame rates. The Voodoo range blew all existing cards out of the water, though 3dfx would go on to see huge competition as DirectX established itself and enabled other manufacturers to offer cards that could accelerate an ever-growing PC gaming line-up.

From this point graphics card development went in tandem with DirectX releases, ensuring that new developments would be supported from the day of release. While this was largely hardware agnostic, occasionally a DirectX release would offer a specific feature only available from AMD or Nvidia, that is until the other could update its range.

While we talk of DirectX, of course OpenGL remained and due to its history initially supported many hardware features DirectX didn't, such as Transform and Lighting. Moving towards the present day, all the baggage that OpenGL still had attached from the past saw the move to release Vulkan. This was a performant, low-level API that while initially envisioned for Android provides equal advantages for PC gaming. Now with Proton, the start of an open source Nvidia driver and Wayland largely working, graphics on Linux systems have a much brighter and flashier future. LXF

The introduction of GPGPU shaders revolutionised the GPU market.

1997

DirectX 5
GPU: Nvidia Riva 128
Features: 800x600, 16-bit, 100MHz, 4MB SGRAM, AGP 1x

1999

OpenGL 1.2
DirectX 7
GPU: Nvidia Geforce 256
Features: Transform & Lighting, Cube mapping, 120MHz, 32MB DDR, AGP 4x

2001

OpenGL 1.3
DirectX 8
GPU: Nvidia Geforce 3
Features: Pixel and Vertex Shaders v1, 200MHz, 64MB DDR, AGP 4x

2002

DirectX 9
GPU: Radeon 9700 Pro
Features: Pixel and Vertex Shaders v2, 1323MHz, 128MB DDR, AGP 8x

2004

OpenGL 2.0 (with GLShaderL 1.1)
DirectX 9c
GPU: GeForce 6800 GT
Features: Unified Shader Model v3, GPGPU introduced, 350MHz, 256MB GDDR3, AGP 8

2015

DirectX 12,
Vulkan 1.0, OpenCL 2.1
GPU: Radeon R9 Fury
Features: Unified Shader Model 6, 1GHz, 4GB HBM, PCIe 3 x16

2018

OpenGL 4.6
DirectX 12 ray tracing
GPU: GeForce RTX 2080
Features: RTX raytracing, 1.5GHz, 8GB GDDR6, PCIe 3 x16

2020

Vulkan 1.2, OpenCL 3.0
GPU: Radeon RX 6800
Features: Ray Accelerators, 1.7GHz, 16GB GDDR6, PCIe 4 x16

2023+

Vulkan 1.3, DirectX 13
GPU: Nvidia RTX 4000
Speculation: 2.5GHz, 24GB GDDR6, PCIe 5 x16, 450W
GPU: AMD RDNA 3
Speculation: 2.9GHz, 16GB GDDR6, PCIe 5 x16, 450W

Raspberry Pi Pico W

Solid hardware, good software and Wi-Fi access see this incremental update elevate the Pico to a truly versatile dev board – just how **Les Pounder** likes it!

With a surprise June 2022 release, we see a new Raspberry Pi Pico W – correcting one of the original's omissions – this $6 microcontroller looks much like its predecessor, but under a tiny silver case lies a Wi-Fi chip that takes the Pico into the world of IoT. It may not be the first RP2040 board to offer Wi-Fi – but we're sure that the Raspberry Pi Pico W is going to go straight on to our list of best RP2040 boards.

There isn't really much difference between the original Pi Pico and the new Pico W. There's the same GPIO, microUSB port, dimensions and SoC. The only difference is the inclusion of Infineon's CYW43439 2.4GHz Wi-Fi chip. Looking at Infineon's datasheet for this chip, we spotted that it's also capable of transmitting Bluetooth 5.2, but in the Pi documentation there's no mention of this. Eben Upton told us that, as yet, Bluetooth isn't enabled in the Pico W's firmware. How long until some enterprising hackers enable Bluetooth on the Pico W for themselves? We shall wait and see.

Getting online

If you have ever used an ESP8266, ESP32 or other MicroPython-compatible Wi-Fi dev board, then you'll notice that the Pico W is exactly the same. We followed the example script to connect to our Wi-Fi network and noticed that it was stock MicroPython. All it took were five lines of MicroPython and our Raspberry Pi Pico W was connected to the Internet.

We then tested our connection using *uPing* from Shawwwn, which mimics the ping command. We confirmed that we had a connection to the wider world, and this really opens up the possibilities of the Pico W.

Taking the test further, we used *urequests*, a network request module, to download data on air-raid siren status in Ukraine. We got the data as JSON, saved it into a dictionary. We then used the Ukrainian city, Kharkiv as the key to search the dictionary for its status. This status was then displayed in the Python REPL, and via a strip of NeoPixel LEDs.

Why is this important? Well, now we have a $6 IoT board from Raspberry Pi Ltd that can use a plethora of sensors and then send the data over a reliable network, where it can be collated and worked on. We no longer have the overhead of a full Linux OS to deal with. All of this is done on a $6 microcontroller instead of a $10 to $15 single board computer.

» WI-FI DEV BOARDS

In the time that the Pico's been with us, there've been a few notable attempts to add Wi-Fi connectivity to Raspberry Pi's RP2040 CPU. The first was via Adafruit's Airlift Featherwing Co-Processor. We managed to connect to and use this board with a Raspberry Pi Pico. A more refined alternative came in the form of Pimoroni's Pico Wireless, which came as a pack that slotted over the entire GPIO. While easy to use, we had no GPIO to play with.

Until now, the most obvious choice was Arduino's Nano RP2040 Connect, which retails for $28, over four times the cost of the Pico W. Arduino's board comes with Wi-Fi and Bluetooth, an IMU, microphone and a cryptographic coprocessor which does offset the higher cost.

If you need professional features, by all means go for the Arduino Nano RP2040 Connect, but most folks will probably find the Pico W a better value. Some makers may now be thinking that you can buy an ESP32, ESP8266 or W600, say, for around the same price. And they'd be right. This author is also part of those communities and shares a love for those boards, one of which powers the decorations at Christmas.

Les Pounder works with groups such as the Raspberry Pi Foundation to help boost people's maker skills.

» DOING MORE WITH LESS

Ten years ago, the Raspberry Pi burst on to the scene and offered low-cost computing for everyone. Educators lapped up the Pi as a means to get kids coding, creating electronics projects while reusing old equipment. Makers gorged themselves on the cheap and accessible hardware. Industrial and embedded users took a more cautious approach, but ultimately the Pi made quite an impact in that sector.

Ten years later, the Raspberry Pi sells approximately the same percentage of units (50/50 split) to the industrial and embedded sector. Raspberry Pi Ltd makes no secret of this, but we've noticed two new Compute Modules made exclusively for the industrial market too.

The Raspberry Pi Compute Module 4S is most of the Raspberry Pi 4 SoC, but housed in the older SO-DIMM connector for legacy compatibility with carrier boards designed around older Compute Modules. The CM4S is there to help customers upgrade or transition to a newer and more powerful Pi, but everyday customers can't order their own units, sadly. The same is true of the Compute Module 3E, a mysterious module that again uses the SO-DIMM form factor. This time the SoC is based around the Raspberry Pi Zero 2 W, roughly the same specification as a Raspberry Pi 3B.

Will these new Pi variants ever make it for general sale? Not likely. But chip shortages can't stop the Raspberry Pi Foundation innovating, and as you can see from our review of a refreshed Pi Pico W model, helping us all keep on doing more with less!

The bare base Pico W is just 51mm long – that's just six barleycorns in the newly introduced UK Imperial measures!

The ease with which we got online was astounding. Even a beginner would be able to write and understand the five lines of code needed to make the connection. The nearest Raspberry Pi we can compare to the Raspberry Pi Pico W is the original Raspberry Pi Zero W and its newer incarnation the Raspberry Pi Zero 2 W.

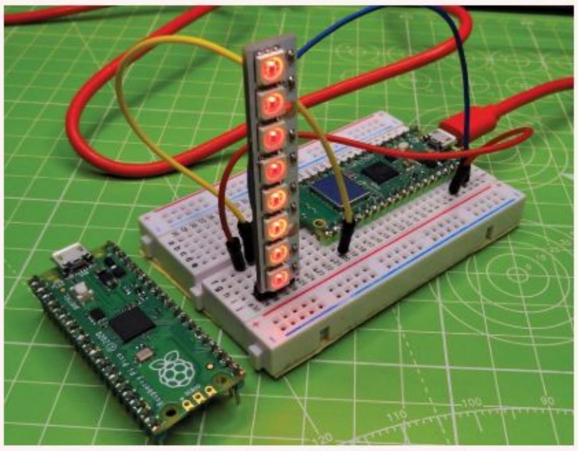
Computationally, the Pico W is slow, even compared to the Zero W. However, if we don't need the horsepower and "bloat" of a full Linux OS, or if your project doesn't need a camera, then the Pico W makes sense. It's cheap, low power and can be used with many of the sensors and inputs used on the Zero W.

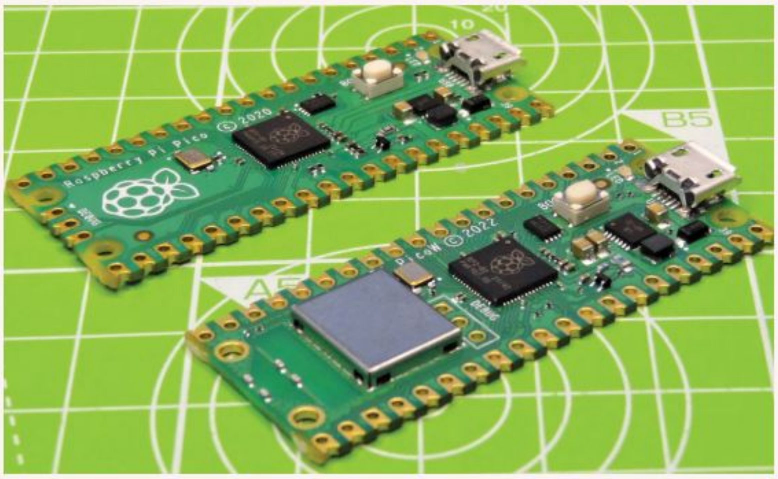
Project use

The Raspberry Pi Pico W comes just like any other Pico – with no header pins. This means that we need to break out the soldering iron and solder up the 40 GPIO pins ready for the breadboard. We did just that using our Pinecil soldering iron and after connecting via the micro USB port (we still wish that the Pico came with a USB-C port), we then hooked up a strip of WS2812B (NeoPixels) for a project. Using a community-created MicroPython library we soon had those RGB LEDs colour changing and pulsing with lights.

For basic electronics and breadboarding projects, the Raspberry Pi Pico W behaves exactly like its predecessor. We wrote a few lines to 'blink' the LED on and off to prove that our hardware was working. The Pico W GPIO pinout is the same, so all you need to do is flash your code on to the Pico W, replace the older Pico and away you go. We also tested I2C connectivity using an I2C HD44780 16 x 2 LCD screen and it worked without problems.

The Raspberry Pi Pico introduced analog inputs to a Raspberry Pi board, and the Pico W retains those same three analog inputs. We mocked up a quick demo using a 10K Ohm potentiometer (a variable resistor with resistance between 0 and 10kilo ohm) and were able to see raw ADC values scrolling along the REPL.


This is what happens to lovely hardware that's been left alone with Les.


The Pico W launch is just an elaborate spot-the-difference game from the Pi Foundation.

Software support

The Raspberry Pi Pico may have been released with support for just MicroPython and C++, but in a very short amount of time we saw support for many alternative programming languages. First off there was CircuitPython, a fork of MicroPython under the stewardship of Adafruit and Scott Shawcroft.

CircuitPython is the pinnacle of easy-to-use programming languages and it's become our preferred means to code for the Pico. At the time of writing there's no version for the Raspberry Pi Pico W, but we believe that will change once the board is officially released.

The Raspberry Pi Pico W does support C/C++, but we've been unable to test this, given time constraints. While C/C++ does offer better performance, many users will favour MicroPython/CircuitPython for its ease of use. The version of MicroPython provided on our review model was "MicroPython v1.18-673-gdf8d97171 on 2022-06-24", a fairly recent release that supports the Pico W hardware. MicroPython on the Pico W worked well with *Thonny*, the chosen editor for Pico MicroPython development.

The addition of Wi-Fi will see the Raspberry Pi Pico W powering many projects that ultimately would have been powered by the $10 Raspberry Pi Zero W. The Pico W will be an even more compelling board if and when someone enables its Bluetooth 5.2 capability. With Bluetooth, one could turn the Pico W into a wireless computer peripheral such as a game controller or mouse. LXF

VERDICT

DEVELOPER: Raspberry Pi Foundation
WEB: www.raspberrypi.com/products/raspberry-pi-pico
PRICE: $6

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

The Raspberry Pi Pico W is the ideal starter microcontroller. While $6 doesn't buy you much in the supermarkets, purchasing the $6 Raspberry Pi Pico W gives you the world.

» **Rating 9/10**

SPECS

SoC: RP2040 Arm Cortex M0+ Dual Core at 133MHz
Memory: 264KB SRAM,
SSD: 2MB Flash
Comms: Infineon CYW43439 2.4GHz Wi-Fi with onboard antenna
GPIO: 40-Pin GPIO 26x Multi-function pins 23x Digital I/O 3x Analog
Inputs: 2x I2C, 2x SPI, 2x UART, 1x Arm Serial Wire Debug (SWD)
Power: Micro USB for data and power
Size: 51x21mm

ULTRASONIC

Adding distance sensors to projects

Les Pounder merges two low-cost, yet complex ultrasonic sensors to create one must-have Pi-powered gadget that anyone can use.

OUR EXPERT

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

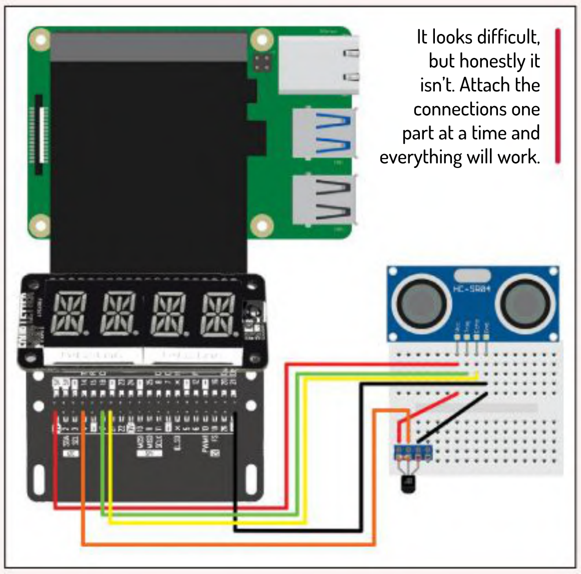
YOU NEED

- **Pimoroni 4 Letter pHAT**
- **HC-SR04P**
- **DS18B20**
- **4.7K Ohm resistor**
- **Breadboard**
- **Four-pin screw terminal**
- **Pimoroni Hacker HAT**
- **5x M2F jumper wires**
- **2x M2M jumper wires**
- **Code: https://github.com/lesp/LXF-Four-Letter-Sensors/archive/refs/heads/main.zip**

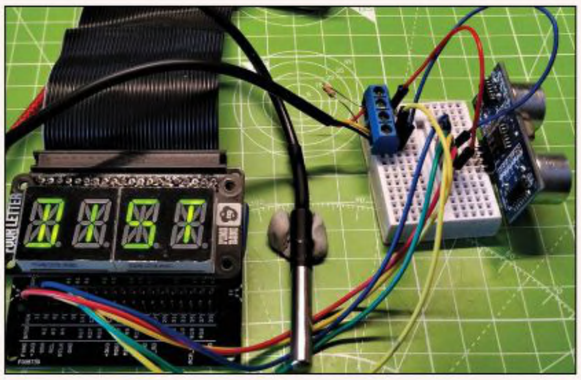
The Raspberry Pi is blessed with a plethora of sensors, but there are two that sum up how easy it is to work with them on the Pi. Ultrasonic sensors measure distances using a pulse of sound. A little science and maths can give us fairly accurate distances, enabling our robots to "see" the world, and providing contactless user interfaces for projects. Temperature sensors, such as the DHT11, DHT22 and the DS18B20 are easy-to-use instruments for gauging temperature and humidity. The DS18B20 also has a waterproof variant that can be submerged in liquids (what's the decay heat of a cup of tea, say?).

These sensors can be used to log data to graphs, databases or even used as inputs for a game. But in this tutorial we want quick hits of data, and for that we need a simple output device. This is where the trusty seven-segment display comes into its own. Pimoroni's Four Letter pHAT is not a new HAT for rude words (*I'm disappointed – Ed*), rather we have four seven-segment LED displays that can be used to show alphanumeric characters. In this tutorial we'll display the distance from an object, and the current temperature.

This is a multi-part build process and so we need to break it down into stages. The first is connecting our Hacker HAT to the Raspberry Pi. The ribbon cable



It looks difficult, but honestly it isn't. Attach the connections one part at a time and everything will work.



Underneath the wires is a custom Python-powered temperature monitor and distance calculator.

connects to all 40 GPIO pins, and extends over the HDMI ports. It then connects to the Hacker HAT without twisting or folding. Note that we're using an older Mini Black HAT Hacker, but it's the same as the HAT Hacker HAT. On one set of Hacker HAT pins, firmly push your Four Letter pHAT.

We'll use the remaining header to connect our HC-SR04 ultrasonic sensor and DS18B20 temperature sensor. Push the HC-SR04 into a breadboard and, using a male to female jumper wire, connect VCC of the sensor to 3v3 of the Pi. Connect GND in the same way to any GND pin on the Pi. Connect the sensor pins Echo to GPIO17 and Trigger to GPIO27.

On the other side of the breadboard press in a screw terminal block. From the same breadboard row as VCC on the HCSR04, use a male-to-male jumper wire to provide 3v3 to one screw terminal. Use another jumper wire to run GND from the HCSR04 to the opposite end of the screw terminal.

If you're using a waterproof DS18B20, strip the wires to show approximately 5mm of bare wire. Twist each wire into a neat end. Insert the red wire into the screw terminal for 3v3, while the black goes to GND. The yellow (data) wire goes to a spare screw terminal, but insert one leg of the 4.7K Ohm resistor into the yellow screw terminal. The other leg goes into the red (3v3) screw terminal. Finally, run a female-to-male jumper wire from the data screw terminal to GPIO4 of the Raspberry Pi. See the diagram (*left*) for more details.

Now you're ready to connect up your peripherals and power up the Raspberry Pi to the desktop.

Software setup

We need to install two pieces of software for this project. The first is for Pimoroni's Four Letter pHAT, which has its own installation script. Open a terminal and use this command to download and run the installation script. When prompted to install the examples, select Yes:

```
$ curl https://get.pimoroni.com/fourletterphat | bash
```

To install the software for the DS18B20 we need to use the Python package manager, *pip*. In the terminal use this command:

```
$ sudo pip3 install w1thermsensor
```

Next we need to set up I2C and the 1-wire interface for the DS18B20. In the terminal type the following:

```
$ sudo raspi-config
```

Scroll down to Interface Options and press Enter. Scroll down to I2C and press Enter. Select Yes and press Enter to enable the I2C interface. When prompted press Enter to go back to the main screen. Go to the Interface Options, but this time select 1-Wire and press Enter. Select Yes and press Enter. Reboot when prompted.

The project code

The next step is to open the *Thonny* editor, found under the Programming menu, to start writing Python code. We start the code with four imports. These are modules of pre-written code that contain functionality to use the ultrasonic sensor (`DistanceSensor`), DS18B20 (`W1ThermSensor`), `fourletterphat` and to add pauses to the code (`sleep`)

```
from gpiozero import DistanceSensor
from w1thermsensor import W1ThermSensor
import fourletterphat
from time import sleep
```

Create two objects, `sensor` and `ultra` . These objects create connections to the sensors that we wish to use. Note that for the HC-SR04 we pass two values: `17` and `27` . These are the echo and trigger pins of the sensor.

```
sensor = W1ThermSensor()
ultra = DistanceSensor(17,27)
```

Create a loop to contain the main body of the code. Note that any code within the loop will be indented.

```
while True:
```

To store the HCSR04 distance measurement create an object: `distance`. In there we oddly store the distance, which is set to metres. We can format the output to centimetres. We also apply rounding, to one decimal place, and convert the returned float value into a string.

```
    distance = str(round(ultra.distance * 100,1))
```

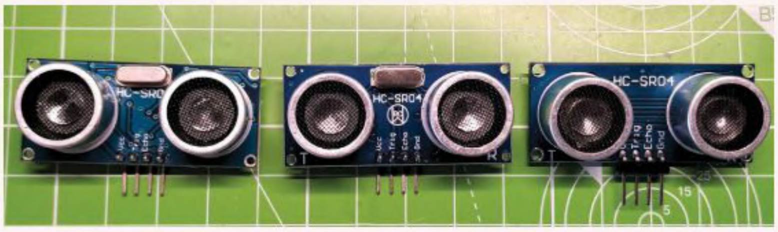
The next three lines will print `DIST` to the Four Letter pHAT, and wait for two seconds.

```
   fourletterphat.print_str("DIST")
   fourletterphat.show()
   sleep(2)
```

A quick print function shows the distance in the Python shell, which is useful for debugging.

```
   print('Distance to nearest object is', distance, 'CM')
```

The next four lines will clear the Four Letter pHAT display, show the distance across all the displays, and then pause for one second.


(From left to right) HC-SR04 (5V), HC-SR04+ (3&5V), HC-SR04P (3&5V). If you're going to buy one, buy the + or P as they also work on both the Pi and Arduino.

```
   fourletterphat.clear()
   fourletterphat.print_str(distance)
   fourletterphat.show()
   sleep(1)
```

Now we move into the temperature sensor portion of the code, and we print `TEMP` to the Four Letter pHAT to show that we're now using the DS18B20 sensor.

```
   fourletterphat.print_str("TEMP")
   fourletterphat.show()
   sleep(2)
```

Store the temperature as a string inside the variable `temp` and then print the temperature to the Python shell for debug.

```
   temp = str(sensor.get_temperature())
   print('The Temperature is', temp, 'C')
```

Lastly, we clear the Four Letter pHAT display, show the temperature using the four displays and then pause for one second. The code will now loop back to the start and take another distance reading.

```
   fourletterphat.clear()
   fourletterphat.print_str(temp)
   fourletterphat.show()
   sleep(1)
```

Save the code as **ultrameasure.py** and then click the Run button to start the project. Place something in front of the sensor to change the distance reading, and to change the temperature simply put the sensor in your hand. LXF

QUICK TIP

The DS18B20 uses a 4.7K Ohm resistor from 3v3 to the data pin to "pull up" the pin. This maintains a constant high state for the pin and means the 1-wire serial interface is fed a constant stream of data.

» SENSORS WORKING OVERTIME

The two sensors used in this tutorial, the ultrasonic distance sensor HC-SR04 and the DS18B20, are two common, cheap sensors. The HC-SR04 comes in at least three versions. The stock HC-SR04 is a 5V-only component that can't be directly used with the Raspberry Pi's 3V GPIO. To use this version we need to create a resistor voltage divided between the Pi and the Echo pin. Otherwise, we risk sending 5V to the GPIO, which can damage the Pi. The safer option is to use the HC-SR04+ or HC-SR04P, which look almost identical to the 5V version. See the author's blog post on how to spot them: **https://bigl.es/tooling-tuesday-hcsr04p-ultrasonic-sensor**.

The DS18B20 is a great temperature sensor. We sourced ours from the CamJam Kit 2. This is a kit produced between Cambridge Raspberry Jam and The Pi Hut as a way to get sensors and inputs in the hands of young makers. Those kits are great value for money. Our DS18B20 is waterproof, protected by lots of insulation and this means it can be dipped into water. At a Picademy in 2016 we used one to change a *Minecraft* world based upon the temperature. Cold changed the world to snow, while hot created a desert.

WLED

Credit: https://kno.wled.ge

Build light shows with multi-coloured LEDs

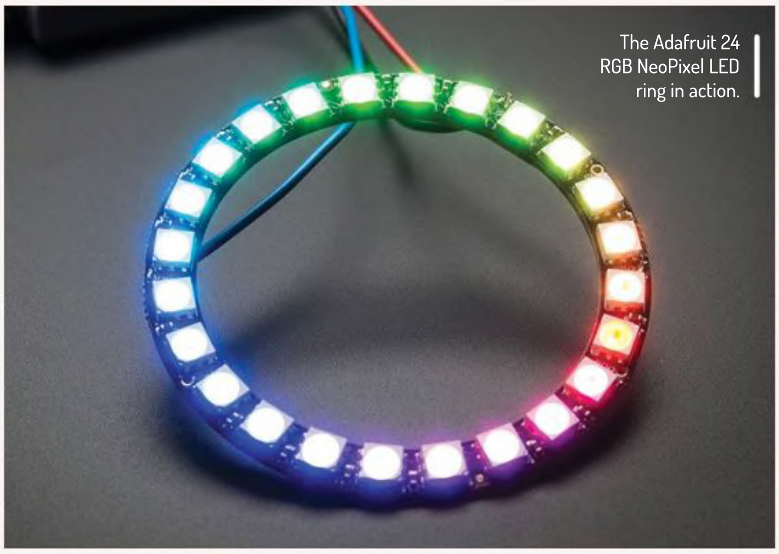
Use WLED firmware and a string of LEDs to create eye-catching light shows to display around your home, with **Matthew Holder**.

OUR EXPERT

Matt Holder has been a fan of the open source methodology for over two decades and uses Linux and other tools where possible.

YOU NEED

- **1x ESP8266 WeMos D1 mini or ESP32 WeMos LOLIN32 lite**
- **1x Panel mount 10K linear potent-iometer**
- **1x LED Strip ie WS2812B 5050 RGB 30+ LEDs**
- **1x Momentary surface mount switch**
- **1x 1K 1/4W resistor**



The Adafruit 24 RGB NeoPixel LED ring in action.

CREDIT: Adafruit

This month we'll be looking at the *WLED* firmware and what it's used for. We'll be building a project that features LEDs being controlled via a web interface and home automation software. We'll also discover ways to make the lights react to music.

This type of LED are commonly known as NeoPixels. They're strung together and are usually multi-coloured and can be addressed individually. Versions of the LEDs are available that take different input voltages and they can all be controlled in the same way. NeoPixel is the name given to this style of LEDs by Adafruit (that offers a wide--range of models), but the same driver is used for a large number of them (WS2811/WS2812), whoever the manufacturer or supplier happens to be. Models are available with separate RGB channels, so that any colour can be made by mixing the values (think of HTML colours). Furthermore, some have a W channel as well, which is an LED that outputs white light.

WLED is open source firmware, which can be installed on ESP8266 and ESP32 microcontroller boards. The ESP8266 is the older board of the two and has lower processor power. The ESP8266 is powerful enough to run the firmware and a smaller number of LEDs. When considering a larger number of LEDs, it's a good idea to spend a little extra money on an ESP32 board. When installed, configuration takes place using an integrated web interface, integration with *Home Assistant*, or the lights can be controlled with software so that they react to music being played on a PC.

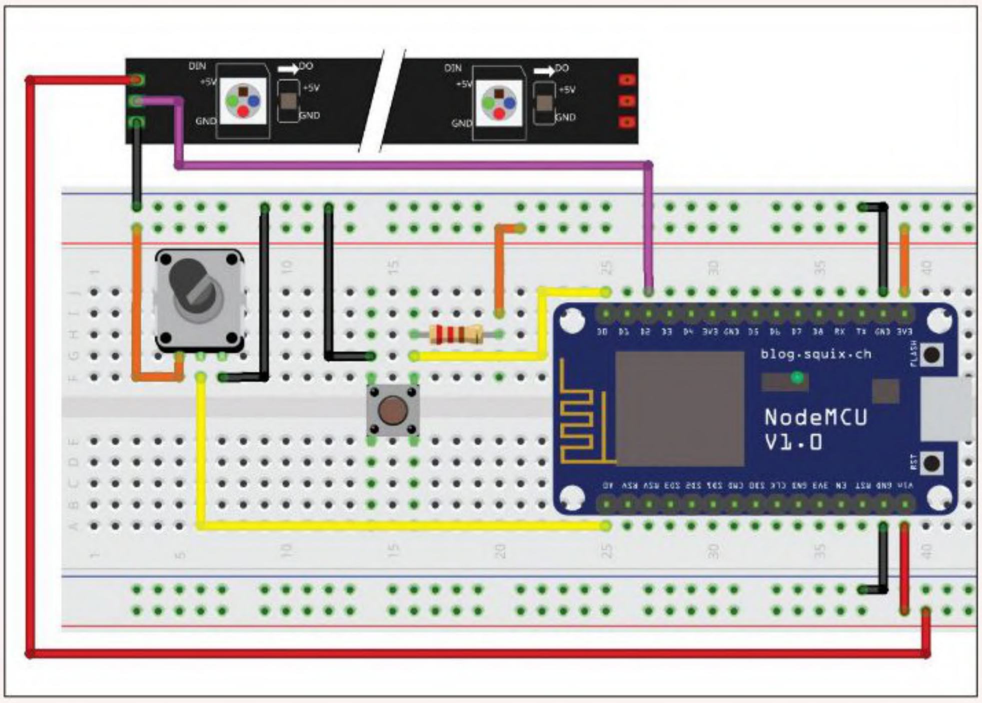
In this tutorial, we'll install *WLED* on an ESP8266, NodeMCU board, investigate the web interface, briefly talk about the *Home Assistant* integration and then describe methods of enabling the lights to be reactive to music. Method one is to use an alternative version of *WLED*, while method two is to use another piece of software that communicates with *WLED* and takes over the light control.

First of all, *WLED* needs to be installed. This is a trivial task, if using a recent version of *Google Chrome*. First, plug the microcontroller board into a device running Linux. Open a terminal and enter `dmesg -w`. This may need to be prefixed with the `sudo` command to run as an elevated command. Towards the bottom of the output, there should be an entry referring to a new serial port, which will have the designation of **/dev/ttyUSB0** (the USB number may very well vary).

Next, the currently logged-in user will need to be given permission to write to the serial port. Adding the user to the dial-out group and logging out and in again should fix this issue (`sudo usermod -a -G dialout USERNAME`). A more temporary solution is to enable any user to access the port. This can be accomplished by running `sudo chmod o+rw /dev/ttyUSBX`, where X is changed to the number identified earlier.

Once permission has been granted, visit **https://install.wled.me**. Select the version to install (at the time of writing, version 0.13.1 was the most recent) and then click the Install button. Select the relevant `ttyUSB` device and click Connect. Once connected, select Install WLED. Click Install one final time and the firmware will be installed on the ESP board. Once the firmware has been installed, click the Next button and enter your network details. If successful, your ESP device will now be connected to your home network. If this step fails, then follow the steps in the next paragraph.

If the network configuration fails, connect your Linux device to the WLED-AP wireless network with the password of **wled1234**. Next, use a browser to connect to **http://4.3.2.1** and use the Settings Cog to change the network settings to connect to your home Wi-Fi network and set a static IP address if required. The device should now restart to apply the new settings, and it can be connected to using an IP address in your home network's default range.



Circuit diagram for the connection of a strip of LEDs and input devices to control their operation.

Hardware configuration

With the software installed we can move on to configuring the hardware. Unplug the microcontroller and gather together your breadboard, power supply, resister and LEDs. It's unlikely that the ESP device can power the LEDs directly, so a connection will need to be made directly to the power supply. The supplier of the LEDs should be able to advise on a suitable power supply unit. The NeoPixel LEDs have three or four connections. One is 5V, the second will be 0V and the third will be a DataIn connection. If your LEDs have a fourth connection, this will be DataOut and is used to chain together multiple strips of LEDs. The DataOut of one set connects to the DataIn of the second set. Follow the diagram (*above*) to connect the LEDs and microcontroller to the power supply and the DataIn pin of the strip to the pin labelled D2. This diagram is also referred to later in the tutorial when input devices are added. It's important to note down which GPIO pin is used so that this can be selected from within the *WLED* web interface.

The circuit can either be powered directly from the Micro-USB socket on the board if the strip of LEDs is very small, or use the rails at the bottom of the breadboard to connect the 0V and 5V supply lines (note that you may need to connect 5V to the LEDs rather than using the breadboard, should they draw a large amount of current and need a larger gauge of cable).

Once connected, switch on the newly connected circuit and after a minute or so, navigate to the web interface from within a web browser. At this stage don't be alarmed if nothing happens. Within *WLED* select the Config page again and navigate to the LED preferences. From within here the pin can be selected that's used to communicate with the strip of lights. This pin is the GPIO number, so look this up by searching for the board's pin out. In this case, D2 corresponds to GPIO 4.

Make a note of the option relating to colour order. Should the LED colours not match with what's selected, the order can be changed from RGB to RBG and so on.

QUICK TIP

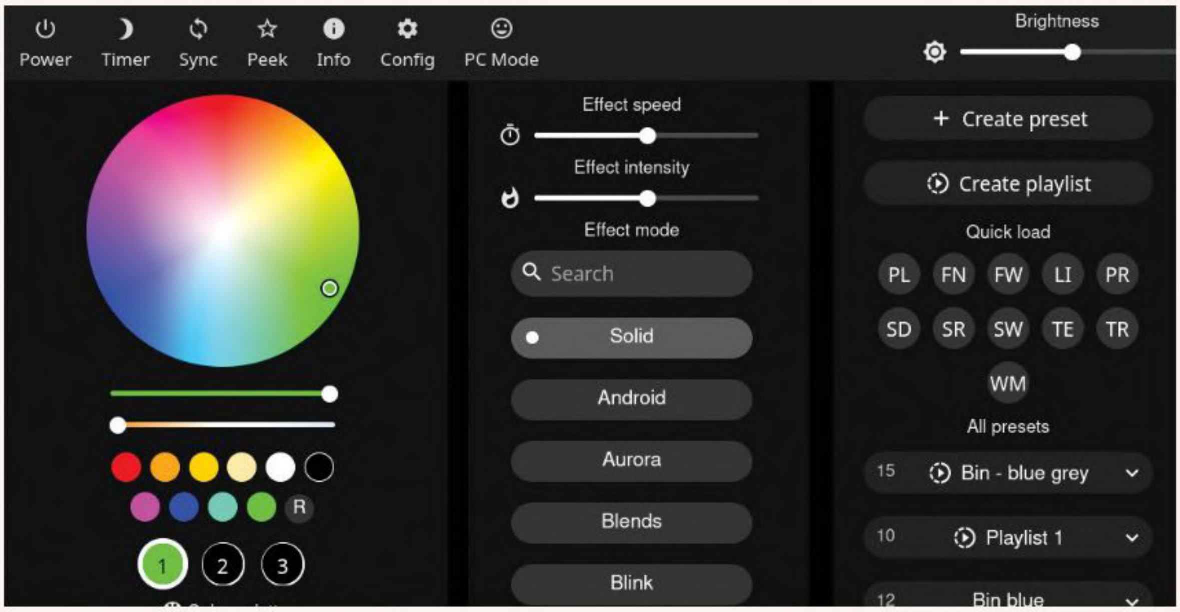
Useful WLED documentation can be found at https://kno.wled.ge.

» MICROCONTROLLERS AND BREADBOARD BASICS

Microcontroller boards are devices that run with relatively low power requirements, but have a relatively generous amount of processing power and RAM. Firmware is loaded on to the devices and they run with this single purpose, rather than general-purpose PCs/Raspberry Pi devices, which run an operating system and then software on top of it. ESP devices have enough resources to run a basic web server, interact with General Purpose Input Output (GPIO) pins and interact with the Wi-Fi networks to report states from the GPIO pins to various services.

Many types of firmware exist, including *MicroPython*, a version of the Python programming language; *CircuitPython*, which is a version of *MicroPython* with support for more hardware integrations; *Tasmota*, which provides a web interface for configuration; *ESPHome* which enables configuration files to be written in YAML; and, of course, *WLED*.

When selecting hardware to pair with a microcontroller it's important to research the voltage requirements for peripherals. Selecting 5V when 3.3V is required or vice versa is a recipe for disaster. At the top and bottom of the breadboard are lines that are marked in red or blue. These lines can be used to carry 3.3V from the microcontroller along with the 0V signal. Convention would be to use the red line for a positive voltage and a blue line for 0V. Between the sets of horizontal lines are a series of vertical lines. The vertical lines are used to build our circuit.



The WLED interface enables you to customise the behaviour of the LEDs.

QUICK TIP

LEDfx can be used to visualise music with a strip of LEDs. More details at www.ledfx.app.

Then enter the number of LEDs on the strip. Save the settings and let the board restart. Reload the *WLED* GUI and select the power button. The LEDs should then light up. Select a different colour and the colour of the LEDs will change accordingly. On the bottom of the interface, select Effects and explore the different options. Within the Presets tab, numbered presets can be chosen based on different brightnesses, colours and effects. These can then be added into a playlist.

Save time with presets

To create a preset, first set the LEDs to display the colour and effect required, as well as the speed and brightness. Now add a new preset from the Presets tab and select the option to store the current values. Finally, save the options and repeat the operation for each preset. To create a playlist of presets, create a new one from the same tab and select which preset numbers you want to include, before saving the preset. Within the Settings page these presets and playlists can be scheduled to play on different days and times of day, using the Time and Macros options.

Another feature is the Segments option. For example, 300 LEDs can be split into two groups of 150 and the effect selected can be reversed on one of the segments. This means that an effect can start in the middle of a strip and travel out to each end, before "bouncing" at the end and travelling back to the centre again.

WLED also has built-in support for using the GPIO pins to make changes to the effects being displayed. For example, a passive infrared sensor could be used to brighten the lights when motion is detected; a button press can change to a new playlist of effects; or a potentiometer can be used to change the brightness or colour range of the LEDs.

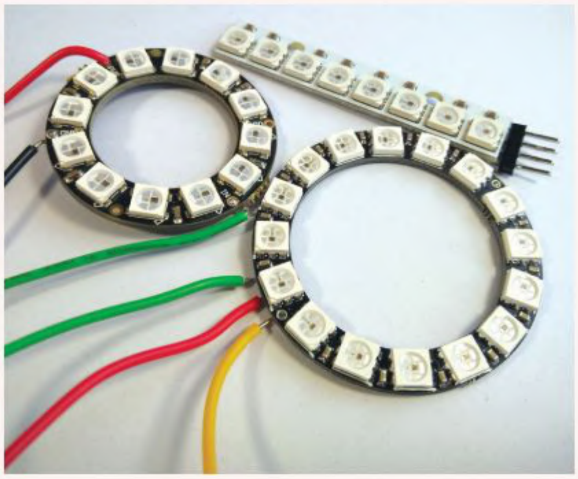
Disconnect your ESP board from the PSU and make changes to the circuitry on your breadboard (*see the earlier diagram*). Connect one leg of a resistor to 3.3V and the other leg to one of the connections on the switch you're using. Now take a cable and connect from the pin that's just been connected and add it to D0 on the microcontroller board. The second pin of the switch should be connected to 0V. Now, when the switch is

» LEDS AT SCALE

While this tutorial demonstrates how *WLED* and the relevant circuitry connects together with a small bank of LEDs, this can be much larger. During the autumn months of 2021, this author installed a 5m reel of LEDs (60 per metre) on to the front of his house and connected them to an ESP32 board and an ATX power supply to provide the power to the LEDs. This is a permanent installation, which is mainly used to provide a light display during the winter months.

Considerations made on a project of this size were that the ATX power supply could supply a high-enough wattage to the LEDs when they're all fully illuminated. Other considerations were to ensure that the cabling was thick enough to support the current being drawn, while a 12V model was used to keep the current in the cable lower than if a 5V model had been used.

ATX power supplies have a standby 5V line, which was used to power the microcontroller. One of the GPIO pins was used to connect to the relevant pin on the PSU, which turns the power supply's main output on when the LEDs are switched on from within *WLED*. This is an important consideration for energy saving when the LEDs aren't being used.



A selection of NeoPixel devices including a stick of eight RGB LEDs from Adafruit.

pressed the microcontroller will detect a change in voltage from 0V to 3.3V. In addition, connect a potentiometer's left-most pin to 3.3V, the right-most pin to 0V and the middle pin to an analogue input on the board (A0 on the NodeMCU or Wemos D1 mini).

The board can now be connected again and the pins configured within the web GUI. Step one is to navigate to Settings>LED Preferences. Navigate to the Colour Order Override section and then below this select pin 16 with the option of Pushbutton and for the second button, select the relevant analogue pin number before selecting the option of analogue. If you're using a NodeMCU/Wemos D1 Mini or other ESP8266 board with a single analogue input, simply select a GPIO number that isn't being used and *WLED* will automatically select the analogue input.

Step two is to navigate to the Time and Macros options and for each pin and select what changes should be made (*see top of page 47*). When selecting the action to apply, there are three different options: one for a short press, one for long press and one for a double press (if supported by the board being used). These three options best apply for buttons. When using the analogue control a number need to be entered into the double text box (the easiest option to enter is 250; then changing the position of the potentiometer will change the brightness of the LEDs). For other options see **https://kno.wled.ge/features/macros/#analog-button**. When referencing switches the numbers entered in the three boxes correspond to the numbers for each defined playlist or preset.

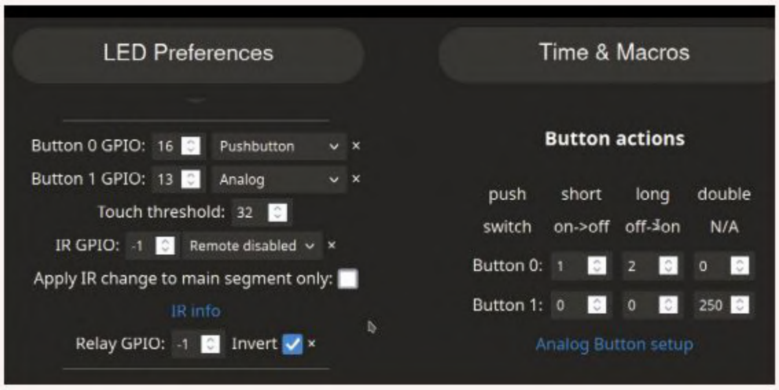
Once the components have been connected and the settings selected, test the button and potentiometer to ensure that everything works. Note that the potentiometer control will switch the LEDs off when the position is set to fully on and fully off. This can be changed by adding a 1k resistor between the left-most pin and 0V, and the right-most pin and 3.3V.

Vary your input sources

An integration option for *Home Assistant* enables lights to be controlled using data from many different sources. For example, the colour can be set based on external temperature. For example, the colder the temperature, the bluer the lights should be.

To install the integration from within *Home Assistant*, (*see* **LXF287**) navigate to the Settings>Devices & Services and select the Add Integration button. Search for *WLED* and add it. Configure the integration accordingly and then select the Device link, which will show in the integration's tile. Within this tile, there will be options to switch on the lights, set colours and brightness or set presets and playlists. These settings can all be configured using automations, scripts or both.

The first method – reacting to music – will use a piece of software called *LedFx*. This is open source and cross platform. Once installed and running, the audio output of the device being used will be monitored and effects then generated, which are sent to *WLED* as E1.31 commands. Further information about *LedFx* can be found at **www.ledfx.app**. The documentation is extensive and covers usage on multiple operating systems. This documentation is important because options may need to be enabled within your operating system, so that the audio output can be monitored,


LED Preferences

Button 0 GPIO: 16 Pushbutton ×
Button 1 GPIO: 13 Analog ×
Touch threshold: 32
IR GPIO: -1 Remote disabled ×
Apply IR change to main segment only: ☐
IR info
Relay GPIO: -1 Invert ✓ ×

Time & Macros

Button actions

	push	short	long	double
	switch	on->off	off->on	N/A
Button 0:		1	2	0
Button 1:		0	0	250

Analog Button setup

This screenshot demonstrates how to configure the GPIO to control presets.

which *LedFx* will then utilise. Within the Config>Sync options of *WLED*, various protocols can be enabled or disabled to support external operation.

The second option for reaction to music is to use an alternative version of *WLED*. This has built-in support for a microphone to be added to the circuitry and when sound is detected, the LEDs will be switched on accordingly. The team behind the project recommend the ESP32 because of its enhanced processing power for the real-time processing of the audio being input to the board. See the circuit diagram (*below*) detailing the addition of a microphone to the microcontroller board. In this example, the NodeMCU board is being used, but only an older version of the firmware is supported.

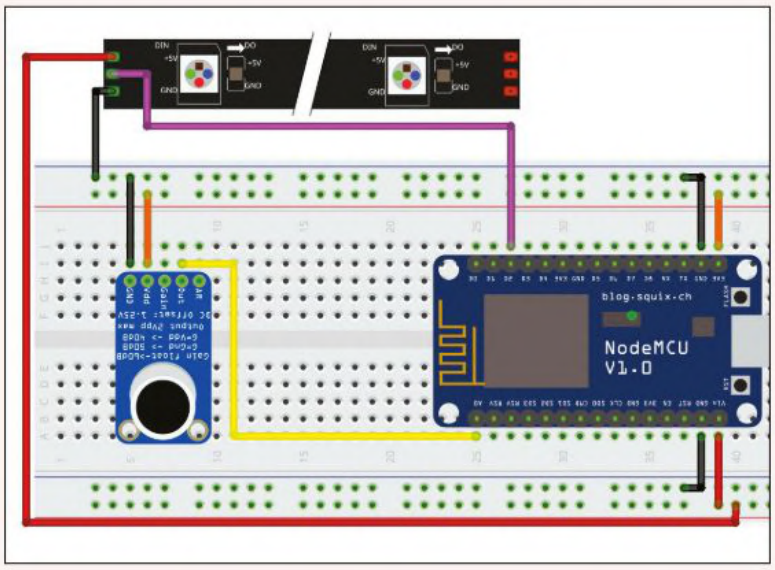
The firmware can be downloaded from **https://github.com/atuline/WLED/wiki** and can be installed on the board using the method described above and selecting the sound reactive firmware version from the pull-down menu. Once installed, configure the LEDs in the same way as earlier and select the pin that the microphone is connected to within the settings and then select the effects beginning with a *. These are the sound reactive effects. The LEDs will then react to sound captured by the microphone.

As can be seen, what seems at first glance to be a simple-to-use interface is actually incredibly powerful. Using a combination of *Home Automation* control, presets and playlists, incredibly complicated light shows can be created. Enjoy experimenting! LXF

QUICK TIP

A useful button at the top of the screen is Peek – this is used to display a representation of the chosen effect on the web GUI.

This circuit diagram details how to connect a microphone to the microcontroller, so that sound activates the lights.



Pi PICO

Build a "rubber ducky" device from a Pi Pico

Discover why the classic fake keyboard hacking device is ideal for the Raspberry Pi Pico and **Matthew Holder's** very particular set of skills...

OUR EXPERT
Matt Holder has worked in IT support for over a decade. As a long-time open source fan, he's always tried to utilise Linux alongside other systems.

Let's have some fun with the Raspberry Pi Pico. We're going to try a couple of basic projects and we'll discuss how to install different types of firmware on the board, write a program to flash an LED and then discuss the configuration required to set up a board as a so-called rubber ducky device. We'll configure our rubber ducky to enter simple keypresses.

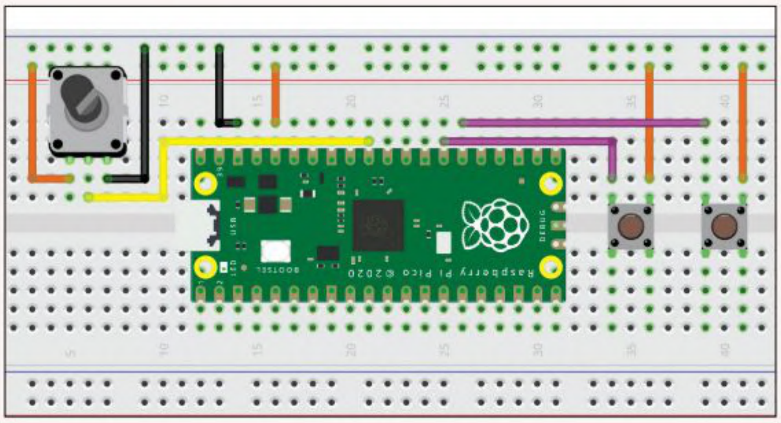
During this tutorial, we will discuss how to install different types of firmware on the board. Examples of firmware are micropython, CircuitPython, C/C++ and the Arduino language and programming tools. Following firmware installation, we'll write a program to flash an LED and then discuss the configuration required to set up our board as a rubber-ducky, style of device. In this example, we will configure the rubber-ducky to enter simple keypresses. With a little more imagination, it would also be possible to configure a device to operate in a similar manner to the streamdeck shortcut devices.

Before beginning with the projects, we'll install *CircuitPython* on the board and install the *mu* editor on the device we'll be programming from. Open a terminal on the device used to create the program and enter `sudo apt install mu-editor` on an Ubuntu-based distribution and `sudo dnf install mu` for Fedora.

To install *CircuitPython* on the board head to **https://circuitpython.org/board/raspberry_pi_pico** and download the latest version. Hold down the BootSel button on the Pi Pico and connect to USB. The uf2 file that you recently downloaded can then be copied to the drive that's mounted. When completed, the device will restart and a new drive will be shown, which is where the **code.py** file can be copied to for the device to run.

To be able to write our programs, open *mu*. On first load this will ask which mode it should operate in. Select CircuitPython and if all is well the text editor will finish opening with a text editor screen in the middle. This is where the Python code will be entered. Also note the button to open a terminal to the device, which is where Python can be entered manually and where programs can be interacted with. The Serial button on the toolbar provides access to the board's REPL, where the board and program can be interacted with directly.

Here's the circuit diagram for both projects. See the left-hand side for the LED project and right hand side for the rubber ducky.



Flashing an LED

This is the Hello World of hardware programming. We'll add a small twist to the usual program and use a potentiometer to control the speed of which the LED flashes. We'll also discuss the usage of Pulse Width Modulation (PWM) to control the brightness of the LED.

To create this functionality requires a small amount of code. First, we must import the libraries we wish to use and define the output pin for LED control, with a type of PWM and an input pin on one of the analogue compatible pins for the potentiometer to connect to. Next we define a while loop, which runs forever and sets the **duty_cycle** of the LED (the ratio of On to Off, which controls the brightness) to the value as read from the analogue input. The **duty_cycle** value needs to be in the range of 0-65,535. This is also the value read in from the analogue port. If we wanted to see the voltage of the pin, rather than a number, we divide 65,535 by the value read from the input and then multiply by 3.3V. Turning the potentiometer will now change the LED's brightness.

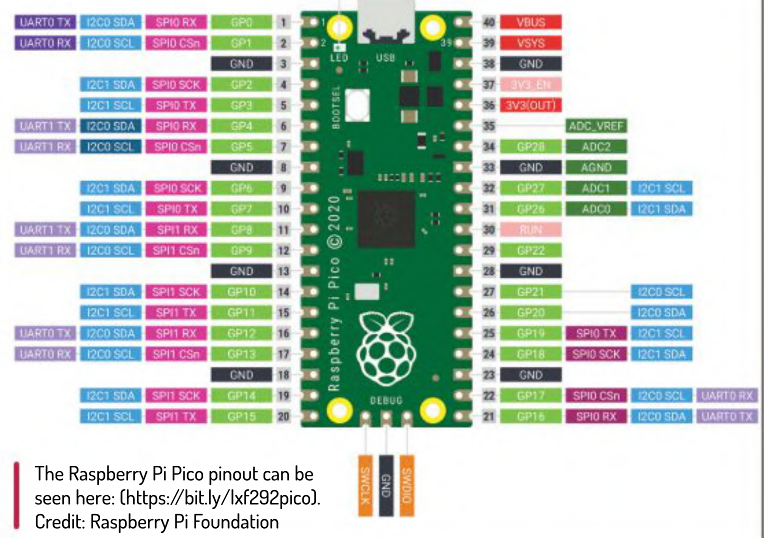
Enter the code below into the text editor and save to the RPi Pico's USB drive as **code.py**. The program will then execute with the functionality described above.

```
#Use a potentiometer to control the brightness of the
built-in LED
import board
import analogio
import pwmio

led = pwmio.PWMOut(board.LED, frequency=1000)
potentiometer = analogio.AnalogIn(board.GP26)
```

```
while True:
  led.duty_cycle = potentiometer.value
```

Second, we'll use the potentiometer to control the rate that the LED flashes. Here, we define the pin to use for the LED and set the pin to be the type of output. Again, this uses the LED on the board. We define a variable to represent the analogue input, which is linked to GPIO pin 26. Finally, we define a while loop, which runs forever. To start, it reads in the analogue port and divide the value by 65,535. This gives a value between 0 and 1. Next, the LED is switched on and then the code is set to wait for the value calculated by reading the value of the input pin and dividing by 65,535. We then switch the LED off before waiting again. Turning the potentiometer changes the frequency of the LEDs flashes. As before, enter this code into the text editor, replacing what was there before and save as **code.py**.

```
#Use a potentiometer to control the frequency that the
LED flashes
import time
import board
import digitalio
import analogio
# Pulse an LED with frequency controlled by a
potentiometer
led = digitalio.DigitalInOut(board.LED)
led.direction = digitalio.Direction.OUTPUT
potentiometer = analogio.AnalogIn(board.GP26)

while True:
  flashFreq = float(potentiometer.value) / 65535
  led.value = True
  time.sleep(flashFreq)
  led.value = False
```



The Raspberry Pi Pico pinout can be seen here: (https://bit.ly/lxf292pico). Credit: Raspberry Pi Foundation

```
  time.sleep(flashFreq)
```

One of the interesting design choices of the Pico board is the addition of two programmable I/O blocks. Each block can run four-state machines and can be used to offload some program functionality from the main CPU cores. Functionality can be written for the PIO slots using *MicroPython*. An example of this, which blinks an LED can be seen at **https://docs.micropython.org/en/latest/rp2/tutorial/pio.html**. This doesn't look like usual Python code and this is because the PIO instruction set consists of only nine commands.

We've merely scratched the surface of what this inexpensive, yet powerful microcontroller board can do. Enjoy experimenting! LXF

QUICK TIP

A more feature packed rubber ducky project can be found at https://github.com/dbisu/pico-ducky

» CREATING A SIMPLE RUBBER DUCKY

A rubber ducky is a device that imitates a keyboard. In this example, we'll connect two switches to the Pico device and program one to imitate the Super-Key and the other to imitate the Print Screen key. First we add the USB Human Interface Device (HID) libraries to the Pico. Download the library files (**https://circuitpython.org/libraries**), extract the archive and copy the **adafruit_hid** folder to the **lib** folder on the Pico.

First we import the libraries for digital IO and USB HID functionality. Second, we define a variable to represent the keyboard we will emulate. Next, the buttons and LED are initialised as three of the GPIO pins. Now we start an infinite while loop and read the state of the buttons. If the pin associated with button 1 goes high (showing the button has been pressed) we send the Super-Key scan code. Similarly, the pin linked to button 2 is scanned and when this goes high, the scan code for the print screen key is pressed. Test this by pressing the buttons and noting what happens. See the right-hand side of the circuit diagram (*left*) to view the wiring diagram for this project.

```
import usb_hid
from adafruit_hid.keyboard import
Keyboard
from adafruit_hid.keycode import
Keycode
import time
import digitalio
import board

kbd = Keyboard(usb_hid.devices)
button1 = digitalio.DigitalInOut(board.
GP20)
button1.direction = digitalio.Direction.
INPUT
button1.pull = digitalio.Pull.DOWN
button2 = digitalio.DigitalInOut(board.
GP21)
button2.direction = digitalio.Direction.
INPUT
button2.pull = digitalio.Pull.DOWN
led = digitalio.DigitalInOut(board.LED)
led.direction = digitalio.Direction.
OUTPUT

def flashLed():
  led.value = True
  time.sleep(0.1)
  led.value = False

while True:
  if button1.value == True:
    kbd.press(Keycode.WINDOWS)
    kbd.release_all()
    print('Button1')
    flashLed()
  if button2.value == True:
    kbd.press(Keycode.PRINT_SCREEN)
    kbd.release_all()
    print('Button2')
    flashLed()
  time.sleep(0.20)
```

LOCAL LINUX FOR LOCAL USERS

You're not from around here, says **Aaron Peters** but that's fine! As he's here to explain how your distro of choice gets to work in any language.

Linux is the result of the work of thousands of developers worldwide. It's only right that it has support for the languages all those volunteers speak, to say nothing of its users. The multi-lingual question is how does open source software support not just translations for all the different languages of the world, but its different and varied alphabets alongside notations. How you represent decimal places or thousandth separators, for instance?

Translation work is a vital aspect of enabling the take-up of open source software and Linux-based distros around the world. While English comprehension is often a perceived requirement for dev work, the truth is much of the world doesn't speak English, let alone read it. So knowing how FOSS handles translations, more often called localisation, is key to growing wider adoption. Being a good open source citizen means being supportive of all these brethren, so let's explore the localisation features of Linux!

The first of the interconnected components is character encoding. Since at the end of it all computers only understand zeros and ones, there needs to be a convention for some number of these zeros and ones to be converted to more meaningful characters. The character encoding is this convention.

Standards, standards, standards

The ASCII standard is one of the earliest standards. In ASCII every seven bits represents a single character, or more correctly, a keypress. If you examine the characters in this encoding scheme, you'll see common keystrokes such as Return (or Line Feed, 10 in ASCII) and Escape (27 in ASCII) alongside characters like uppercase M (77), lowercase g (103) and the closing brace: }, or 125 in ASCII.

So how does this pertain to localising your Linux system? Well, consider that ASCII only contains the lower- and uppercase for the 26 "standard" Roman letters, along with some specific punctuation. But you may speak a language that uses diacritics, such as Spanish. This in turn requires more entries in the standard to represent the letter 'a' both with and without its accent marks (á, à and a, respectively).

This resulted in a large number of standards as part of the IEC/ISO 8859 (commonly written ISO-8859) covering languages primarily in Europe. Some examples are as follows:

- ISO-8859-1 covers the majority of romance languages, Scandinavian languages and Celtic tongues, as well as languages in Africa and Asia.
- ISO-8859-2 supports Eastern European languages based on the Latin alphabet, such as Polish, Czech and Hungarian.
- ISO-8859-5 includes languages using the Cyrillic alphabet – Russian and Bulgarian, for example.

ISO-8859-11 contains the characters needed for Thai.

Add to these a number of legacy schemes created by technology companies like the CP standards of the IBM PC or Windows character sets (Windows-1250 through Windows-1258) and country-specific standards like KOI8-R (Russian), Shift-JIS (Japanese), or ISCII (Indic scripts), and there were at one time a large number of encodings that a system might encounter.

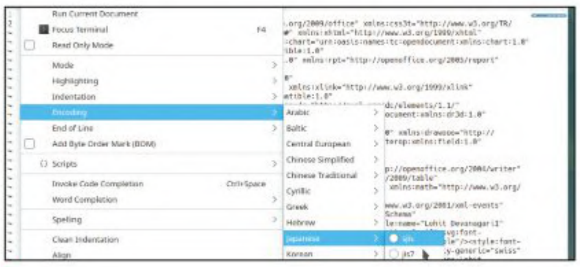
In 1991 the first version of a standard called Universal Coded Character Set, or Unicode, was published. One of the implementations of this standard, titled Unicode Transformation Format or UTF, uses between seven and 32-bits to identify over 144,000 characters, including letters, numerals, pictograph and emoji in its current iteration. Many operating systems, including major Linux distributions, have implemented UTF-8 as the default character encoding where applicable, or at least support it. However, it's still useful to be aware of character encoding.

Most modern applications either set the encoding created within them, or infer it from some sort of standard. Consider *LibreOffice*, for example. The character encoding is included in the standard XML encoding attribute of the **content.xml** file. Likewise, web browsers have become better at guessing the language of the pages they render. *Chrome* even removed the ability to manually select character encoding as of version 55, requiring users to install extensions instead.

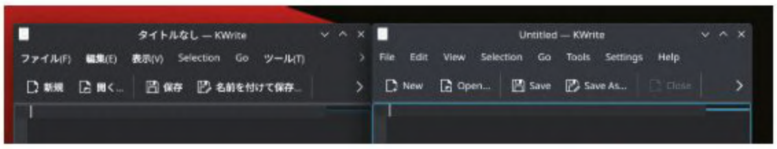
But when dealing with plain text files the encoding standard becomes ambiguous. Again, modern applications have become adept at guessing encoding formats. See the screenshot (*right*) showing two text editor windows: the one on the left shows a text file open with the wrong encoding, and the right shows it correctly. In many programs you can change the display of the characters in text files on the fly.

For example, in KDE's *Kate* text editor, the following steps are used to correct the display of Japanese text:

1 Open the Tools menu.
2 Select the Encoding item.



Showing character encoding in the Kate text editor.


タイトルなし — KWrite
ファイル(F) 編集(E) 表示(V) Selection Go ツール(T)
新規 開く... 保存 名前を付けて保存...
Untitled — KWrite
File Edit View Selection Go Tools Settings Help
New Open... Save Save As... Close

Overriding the locale for the KWrite text editor (left) versus opening normally (on the right).

3 Select the language of the file's text.
4 Select the correct encoding for that language.

However, the text encoding format isn't the only thing that properly localises Linux for its users. Read on for an explanation of locales...

Becoming a locale

Now that your computer can read and write data representing all the languages, let's explore displaying them. Linux uses the concept of locales to adjust the system to the user's native language. The standard notation to designate the value of a locale is as follows:

```
en_US.UTF-8
```

The first two letters indicate the language, while the second two after the underscore represent the region (`en` for English and `US` for the US, in this case). Appended is the character encoding that should be used by default. The `locale` command shows you all the sets currently available on the system (the following list from the author's system is abbreviated for space):

```
locale -a
C
C.UTF-8
en_AG
en_AG.utf8
en_AU.utf8
en_GB.utf8
en_HK.utf8
en_IE.utf8
en_US.utf8
en_ZA.utf8
ja_JP.utf8
POSIX
```

Locales are generated when the system is installed, and occasionally when new packages are installed. Once available, the locale values are assigned to different aspects of the system settings, which are stored as environment variables starting with LC_ except for one, because of course there's always an exception to the rule! The most important for the purposes of this article are as follows:

- `LANG:` Acts as a default when none of the other LC_{*} variables are set.
- `LC_ALL:` Acts as a global override, supersedes any of the below individual variables.

DOUBLE-DUTCH? DOUBLE-BYTE!

The Chinese, Japanese and Korean (CJK) languages include pictographic characters numbering in the tens of thousands. Storing these a digital format requires more than a single byte, because the eight bits can only represent up to 256 characters. As a result multi-byte encoding schemes were created. These included language-specific implementations such as Shift-JIS for Japanese and Big-5 for Chinese, as well as more recent, unified standards such as UTF-8 and UTF-16. These operate by mapping characters using two (or more) bytes. This is crucial when thinking about encoding standards. If your system can't auto-detect the encoding, it may interpret these as separate characters.

Even if you're not someone who interacts with CJK languages, being aware of multi-byte characters is still important. Newer updates to Unicode include entries such as emojis as part of the standard. So make sure to keep multi-byte characters in mind, lest your smiley face become a hospital emoji and cause a panic.

» **LC_CTYPE:** Describes the rules applications use for "character handling" operations, such as converting an uppercase letter to lowercase.

» **LC_MESSAGES:** Arguably the most important, this describes how system messages display.

You can explore the details of what's defined by a locale by looking through the files residing in **/usr/share/i18n/locales/**. Each one contains formatting rules such as ordering the parts of an address (LC_ADDRESS), the unit of money (LC_MONETARY), and the size of print-outs (LC_PAPER). See the box (*below*) for a more detailed explanation of the others.

The installer for a Linux distribution will normally set the language selected at install time. This default value is read at boot time from locations such as **/etc/default/locale** (Ubuntu) or **/etc/locale.conf** (Fedora). You can see the current value of system locale with the following command:

```
localectl status
 System Locale: LANG=en_US.UTF-8
   VC Keymap: n/a
   X11 Layout: us
   X11 Model: pc105
```

Naturally, this can be overridden by you in several ways. For instance, to activate a locale you want in Ubuntu, install its **language-pack** package, which will enable the locale as well as installing the translations (the example below is for Japanese):

```
$ sudo apt install language-pack-ja
```

You can also install the associated file for your desktop environment (**language-pack-gnome**, for example), or alternately add the language using your DE's tools. To set the default locale for the whole system, issue the following command. This will set the system to use Japanese instead of English:

```
$ sudo localectl set-locale ja_JP.UTF-8
```

You can also set locale variables individually with the `update-locale` command. The following will keep all the Japanese aspects of the set by the previous command except for units of measure, which will show in Imperial units rather than metric:

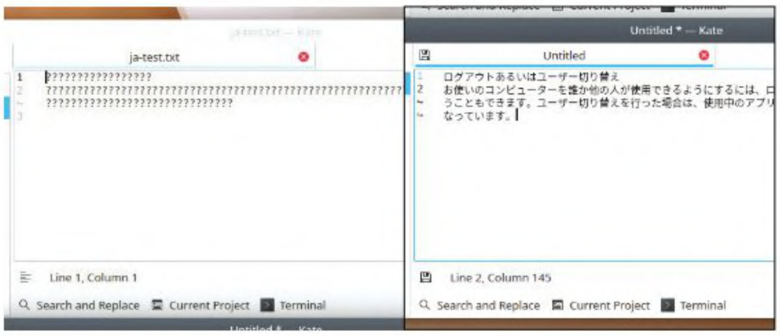
```
$ sudo update-locale LC_MEASUREMENT=en_US.UTF-8
```

But perhaps most interesting of all is the ability to set these as a modifier to a command. Prepending a command will change the locale values for that instance of the command only. Consider the screenshot below, which shows the windows launched by the following commands:

```
$ LANG=ja_JP.UTF-8 kwrite &
$ kwrite &
```

Desktop environments and window managers can customize the locale at the user level. The precise method varies, but your GUI probably contains a screen in its control center application to set these options. In KDE, for example, the Regional Settings > Language screen allows you to add new languages (see the screenshot to the right adding Japanese) as well as set your preference for them. Change the settings here by moving a different language to the top of the list and you should be greeted with a whole new experience the next time you log in.

Now that your machine understands other languages, and you're able to view them properly, all that's left is the ability to enter them. Read on for an explanation of keyboard layouts and input methods.



```
?????????????????
??????????????????????????????????????????????????????????????
??????????????????????????????
```

```
ログアウトあるいはユーザー切り替え
お使いのコンピューターを誰か他の人が使用できるようにするには、ロ
うこともできます。ユーザー切り替えを行った場合は、使用中のアプリ
なっています。
```

Opening a text file with the wrong (left) and correct (right) encoding set.

Linux Format Thai Edition?

Lets make an unfair assumption that you normally write in English, or at the very least, your computer has an English-based keyboard. Unfortunately, we can't cover every combination of "native" and "target" language in this article. So if you're a speaker of Urdu looking to write text in Thai, you'll have to adjust some of the steps below appropriately.

When it comes to actually writing in a given language within your applications, there's one main question: can the language be reasonably represented by Roman characters. "The language" in this context means the actual letters, for reasons that will become apparent shortly.

» LOCALE-RELATED ENVIRONMENT VARIABLES

As mentioned at the outset of this article, locale is more than just a fancy word for location. In tech terms it encompasses not only the base language, but some of the other aspects of the region's culture that have an impact on that language.

Exploring the locale files in **/usr/share/i18n/locales** will reveal a number of data points included, such as:

» **LC_ADDRESS:** the format for addresses, such as whether street and number or state/province comes first, ZIP/Post code formats, etc.

» **LC_COLLATE:** specifies the rules used for sorting.

» **LC_IDENTIFICATION:** contains metadata about the locale.

» **LC_MEASUREMENT:** lists whether the region makes use of metric or imperial (US, UK) values.

» **LC_MONETARY:** describes the prefix and format for expressing monetary values.

» **LC_NAME:** indicates, for example, whether given or surname should come first.

» **LC_NUMERIC:** this value dictates what punctuation should indicate the decimal, separate number groups, and how many places those groups should contains.

» **LC_PAPER:** paper size, in millimeters.

» **LC_TELEPHONE:** the format of international and domestic phone numbers, as well as key code numbers.

» **LC_TIME:** specifies the format of date and time (e.g. YYYY-MM-DD).

Remember that the language you select will have default settings for these, but you can always set these on an individual basis.

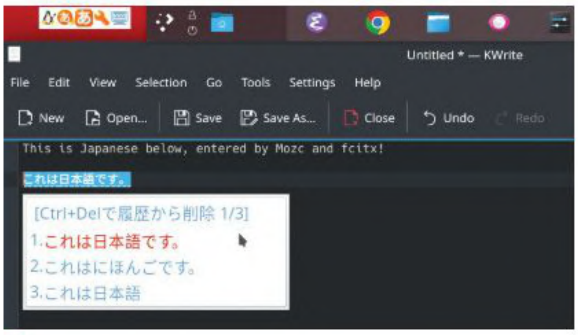
If it can, you can likely enter its text simply by setting a Compose key. Take Italian, for example. Being the homeland of the Romans, it follows that its language uses Roman letters. The main difference, as with many of the western European languages, is the use of the aforementioned diacritics. The Compose key enables you to say "I'm going to compose a character with the character I'm about to type, as well as another."

Start by defining which key on your keyboard will serve as the Compose key. Beware: this means you'll lose use of that key for other purposes. In KDE, you define the Compose key as part of the keyboard layout. On the Input Devices>Keyboard screen, under the Advanced tab, is an option titled "Position of the Compose key" with a number of options; the right-Alt key is a common choice.

Once you've saved your selection, entering the characters is easy once you get the hang of it. The process basically involves holding the Compose key, pressing the letter you want, and then pressing the punctuation mark that looks most like the accent mark. For example, to create the cedilla mark required for the word garçon, press Compose-c, then the comma key.

Keyboard solution

If you work in more alphabetically complex languages, you may need to go a little further. For some languages (such as those with Cyrillic alphabets) it's enough to activate alternate keyboard layouts. In these cases you'll use the keyboard to type the language's letters, though the Roman values you're used to will be replaced by localised ones. If you have a machine from the country in question, it's likely these alternate characters are printed on the key caps. But even if they're not, you can still use the keyboard layout. You'll just need to remember which key is which.

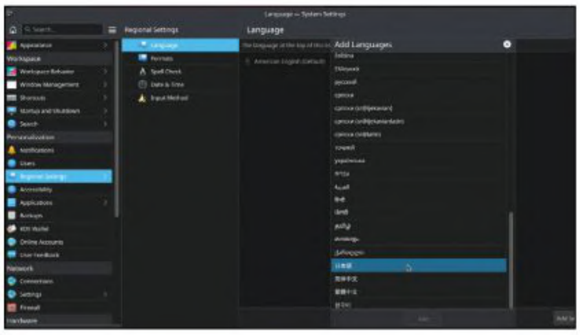


This is Japanese below, entered by Mozc and fcitx!

これは日本語です。

[Ctrl+Delで履歴から削除 1/3]
1.これは日本語です。
2.これはにほんごです。
3.これは日本語

Entering Japanese text with the Mozc IME.



Adding a language in the KDE System Settings application.

Finally, for the languages that use pictographs, in particular Chinese, Japanese and Korean (CJK), you'll need an Input Method Editor, aka IME as we all love a TLA! This is a system that takes a pronunciation you provide, and displays the various characters that could represent it. Again, there are multiple systems available in each of these languages, so we'll demonstrate the process using the combination with which the author is most familiar: using the Mozc IME through the *fcitx* framework to enter Japanese.

First, install *fcitx*. If you're using a desktop environment, it's good to add the integration package for that as well. On Ubuntu, type:

```
$ sudo apt install fcitx-mozc kde-config-fcitx
```

On your next login into your system, the *Systems Setting* tool will have a new option under the Regional Settings menu titled Input Method. Its Input Method tab contains a list of all available input methods, which should now include Mozc under the Japanese section. Click the right arrow to move it to the Current Input Method column.

Next, in the Global Config tab, select a new hotkey to switch between your input methods (the author favours Meta-Space for this). Apply your settings, and on recent versions of Ubuntu, you should be good to go. The only caveat is switching your input method requires that you're in a position to input text. In other words, that you're in a field on a web page or a text editor. But when you are, hit your hotkey, or alternately click the keyboard switcher on your DE's system tray, and you'll activate your IME. In the case of Mozc, you'll be presented with a small, floating window similar to what you see in the screenshot (*left*).

Now, whether you speak English as a first language, are learning Brazilian Portuguese, or have family in Taipei, you know how to get your Linux system set up. So get to it, and get your polyglot on! LXF

» INPUT METHOD EDITORS

Using an IME takes some getting used to, but once you understand what it's doing, it becomes second nature. While each one caters to the unique needs of the language it supports, understanding one can help get you started with the one you need for your locale.

Typically, IMEs are servers that sit behind the scenes. To expand on the example in this article, the Mozc IME has includes packages containing the service, data, and front-end support. These clients of Mozc will send it data, which it evaluates and sends back matching text. In the case of Japanese, you're sending a representation of the pronunciation of the word(s) you want, often termed as "romanization" of the text. Japanese has a relatively small number of syllabic sounds, so it's easy to support their romanization. Mozc sends back a list of matching words written in hiragana (basic Japanese letters), katakana (an alternate alphabet reserved for borrowed words), and Kanji (pictograph characters co-opted from Chinese).

While the input method is active, the Mozc palette appears, and as you type roman letters, potential matches appear in an overlay window. You use the Tab key (or arrows) to select the correct one, and then press Enter. This inserts the text into the active application.

Credit: https://cmus.github.io

Feature-rich music players for minimalists

Simplicity, elegance, usefulness and sheer fun are all qualities that describe **Shashank Sharma**. They're also features he values most in applications.

OUR EXPERT

Shashank Sharma is a trial lawyer in Delhi and an avid Arch user. He's always on the hunt for pocket friendly geeky memorabilia.

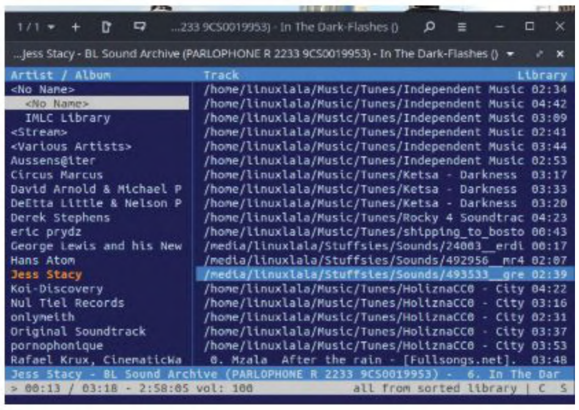
Music players come in all shapes, colours and sizes. Modern ones have a number of common features such as support for multitude of file formats, gapless playback, player queue, search, playlist filters, ability to play mp3 and ogg streams, and a customisable appearance. Released under the GPLv2 and written in C, *cmus* or *C* Music Player* is a nifty little CLI utility that supports all these features and more.

While most modern command-line applications make use of Python, Ruby, or Go, the choice of C for *cmus* should already give you an idea about the project's age. First released just a little over 17 years ago, *cmus* follows the old Unix philosophy of doing one thing well. That one thing being that it's the only music player you'll ever need.

One advantage of being such an old-school utility is that you'll find it in the software repositories of most modern desktop distributions, so installation is as easy as running either the `sudo apt install cmus` or `sudo dnf install cmus` command, depending on your choice of distribution. At the time of writing, the latest release was 2.9.1; however, with the exception of Arch and a few others, most other distributions carry an older version.

Getting started

To install the latest release, you'll have to manually run the classic trifecta of `./configure`, `make` and `make install`. Download the latest tarball from the project's GitHub page (**https://github.com/cmus/cmus**) and extract the files with the `tar xvf cmus-2.9.1.tar.gz` command. Depending on your distribution, it's possible that you'll have to install a number of additional packages for optimum results. You'll find a list of such optional dependencies on the project's website (**https://cmus.github.io/#development**), These dependencies can all be installed using the software repositories on your distribution.



```
1/1 ▾  +          ...233 9CS0019953) - In The Dark-Flashes ()
...Jess Stacy - BL Sound Archive (PARLOPHONE R 2233 9CS0019953) - In The Dark-Flashes ()
Artist / Album             Track                                                Library
<No Name>                  /home/linuxlala/Music/Tunes/Independent Music 02:34
  <No Name>                /home/linuxlala/Music/Tunes/Independent Music 04:42
  IMLC Library             /home/linuxlala/Music/Tunes/Independent Music 03:09
<Stream>                   /home/linuxlala/Music/Tunes/Independent Music 02:41
<Various Artists>          /home/linuxlala/Music/Tunes/Independent Music 03:44
Aussens@iter               /home/linuxlala/Music/Tunes/Independent Music 02:53
Circus Marcus              /home/linuxlala/Music/Tunes/Ketsa - Darkness  03:17
David Arnold & Michael P   /home/linuxlala/Music/Tunes/Ketsa - Darkness  03:33
DeEtta Little & Nelson P   /home/linuxlala/Music/Tunes/Ketsa - Darkness  03:20
Derek Stephens             /home/linuxlala/Music/Tunes/Rocky 4 Soundtrac 04:23
eric prydz                 /home/linuxlala/Music/Tunes/shipping_to_bosto 00:43
George Lewis and his New   /media/linuxlala/Stuffsies/Sounds/24003__erdi 00:17
Hans Atom                  /media/linuxlala/Stuffsies/Sounds/492956__mr4 02:07
Jess Stacy                 /media/linuxlala/Stuffsies/Sounds/493533__gre 02:39
Koi-Discovery              /home/linuxlala/Music/Tunes/HoliznaCC0 - City 04:22
Nul Tiel Records           /home/linuxlala/Music/Tunes/HoliznaCC0 - City 03:16
onlymeith                  /home/linuxlala/Music/Tunes/HoliznaCC0 - City 02:31
Original Soundtrack        /home/linuxlala/Music/Tunes/HoliznaCC0 - City 03:37
pornophonique              /home/linuxlala/Music/Tunes/HoliznaCC0 - City 03:53
Rafael Krux, CinematicWa    0. Mzala  After the rain - [Fullsongs.net].  03:48
Jess Stacy - BL Sound Archive (PARLOPHONE R 2233 9CS0019953) -  6. In The Dar
> 00:13 / 03:18 - 2:58:05 vol: 100                  all from sorted library | C  S
```

The bottom of the screen shows the total run time of all your added tracks. Search for a track by pressing /keyword from views 1 and 2.

» ENHANCING CMUS

The project's GitHub page hosts a wiki where you can find many third-party scripts and utilities that enhance *cmus's* functioning and behaviour.

For instance, if you enjoy singing along with the playing tracks but can't remember the exact lyrics, the *cmus-lyrics* tool might interest you. The installation requires downloading the zipped archive, extracting the *cmus-lyics* script, changing permissions and moving it to the **/usr/local/bin** directory. When you now run the `cmus-lyrics` command, it'll automatically fetch the lyrics for the currently playing track. It does so by reading the track title details from the metadata, so if the information is incomplete or missing, *cmus-lyrics* will be unable to fetch the lyrics. You only have to run `cmus-lyrics` once, and then so long as *cmus* is playing, the tool will keep fetching lyrics.

You'll be able to synchronize your *cmus* library with your Android phone with the *cmus-sync* tool, while *cmus-youtube* is a patch that you can apply before compiling *cmus* from source to be able to play tracks from Youtube.

Many of the third-party scripts haven't seen an update in years, but that's nothing to worry about. The basic functionality provided is such that most of these scripts just work.

With that done, change into the extracted **cmus** directory and run the `./configure` command. The generated output will reveal if any library or package is missing. You can alternatively open the generated **config.mk** file in your favourite text editor. Look for lines starting with `CONFIG_`. You'll have to rerun the `./configure` command every time you install any dependencies. Next, run the `make` and `make install` commands to finalise the installation.

Read up on the music player

To help you acclimatise to its way of doing things, *cmus* ships with a robust man page, which is a recommended read, and also a useful guide to help you make sense of its interface. Run the `man cmus-tutorial` command to launch the tutorial which walks you through the usage. It covers everything from launching *cmus*, adding music, managing playlists, finding tracks and more.

Because it's a command-line utility, you control the behaviour of the tool using defined keyboard shortcuts, so it makes sense to read the man page as well as run through the tutorial.

You can launch *cmus* by running the `cmus` command. You'll be greeted by a two-pane Artist/Album view. Because we've just launched the program, the view is understandably bereft of any information. *Cmus* has a built-in file browser that you can use to identify the songs you wish to listen to. Press 5 to switch to the file-browser view.

You can now use the arrow keys to select the song or directory you wish to add, and press 'a' to add it to the *cmus* library. Press Enter to navigate into a directory and the Backspace key to move up to the parent directory. If all your tracks are sorted neatly into directories, you can add the entire directory into the library by pressing 'a' after selecting the directory.

You can move back to the Artist/Album view by pressing 1. You'll find all the Albums are listed in alphabetical order on the left pane, while the tracks are listed on the right. The left pane, Artist/Album, is selected by default. Press Tab if you wish to navigate the tracks on the right pane. You can alternatively switch to the library view, which lists all the music files listed in alphabetical order, by pressing 2.

The *cmus* player utilises different views, such as Artist/Album, Library, Playlist, Play queue, File browser, Library Filter and Settings. You can switch between these different views by pressing the number keys 1 to 7.

To save your music library, move to any of views 1 to 4, type `:save` and press Enter. This will ensure that you don't have to keep adding your music files to the library every time you run *cmus*.

Let's play

Hit Enter to play the currently selected song from your library. The selected song will be highlighted and the currently playing song is displayed in the status bar at the bottom of the *cmus* interface.

There are a number of keys that control playback. You will find a complete list of keybindings by pressing 7, but the following are all you need to start enjoying your music collection:

Keyboard shortcut	Function
Enter	Play track
c	Pause or resume playback
z	Play previous track
b	Play next track
s	Toggle random playback
X	Restart the track
Right arrow	Seek forward by five seconds
Left arrow	Seek backwards by five seconds
,	Seek backwards by one minute
.	Seek forward by one minute
-	Reduce volume by 10%
=	Increase volume by 10%
D	Remove track from library

In addition to music files, you can also add playlists to your *cmus* library. Even better, you can add your favourite internet radio station as well, but only if it provides a mp3 or ogg stream. Such internet streams are sorted under the <stream> name on the Artist/Album view.

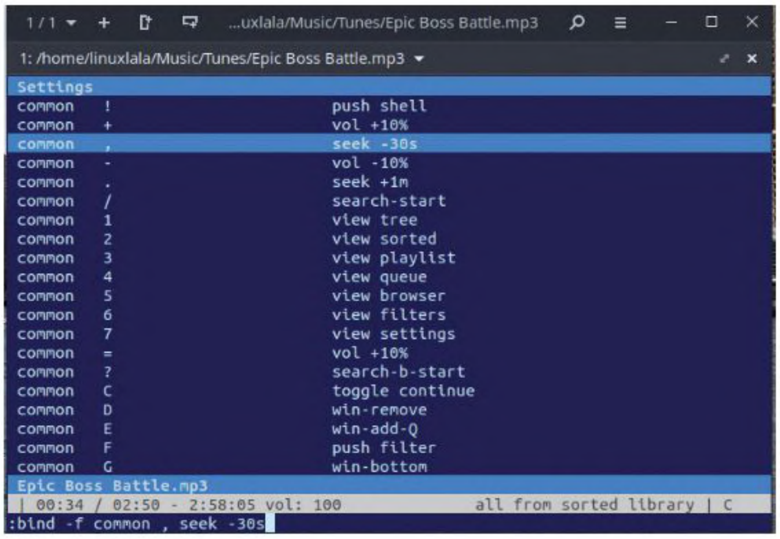
If you want to play certain tracks in sequence, you can select the song from the library view and press 'e'. Keep doing this for all the tracks you wish to play in the order, and these will be added to the player queue, which you can view by pressing 4. To rearrange the songs from the player queue, you can use the P or 'p' keys to move tracks up or down respectively.

While we've only talked of playing music with *cmus*, you can just as readily use the tool to listen to your favourite podcasts, or even audio books. The only downside being that the program won't remember the location in a given chapter.

If you care about speed then you'll enjoy working with *cmus,* which is an incredibly fast and robust music player. We were particularly impressed by the program's search feature, which returns results instantaneously. We recommend giving *cmus* a whirl today. LXF

QUICK TIP

The key bindings on the Settings view (7) are sorted into two categories: browser and common. The browser settings are ones that can only be used from the file browser view, while the others can be used from any of the different views. For instance, pressing 'i' in browser view will display hidden files.


1 / 1
...uxlala/Music/Tunes/Epic Boss Battle.mp3
1: /home/linuxlala/Music/Tunes/Epic Boss Battle.mp3

```
Settings
common   !                          push shell
common   +                          vol +10%
common   ,                          seek -30s
common   -                          vol -10%
common   .                          seek +1m
common   /                          search-start
common   1                          view tree
common   2                          view sorted
common   3                          view playlist
common   4                          view queue
common   5                          view browser
common   6                          view filters
common   7                          view settings
common   =                          vol +10%
common   ?                          search-b-start
common   C                          toggle continue
common   D                          win-remove
common   E                          win-add-Q
common   F                          push filter
common   G                          win-bottom
Epic Boss Battle.mp3
| 00:34 / 02:50 - 2:58:05 vol: 100            all from sorted library | C
:bind -f common , seek -30s
```

You can change cmus's behaviour by editing its settings. Pressing , (comma) seeks backwards by one minute. Select this setting, press Enter, and change -1m to -30s. Press Enter to save.

PHOTOFLARE

Credit: https://photoflare.io

Enhance your photos quickly and easily

Nick Peers tracks down an image editor that strikes the right balance of features and user-friendliness for making quick-fire edits.

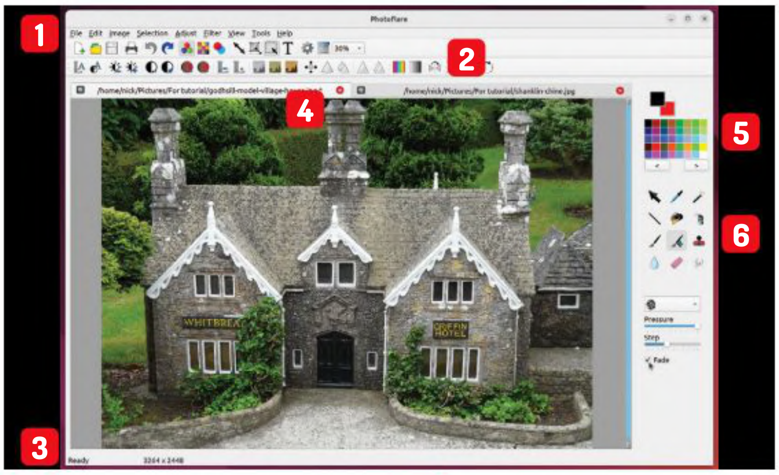
OUR EXPERT

Nick Peers went on holiday recently, giving him some new photos to use in his screengrabs.

Who needs another image editor? You've got *Shotwell* for image cataloguing (and the odd bit of editing) at one end of the scale, and at the other there's *GIMP* to rival the likes of *Photoshop* with its powerful array of tools, filters and plugins. But what about those everyday tasks that you'd like to be able to perform without getting bogged down in *GIMP's* multi-windowed (and occasionally confusing) user interface?

One image editor offering to sit in the gap between the two is *Photoflare* (**https://photoflare.io**). Based on a popular Windows image editor called *Photofiltre*, *Photoflare* combines user-friendliness and simplicity with a relatively powerful set of features that should cover most everyday needs. If it falls short in certain areas then you can, of course, switch back to *GIMP*, but otherwise you'll find that *Photoflare* is able to perform a wide range of image-editing tasks with ease.

NAVIGATING PHOTOFLARE



1 Main toolbar Provides shortcuts to common options, as well as useful tools like the zoom setting, batch processing tool and colour settings.

2 Filter bar Focuses on image correction tools from brightness and colour settings to a selection of soften and sharpening.

3 Status toolbar When performing edits on larger images, keep an eye on this bar. Once it shifts from Working... to Ready, your latest adjustment should be done.

4 Image tabs Makes it easy to work on more than one image at once by providing separate tabs for each. Click one to switch to it.

5 Tool palette Use the colour tools to set the foreground and background colours for your drawing tools, and switch between palettes.

6 Drawing tools Photoflare's drawing tools cover most bases, from various selection tools to Clone Stamp and Eraser. When selected, a tool may display its own options beneath the tool palette.

Getting Photoflare

The current version of *Photoflare* is the open-source Community Edition, which we'll be focusing on in this tutorial. Going forward, the developer is looking to focus more on a paid-for Studio version (*see the box over the page for details*), which might explain why development has slowed down. However, version 1.6.10 was released recently, and added a couple of minor new features – fine-grained image rotation and the ability to unlock the aspect ratio when resizing an image canvas – so it's not been abandoned yet by a long shot.

An outdated version of *Photoflare* is available for Ubuntu and Mint via the main Ubuntu repositories, so be sure to install it from its own repository:

```
$ sudo add-apt-repository ppa:photoflare/photoflare-stable
$ sudo apt update
$ sudo apt install photoflare
```

with a series of icons beneath the program's menus displayed over two levels, plus further tools in the right-hand pane. The annotation (*left*) provides a quick run-through of what goes where.

Your first image

Once installed, open *PhotoFlare* from the Launcher. It'll open to a blank screen, so first choose File>Open in the usual way to select your first image – *Photoflare* supports nine formats: JPG, PNG, GIF, TIF, BMP, ICO, PBM, PGM and PPM. You can open multiple images at once – each in its own tab within the main *Photoflare* window – by holding Shift as you click to the first and last image in the group you want to load.

Once opened, the first image will appear in the centre of the main window. By default, the image is opened quite small – you'll see a percentage drop-down appear on the main toolbar where you can choose a different level of magnification. Alternatively, use the View menu to zoom in, out or view the image at its original size.

The annotation reveals how the main interface works. *Photoflare* places its 'tool palette' (basically its drawing and colour tools) to the right of the main image pane. If this jars, move it to the left via Tools>Preferences>Layout. The tools work much as you'd expect in other applications – there's the Pointer tool, which you use to select part of your image by clicking and dragging around the area you want to grab. You'll also find the Eyedropper and Magic Wand tools for setting and selecting colour, respectively, along with a range of tools for drawing on or altering the image: two brushes, a Line tool, Fill tool (bucket), Spray Can, Clone Stamp, Blur, Eraser and Smudge.

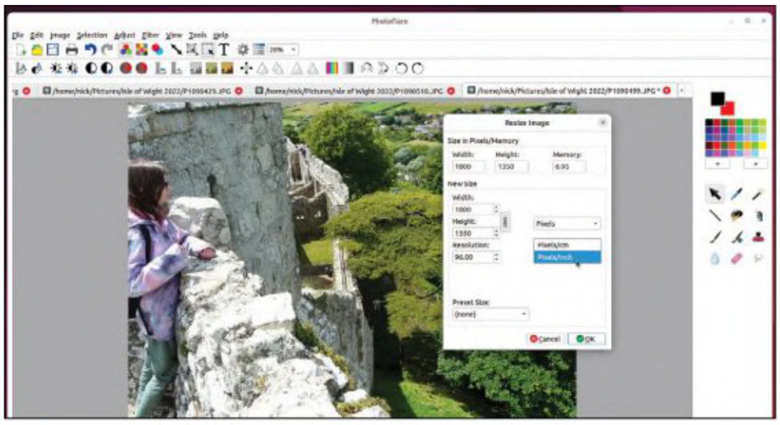
Select one and you may be shown additional options beneath it, such as a choice of brush types when choosing the Advanced Paint Brush tool. When configuring a tool using a number picker, you'll come across a strange restriction: *Photoflare* can't handle numbers greater than 99 (or 99.99). It's not a problem where the numbers are arbitrary (such as Brush Hardness), but it becomes an issue when working on larger images with pixel-based options such as Brush Size or Image Resolution. This may be one of those moments where you might have to switch to *GIMP* for certain functions.

Resizing images

The two toolbars at the top contain a range of tools that are replicated on the menus. If we start with the Image menu, you'll find its Transform sub-menu contains tools for flipping and rotating your image, and these can also be found on the Filter toolbar. On the other hand, the Image Size and Canvas Size options are replicated on the Main toolbar.

When resizing images, you'll be shown its current size in terms of both pixels (width and height) and memory (its size when uncompressed, so often it's larger than the file it's housed in). Beneath this are controls for altering the image's size. The Link button to the right of the Width and Height fields ensures the image is resized proportionally, so when you set either a new height or width, the other value is automatically calculated to preserve its current aspect ratio. If you want to stretch or squash an image for any reason, click the Link button to deselect it before manually entering both values.

By default, you'll be prompted to resize your image in pixels, but the drop-down menu makes it possible for you to specify a size in centimetres or inches instead. You'd usually want to do this when gauging an image's size for printing on paper, and it works in tandem with the resolution figure beneath.


Resize Image

Size in Pixels/Memory

Width: 1000 Height: 1350 Memory: 6.95

New Size

Width: 1000

Height: 1350

Resolution: 96.00

Pixels

Pixels/cm

Pixels/inch

Preset Size: (none)

Photoflare enables you to make quick and easy adjustments, such as resizing your image by pixels, inches or whatever centimetres are!

As we've hinted, *Photoflare's* resolution maxes out at 99.99 for both pixels/inch and pixels/cm. 99.99 is well short of the industry standard of 300ppi for printing photos, but if you switch to pixels/cm and set it to its maximum of 99.99, you can create the equivalent of 250ppi, which provides you with a more accurate estimate of how large your photos are when printed.

Cropping tools

Photoflare's Resize Canvas function works in a similar manner, but bear in mind that you're not resizing the image, simply resizing the canvas it resides on. If you shrink it, it'll be cropped; if you expand it, you'll surround it with blank space that can be left white or filled with a colour. The Position section enables you to decide where your image will reside on the canvas or where it'll be forcibly cropped.

When clicking any of *Photoflare's* Colour drop-down menus – both here and elsewhere – you might be forgiven for thinking that you've only got a choice of 10 colours and shades of grey to choose from. To access a wider range, click the white colour itself to bring up the Select Color dialog with a choice of 54 shades, together with a '+' button. Pressing this enables you to your own custom choices using a colour chart, your chosen colour's six-digit hexadecimal code (if known), or by

QUICK TIP

When resizing images, look out for the Preset Size drop-down menu on the Resize Image dialog. You'll find a range of commonly selected presets here, from 100x100 pixels for icons to 1920x1280 pixels (Full HD) and even 3840x2160 (the equivalent of a 4K image).

» ALTERNATIVES TO PHOTOFLARE

If you find yourself frustrated with *Photoflare* – a lack of layer support and non-interactive filters are two notable shortcomings – then there are other tools are out there that serve as user-friendly alternatives to *GIMP*.

One such program is *LazPaint* (**https://lazpaint.github.io**), which you can install through the *Software Centre*. It's closer to *GIMP* in terms of functionality – full layer support and a wider choice of controllable filters – but manages to keep everything grouped together in a single window. It's more sophisticated than *Photoflare* and so requires a steeper learning curve, but is still relatively simple to navigate.

Another tool worth looking at is *Glimpse* (**https://glimpse-editor.org**).

If you find Photoflare lacks certain features, try another tool like Glimpse, a direct fork of GIMP.

This is an actual fork of *GIMP*, with the developers looking to provide a more 'professional' user interface for those switching from commercial tools. The obvious benefit is that you're using a tool with close links to *GIMP* wrapped up in a more accessible GUI, which means *Glimpse* could be used as a stepping stone to full-blown *GIMP* or simply as a more accessible alternative to it.

Development on the tool has stalled since 2021, but it's still available to install as a snap. In addition to supporting many of *GIMP's* advanced features, *Glimpse* is also compatible with its plugins.

QUICK TIP

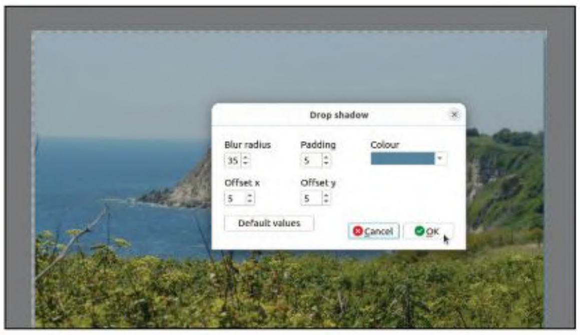
Visit https://photoflare.github.io/photoflare for a basic user manual of Photoflare's main functions. You can also post comments on the tool's official forum (https://discuss.pixls.us/c/software/photoflare/32) or report issues and bugs at the project's Github page (https://github.com/PhotoFlare/photoflare/issues).

capturing a specific colour from anywhere on your desktop using the Eyedropper tool.

If you're looking to crop the image by selecting part of it, first make sure the Pointer tool is selected, then click and drag a bounding box around the area you wish to preserve. You can resize or move it if required, but once selected, right-click inside the box and choose Crop. If you're looking to crop out a frame or other element surrounding your image, try selecting Image>Automatic Crop instead to see if *Photoflare* can do the hard work for you.

Speaking of frames, the Image menu is also where you'll find two options for enclosing your photo in a frame. Both expand the canvas size so that the frame fits around the image rather than sitting on top of it. The Frame option provides you with a simple choice of frame width and colour, while Drop Shadow gives you the opportunity to create a more nuanced effect using a combination of options, including blur radius, padding and offset.

If you want to add text to your image, choose Image>Text to open a dialog where you can compose your text, and choose a font, size, style and alignment. After clicking OK, the text appears on your image as a floating, movable box. Again, because *Photoflare* doesn't support text sizes larger than 99 pixels, you'll find text gets lost on larger images, so if resolution isn't an issue, consider shrinking the image before labelling it.



For a less-brash way to frame your image, choose Image>Drop Shadow or try one of the Filter> Frame options.

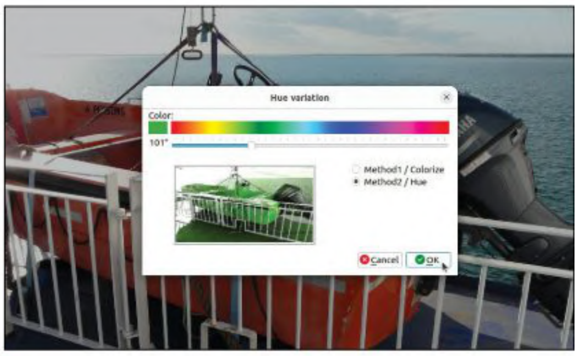
» THE FUTURE OF PHOTOFLARE

Hidden away on the *Photoflare* website (**https://photoflare.io/v2**) is a planned roadmap for the future of the project. It reveals that version 2 will move away from the free Community Edition to a paid-for Studio Edition, but does add that many of the big new features being planned will turn up in the Community Edition as add-ons.

Those forthcoming features include support for layers, the ability to add new advanced paint brushes using the libmypaint (**https://github.com/mypaint/libmypaint**) engine, and the ability to draw shapes and create different selection types using the Pointer tool. There are other noteworthy improvements planned, too: the ability to erase parts of your image using transparent pixels, and options for loading and saving custom colour palettes.

Users can gain early access to the forthcoming Alpha build of *Photoflare Studio Edition* by becoming a paid advocate ($10/month) and then logging into the Studio downloads section of the website (**https://photoflare.io/studio-downloads**).

You'll also notice how there are currently no plugins available for the program. The author recently added a new sponsor in the form of cutout.pro, an online AI tool that automatically removes unwanted backgrounds from images. He plans to integrate the tool into a future version of *Photoflare* as a plugin, perhaps showcasing the way that the Community Edition will continue to be expanded in the future.



Photoflare's adjustment tools are perhaps one of its weakest elements, but you can still produce interesting effects.

Make an image transparent

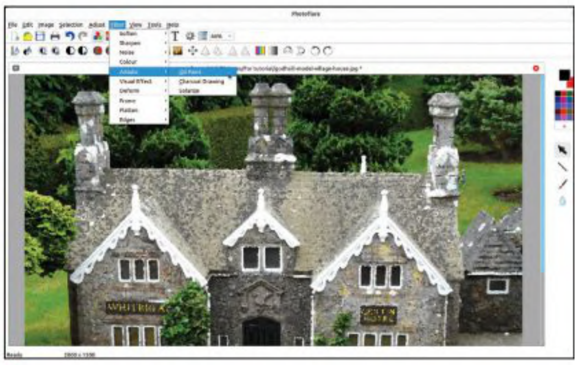
Photoflare makes it possible for you to take advantage of PNG and GIF images' support for transparency by specifying which colour will be transparent. This works best where a single solid colour currently serves as the background of your image. With your image open, select Image>Transparent Colour... A dialog will pop up and the cursor will change to the eyedropper. Position this over the colour you'd like to make transparent and then click to select it.

Next, click Preview and wait a few seconds (or longer with larger photos on slower machines) to see the effect your changes will have. A speckled effect shows which parts of the photo are now transparent. If the background isn't perfectly lit then you may find parts of it don't disappear. Increase the tolerance slider up to 20 and click Preview again. More of the unwanted background should disappear. Keep moving the slider up to a maximum of 99. Click OK when you're happy.

Luminance and colour effects

Photoflare offers a handful of options for attempting to correct poor lighting or colour balance. There are Auto Levels and Auto Contrast options for quick and dirty fixes, along with Negative to convert your photo into a replica of a film negative. The Opacity option (choose a figure between 0 which is completely transparent, and 1 for solid) enables you to transform your image into a watermark for overlaying on to other images via copy and paste.

To make manual colour adjustments, choose Adjust> Hue Variation. This provides you with two sliders:



Try reducing the size of your image to strengthen the effect of a selected effect, like this oil painting filter.

method one (Colorize) retains the original colour to change its 'temperature' through mixing in another colour such as blue, green or yellow. Method two (Hue) simply colourises the image according to whichever colour has been selected. A preview window gives you an idea of what's going on. Click OK to apply the effect.

Quick-fire effects

Photoflare's Filters menu is home to a range of tweaks and special effects. Unlike more sophisticated tools like *GIMP*, these are blunt instruments: select the filter and it's applied with no controls for tweaking its effects. Divided into nine categories, they're self-explanatory on the whole: Soften is less drastic than Blur for example. If an effect isn't as strong as you'd like it to be, try applying it a second (or even third) time.

The first group of categories – Soften, Sharpen, Noise and Colour – are aimed at image correction. If you're unable to fix problems like colour balance using the Adjust menu, try the Equalize and Normalize filters. Beneath this are three categories for transforming your images using effects like Artistic (Oil Paint, Charcoal Drawing and Solarize), Visual Effect (Motion blur) and Deform (Swirl, Wave, Implode and Explode).

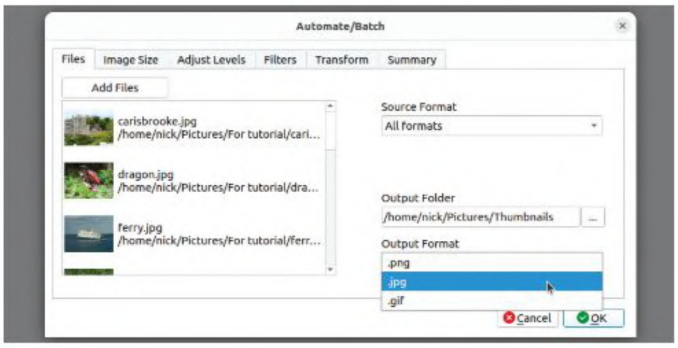
If *Photoflare's* frame options leave you uninspired, you'll find three more frame options under the Filter menu. Again, none offer you any form of control over the finished effect. Rounding out the filters on offer are Emboss and Monochromatic.

Batch processing

Photoflare's final feature of note may be the most compelling of all: select Tools>Automate/Batch and you'll be able to select a group of PNG, JPG and/or GIF files and perform the same edits on them with just a few clicks. Options include converting to a different format, resizing and even applying multiple filters The step-by-step guide (*below*) reveals how it works. LXF

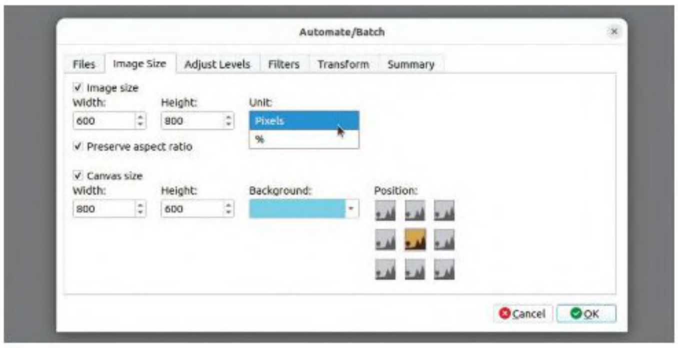
QUICK TIP
Photoflare supports two colour modes: RGB and Indexed. RGB mode is 16-bit and supports millions of colours, while Indexed mode is 8-bit, which limits images to 256 colours.

SAVE TIME BY BATCH PROCESSING IMAGES


Automate/Batch
Files | Image Size | Adjust Levels | Filters | Transform | Summary
Add Files
carisbrooke.jpg /home/nick/Pictures/For tutorial/cari...
dragon.jpg /home/nick/Pictures/For tutorial/dra...
ferry.jpg /home/nick/Pictures/For tutorial/ferr...
Source Format: All formats
Output Folder: /home/nick/Pictures/Thumbnails
Output Format: .png / .jpg / .gif

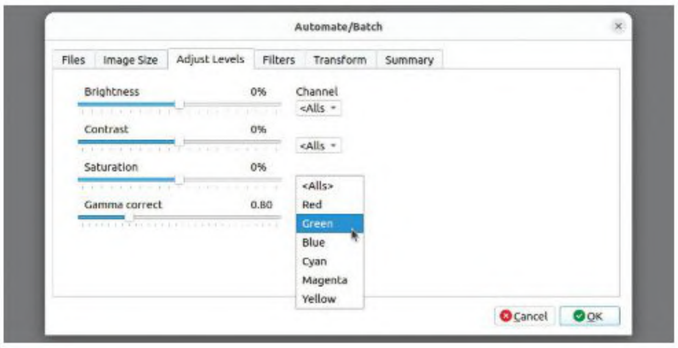
1 Select files and output format

Choose Tools>Automate/Batch... to open the six-tabbed dialog. Click Add Files to add your files (only PNG, GIF or JPG files will be recognised for batch operations). Click ... to set the folder where your converted files will changed, and the Output Format drop down to choose what file format they'll be saved in (again, PNG, GIF or JPG) based on the source format.


Automate/Batch
Files | Image Size | Adjust Levels | Filters | Transform | Summary
✓ Image size
Width: 600 Height: 800 Unit: Pixels / %
✓ Preserve aspect ratio
✓ Canvas size
Width: 800 Height: 600 Background: Position:

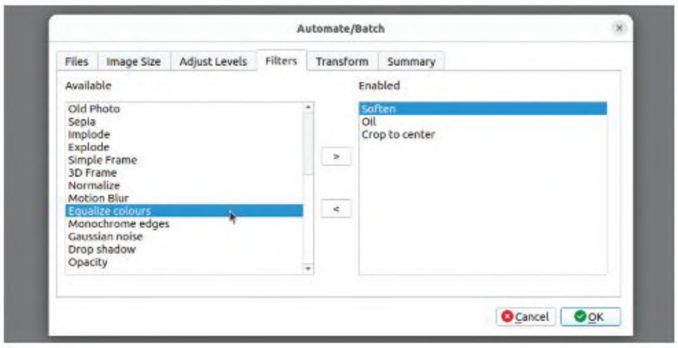
2 Choose your resize options

Use the Image Size tab if you're shrinking the images to a set size (say, for thumbnails). You can resize by pixels or percentage – click Preserve Aspect Ratio if your images aren't all the same. Tick Canvas Size to place your images within a frame. Make sure the canvas size is larger than your images, set your background colour and choose where to position the image within the frame.


Automate/Batch
Files | Image Size | Adjust Levels | Filters | Transform | Summary
Brightness 0% Channel <All>
Contrast 0% <All>
Saturation 0%
Gamma correct 0.80 <All> / Red / Green / Blue / Cyan / Magenta / Yellow

3 Adjust levels and transform

Use the Adjust Levels tab to alter the images' brightness, contrast, saturation and luminance (gamma correct) using a combination of the sliders and the Channel drop-down to apply the changes to all colours or just a specific colour. Switch to the Transform tab to find options for rotating or flipping all the images in the selection.


Automate/Batch
Files | Image Size | Adjust Levels | Filters | Transform | Summary
Available: Old Photo, Sepia, Implode, Explode, Simple Frame, 3D Frame, Normalize, Motion Blur, Equalize colours, Monochrome edges, Gaussian noise, Drop shadow, Opacity
Enabled: Soften, Oil, Crop to center

4 Pick filters and process

Switch to the Filters tab, where you'll see a complete list of filters to match what's on the Filter menu. Select a filter and press the '>' button to have it applied to your selected images. You can choose multiple filters, which will be applied in the order they appear in the Enabled list. Finally, review your proposed changes on the Summary tab before clicking OK to apply them.

RISC OS

Credit: www.riscosopen.org

Relive the classic ARM-based RISC OS

RISC OS is an operating system with a fascinating history that dates back decades. **Michael Reed** tries it out in its current incarnation.

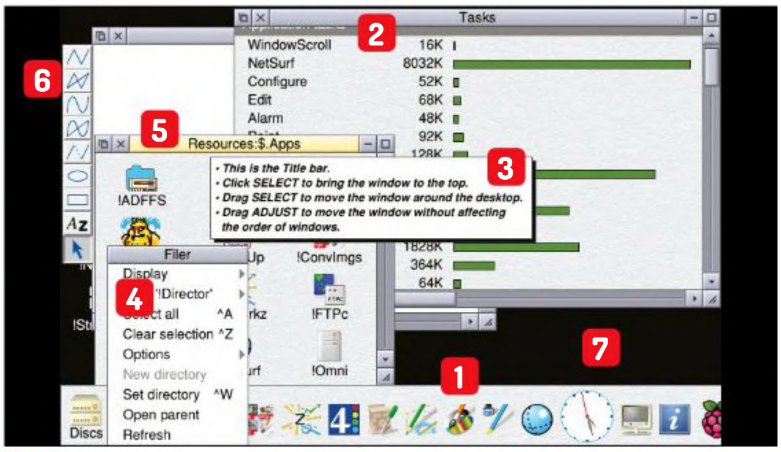
OUR EXPERT

Michael Reed has been RISCing his OS and chancing his ARM since the early 90s.

Back in 1987, British computer manufacturer Acorn released its Archimedes series of workstations that were aimed at educational establishments and home users. Rather than using an off-the-shelf CPU, Acorn designed its own – the ARM processor – which now powers the majority of mobile phones and portable devices.

However, we're interested in the other half of that legacy: the software part. Specifically, this is RISC OS, the operating system that Acorn created for the Archimedes. We're going to look at RISC OS 5, an open source version of the operating system that's currently maintained and can be run natively on modern ARM hardware such as a Raspberry Pi. RISC OS was cutting edge when it was first released, but it's now quite an old operating system, and it's deficient in many areas compared to more modern offerings such as Linux or Windows. However, it's fascinating to look at and use, and it may have a few features that we'd like to see in modern systems.

EXPLORE THE RISC OS INTERFACE



• This is the Title bar.
• Click SELECT to bring the window to the top.
• Drag SELECT to move the window around the desktop.
• Drag ADJUST to move the window without affecting the order of windows.

1 Icon bar
Drives and other hardware are on the left, applications are on the right, with system tools located on the far right.

2 Task Manager
A click on the Pi logo (on this distribution) opens up the Task Manager that neatly shows resource usage on the machine.

3 !Help
RISC OS's help application is system-wide and provides pop-up descriptions of any GUI element under the mouse pointer.

4 Pop-up menu
This is the style of menu that RISC OS has instead of the more common pull-down menus of other systems.

5 A filer window
This is where you launch applications and load and save files. Filetype is stored within the metadata.

6 Toolbox
RISC OS makes quite a lot of use of toolboxes, which are either glued to the side of an application window or free-floating.

7 Adaptable desktop
The backdrop itself can be used as an application launcher or to store minimised application icons.

Obtaining RISC OS 5

On the original Archimedes computers that RISC OS ran on, the operating system was largely contained within ROMs on the motherboard. This meant that they could at least boot without a floppy or a hard drive. If you want to run RISC OS 5 on modern ARM hardware, you can obtain it from RISC OS Open (**www.riscosopen.org**), a not-for-profit organisation that carries out most of the maintenance of RISC OS. There are various distros that support different ARM platforms, and in our case we're going to run it on a Raspberry Pi 400. You can download it as an image file to be written to an SD card, and it's even offered as an alternative OS in the Raspberry Pi Imager application.

When you first run RISC OS, there's a boot-up screen with a progress bar. This takes about 35 seconds, but most of that time is waiting for the network to serve an IP address through DHCP. This leads us to the first problem we encountered: due to hit-and-miss hardware support, the Wi-Fi chipset of the Pi 400 isn't supported. This means you have to use a wired Ethernet cable for the moment. RISC OS is intended to be operated largely from the GUI (although there is a command line), and it's a mixture of familiar and unique elements.

User interface

Interacting with the RISC OS GUI is the highlight of using the system. The first thing that you may notice is the icon bar at the bottom of the screen. As often as not, when you open an application, it doesn't open a blank document like on other systems. Instead, the icon simply adds itself to the icon bar.

From here, you left-click the icon to create a blank document. You can also drag documents onto these icons, a feature that's often missing on modern docks. A middle-click on the icon brings up a menu with features related to that program. Overall, this adds up to dock-style functionality that's highly logical and a bit richer than the Gnome dock, for example.

Once you have a blank document open, another thing that strikes you about the user interface is that there are no pull-down menus. Instead, RISC OS relies exclusively on pop-up menus and these are triggered by pressing the middle-mouse button. Menus are also context-sensitive, so what pops up can vary according to what you're clicking. A great feature here is that when you right-click within a menu, the menu stays open, enabling you to make multiple selections.

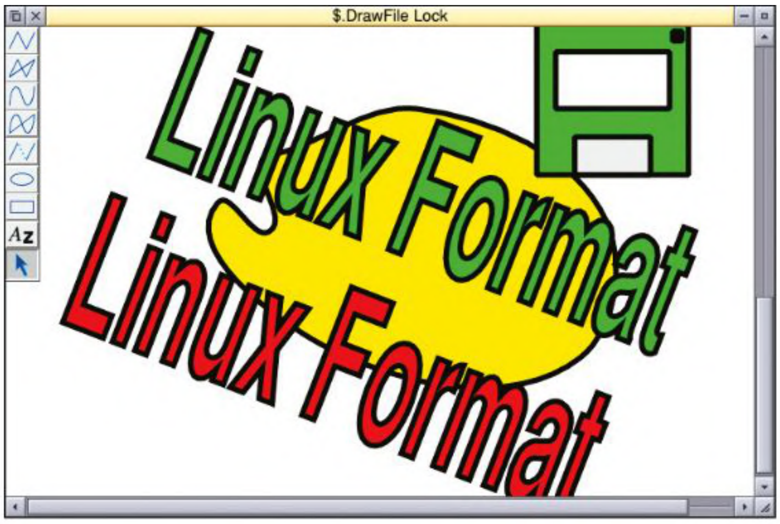
The mouse buttons are assigned as Select, Menu and Adjust. For example, you can drag over some text in a text document with the left button, much like on other systems, but you can also tweak that selection with the right-mouse button. You could also select points on a diagram with left-mouse button clicks and move those points around with the right button. Acorn had the benefit of starting with a three-button mouse, and use of the mouse for basic operations is arguably better than the setup that's standard on today's computers.

All of the RISC OS user interface is, for better or worse, extremely consistent, and the mouse button layout is an example of that consistency. However, RISC OS applications often use multiple windows, and generally you have to get used to positioning them yourself, which can sometimes be awkward.

There are no file saving or loading dialogs because all file operations are carried out through drag-and-drop operations. This makes a lot of sense. Think of all of the times that you've had to dig down into the directory structure to save a file when using other operating systems. Following that, you usually have to dig down again to work in that directory with the file manager or other applications.

On RISC OS, all file operations are carried out through drag and drop within 'filer windows' – RISC OS's equivalent of a file manager. You can even save a file from one program to another. For example, you can drag the save icon from an image editor on to the frame of a DTP package to set up the equivalent of an OLE link.

Ubiquitous drag and drop isn't perfect. The main downside with this approach is that you sometimes have to do some window-manoeuvring to be able to save a file. In addition, most Linux users have now rejected the convention of every directory opening in its own, minimalist Window. However, it's an efficient and



This is !Draw in action. The vector drawing application that comes with RISC OS is a useful application in its own right, and has established a standard vector format.

logical system, and it offers a glimpse of how the Linux desktop might have evolved if the popular GUI toolkits hadn't adopted the file dialogs of Windows and MacOS.

Installing software

If you're delving into the world of RISC OS for the first time, you may notice much use is made of the exclamation mark (called the 'pling' in Acorn jargon). In RISC OS, if you create a directory with a ! at the start, it doesn't open when you double-click it. Instead, a script called **!Run** within the folder is run. This enables you to create application folders, which you can drag directly on to your hard disc to install. If you want to see inside an application folder, shift-double-click it. This gives you access to its internal resources, such as configuration files. Having a pling at the front of certain files such as **!Readme** makes sense because it pushes them to the front when alphabetical sorting is in use.

It's a simple system that works well, but it comes with a few downsides. For instance, a script called **!Boot** is run whenever a filer window sees the application directory, creating a potential security nightmare. Speaking of which, RISC OS doesn't have many security features, and it's not a multi-user operating system. Like most of RISC OS, application installation is elegant but overall we prefer a Linux-style package manager that

QUICK TIP

Acorn ceased development of RISC OS in the late 90s. However, it was developing the successor, called Galileo, with an improved kernel, better multi-tasking and features such as virtual memory. Imagine how good the next version of the OS could have been if Acorn had kept pushing ahead with it...

» WEB BROWSING DIFFICULTIES

Unfortunately, web browsing is an area in which RISC OS is lacking. RISC OS comes with *NetSurf* (**www.netsurf-browser.org**), a small footprint browser that uses its own layout engine and can be compiled on Linux. Most sites concerned with RISC OS have been tested on *NetSurf*, so they run fairly quickly and display properly. Once you get away from RISC OS-centric sites, the problems become apparent. A few sites work, but most sites suffer from a broken layout and missing features. Other sites are impossible to navigate.

There are plans to improve the situation and there are webkit-based browsers under development. We tried a couple of these, but they also exhibited layout problems along with poor performance. The truth is that the RISC OS kernel doesn't support multi-threading or multiple cores. So, the performance of even a complete *Firefox* port would never be as good as, say, running it under Linux on the same ARM hardware. For general web browsing, you'd be better off using a tablet (which, ironically will be running on an ARM chip).

!QupZilla, a webkit-based browser. Compatibility is better than !NetSurf but it's a bit of a slug, speed-wise.

»

QUICK TIP

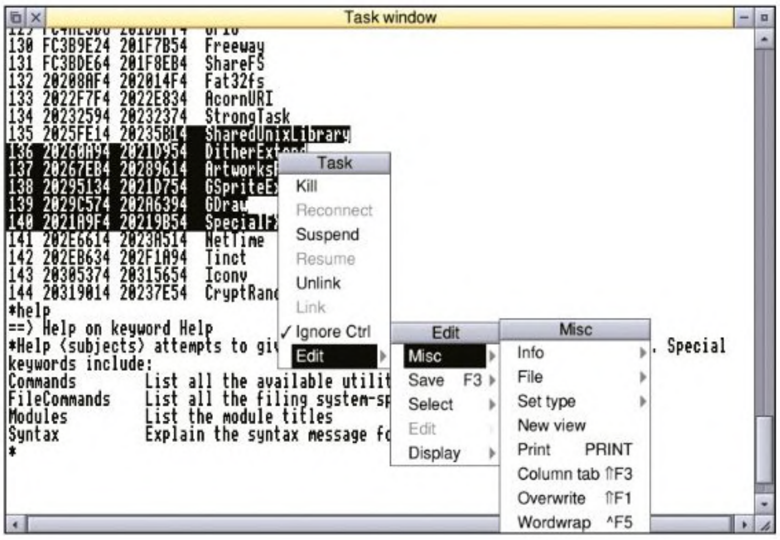
RISC OS was always designed to be fast, and it's faster still on modern hardware. Indeed, you may think an operation, such as launching an application, hasn't worked at first because it happens so quickly. Generally, applications don't access the hard drive once they're loaded.

offers easy installation and maintenance combined with fine-grained facilities to find out what resources have been installed. Thankfully, RISC OS 5 now comes with a package manager (*!PackMan*) and a store (*!Store*), and these can take you quite a long way when it comes to installing the software you need.

You can extend the operating system with the use of 'relocatable modules', which are a bit like Linux's kernel modules. You can either double-click these to load them or place them into the **!System** directory in the root of the filesystem, to be loaded when required. They offer facilities such as new filing systems that enable the system to access storage resources such as network shares, archives and foreign disk formats as though they're native drives. Relocatable modules are how the shared C library and hardware drivers are implemented. In fact, most of the operating system is implemented in this way, meaning that you can easily softload updated components, usually on the fly and without restarting the system, either.

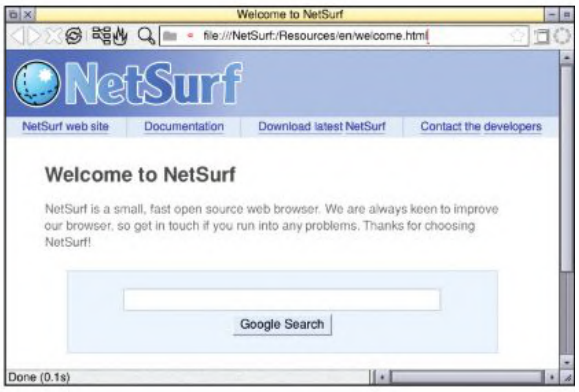
The architecture

A lot of the RISC OS user interface is highly efficient and it even contains some features that we'd like to see in modern operating systems. However, the RISC OS



```
130 FC3B9E24 201F7B54  Freeway
131 FC3BDE64 201F8EB4  ShareFS
132 20208AF4 202014F4  Fat32fs
133 2022F7F4 2022E834  AcornURI
134 20232594 20232374  StrongTask
135 2025FE14 20235B14  SharedUnixLibrary
136 20260A94 2021D954  DitherExt
137 20267EB4 20289614  ArtworksR
138 20295134 2021D754  GSpriteEx
139 2029C574 202A6394  GDraw
140 2021A9F4 20219B54  SpecialFX
141 202E6614 2023A514  NetTime
142 202EB634 202F1A94  Tinct
143 20305374 20315654  Iconv
144 20319014 20237E54  CryptRand
*help
==> Help on keyword Help
*Help <subjects> attempts to gi                              . Special
keywords include:
Commands        List all the available utilit
FileCommands    List all the filing system-sp
Modules         List the module titles
Syntax          Explain the syntax message fo
*
```

The windowed command line of RISC OS is a mode of !Edit, the text editor. It's an interesting approach and quite useful.



Welcome to NetSurf

NetSurf is a small, fast open source web browser. We are always keen to improve our browser, so get in touch if you run into any problems. Thanks for choosing NetSurf!

!NetSurf is a basic web browser. It's good for browsing RISC OS-related sites, but not comprehensive enough for general use.

kernel and core system are more of a mixed bag, as fascinating as they may be to the OS enthusiast.

RISC OS was based on a stop-gap Archimedes operating system called Arthur. This, in turn, was based on the operating system of the BBC Micro series of 8-bit computers. The kernel doesn't support multithreading, virtual memory or comprehensive memory protection. GPU support is all but non-existent, and RISC OS has no awareness of multiple CPU cores. Worst of all, multi-tasking is what is called 'cooperative multitasking', like classic versions of MacOS, rather than the preemptive multitasking that was favoured by the rival Commodore Amiga computers and a feature of modern operating systems such as Linux and modern Microsoft Windows.

In a cooperative multitasking system, each application has to give control back to the operating system once it's finished doing its processing. If that processing takes five seconds, every other application on the system stops for five seconds. If the application itself hangs then generally the entire operating system hangs. There are some workarounds and kludges, so that things like music players and the TCP/IP stack can operate in the background, but it's not how most of the operating system runs.

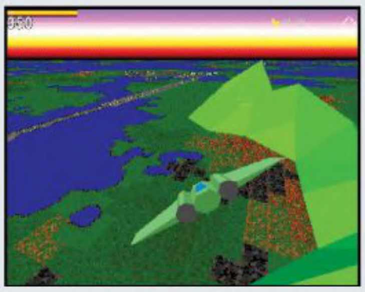
In use, RISC OS seems like a fast system, and at first, you hardly notice that the multitasking is primitive compared to what we're used to today. The truth is that by the late 90s the average user wanted to run media-heavy tasks and go online with good security and

» RISC OS GAMING

As a gaming platform, the Archimedes never gained the same popularity as its rivals such as the Commodore Amiga and Atari ST, but it did have a healthy – if small – games scene. Like those two platforms, the bulk of the gaming library was aimed at the lowest spec baseline machines. There are some exclusive titles that are worth seeking out, and most of the big hits of the day, such as *Lemmings* (1991), were ported to the platform.

Because of its simple but fast graphics hardware coupled with 256-colour screen modes, the Archimedes was a good platform for smooth 3D games. The Archimedes was also a highly competent machine for classic game types such as sideways scrolling shooters and platform games. Overall, it's an underrated classic system with some gems that are worth looking up.

If you install *ADFFS*, about 80 games are installable through the package manager. Although some of the games can be made to run under RISC OS 5, exploring the games library with a dedicated Archimedes emulator, such as *Arcem* (**www.arcem.sourceforge.net**) or *Arculator* (**http://b-em.bbcmicro.com/arculator**), may be a better approach to take.



Star Fighter 3000 was a later-era Archimedes game and it was ported to other platforms.

smooth multitasking, leaving RISC OS behind. These technical shortcomings are collectively the Achilles heel of RISC OS.

Much of the operating system was hand-coded in assembly language, as opposed to a more portable language such as C, and this approach contributed to the Archimedes' reputation as an absolute speed-monster. However, this also means that there's very little prospect of it ever being ported to anything other than ARM hardware.

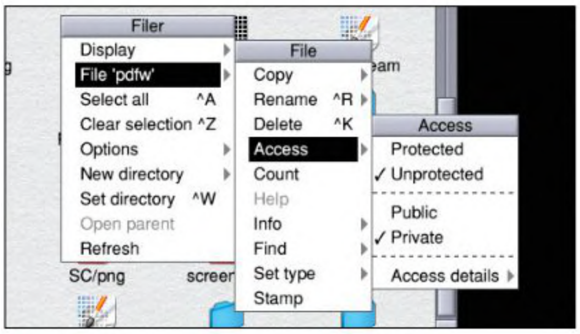
Talking of programming languages, although it may seem like an eccentric addition to a powerful workstation of the time, RISC OS has a built-in version of BASIC, the most common programming language of 80s home computers. It's a particularly good version, too – an improved version of BBC Micro BASIC that was widely employed in British schools. Like a lot of computers of an earlier era, you could start learning to program with nothing more than a blank disc to save your work on.

The software

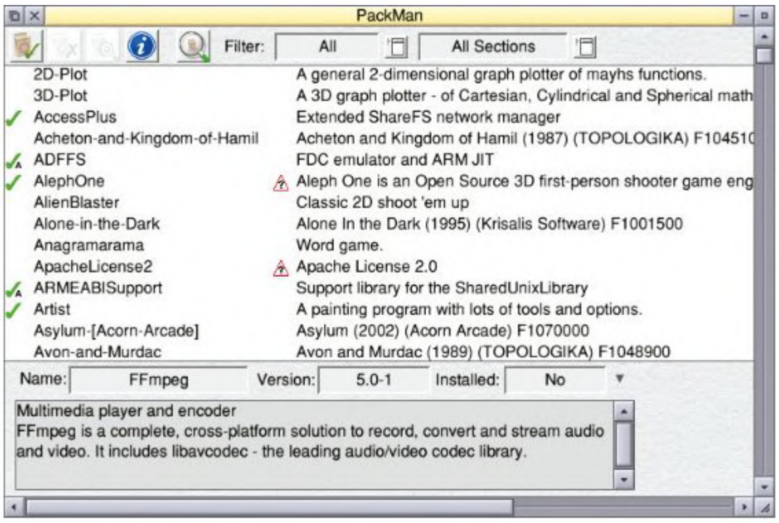
In its day, RISC OS had some outstanding applications and games. Much of that older software is now freely available, but as you might imagine, most of it is considerably out of date. The software that's available for the platform consists of former commercial software, freeware, ported open source software and some shareware that's still being maintained. The RISC OS 5 distribution comes with some software already installed so that you start off with the basic tools such as a word processor, a PDF viewer, image viewer and some programming tools.

Even the mighty ARM chip itself suffered from some corner-cutting when it was first conceived. The program counter – one of the most fundamental parts of a CPU – was originally 26-bits, rather than 32-bits like the rest of the CPU. This means that modern ARM hardware can't run the classic-era software.

Fortunately, much of the old software has been patched or recompiled to run on modern ARM systems, and a lot of the legacy software such as games can be forced to run in other ways. *ADFFS* is one such tool that can be used to persuade some of the old applications and games to run on RISC OS 5 on a Raspberry Pi. Look for the JASPP Forum (**https://forums.jaspp.org.uk**) for instructions on how to obtain and install it. The most common method for installation of this tool will appeal



Pop-up menus are the bread and butter of the RISC OS GUI. If you right-click an item, the menu remains open.



PackMan

Filter: All | All Sections

	2D-Plot		A general 2-dimensional graph plotter of mayhs functions.
	3D-Plot		A 3D graph plotter - of Cartesian, Cylindrical and Spherical math
✓	AccessPlus		Extended ShareFS network manager
	Acheton-and-Kingdom-of-Hamil		Acheton and Kingdom of Hamil (1987) (TOPOLOGIKA) F10451
✓A	ADFFS		FDC emulator and ARM JIT
✓	AlephOne	⚠	Aleph One is an Open Source 3D first-person shooter game eng
	AlienBlaster		Classic 2D shoot 'em up
	Alone-in-the-Dark		Alone In the Dark (1995) (Krisalis Software) F1001500
	Anagramarama		Word game.
	ApacheLicense2	⚠	Apache License 2.0
✓A	ARMEABISupport		Support library for the SharedUnixLibrary
✓	Artist		A painting program with lots of tools and options.
	Asylum-[Acorn-Arcade]		Asylum (2002) (Acorn Arcade) F1070000
	Avon-and-Murdac		Avon and Murdac (1989) (TOPOLOGIKA) F1048900

Name: FFmpeg Version: 5.0-1 Installed: No

Multimedia player and encoder
FFmpeg is a complete, cross-platform solution to record, convert and stream audio and video. It includes libavcodec - the leading audio/video codec library.

We were pleased to see that RISC OS 5 comes with a Linux-style package manager called !PackMan.

to Linux users because it involves adding the repository to the package manager.

Acorn shipped some standard applications with RISC OS. They were built into the ROM on later Archimedes models, and you can access them by clicking the Applications icon on the icon bar. *!Draw* is a vector drawing package, and it's the standout of the built-in applications. It's a handy tool for knocking together a quick diagram or two, and it established a standard vector drawing format for the platform right from the beginning. *!Paint* is a graphics editor, and again, it established a standard format that was supported by all software on the platform. *!Edit* is the built-in text editor.

Back in the day, speed-wise the fastest Archimedes machines could go toe to toe with pretty much every platform out there, when used as a DTP workstation. So it's hardly surprising that DTP software was popular on the platform. There is some commercial software still available in these areas, such as *ArtWorks*, a vector drawing application and *OvationPro*, a DTP package, but it tends to be quite expensive, sitting around the £100 mark for a copy of each of them. We're not going to put it down, if the users are willing to pay to keep these old projects alive.

A flawed gem

In our estimation, RISC OS 5 is a couple of notches below being usable as a daily driver. The biggest shortcoming in terms of essential software is the lack of a fully featured, modern web browser. Various browsers exist, but it would be hard to imagine carrying out a typical desktop computer workload with them. It's certainly a fascinating OS, and the GUI even contains a few niceties that it would be nice to see ported to a more modern system. As charming as the system is, beneath the surface is where the real problems lie, because the kernel technology is terribly out of date.

It's a fun system to play around with, and it still has a small community of developers and users attached to it. If it takes your fancy and you have some suitable hardware such as a Raspberry Pi and a blank SD card, installing it is a quick job, and there's a lot about the system and its software to explore and admire. LXF

QUICK TIP

Hardware support of RISC OS 5 is basic. It supported everything that we tested on a Raspberry Pi 400 other than the Wi-Fi networking. Plugging in a flash drive worked as expected.

BACK ISSUES » MISSED ONE?

ISSUE 291
August 2022

Product code:
LXFDB0291

In the magazine
Learn how to make your virtual machine go faster and run a virtual GPU. We show you how to compose music using algorithms, back up and copy entire hard drives, and advise on best password practice. In the market for a new distro for your Pi? We test five of them, plus check out the story behind Bodhi Linux, emulate the first portable computer, and take a deep dive into World Web 3.

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.

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

USA? EU? THE MOON?

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!

3 GREAT WAYS TO SUBSCRIBE
Print, digital-only, and print+digital bundles!

» USA
From $132
For 13 issues

» REST OF THE WORLD
From $132
For 13 issues

» EUROPE
From €100
For 13 issues

IT'S EASY TO SUBSCRIBE!

Visit **www.magazinesdirect.com/linux-format**

Call **+44 0330 333 1113**

Lines open Monday-Friday, 9am-5pm, UK time

BIMP

Credit: https://alessandrofrancesconi.it/projects/bimp/

GIMP batch processing

Tackling mass image processing is a walk in the park when you have **Karsten Günther** on your side and BIMP installed alongside GIMP.

OUR EXPERT

Karsten Günther is a long-time LaTeX, Linux and GIMP user, and has written a number of books on these topics.

BIMP is located in GIMP's File menu. In addition to the File list, the dialog contains previous Functions and the output directory.

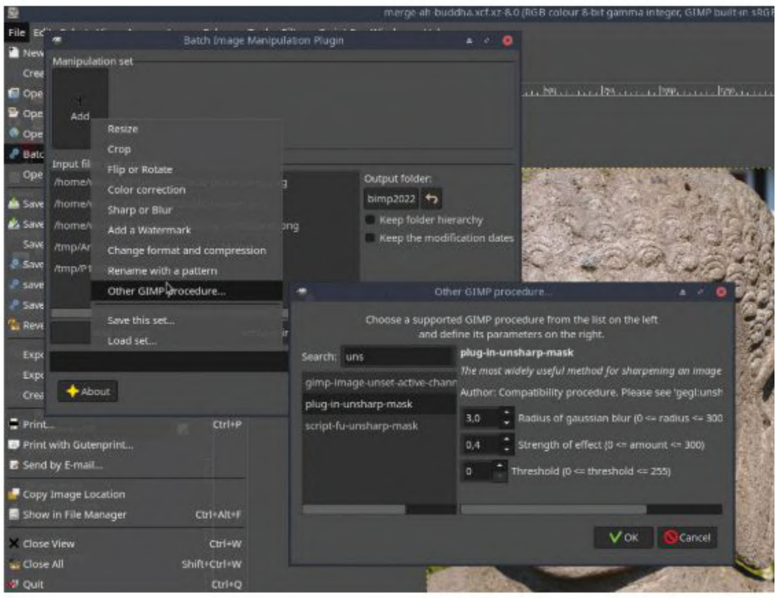
G*IMP* is excellent for editing individual images, but what about editing multiple images at once? The answer usually involves either *ImageMagick* or *GraphicsMagick*, which are used with 'for' loops on the shell. This can be done quickly if shell-savvy users can handle the syntax, but this is not necessarily the case. Both programs offer everything that *GIMP* can do – and more – but they don't show a preview and take longer to become familiar with.

This is where *BIMP* comes in. The program offers a simple interface for applying *GIMP's* functions to any number of images. Only a few steps are required to get it up and running. The *Batch Image Manipulation Plugin* option appears in the File menu after installation (*see the box, facing page, for installation advice*).

For all of *GIMP's* functions that are used in *BIMP*, the parameters with which they're applied on the images must first be determined. When used interactively in *GIMP*, they can be found in the dialog window. In addition to many GEGL (Generic Graphics Library) functions, many script-based functions can also be used. However, not all of the functions within *GIMP* are available. *BIMP* collects all the functions used for editing as a Recipe, which is then applied to the selected images individually.

Selecting images

The first step is to select and load images. Normally these will be bitmap images, which *GIMP* recognises.



Choose a supported GIMP procedure from the list on the left and define its parameters on the right.

plug-in-unsharp-mask

The most widely useful method for sharpening an image

Author: Compatibility procedure. Please see 'gegl:unsh

3,0 Radius of gaussian blur (0 <= radius <= 300

0,4 Strength of effect (0 <= amount <= 300)

0 Threshold (0 <= threshold <= 255)

But XCF files – *GIMP's* own complex format – can also be used; *BIMP* then processes all layers individually. The most common selection methods are Add Single Images to select images individually and Add Folder to process all images in a folder. Choosing Add All Open Images uses the images that are currently open. Note that if many images are open then RAM can become scarce. These methods can also be combined; *BIMP* always creates a list of images first. Unfortunately, drag-and-drop doesn't work from the file manager.

If too many images are selected by mistake, they can be removed from the image list. Choosing Remove Images enables you to delete specific images from the list. In the list window there's also a preview that appears when you click it to identify the images. After creating the image list, an output directory should then be specified.

Functions, functions, functions

The next step is to focus on the functions used. They can be called up and configured individually, and you can decide on the order in which they're used. The button labelled Add leads to a corresponding menu. This menu consists of three parts: a series of functions that are already built into *BIMP*; the universal function Other *GIMP* Procedure...; and options for saving and loading Recipes.

The *BIMP* built-in functions cover some of the most frequently used functions, such as scaling, rotating, mirroring, sharpening and applying a watermark. *BIMP* uses *GIMP's* preset algorithms in the corresponding functions, which leads to results that are indistinguishable from those created manually.

When creating watermarks, *BIMP* offers two variants. By default, the watermarks are short lines of text, as created by the Text tool, which are then inserted on a separate layer. Alternatively, an existing image can also be used as a watermark.

The Sharpening and Blurring menu reveals a typical problem of the built-in functions. *GIMP's* standard functions – which *BIMP* uses in this menu – are significantly less powerful than the functions specialised for a task. These can then be activated via the menu option Other GIMP Procedure... There, for example, the above-mentioned unsharp masking with the function plug-in-unsharp-mask is offered. The same applies to blurring, where several variants of the blur functions are available here.

An output format should always be explicitly defined as the final function. The corresponding entry can be found in the button menu under Change Format and Compression. There you can select various bitmap formats and choose from a range of settings.

There's an additional option to name the newly created files according to a pattern. *BIMP* supports three placeholders: $$ stands for the original file name without its extension, ## becomes a consecutive number and @@ replaces BIMP with the current date plus time, to the minute. Each pattern defined with this function must contain at least one of the first two placeholders. This means that several files with the same name can't be created.

The most important function in this menu is Other GIMP Procedure... This option provides you with access to the *GIMP* functions that *BIMP* uses in its various tools and menus.

What functions are then available? Well, what *BIMP* offers is only a small part of all the functions of *GIMP*. In particular, most complex functions are unavailable and some of the simpler ones are also missing.

Almost all functions are configured using parameters. Good preparation pays off here: these parameters should have been determined manually in advance on some sample images. Although *BIMP* finds out the required parameters of the functions, the plug-in doesn't know the default settings that *GIMP* Filter starts with. Users have to enter them each time. The parameters of functions can be readjusted later with a mouse click on the button.

The order of the functions should also be determined when building a *BIMP* Recipe. For example, denoising is always done first, followed by sharpening – if both actions are really necessary – so that the image noise isn't sharpened as well. With *BIMP*, there's no easy way to change the order of the functions afterwards.

The context menu of *BIMP* contains two menu items in the lower section to save the defined Recipes or to reload saved Recipes. This saves a lot of time and work, because not all functions have to be determined again, which would mean all parameters have to be entered repeatedly. It makes sense to name the Recipes after the most important functions used. For example, "upscale2+usm".

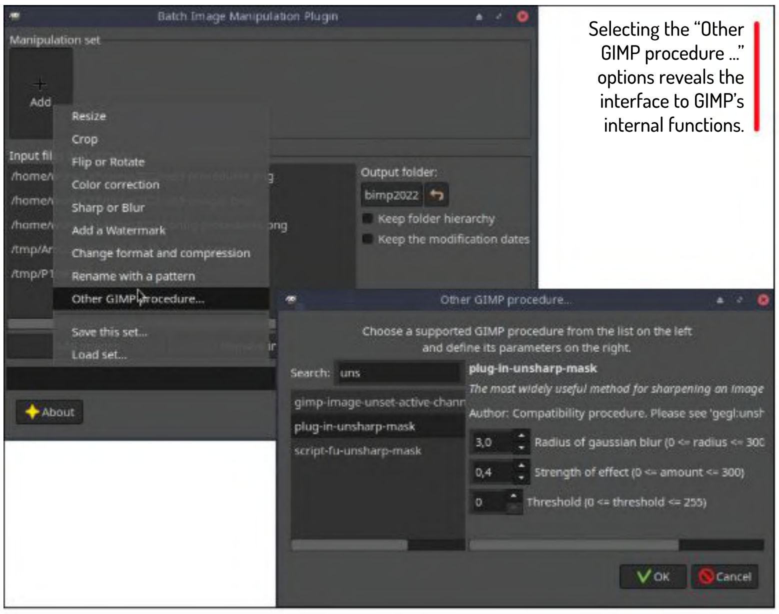
The Recipes used by *BIMP* are simple plain text files with the extension .bimp and can be edited later with a text editor if necessary. With a little skill, several scripts can be merged and then executed together.

The selected images and the output directory aren't included in the Recipes. However, *BIMP* doesn't "forget" the information once it's been entered as long as the dialog is still open.

And what happens if files are created during a run that already exist, for example because a Recipe is called a second time with changed parameters? In this case, *BIMP* notices the presence of the target files and asks how to proceed. To ensure that the overwriting doesn't have to be confirmed again for each file, there's an "Always apply this decision" button in the dialog.

Tips for practice

Provided that all selected images are similar and can be processed with the same settings, it's sufficient to create one batch job. In reality, however, this is often not the case. There are then two possibilities. You can group similar images together and process them in one go.



Output folder: bimp2022
Keep folder hierarchy
Keep the modification dates

Choose a supported GIMP procedure from the list on the left and define its parameters on the right.

Search: uns

plug-in-unsharp-mask

The most widely useful method for sharpening an image

Author: Compatibility procedure. Please see 'gegl:unsh

3,0 Radius of gaussian blur (0 <= radius <= 30C

0,4 Strength of effect (0 <= amount <= 300)

0 Threshold (0 <= threshold <= 255)

Selecting the "Other GIMP procedure ..." options reveals the interface to GIMP's internal functions.

Alternatively, if there are only a few problematic images, they can be processed manually as usual.

It's useful to carry out this manual processing before the batch job, because it's been shown that some of the solutions found here can also benefit the other images. In principle, this is comparable to the batch processing of RAW photos, where similar problems occur.

It's tempting to use *BIMP* to apply many functions to the selected images. However, the end results often aren't worth this extra effort. It makes more sense to first use selected examples to clarify what's necessary and in what order to apply the required image edits.

QUICK TIP

Applying too many functions spoils the photo. Do only what is necessary, as well as possible, but nothing more. Try to use the right functions for each task.

Master of none

BIMP can't replace a "real" macro recorder, nor can it replace real script development. Too many functions are missing for that and more complex structures such as loops and decisions aren't available. But *BIMP* doesn't have to replace a recorder. The plugin is easy to use and it quickly delivers solid results for simple tasks. In addition, there are the good "recycling" possibilities of *BIMP* Recipes, which make it easier each time to develop a new Recipe. LXF

» FROM GIMP TO BIMP

Some distributions offer ready-made packages of *BIMP* or – like Arch Linux and friends – provide automatic installation scripts. However, even if both are missing, installing *BIMP* is not difficult. The source code is available at **https://alessandrofrancesconi.it/projects/bimp**. There's also a reference to the well-known method for installing *GIMP* plug-ins.

First, the archive with the source code is loaded and unpacked:

```
$ cd gimp-plugin-bimp-2.6 && make
```

This creates the plugin. It can be installed in the user's home directory as follows:

```
$ make install
```

Or for all users system-wide with:

```
$ sudo make install-admin
```

After successful installation and a restart of *GIMP*, the entry Batch Image Manipulating... appears in the File menu.

ONLYOFFICE

Credit: www.onlyoffice.com

Using WordPress with OnlyOffice

With OnlyOffice Docs connected to WordPress, authors are able to insert documents into online posts. **Kseniya Fedoruk** explains how it works.

OUR EXPERT

Kseniya Fedoruk is a document expert at OnlyOffice and spends her waking days and even many nights dreaming about documents!

Many *Linux Format* readers will either have heard about *WordPress* or are already using it. It's an open-source content management system that's widely used for hosting blogs and websites. One of the advantages of this publishing service is the ability to extend its functionality and usability with numerous themes and plugins.

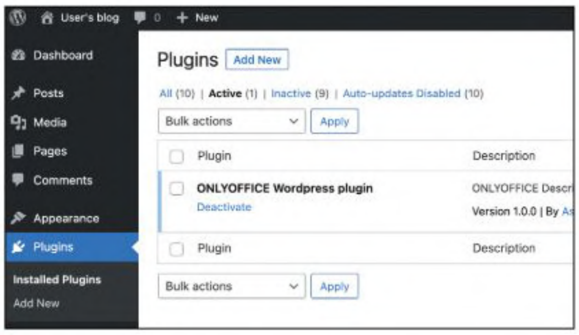
One such plugin is OnlyOffice, that's used for connecting instances of *OnlyOffice Docs* and *WordPress*. It enables *WordPress* administrators to edit and co-author office documents from the admin dashboard, and also add *OnlyOffice* blocks to posts to enable site visitors to view the inserted file without downloading it.

OnlyOffice Docs is an open-source office suite that's made up of online viewers and collaborative editors for text documents, spreadsheets, presentations, fillable forms and PDF files. The core format that's used is . ooxml, so the editors are highly compatible with .docx, .xlsx and .pptx files.

You can use the suite within multiple cloud services: CMS frameworks (*WordPress*, *Strapi*); e-learning solutions (*Moodle*, *Chamilo*, *HumHub*); collaboration platforms (*OnlyOffice Workspace*, *Nextcloud*, *Seafile*, *Confluence*, *Alfresco*); issue trackers (*Jira*, *Redmine*), and many others. You're also able to embed the editors into your own service and deliver it to your end users.

Installing WordPress

According to the *WordPress* team, the easiest way to launch your own instance is to get it through a hosting provider. You can also use *WordPress* through a web browser, or download and install it yourself (see tutorials

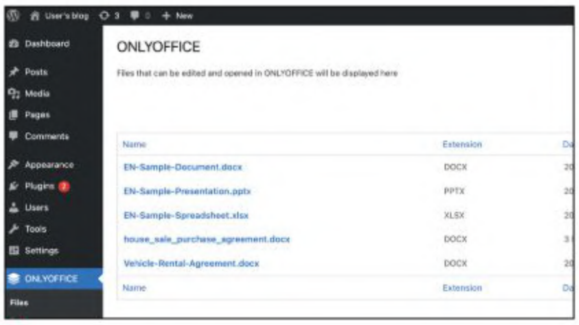


Plugins

All (10) | Active (1) | Inactive (9) | Auto-updates Disabled (10)

	Plugin	Description
☐	ONLYOFFICE Wordpress plugin Deactivate	ONLYOFFICE Descr Version 1.0.0 | By A
☐	Plugin	Description

Activating the OnlyOffice plugin within the WordPress settings.



ONLYOFFICE

Files that can be edited and opened in ONLYOFFICE will be displayed here

Name	Extension	Da
EN-Sample-Document.docx	DOCX	20
EN-Sample-Presentation.pptx	PPTX	20
EN-Sample-Spreadsheet.xlsx	XLSX	20
house_sale_purchase_agreement.docx	DOCX	3
Vehicle-Rental-Agreement.docx	DOCX	20
Name	Extension	Da

OnlyOffice Files page in WordPress enables you access your shared documents directly.

in **LXF248** or **LXF237** for dedicated tutorials). To do so, download and unzip the *WordPress* package from **https://wordpress.org/download**. Next, create a database on your web server, as well as a MySQL/MariaDB user with Access and Modify privileges. Upload the *WordPress* files to a suitable location on your web server. Once ready, run the *WordPress* installation script by accessing the URL in a web browser: **http://example.com/wp-admin/install.php**.

Install OnlyOffice Docs

As with *WordPress*, you can easily obtain the office suite via hosting providers, the number of which is constantly growing. However, if you can't find an appropriate hosting option then you can install the suite using *Docker*, which is officially recommended by the developers. If you don't have *Docker* installed, get it using the command `sudo apt-get install docker-ce` .

Once you have the latest version of *Docker*, install *OnlyOffice Docs* with all the dependencies with:

```
$ sudo docker run -i -t -d -p 80:80 --restart=always
onlyoffice/documentserver .
```

If you want to change the port, use the `-p` command. Here's an example for switching to port 8080:

```
$ sudo docker run -i -t -d -p 8080:80 --restart=always
onlyoffice/documentserver
```

All the data, including logs, certificates, file cache and database, is stored in the specially designated directories called data volumes. It's a good idea to mount those you need to your hosting machine. Use the `-v` option in the `docker run` command:

QUICK TIP

OnlyOffice is compatible with WordPress v.5.9 or higher. The integration plugin is also available in the official WordPress Plugins Directory at https://wordpress.org/plugins/onlyoffice.

```
$ sudo docker run -i -t -d -p 80:80 --restart=always \
   -v /app/onlyoffice/DocumentServer/logs:/var/log/
onlyoffice  \
   -v /app/onlyoffice/DocumentServer/data:/var/www/
onlyoffice/Data  \
   -v /app/onlyoffice/DocumentServer/lib:/var/lib/
onlyoffice \
   -v /app/onlyoffice/DocumentServer/db:/var/lib/
postgresql  onlyoffice/documentserver
```

If you delete the container or something goes wrong during the update, you won't lose your data. You'll also be able to update your certificates without messing with the container. Then you can switch *OnlyOffice Docs* to HTTPS — automatically get Let's Encrypt SSL Certificates using *certbot*.

Install certbot:

```
$ sudo snap install --classic certbot
$ sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

Run the *OnlyOffice Docs Docker* container. Specify ports 80 and 443 and set your domain name and email:

```
$ sudo docker run -i -t -d -p 80:80 -p 443:443 \
   -e LETS_ENCRYPT_DOMAIN=yourdomain.com -e
LETS_ENCRYPT_MAIL=email@example.com
onlyoffice/documentserver
```

Once done, *OnlyOffice Docs* will be available under https://yourdomain.com.

Integration plugin

OnlyOffice's integration plugin for *WordPress* is free and distributed under the GNU GPL v2.0 licence. To install it, clone the plugin master branch:

```
$ cd wp-content/plugins
$ git clone https://github.com/ONLYOFFICE/
onlyoffice-wordpress
```

Once ready, navigate to the Plugins section in your *WordPress* administrative dashboard and activate the OnlyOffice plugin. When you have the plugin installed, configure *OnlyOffice* via the *WordPress* interface. Go to the administrative dashboard and switch to the *OnlyOffice* Settings page. You need to specify the Document Editing Service address – this is the URL of the installed *OnlyOffice Document Server*.

Enhancing data security

To protect your documents from unauthorised access, enter the Document server JWT secret key. The JSON Web Token (JWT for short) helps to verify users who enter the editing sessions in the editors and attempt to perform certain actions within documents. You need to specify the same secret key in the *OnlyOffice Document Server* configuration file to enable the validation. Remember to save your settings after entering the required credentials.

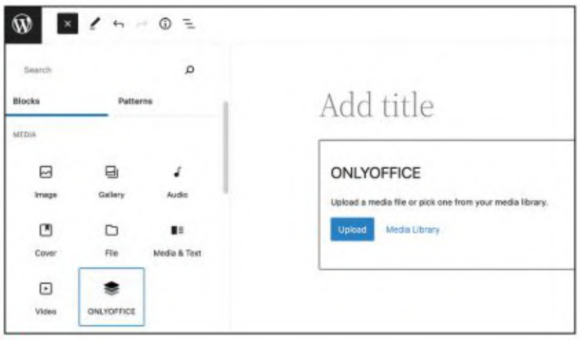
Once you've completed the installation and configuration process, go ahead and start your work. All uploaded files from the Media section appear on the *OnlyOffice* Files page within your admin dashboard.

Users with administrator rights are able to edit and collaborate on .docx, .xlsx and .pptx files using real-time and paragraph-locking co-editing modes, Track Changes, comments and the built-in chat facility. The corresponding *OnlyOffice* editor opens in the same tab when you click the file name. All the changes are saved in the same file after each editing session (when all co-authors leave the document).

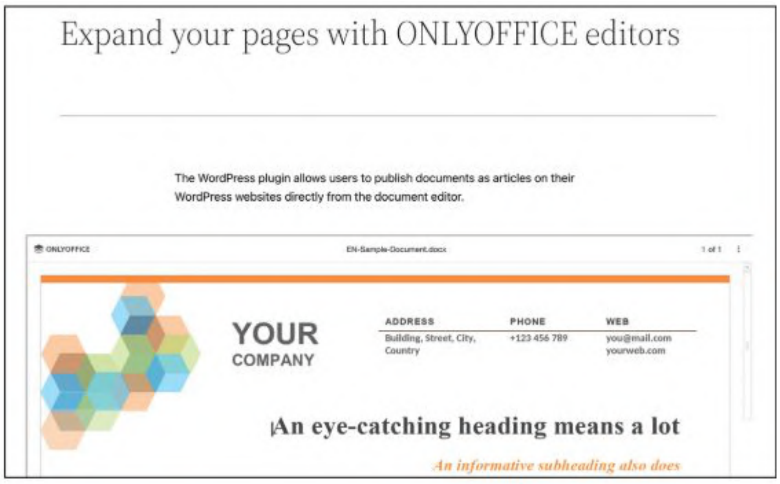
When creating a post, you can add the *OnlyOffice* block and then upload a new file or select one from the Media Library. *OnlyOffice* supports a range of document file formats, including the standard .doc, .docx, .odt, .pdf and many more.

The new file is displayed as the *OnlyOffice* logo with the file name in the currently edited post. After the post is published, *WordPress* site visitors have access to this file for viewing in the embedded mode. This enables them to view the inserted file without downloading it.

That's it! As you can see, connecting *OnlyOffice Docs* and *WordPress* is a straightforward process thanks to the OnlyOffice integration plugin, which acts like a bridge between two services. Such integration makes it possible for you to work with office documents right in your *WordPress* session without having to switch between several applications. LXF



Here you can see an OnlyOffice block being added to a normal WordPress post.


Expand your pages with ONLYOFFICE editors

The WordPress plugin allows users to publish documents as articles on their WordPress websites directly from the document editor.

ONLYOFFICE
EN-Sample-Document.docx
1 of 1
YOUR COMPANY
ADDRESS
Building, Street, City, Country
PHONE
+123 456 789
WEB
you@mail.com
yourweb.com
An eye-catching heading means a lot
An informative subheading also does

You can see the inserted OnlyOffice file in the final published post.

» CHOOSE YOUR CLOUD PLATFORM

WordPress is just one of the many services that you can connect *OnlyOffice* with. You are also able to use *OnlyOffice Docs* with other popular cloud platforms such as *Strapi*, *Jira*, *Moodle*, *Nextcloud*, *Alfresco*, *ownCloud*, *Seafile*, *Plone*, *Confluence*, *Liferay* and *HumHub*. You can either install the free version of *OnlyOffice Docs* or choose the scalable enterprise build, and then just connect the editors with the required service using ready-to-use integration tools. If you need professional editing features for a family-size team, opt for Home Server, a special tariff plan that's designed especially for non-commercial home use. You can find detailed information about *OnlyOffice Docs* on the official website – see **www.onlyoffice.com**.

GITPOD

Credit: www.gitpod.io

Online reproducible dev environments

Stuart Burns is your guide to Gitpod, a new 'work from anywhere' development platform that could revolutionise test environments.

OUR EXPERT

Stuart Burns is a Linux enthusiast and code wrangler. He specialises in virtualisation at scale and cloud infrastructure, management.

Gitpod is a web-based development platform for developers that's stored in the cloud in Git. It enables transparent local development and uses the cloud as the backend. Think of it as a disposable build and test environment that's pre-configured to your liking. Gitpod stands apart from other products on the market that make it a useful solution for some. For starters, it enables an on-demand reproducible web-based development and test environment every time.

Configure Gitpod once and away you go. It's more than that, though. You can create new projects/branches and have them in a new, isolated workspace every time with different configurations for when you want to utilise different setups. Most languages and development are supported. You can even run *Visual Studio Code* in a browser environment, so there's no need to have a laptop or such, nor the power to build locally. There's also a multitude of other browsers that can be consumed within the environment. In summary, it makes it possible to develop, build and test code in a private cloud space that's always available and is ephemeral in nature.

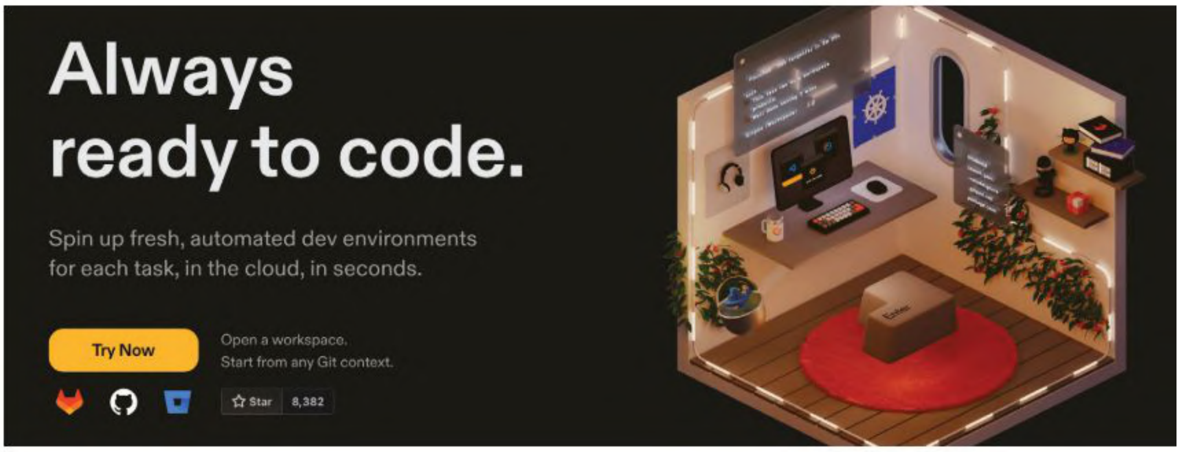
In reality, the underlying environment is a highly optimised Docker image built for running a customised development environment that you have console access to. Think of it as a versatile, mobile development environment. It also supports development in most modern languages. Because of how it's been designed, It does require a constant internet connection.

Get up and running with GitPod

To use GitPod you'll need a GitHub, Gitlab or BitBucket account, because some of the technology requires version control underneath. For this example, GitHub is being used. Assuming the reader already has a GitHub account, go to the Gitpod home page (**www.gitpod.io**).

Select GitHub as the backend choice. It'll pop up a new window for login credentials followed by a request to authorise GitPod to interact with it. Confirm the authorisation by clicking the authorisation approval. Finally, it'll ask you to select an Editor. This author prefers *VS Code*, so we'll use VS Code Browser. Select that option, followed by continue. *VS Code* will enable the user to create code in the full fat editor if desired and it will offer that option.

After the initial login the web browser will show "No Workspaces". To create a new workspace, click the green "new workspace button". Gitpod will offer several quick-start solutions but it's possible to just paste in your own existing repository, as long as it's supported it'll import without issue and Gitpod will auto populate/import as required. For this example we're using the simple Vet and Pet database from Spring (**https://bit.ly/3zZHTTI**). Get the full URL and paste it in. This will create a brand new custom environment with a running webserver and application as is shown in the screenshot (*right*). This is your own personal copy to change as you wish.


Always ready to code.
Spin up fresh, automated dev environments for each task, in the cloud, in seconds.
Try Now
Open a workspace.
Start from any Git context.
Star
8,382

Get all the information you need at gitpod.io.

Obviously, one of the first things to customise is the look and feel. To customise the basic environment there's an important settings file called **gitpod.yml**. This must be stored in your Gitpod home folder.

The **gitpod.yml** file is central to most of the configuration settings. This file defines web port configurations, dependencies and such that need to be installed in the workspace, tweaks and other important things such as well as enabling the developer to auto-open ports. By default, on new projects this file doesn't exist. It can be created from inside the command line (the docker image in other words) by using the following command:

```
$ gp init
```

Because it already exists in our example application you can use the editor to view the **gitpod.yml** file that was used to create the settings for the application. To have more control over the initial setup the user can use `gp init -i`. This will walk through some of the options, enabling you to fine-tune some of the options.

Don't forget that the whole reason for doing this is that the workspaces are ephemeral and only last for the duration they're used. As such, manual reconfiguration for each instance isn't a route you want to go down.

At a more fundamental level the Dockerfile (the environment as a whole) can be customised to a great extent. A detailed (classic) Dockerfile can be created and placed in the home directory of the user. It must be called **.gitpod.Dockerfile** (note the leading dot).

The reason to use the custom Dockerfile option is that it can be treated as a normal Dockerfile and can use the normal framework tools of RUN, COPY, MOUNT and all the other commands, rather than just telling the system to pull the current default image in what could be a stale image or missing some components.

One can just specify the (publicly available) image to use in the **gitpod.yml** (without the ability to add in tweaks). To do this just change the image line:

```
image: node:buster
```

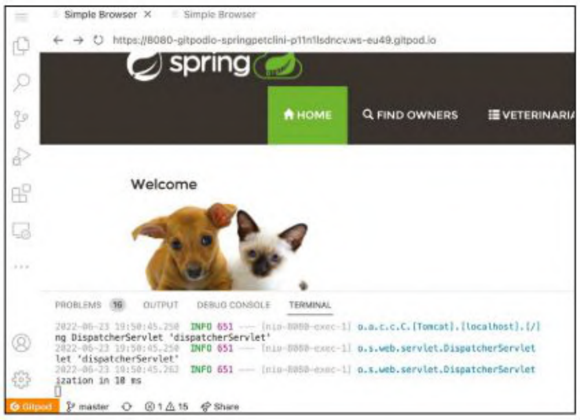
Below is a simple example of a Docker setup (taken from the Gitpods example):

```
FROM gitpod/myurl
RUN sudo install-packages binwalk clang tmux

```

Once a user has written their own Dockerfile they can build the Docker image using the command below

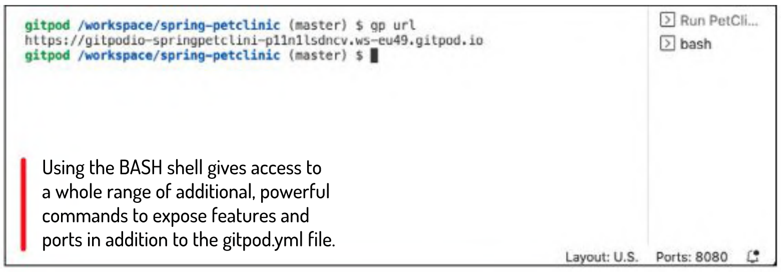
```
$ docker build -f .gitpod.Dockerfile -t dockerfile-test .
```

Test it by running interactively (basically running a



```
2022-06-23 19:50:45.250  INFO 651 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]
ng DispatcherServlet 'dispatcherServlet'
2022-06-23 19:50:45.250  INFO 651 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet
let 'dispatcherServlet'
2022-06-23 19:50:45.262  INFO 651 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet
ization in 10 ms
```

An example of using one of the pre-defined applications with GitPod that enable you to run the program and debug in the same window.



```
gitpod /workspace/spring-petclinic (master) $ gp url
https://gitpodio-springpetclini-p11n1lsdncv.ws-eu49.gitpod.io
gitpod /workspace/spring-petclinic (master) $
```

Using the BASH shell gives access to a whole range of additional, powerful commands to expose features and ports in addition to the gitpod.yml file.

bash session into the container):

```
$ docker run -it gitpod-dockerfile-test bash
```

To exit that container type `exit`. All this is great, but being able to access the application via web/http is key for development. Again, this is configured in **gitpod.yml** by looking at the **gitpod.yml** file for the PORTS section and actions section. The available actions are documented here. A snippet of the open ports is:

```
ports:
 - port: 8080
   onOpen: open-browser
```

Ranges can also be specified:

```
ports:
 - port: 3000-8999
   onOpen: Open-Browser
```

Obviously, without knowing the IP address you can't access the application. GitPod has thought about this and provided the useful `gp url` command that will give you the full URL. Alongside that there's a *gp* ports list that provides the port and URL for access. If you don't have the console that can be typed in, you may be looking at the wrong window. Make sure that the Bash option is selected when the commands are typed. This is distinct from the general console. To save these changes be sure to commit the changes to the GIT environment so that they persist. This is done within *VS Code* in the normal fashion.

In summary, Gitpod can be used to develop code remotely, with the ability to run and test in an isolated but network available environment. It may not be for everyone but is certainly worth checking out. LXF

QUICK TIP

By default, the VS Code editor in the web interface defaults to US English. To change it just click the layout bar at the bottom of the page and select British to get a British keyboard or other locale as desired.

» KEEP THINGS LOCAL

Gitpod.io is all about being able to code on almost anything from almost anywhere, with an ephemeral design but sometimes, for reasons of information security and corporate policy, all the components must stay under local control.

Luckily **Gitpod.io** does offer a locally hosted version. As you'd expect it comes as a container image to download and use. It is free to download and use for up to three users. The download and instructions for use can be found at **www.gitpod.io/docs/self-hosted/latest**. It does come with a caveat that users need to be experienced in order to use it...

Essentially it comes down to running two commands, `curl https://kots.io/install | bash` which will download and run the required install on a local Kubernetes server. It should be obvious that the server in question needs to be running Kubernetes and also have a substantial amount of resources available.

The second command `kubectl kots install gitpod` will then allow you to enter configuration details and enable any required configuration. Once it completes, it will present a web administration GUI for fine grained control and configuration.

WIREGUARD

Credit: www.wireguard.com

Host your own secure VPS-based WireGuard

David Rutland takes the LXF VPS server for a final spin. This month, he's protecting his privacy and watching Mexican TV with a self-hosted VPN.

OUR EXPERT

David Rutland is a man of mystery, and all of his online activities are shrouded under a layer of VPN enabled privacy. Is David Rutland even his real name? No-one knows... and he's not saying.

Virtual Private Networks are everywhere these days. Search any query of a vaguely technical nature, and most of the time, you'll end up on a cunningly disguised VPN affiliate site where you'll find that the answer to your problem is to use a VPN. This is largely due to exhaustive keyword research by VPN companies and pseudo-tech websites that take a substantial commission for every VPN sold.

If there's an internet-connected device within range, try this query: "How to File Taxes Online Securely in 2022". The top result on Google, Bing and DuckDuckGo is for a VPN company. With a UK IP address, it's above the **gov.uk** result and above results for genuinely useful tax software. This is clearly nonsense: VPNs, if anything, will bork the process entirely. This is because any public-facing organisation of decent size knows about VPNs, and may or may not choose to take proactive measures to prevent their effective use on that organisation's sites and services.

This author has written about them before in **LXF279**, and Jonni Bidwell did a more extensive feature on several providers in **LXF286**.

The problem

It's a sad fact that no commercial VPN company will tell you the true use to which punters put its VPNs, and its affiliates won't either – nor will we. The reason for this is simple: copyright bodies will sue the pants, shirt and shoes off anybody who they deem to be encouraging software, film or music piracy. No one wants to be named as a co-defendant in a lawsuit for contributing to IP theft, so we'll leave that part to your imagination.

Instead, the advertised features are for "protecting your privacy", or for accessing geoblocked content, which is still legally iffy, but it's unlikely that any Mouse-paid lawyer is going to get too sniffy about it. The problem is that it doesn't actually work.

This writer occasionally uses a VPN for purposes that are far too obscure to mention here, and as he does so, he likes to pop along to see what's happening on the *BBC News* homepage. Again, this is something that you can try at home.

Commercial VPN companies provide IP addresses for virtually anywhere in the world. You can physically be in Woking, while your IP address indicates that you're on a large-ish island off the Texas mainland. You can be sunning yourself on the Amalfi coast with an IP address that will swear blind that you're in a Scunthorpe basement flat.

However, if you try to catch up with the latest occurrences on *The Great British Sewing Bee*, you'll find yourself blocked – even with a UK IP address. This is because the clever tech bods down at Broadcasting House have a record of which IP address blocks are owned by VPN companies and will treat them as belonging to people who don't pay the license fee and are therefore not entitled to access Auntie's output.

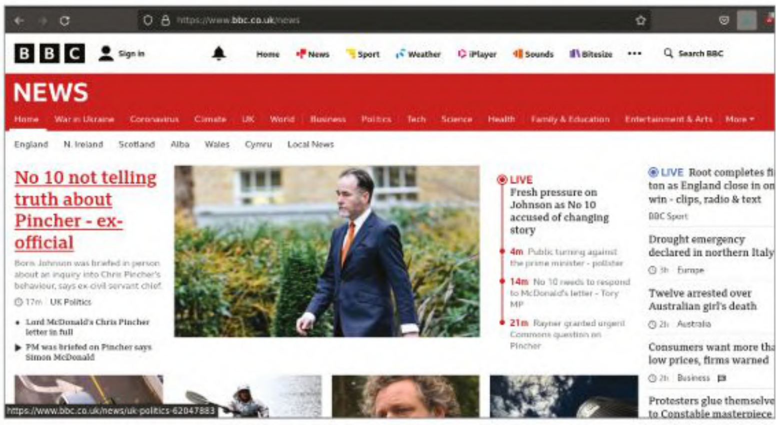
Is self-hosted better?

So when this humble scribbler (who is physically in the UK) has his VPN active and is using a UK IP address, he's directed to **bbc.com** rather than **bbc.co.uk**. It's a small difference that enables the Beeb to show advertisements and to engage in questionable tracking practices.

Netflix, Disney and other streaming services have the capability of doing this, too. They don't (for now at least) because it would probably damage subscription numbers and the bottom line. Your VPN access to US-only Netflix shows exists solely at the whim of Netflix – it knows who you are and where you are. It just doesn't care that you're cheating.

The point we're making here is that for their headline advertised features, commercial VPNs are a bit rubbish.

Aside from being able to take pride in having not



No 10 not telling truth about Pincher - ex-official

Boris Johnson was briefed in person about an inquiry into Chris Pincher's behaviour, says ex-civil servant chief

17m UK Politics

- Lord McDonald's Chris Pincher letter in full
- PM was briefed on Pincher says Simon McDonald

LIVE Fresh pressure on Johnson as No 10 accused of changing story

4m Public turning against the prime minister - pollster

14m No 10 needs to respond to McDonald's letter - Tory MP

21m Rayner granted urgent Commons question on Pincher

LIVE Root completes fi ton as England close in on win - clips, radio & text

BBC Sport

Drought emergency declared in northern Italy

3h Europe

Twelve arrested over Australian girl's death

2h Australia

Consumers want more tha low prices, firms warned

2h Business

Protesters glue themselve to Constable masterpiece

https://www.bbc.co.uk/news/uk-politics-62047883

Running your own VPN on your own VPS means you can view UK versions of websites while in the UK – a difficult task with commercial VPNs.

fallen for the deceptive marketing techniques used by commercial VPN providers, the benefits of hosting your very own VPN on your own VPS are marginal at best. While commercial VPNs claim to offer anonymity and privacy protection that will stand up to the worst a surveillance state can throw at you, for example, your VPS provider (Digital Ocean in our case) is unlikely to make such claims, and will surrender your subscriber details to any enforcement agency or rights holder that shows up with the correct legal paperwork. Don't do anything illegal. Mmkay?

A VPS VPN will keep your internet traffic safe from your ISP and from digital eavesdroppers desperate to know what sites you're visiting, and they'll only be able to see that you're connected to your VPN. This will in no way protect you from web-based trackers from the likes of Google, Facebook and other evil entities.

It's possible to change IP addressed on your VPS VPN project. It's simple, but it's also time-consuming. This means that it's possible to consume streaming media from around the world. You can even catch up with *Sherwood* on the BBC iPlayer while physically located in the UK! Because while the BBC IT department keep records of the IP address pool used by VPN providers, it doesn't appear that it pays attention to the ones leased by VPN companies – at least for the time being.

The final reason to do this VPN project is because you can. It's fun, and will give you enormous bragging rights at the next Sunday School barbecue. It'll also make you at least seven times more attractive to your gender(s) of choice.

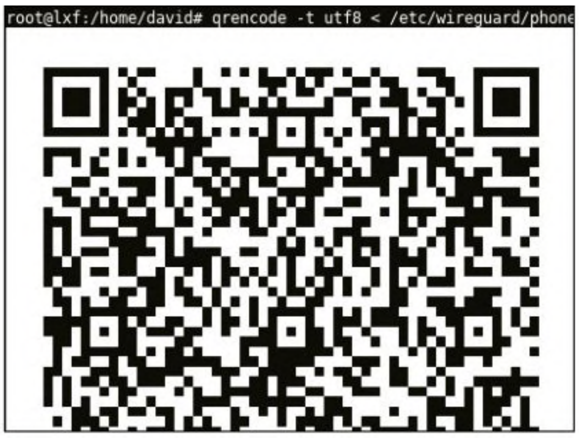
In the self-hosted VPN arena, there are two main contenders worthy of your consideration: *WireGuard* and *OpenVPN*. Both of these have their advocates and both are used by commercial VPN providers. After some humming and hawing, we eventually settled on *WireGuard*, which is faster, has better clients, handles network roaming better, and is easier to set up. The community version of *OpenVPN* is also limited in the number of devices it'll support.

Installing is easy

First, connect to your VPS using a non-root account by popping open a terminal and typing:

```
$ ssh your-user-name@your-vps-ip-address
```

and enter your password.



```
root@lxf:/home/david# qrencode -t utf8 < /etc/wireguard/phone
```

Scanning a QR code with your phone is significantly easier than inputting a 44- character random key.

Make sure that your system is up to date with:

```
$ sudo apt update && sudo apt upgrade -y
```

and reboot if there are update that require it. Fortunately, *WireGuard* is included in the default Ubuntu repositories, so to install the software, all you need to do is pop the following command into your terminal:

```
$ sudo apt install wireguard
```

and the package manager will install *WireGuard* and the *WireGuard* tools which will be used to manage the tunnel interface. Now open another terminal on your local machine, but don't connect to your VPS. Instead, you'll again tap in:

```
$ sudo apt install wireguard
```

As simple and smooth as the actual software installation was, there's still work to do, and we'll begin by creating private and public keys to establish a secure connection. Hop back into the terminal on your VPS and

```
$ wg genkey | tee privatekey | wg pubkey > publickey
```

The first part of this command cause *WireGuard* to generate a private key, which is then piped to the `tee` command, which writes the private key to a new file called **privatekey**. The private key is then piped to **pubkey**, which generates a new public key that's echoed to a new file called **publickey**. You don't need to name your keys anything so obvious, but it helps to keep track of them. `ls` to quickly check that your newly generated key files are saved in your current directory.

It should go without saying that your private key should be kept super secret and never revealed to anyone. Not even if they ask nicely. With it, your server can decrypt all packets sent to it by clients, and if a third

QUICK TIP

Read our original VPS features on how to choose and set up one in LXF281 and LXF282 at https://bit.ly/lxf281lxfserver and https://bit.ly/lxf282lxfserver.

» COMMERCIAL VPNS

Believe it or not, we're not shills (*open to considering offers! – Ed*) for VPN companies, although among tech news and features publications we're probably alone in this. Nonetheless, a VPS is hardly ideal for the uses to which most people would put a VPN.

As stated in the article, your VPS is inextricably linked to you, and any nefarious activity you get up to – whether you're leaking state secrets to journalists, sending hate mail to politicians, taking a deep dive into GCHQ archives, or being alarmingly politically incorrect on Twitter – can be traced back to you. Some VPS providers offer completely anonymous access, payment via Bitcoin, server locations scattered across the planet, and encrypted disks. They'll swear blind that they don't keep logs and don't cooperate with the authorities.

Regardless how true these claims are, the costs associated with this type of VPS are astronomical. They're far more than the $5/month *Linux Format* VPS, and orders of magnitude more than the cheapest offerings from an established commercial VPN provider.

Running your own VPN on your own VPS is a fun project, but if you're planning on doing something that could lead to you losing your liberty, we'd suggest using an off-the-shelf offering instead. Most claim not to keep logs (for whatever that's worth) and several accept payment by cryptocurrency. Visit any tech site on the internet and we're sure there will be thousands of articles extolling their benefits.

»

QUICK TIP
If you're going to be using your VPN a lot, consider mapping wg-quick up and wg0 wg-quick down wg0 to shortcut keys – it'll save wear and tear on your fingers.

party manages to get hold of your private key, they can decrypt your packets, too. Next, type the following:

```
$ cat privatekey
```

and copy the output to your clipboard. To configure the *WireGuard* tunnel interface create a new file with:

```
$ sudo nano /etc/wireguard/wg0.conf
```

and enter:

```
[Interface]
```

at the start of the file. On the next line you'll enter the private key you copied earlier. In our (now destroyed) VPS, the private key was the following:

```
sKlPaO2gABnus6x9u4hN4VRMiop
YHEKIbxEAPNzEyFU=
```

so we entered:

```
PrivateKey=sKlPaO2gABnus6x9u4hN4VRMiopYHEKIb
xEAPNzEyFU=
```

The next line should contain the private IP address range you want to use. Type `Address=10.0.0.1/8` and then `SaveConfig=true`.

Now come two commands that set *iptables* rules to accept every incoming packet on the tunnel interface and forward outgoing packets, while keeping them hidden behind the IP address of your VPS server.

```
PostUp=iptables -A FORWARD -i wg0 -j ACCEPT;
iptables -t nat -A POSTROUTING -o eth0 -j
MASQUERADE;
PostDown=iptables -D FORWARD -i wg0 -j ACCEPT;
iptables -t nat -D POSTROUTING -o eth0 -j
MASQUERADE;
```

The standard listening port for *WireGuard* is 51820. You can change this if you want to, but for the sake of simplicity, we went with the defaults.

```
ListenPort=51820
```

Save and exit with Ctrl+O then Ctrl+X.

You should now start the network interface with

```
$ wg-quick up wg0
```

Entering `sudo wg` should show the interface name, your (hidden) private key, your public key, and the port that *WireGuard* is running on. If it doesn't, something has gone wrong.

Assuming that everything is as it should be and you don't need to go back and double-check your **wg0.conf**, head back over to the terminal for your local machine where you'll generate yet more private and public keys – this time for the client:

```
$ wg genkey | tee privatekey | wg pubkey > publickey
```

Again, copy the private key to your clipboard, then

```
$ sudo nano /etc/wireguard/wg0.conf
```

The contents of this file will be similar but different to the equivalent file on your VPS:

```
[Interface]
PrivateKey=your-locally-generated-private-key
Address=10.0.0.2/8
SaveConfig=true
[Peer]
```

Now append the public key of your *WireGuard* server and the IP address of your VPS, together with the *WireGuard* port, as the endpoint. In our case, this means adding the following:

```
PublicKey=Hi9SEoymiAzmCKaMOzftfqBrTGNeELq6oJ
pY79z7TEI=
Endpoint=206.189.30.160:51820
```

To forward all traffic from the client through the tunnel to your VPN, add:

```
AllowedIPs=0.0.0.0/0
```

Save and exit with Ctrl+O then Ctrl+X. Type:

```
$ wg-quick up wg0
```

This will create the tunnel, although it isn't yet connected to the *WireGuard* server, and you'll notice that you're no longer able to visit any websites. Shut down the tunnel with:

```
$ wg-quick down wg0
```

Now we need to tell the server that your local machine is allowed to connect to and through it. Copy the client's public key, and once again, hop back over to the terminal for your VPS. Enter:

```
$ sudo wg set wg0 peer your-client-public-key allowed-
ips 1.0.0.2/32
```

By default, *WireGuard* won't try to keep connections alive, and this can cause all kinds of problems – especially if you're using firewalls or any other kind of gateway. If the connection closes, *WireGuard* won't try to reopen it. But all is not lost and we can instruct the client to keep the connection to the server alive.

Back in the terminal of your local machine, type:

```
$ sudo nano /etc/wireguard/wg0.conf
```

and then add to the config file:

```
PersistentKeepAlive=10
```

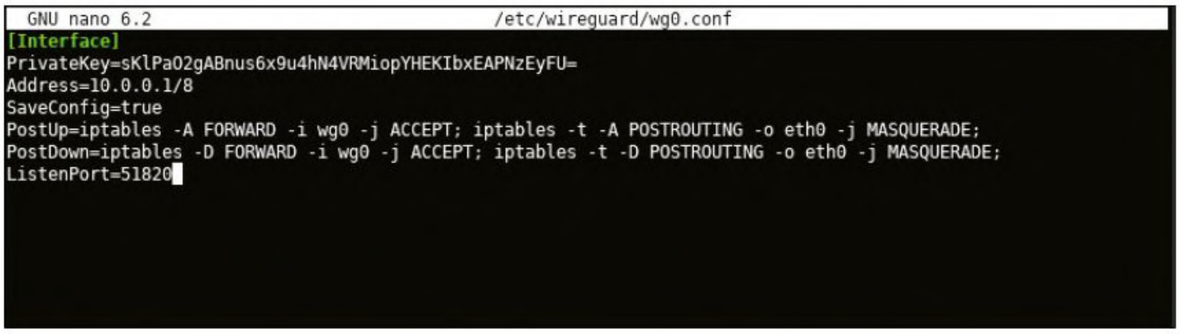
The '10' figure is arbitrary and means that the client will send a keep alive request to the server every 10 seconds. In all honesty, every 10 seconds is probably overkill. You can change it to whatever you want and fine-tune for whatever works best for you.

Save and exit with Ctrl+O then Ctrl+X and start the tunnel again with:

```
$ wg-quick up wg0
```

All internet traffic from your local machine is now passing through your very own VPN on your very own VPS. How neat is that?

If the IP address of your VPS is in another country (as ours is), you'll start seeing adverts and content specific to that location. Blocked from California local newspapers because of GDPR concerns? Not any more, and you can snarf down celebrity gossip to your heart's content, safe in the knowledge that your ISP will never have a clue what you're up to!



```
  GNU nano 6.2                                    /etc/wireguard/wg0.conf
[Interface]
PrivateKey=sKlPaO2gABnus6x9u4hN4VRMiopYHEKIbxEAPNzEyFU=
Address=10.0.0.1/8
SaveConfig=true
PostUp=iptables -A FORWARD -i wg0 -j ACCEPT; iptables -t -A POSTROUTING -o eth0 -j MASQUERADE;
PostDown=iptables -D FORWARD -i wg0 -j ACCEPT; iptables -t -D POSTROUTING -o eth0 -j MASQUERADE;
ListenPort=51820
```

The server conf file tells WireGuard who it is, verifies its identity, and tells it what it's supposed to be doing with incoming and outgoing packets.

Google
How to File Taxes Online Securely in 2022
All Videos News Shopping Images More Tools
About 1,110,000,000 results (0.44 seconds)
https://www.privateinternetaccess.com › blog › how-to-...
How to File Taxes Online Securely in 2022
13 Apr 2022 — You can **file** your **taxes** through old-school snail mail or **online**. Paper **filing** is more **secure**, but is much more difficult. You'll need to print ...
When can you start filing your taxes in 2022?
How do I start filing taxes?
People also ask

VPN companies are so efficient at deceptive SEO marketing that they've rendered search engines effectively useless.

Changing locations

The great thing about a VPS is that it gives you a static IP address. This makes it super easy to do things like setting up email servers that are trusted by other providers, deploy websites easily, and securely deploy a NextCloud instance. It's not, however, great for deploying a VPN if your main purpose is to watch TV shows from other parts of the world.

With a regular commercial VPN, changing server locations is simple. You simply select your preferred virtual domicile from a drop-down list, hit a button, and boom, you're in San Cristóbal de las Casas streaming reruns of *Una familia con suerte*.

Alter your IP with care

Changing the IP address of your VPS is not so easy, and it's probably not something that you particularly want to do if you're hosting anything else on the VPS. Any websites you host will immediately 404, your emails will be marked as spam, and NextCloud won't even trust itself. This will take a very long time to sort out and make you grumpy for the rest of the day while you repair your cloud infrastructure. Not optimal.

To change your VPS IP address with Digital Ocean, you first need to take a snapshot of your existing droplet. You can do this by selecting the Snapshots tab in the server control panel. Now destroy the droplet, and select Recreate from Snapshot.

The entire process takes several minutes, and you don't have a vast choice as to where the new IP address of your VPS will be located. All in all, it's not ideal – try to pick a VPS provider that has server locations in multiple countries or you'll be out of luck.

WireGuard is go

Very few people access the internet solely through their desktop Linux system, whereas most people will have a mobile phone about their person 24 hours per day – ever-ready to query Wikipedia on popular trivia in the pub toilet.

Whatever paranoia caused you to protect traffic from your main machine with a VPS-hosted instance of *WireGuard* will doubtless be anxious to see your mobile devices protected, too.

Fortunately, there are a number of mobile apps available which mean that you won't need to force your oversized fingers to interact with the teeny tiny keyboard on your smartphone – at least too much.

The best app for Android – at least in our opinion – is the official *WireGuard* app, which is available from the iOS store, the Google Play store, and for privacy minded people like you, from F-Droid.

Before you grab the APK, you'll want to generate (yet more) keys for your mobile devices. On your VPS type

```
$ wg genkey | tee phone-privatekey | wg pubkey >
phone-publickey
```

Copy both the public and private keys somewhere safe for now. Create a new conf file for your device with:

```
$ sudo nano /etc/wireguard/phone.conf
```

and much as before, type:

```
[Interface]
PrivateKey=your-locally-generated-server-private-key
Address=10.0.0.2/8
SaveConfig=true
[Peer]
PublicKey=public-key-for-your-phone
Endpoint=206.189.30.160:51820
AllowedIPs=0.0.0.0/0
```

Save and exit with Ctrl + O then Ctrl + X. Next, add your phone's public key:

```
$ sudo wg set wg0 peer your-phone-client-public-key
allowed-ips 1.0.0.2/32
```

Now restart *WireGuard* with:

```
$ wg-quick down wg0
```

then type:

```
$wg-quick up wg0
```

Make the connection

To generate a QR code to make it easy to connect your Android or iOS device to your VPN, you'll need an extra piece of software:

```
$ sudo apt install qrencode -y
```

Once it's installed, become root:

```
$ sudo su
```

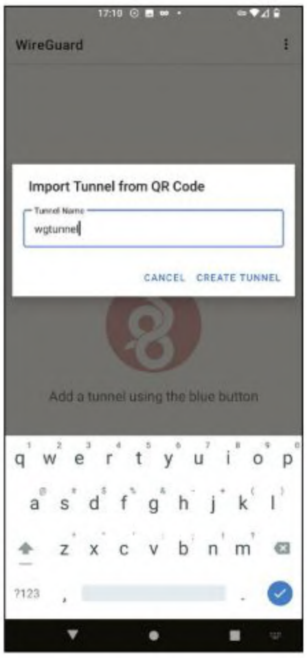
then `cd` to **/etc/wireguard** and:

```
$ qrencode -t utf8 < /etc/wireguard/phone.conf
```

As if by magic, a QR code will appear in your terminal. Pick up your phone handset, and once you have downloaded and installed the *WireGuard* app from your store of choice, tap on the blue button to add a tunnel. The app offers you the convenience of scanning a QR code, importing from a file, or creating from scratch. Click the QR code option, and when prompted, name the tunnel and click the toggle to ON.

That's it. Your phone, and as many computers as you like are now protected by your VPN. LXF

QUICK TIP
We can't state this enough – don't engage in any illegal activity on your VPS-hosted VPN. You will get caught. Don't say we haven't warned you – we have. Repeatedly.


WireGuard
Import Tunnel from QR Code
Tunnel Name
wgtunnel
CANCEL
CREATE TUNNEL
Add a tunnel using the blue button

Once you've scanned the QR code you generated earlier, all you need to do is name your new tunnel and click a toggle to connect.

» **GIVE ALL YOUR DETAILS TO US!** Subscribe now at **http://bit.ly/LinuxFormat**

AWOOGA, AMIGA

Jonni Bidwell's desk has become a retro gaming station, thanks to Dimitris Panokostas and his Amiberry emulator.

A huge number of readers, we reckon, will fondly look back on the Commodore Amiga 500. Sure, some may remember further back, to the C64s, Spectrum or even the BBC Micro. But the A500, especially when viewed through the roseate lens of nostalgia, was truly a marvel. Around £300 in 1992 would get you one of these state-of-the-art machines, and you'd be able to enjoy great games from The Bitmap Brothers, Gremlin Interactive, Team17 and other great British software houses. Thirty years hence, and no machine (*the Deli's free coffee machine?–ED*) has ever made us at *LXF* Towers so happy.

But that's no matter, because while our younger colleagues discuss memes and web3 on social media platforms that make no sense to us, we can content ourselves with using our new-fangled hardware to emulate the machines of old. Amiga emulation has been popular, though not necessarily easy, since 1995 when *UAE* (originally the *Unusable Amiga Emulator*) was first brought to life. Today, *UAE* lives on, having made it from Linux to Windows (in the form of *WinUAE*) and from x86 machines to ARM, by way of *UAE4ARM* (the first Amiga emulator to run on the Pi).

Now, thanks to the Pi 4, and in no small part to Dimitris Panokostas' hard work creating *Amiberry* (a lighter, faster *UAE4ARM* fork) Amiga emulation is in better shape than ever. Dimitris was good enough to spend some time talking to us about the project's origins, its current state of play and its plans for the future.

We're big Amiga fans here at Future Towers. We all read *Amiga Format* growing up, and somehow went on to write, layout and edit *Linux Format* today. It's good, then, that the Amiga emulation scene (unlike the current state of specialist tech magazines) is so alive and well. The original *UAE* (then the *Unix Amiga Emulator*) was written in 1995 and has been ported and forked numerous times. Today, that name lives on in the forms of *FS-UAE*, *WinUAE*, *PUAE* and *UAE4ARM*, although *UAE* doesn't really stand for anything anymore.

Dimitris had been a fan of Amiga emulation long before he started hacking on *UAE4ARM* in the mid-2010s, as he explains. "When the first Amiga emulators came out, we mostly laughed at it and saw how hard it was to emulate a humble A500 at decent speed, even on the fastest Intel processors of the time. It's no wonder the

UAE acronym stood for *Unusable Amiga Emulator* in the beginning. But of course, as the hardware got better and faster, and the software matured further, it eventually changed from unusable to usable, to not bad, to quite good, leading to pretty cool and faster than the real thing, nowadays – for example, if you use the JIT (Just In Time) and RTG (Retargetable graphics) options."

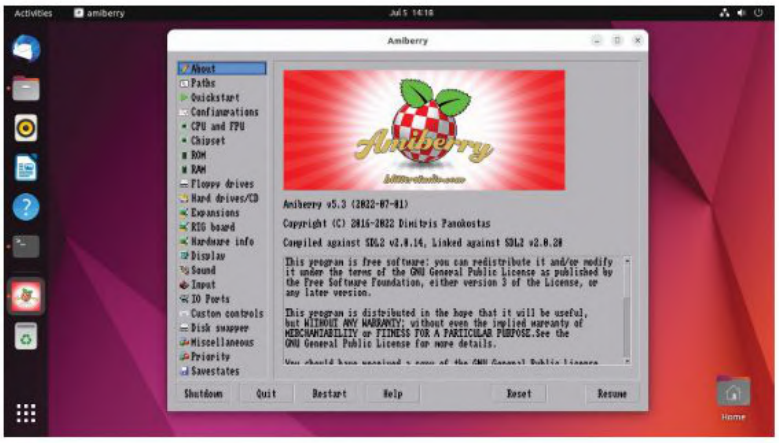
Much of the early emulation relied on developers having access to hardware, which (since no Amigas were produced after Commodore's liquidation in 1994) meant scouring auction sites and yard sales in the hopes of finding working machines. As time went on, this became increasingly difficult. "In the meantime, the original Amiga hardware got older, more fragile and harder to find, which made it more expensive, too. You can maintain it by recapping of course, and some third-party hardware upgrades help to keep things up to date (for example, USB ports, modern graphics cards, new accelerators, SD card drives), but it gets quite expensive quite fast."

A Raspberry Pi ripe for emulation duties

Like many of us, Dimitris (aka midwan) was taken with the original Raspberry Pi. But while most of us foundered over the dual pin-numberings and became frustrated by slow SD card access, he was devising a plan. "I thought that it would be really cool if we could get Amiga emulation running on them. They were quite affordable, easy to find and with very good software support. The problem was, they weren't powerful enough in their first iterations, so we were back to the old problem again. It wasn't until the Pi 2 came out, when it started becoming feasible to run ARM-specific versions of *UAE* with some decent speed. The Pi 3 made things better, but some cases were still too slow – the Pi 4 was fantastic in that respect, since almost everything can run full-speed now, and we could even add more features (like Cycle-Exact emulation for the 68000)."

Retro gaming, even on the first Pis, very quickly became a popular pastime "When I first got involved in the scene of Amiga emulation on the Raspberry Pi, things were still very young. Lots of different distros were coming out, customised specifically for running Amiga emulation.

"I first got involved in those, trying to make them better, or at least closer to what I'd like to use. But eventually it became clear that it was the emulator itself that needed the most work. I didn't want to do everything myself, so I left the distro part to others, and I decided to start working on the *UAE4ARM* code and see if I can improve things further there.



Amiberry v5.3 (2022-07-01)

Copyright (C) 2016-2022 Dimitris Panokostas

Compiled against SDL2 v2.0.14, Linked against SDL2 v2.0.20

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.See the GNU General Public License for more details.

Dimitris has styled the Amiberry UI in the classic Topaz font, so you feel like you're in Workbench before you've even started.

The *Amiberry* project began in earnest in 2016. At the time *UAE4ARM* was based on an old version of *WinUAE*, so Dimitris decided to modernise it. "My fork of *UAE4ARM* started evolving much faster than the original. I wanted to bring it closer to the latest *WinUAE* standards, besides the numerous bug fixes that I was implementing. So, to avoid confusion, I named my project *Amiberry* to separate it out from the original *UAE4ARM*, and continued development under that name. I had to freshen up my C/C++ skills, but that was one reason I wanted to get involved in this project to begin with. I usually try to find a real-life project that I can use, to learn or improve my skills on something."

We asked Dimitris if he also had to learn 68k assembly for his new mission, to which he replied, "No, luckily. I much prefer C/C++ syntax. However, there were some parts written in ARM assembly, for maximum performance. I didn't write those myself, and some of them ended up having bugs so I replaced them with their standard C/C++ versions eventually." But the UAE codebase is sprawling, sometimes unwieldy and as, he explains, not pretty.

"One thing that was (and still is, to be honest) a bit hard, is that the codebase goes back many years, and comes from various people. There's a lot of ugly code in

» WHAT THE WHDLOAD?!

"*WHDLoad* is an excellent tool that was written by Bert Jahn (**https://whdload.de**) and it's essentially a system-friendly hard disk installer," says Dimitris. "It allows you to install a lot of games and demos, which were originally designed to run only from floppy disks.

"The *WHDLoad* booter in *Amiberry* is a great feature that was made possible by community contributions (specifically, HoraceAndTheSpider). The idea is to load a pre-installed *WHDLoad* game from an .LHA archive, in a similar way as you would from an ADF disk image. Only with *WHDLoad* there's no disk-swapping, of course – and in many cases the games are patched to fix known issues that the ADF versions might have had.

"Additionally, there's a mechanism in *Amiberry* that will try to detect the *WHDLoad* title you're loading against a list of known titles, and if found, it'll change the emulator settings to the best-known for that specific case. Amiga emulation covers many different models and each one had different hardware. Some games were designed to run on A500 models, while others were meant for the A1200 or the CD32.

"Using this mechanism allows you to have a single click solution to run your games, without worrying about swapping disks or tweaking emulator settings for that game. It works great with front ends like *RetroPie*, since it can be triggered from the CLI. But it also works the same way from the *Amiberry* GUI, since you can choose a *WHDLoad* title from the Quickstart panel and click Start."

»

there, which I can't really change a lot or risk merge-hell when there's a new update from *WinUAE*. I try to improve things on the *Amiberry*-specific parts, and report issues back to *WinUAE*, but it's still far from the 'clean code' standards that I normally would write."

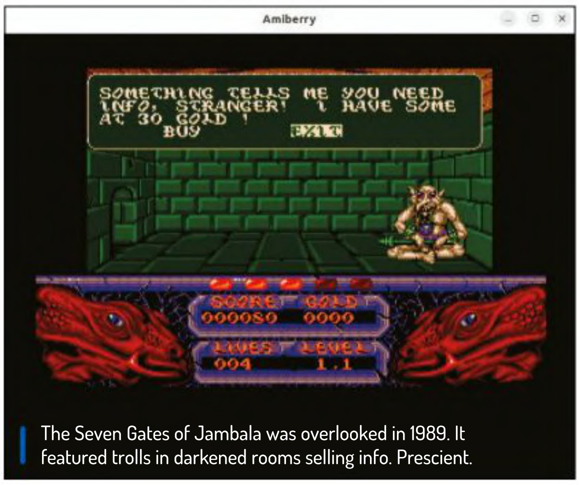
WinUAE is almost certainly the most popular Amiga emulator on Windows. It's fairly fully featured, but as Dimitris explains some of its features weren't included in *Amiberry*. "The intention was to implement as much as possible, while keeping the performance at acceptable levels. Even on the latest and fastest ARM boards, we cannot have everything that *WinUAE* has – for example, PPC emulation is a no-go (would be way too slow to be useful).

"The JIT implementation is very different, since it's designed for ARM CPUs instead of x86. It originally comes from the Aranym project (**https://aranym.github.io**), modified for the Raspberry Pi specifically by TomB, on his own *UAE4ARM* fork. I've tried to keep the core emulation almost identical to the latest *WinUAE* versions, minus a few things that wouldn't make much sense on devices like the Raspberry Pi: multiple monitor support, multiple expansion cards emulation, multiple RAM boards, debugger, variable sync rates, beam-racing techniques… all of these are not very useful here."

But *Amiberry* is far from just being a cut-down *WinUAE*. Rather, it aims to improve on some of the latter's shortcomings. "I've tried to improve on areas that I think *WinUAE* could do better: there's a *WHDLoad* booter, better support for command line options, direct support for RetroArch configurations, controller Custom mappings, Auto-Crop, and many more features that are unique to *Amiberry*. There are also things that *WinUAE* offers that might eventually make it into *Amiberry*, as newer/faster hardware becomes available. For example, better CPU compatibility (we currently have 68000 cycle-accurate mode only, but we could have it for 68020 and other models as well in the future), more accurate chipset emulation (for example, cycle-exact memory emulation), and so on.

If it weren't for Dimitris Panokostas, Jonni might have actually had to do some work for this feature.

"The general rule is that if I can implement something from *WinUAE* which doesn't affect performance, I'll try to do it. And if there's a better way to do something, I'll look into that and also offer it upstream to *WinUAE*. It's a continuous journey of iterations, which make the software better over time. You could say it's a full-time job." It sure sounds like it.


Amiberry
SOMETHING TELLS ME YOU NEED INFO, STRANGER! I HAVE SOME AT 30 GOLD !
BUY
EXIT
SCORE GOLD
000080 0000
LIVES LEVEL
004 1.1
The Seven Gates of Jambala was overlooked in 1989. It featured trolls in darkened rooms selling info. Prescient.

CREDIT: Thalion Software (closed)

How to handle 68k emulation

We asked Dimitris if the x86 architecture is better equipped to handle 68k emulation than ARM machines, or whether it boiled down to memory and megahertz. The game (and emulation thereof) is changing, he explains: "Memory isn't that much of an issue anymore, since even a 1GB board can do it (at least for the basics).

"The biggest bottleneck currently is single-core CPU performance, and GPU performance. Although many parts of the code are running on separate threads (where it makes sense), there are critical pieces in custom chipset emulation that run on the same thread. If a game does fancy stuff and makes heavy use of certain features, the performance will be held back by

» YOU WOULDN'T DOWNLOAD AN AMIGA…

The Amiga Kickstart ROMs are still copyrighted, but can be legally purchased from Cloanto's Amiga Forever. You need these files to do almost anything with an Amiga emulator. As well as that you'll need some disk images (most commonly in the form of ADF images). Some websites have been given permission by the original developers to distribute their games. For example, Team17 titles are available from **http://dream17.abime.net** and Gremlin games are available at **http://gremlinworld.emuunlim.com/amiga.htm**. And of course, *Amiga Format* cover discs from back in the day are easy to find.

We asked Dimitris where else interested readers can find Amiga software, legally. "Amiga Forever is the legal way of obtaining Amiga Kickstart ROMs (and AmigaOS 1.3-3.1), if you don't have them already. Another way is to copy them from the real hardware, if you have it around (there are special tools for this task available on Aminet).

"There's a new version of AmigaOS 3 out by another company (Hyperion), if you want to explore that as well. The current version is AmigaOS 3.2.1, and it comes with its own Kickstart ROM.

"If you want to emulate a graphics card and use a desktop with high-resolution 24-/32-bit colour, then you'll also need *P96* installed – there's a free version available on Aminet, while Individual Computers continues to develop newer versions commercially, as well. Both will work with *Amiberry*. And then there's the free and open-source AROS project, which also works fine on *Amiberry*. The AROS kickstart ROM is included in the *Amiberry* repository.

"Regarding games, there are also the Factor 5 classics that are available for free at **www.factor5.com/downloads.shtml**. And of course, Aminet has a big collection of games, among other things: **https://aminet.net**."

Amiberry is included in popular Pi-gaming distros such as RetroPie and Batocera (pictured here running the 1992 classic Agony).

CREDIT: Psygnosis, Sony

how fast that one CPU core can execute those operations. A classic example of this scenario is the game *Jim Power* and specifically the title screen, which uses a parallax scrolling technique in the background. That calls some sprite handling routines quite heavily, so one CPU core is trying to cope with it and could hit 100 per cent usage, while the others are at 20-30 per cent. On anything less than a Pi 4, that usually struggles to keep up with the expected 50 frames per second that the PAL Amiga emulation requires.

"Similar with the GPU performance: we draw pixels line by line in a memory buffer and copy that buffer as a texture to the GPU for display. That needs to happen at least 50 times per second (for PAL emulation), otherwise the emulation will be slower than expected. Depending on the GPU driver and back-end used (through SDL2), the performance from platform to platform differs quite a lot. X86 is usually more powerful than ARM devices in that respect, but things are changing quite fast lately. I expect that it won't be much of an issue going forward, judging by the latest generation of platforms."

Anyone who's dabbled with Amiga emulation will know that one memory it stirs is that of endless disk swapping. The various *UAEs* can emulate up to four floppy drives, but that won't really help if the game your trying to play only supported one or two.

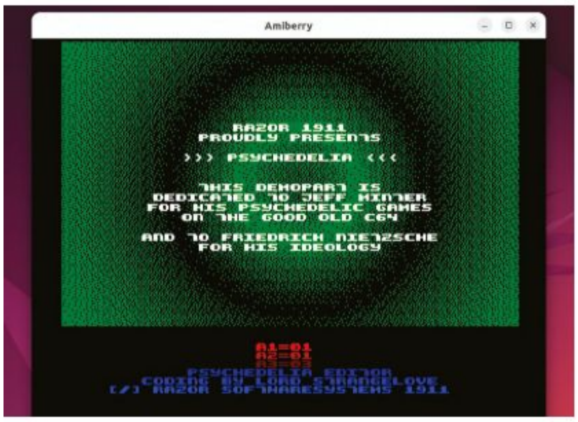
Dimitris explains that swapping disks is a little easier in *Amiberry* than it was in real life. "It has a GUI that by default opens with F12, and it includes a Disk Swapper panel as well (straight from *WinUAE*) in which you can add multiple ADF/IPF/DMS/etc. disk images and use keyboard shortcuts to insert them in the emulated floppy drives. But the easiest and best solution would be to use *WHDLoad* games." See the box (*page 79*) for more on that.

PAL EMULATION AT 50 TIMES/SEC

"We draw pixels line by line in a memory buffer and copy that buffer as a texture to the GPU for display."

Everyone's got their own favourite Amiga titles. We regularly like to unleash our own inner murder owl with Psygnosis' legendary *Agony*, And who can forget the legendary Bomb the Bass soundtrack of *Xenon 2: Megablast*? Anyway, rather that segue into a nostalgic reverie we figured we should instead ask Dimitris about any obscure games he'd encountered (so that we too could play them).

"There have been a few over the years, that I liked. Since we're talking about obscure ones, *Doman* was one of them (**https://en.wikipedia.org/wiki/Doman:_Grzechy_Ardana**). I enjoyed playing this in two-player co-op mode with my brother. We didn't understand any of the text (it was in Polish), but it didn't matter. Another obscure favourite one was *Powerplay: The Game of the Gods*, for which there was a Greek version out also (**www.lemonamiga.com/games/details.php?id=1456**). It was a game of questions, with a chess-like board where the pieces originated from Greek mythology. Answer a few questions correctly, and your piece would get upgraded to a higher-level entity, great stuff!" LXF



```
RAZOR 1911
PROUDLY PRESENTS

>>> PSYCHEDELIA <<<

THIS DEMOPART IS
DEDICATED TO JEFF MINTER
FOR HIS PSYCHEDELIC GAMES
ON THE GOOD OLD C64

AND TO FRIEDRICH NIETZSCHE
FOR HIS IDEOLOGY

A1=01
A2=01
A3=03
PSYCHEDELIA EDITOR
CODING BY LORD STRANGELOVE
(C) RAZOR SOFTWARESYSTEMS 1911
```

The Razor 1911 demo group is still around today, but this humble message comes from their 1990 release, Psychedelia.

72
techradar.

HotPicks

Speek » zuluCrypt » BpyTop » Converseen » inxi » Qrencode » Espanso » nudoku » Powder Toy » Cerebro » Flatseal

Mayank Sharma
After being given an offer he couldn't refuse, has the delight of balancing his first-born on one knee and the ever growing LXF workload on the other…

SECURE MESSENGER

Speek

Version: 1.6.0
Web: https://speek.network

Speek is a privacy-focussed chat program that ferries messages via the Tor network. For added privacy, all messages sent through the application are end-to-end encrypted.

The program is available as an AppImage, so grab the latest version from its website, make it an executable with `chmod +x Speek*.AppImage`, and then double-click the file to launch it from the file manager or use `./Speek*.AppImage` from the terminal to launch the messenger.

On launch, *Speek* will offer you the option to either bring up the client with the default settings, or tweak them with the Advanced Network Configuration option. Choosing the first option will take you directly to the messenger's chat window, while the latter will drop you on the network settings page, where you can configure settings for proxy type, address, port, username, password and more, based on your requirements.

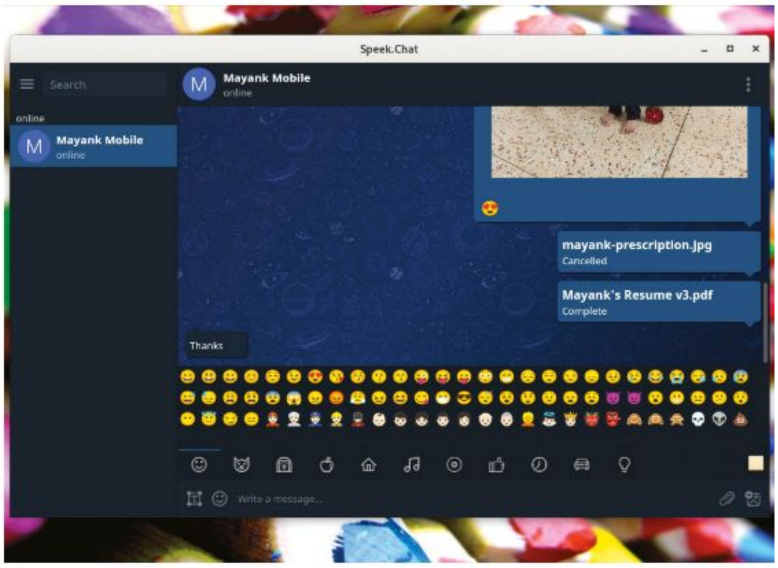
Speek is a decentralised messaging tool that doesn't require you to sign up for a username or ID, using an email address or a phone number. Instead users connect using their contact's *Speek* ID.

To view your ID, click the hamburger menu and select the View Speek ID option. You can then copy it and pass it on to your contacts. And when you receive an ID for a contact, again head to the hamburger menu and select the Add Contact option.

Once you've connected with friends, you can start exchanging messages and files with them, just like you would with any other messaging program.

Speek is cross-platform and besides supporting Linux, also runs on macOS, and Windows, as well as on Android. Its latest pre-release version has also added the option to conduct group chats. Note, however, that the option to initiate and host group chats is currently restricted to desktop clients.

For additional levels of privacy and anonymity, you can run multiple instances of *Speek*, each with a different *Speek* ID.



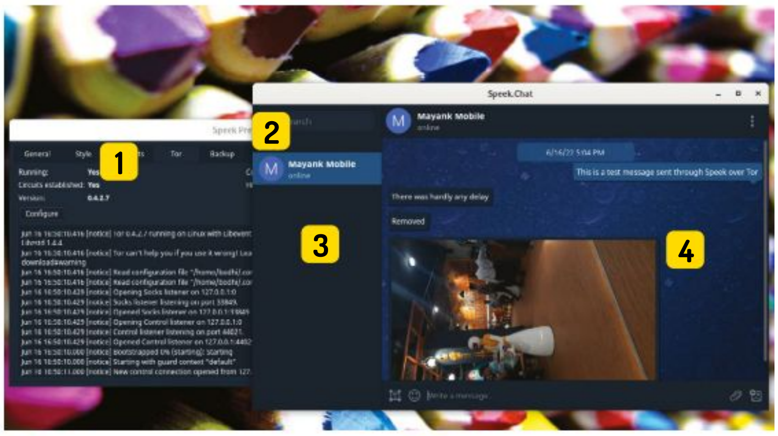
mayank-prescription.jpg
Cancelled

Mayank's Resume v3.pdf
Complete

Thanks

Once connected, Speek behaves pretty much like any other instant messaging client.

FIND YOUR WAY AROUND SPEEK



Running: Yes
Circuits established: Yes
Version: 0.4.2.7

```
Jun 16 16:50:10.416 [notice] Tor 0.4.2.7 running on Linux with Libevent
Libevent 1.4.4
Jun 16 16:50:10.416 [notice] Tor can't help you if you use it wrong! Lea
downloadawarning
Jun 16 16:50:10.416 [notice] Read configuration file "/home/bodhi/.co
Jun 16 16:50:10.416 [notice] Read configuration file "/home/bodhi/.co
Jun 16 16:50:10.429 [notice] Opening Socks listener on 127.0.0.1:0
Jun 16 16:50:10.429 [notice] Socks listener listening on port 33849.
Jun 16 16:50:10.429 [notice] Opened Socks listener on 127.0.0.1:33849
Jun 16 16:50:10.429 [notice] Opening Control listener on 127.0.0.1:0
Jun 16 16:50:10.429 [notice] Control listener listening on port 44021.
Jun 16 16:50:10.429 [notice] Opened Control listener on 127.0.0.1:4402
Jun 16 16:50:10.000 [notice] Bootstrapped 0% (starting): starting
Jun 16 16:50:10.000 [notice] Starting with guard context "default"
Jun 16 16:50:11.000 [notice] New control connection opened from 127.
```

6/16/22 5:04 PM
This is a test message sent through Speek over Tor
There was hardly any delay
Removed

1 Preferences Use this menu to configure the Tor network, and tweak the appearance of the application.

2 Main Menu Head here to the hamburger menu to view your Speek ID and to add contacts, and to switch to a new identity.

3 Contacts This is the list of your added contacts. Right-click a contact to edit their details, and export chats.

4 Chat Window This is the main chat window, which acts just like it does on any other chat program, don't mix up your Whatsapp chats!

ENCRYPTION TOOL

zuluCrypt

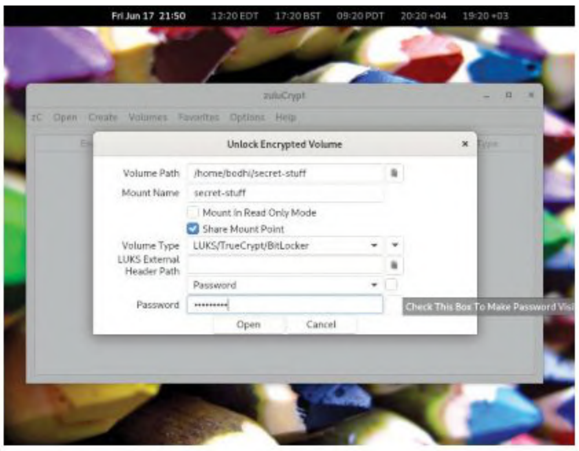
Version: 6.1.0 **Web:** https://mhogomchungu.github.io/zuluCrypt/

The only reliable mechanism to keep your personal data to yourself is to encrypt it. Sure, working with encrypted data is an involved process, but *zuluCrypt* will simplify the process to a large extent, and help you reinforce the security of your data.

While you can find *zuluCrypt* in the repositories of some Linux distros, such as Fedora, and Ubuntu, the project suggests that you skip these and instead use pre-compiled binaries from its downloads page.

zuluCrypt has both a graphical and a command-line instance. The graphical instance has an intuitive, easy-to-follow interface. Using the application you can also create an encrypted disk within a file, a partition and even USB disks. It can also encrypt individual files with the command line tool *GnuPG*.

zuluCrypt can also be used for block device encryption, which means that the program can encrypt everything written to a certain block device. The block device can be a whole disk, a partition or even a file that's been mounted as a loopback device. With block device encryption, the user creates the file system on



Volume Path: /home/bodhi/secret-stuff
Mount Name: secret-stuff
Mount In Read Only Mode [ ]
Share Mount Point [✓]
Volume Type: LUKS/TrueCrypt/BitLocker
LUKS External Header Path:
Password
Password: ••••••••••
Check This Box To Make Password Visi

Use zuluCrypt to create a hidden container inside a hidden volume for added security.

the block device, and the encryption layer transparently encrypts the data before writing it to the actual lower block device.

While encrypted, the storage blocks just appear like a large blob of random data and doesn't even reveal its directory structure. To work with the encrypted container, as with a conventional drive, you must mount it manually on the system. When you double-click a container, *zuluCrypt* mounts it as a new folder; the drive name is used as the folder name in your home directory. You can now work with the container like any conventional directory.

Once you've spent time familiarising yourself with *zuluCrypt*, you can use it through its CLI tool to speed up and script tasks.

RESOURCE MONITOR

Bpytop

Version: 1.0.68 **Web:** https://github.com/aristocratos/bpytop

Think of *bpytop* as a super-charged version of the *top* utility for monitoring the resources on your computer. The CLI utility helps gauge and identify processes that are hogging the resources on your computer.

bpytop is written in Python, and in addition to working on Linux, is also available for macOS. You'll need Python 3, version 3.6 or later, and the *psutil* module installed on your system for *bpytop* to work without problems.

There are various ways of installing the utility. For instance, you can install it using *pip3*, the package installer for Python, with:

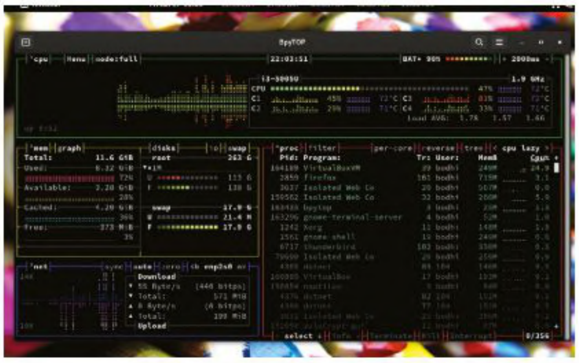
```
$ sudo pip3 install bpytop
```

The tool is also offered as a Snap package. Ubuntu from version 16.04 LTS already ships with Snap support. If you use another distro such as Fedora, you can install Snap from the official repositories with:

```
$ sudo dnf install snapd
```

You can then install *bpytop's* Snap package with:

```
$ sudo snap install bpytop
```



```
cpu  menu  mode:full          22:03:51          BAT+ 90%         2000ms
                              i3-5005U                            1.9 GHz
                              CPU                          47%
                              C1          45%   72°C  C3    83%   72°C
                              C2          29%   71°C  C4    33%   71°C
                                                Load AVG:  1.78  1.57  1.66
mem  graph       disks          swap    proc  filter   per-core reverse tree  cpu lazy
Total:   11.6 GiB  root     263 G        Pid:   Program:              Tr: User:  MemB   Cpu%
Used:    8.32 GiB                        164189 VirtualBoxVM           39 bodhi  249M   24.9
                72%  U         113 G       2859 Firefox               161 bodhi  719M    3.1
Available: 3.28 GiB F          138 G       3637 Isolated Web Co        20 bodhi  507M    0.0
                28%                      159562 Isolated Web Co        32 bodhi  260M    5.0
Cached:  4.20 GiB swap       37.9 G      163428 bpytop                  3 bodhi   23M    1.8
                36%  U       21.4 M      163296 gnome-terminal-server   4 bodhi   52M    1.0
Free:     373 MiB  F         37.9 G        1242 Xorg                   11 bodhi  148M    1.3
                 3%                        1561 gnome-shell            19 bodhi  249M    0.6
net        sync  auto  zero  wlp2s0        6717 thunderbird           103 bodhi  338M    0.3
           Download                       79600 Isolated Web Co        20 bodhi  259M    0.9
           ▼ 55 Byte/s   (440 bitps)
           ▼ Total:       571 MiB
           ▲ 0 Byte/s    (0 bitps)
           ▲ Total:       199 MiB
           Upload                          select  info  terminate  kill  interrupt     0/256
```

You can use the mouse to scroll the list of processes, and double-click any to view additional details.

If you go via the Snap router you'll also need to grant additional permissions with the following:

```
$ sudo snap connect bpytop:mount-observe
$ sudo snap connect bpytop:network-control
$ sudo snap connect bpytop:hardware-observe
$ sudo snap connect bpytop:system-observe
$ sudo snap connect bpytop:process-control
$ sudo snap connect bpytop:physical-memory-observe
```

You can now launch *bpytop*:

```
$ bpytop
```

Despite being a CLI tool, *bpytop* has a visually appealing interface that can be navigated both using the keyboard as well as the mouse. Each section has a number listed next to it, which can be used to either enable or disable that particular section. For instance, pressing 1 will turn off the CPU details, number 2 will hide memory information, and so on.

IMAGE CONVERTER

Converseen

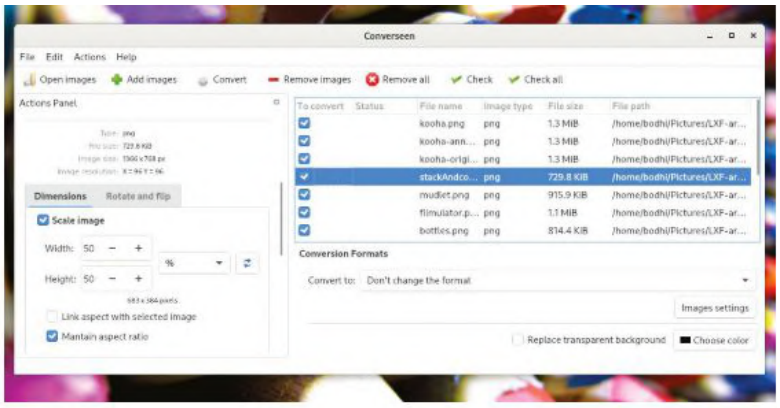
Version: 0.9.9.5 **Web:** https://converseen.fasterland.net

Those readers who've worked with images, whether professionally or just sorting through holiday photos, will know that a lot of image-processing work is monotonous. It involves repeatedly converting, resizing and rotating images to make them suitable for print or for passing around.

Although you can use virtually any image viewing or editing tool, from *GIMP* to *digiKam* for this task, you'll save yourself a lot of hassle by employing a dedicated batch conversion tool such as *Converseen*.

Converseen is a straightforward front-end to the various command-line conversion utilities that are bundled as part of the *ImageMagick* suite. You can use the tool to convert images to and from over 100 formats, rotate and flip them, change their dimensions, and rename them in a fraction of the time it would take to perform these tasks manually.

The program is available in the official repos of most distros, so you can just grab with `sudo apt install converseen` on Ubuntu or `sudo dnf install converseen` on Fedora.


Converseen
File Edit Actions Help
Open images Add images Convert Remove images Remove all Check Check all
Actions Panel
Dimensions Rotate and flip
Scale image
Width: 50
Height: 50
%
683 x 384 pixels
Link aspect with selected image
Maintain aspect ratio
Conversion Formats
Convert to: Don't change the format
Images settings
Replace transparent background
Choose color

To convert	Status	File name	Image type	File size	File path
✓		kooha.png	png	1.3 MiB	/home/bodhi/Pictures/LXF-ar...
✓		kooha-ann...	png	1.3 MiB	/home/bodhi/Pictures/LXF-ar...
✓		kooha-origi...	png	1.3 MiB	/home/bodhi/Pictures/LXF-ar...
✓		stackAndco...	png	729.8 KiB	/home/bodhi/Pictures/LXF-ar...
✓		mudlet.png	png	915.9 KiB	/home/bodhi/Pictures/LXF-ar...
✓		flimulator.p...	png	1.1 MiB	/home/bodhi/Pictures/LXF-ar...
✓		bottles.png	png	814.4 KiB	/home/bodhi/Pictures/LXF-ar...

Besides common image formats, Converseen also works with a variety of RAW image formats.

Fire up the application once it's been installed. *Converseen's* workflow is pretty straightforward – simply click the Add Images button to select any number of images you want to convert. You can also import images from PDF files.

Once the images have appeared in the list, use the Actions Panel on the left to perform any of the common operations. For instance, you can resize the images by a specific number of pixels or scale factor of one to 100 per cent, and flip and/or rotate them.

You can also choose a different directory to save your converted photos in, and rename them by adding a prefix, suffix or progressive number. Once you've configured the actions you want *Converseen* to perform on the images, select an output format for the converted images, then click Convert to get the process underway.

SYSTEM PROFILER

inxi

Version: 3.3.16 **Web:** https://github.com/smxi/inxi

The CLI-based system information tool *inxi* can display all sorts of details about the hardware and software on your computer. The output of the application can be easily influenced, which makes it ideal for everyone from everyday desktop users who would be happy with broad details, to support and QA engineers who are looking for detailed information.

The utility is already in your distro's official repos, so Ubuntu users can grab it with `sudo apt install inxi`, while `sudo dnf install inxi` will install it in Fedora.

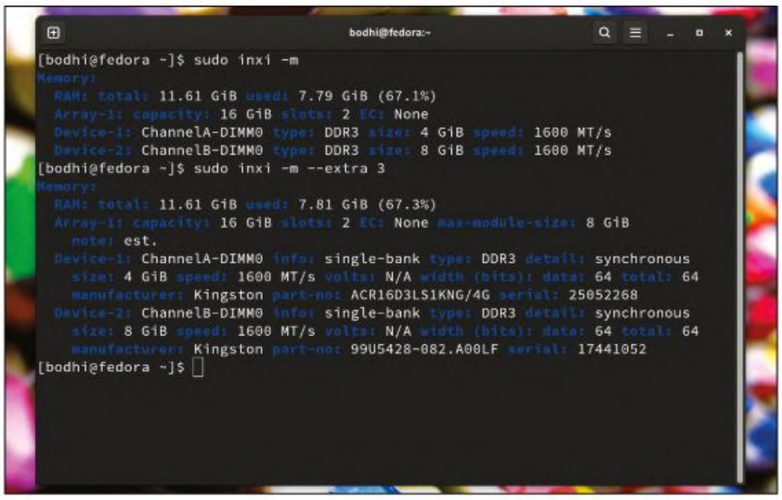
Invoking the tool without any switches will display basic hardware and software details about the installation. Use the `-C` option to extract detailed information about the processor, such as `inxi -C`. For even more details, use `inxi -Cfxa`. This will display information about any CPU vulnerabilities.

If you'd like to see a much more detailed overview of all components in your system, use `inxi -F`. For an even more verbose view, use `inxi -v8`.

Similarly, *inxi* can be made to spit out detailed information about the other hardware that's installed in the machine, including memory, graphics, hard disks and partitions, USB devices, sensors, software repositories, and a whole lot more.

For most common modules, you can use the `--extra` switch along with a number from 1 to 3 to generate more verbose output for that module. For instance, while `inxi -m` displays basic details about the memory, `inxi -m --extra 3` will display the manufacturer's name along with part number for the memory module.

The *inxi* utility features a surprising number of options, and we suggest that you spend some time perusing through its man page to get to grips with all of them.



```
[bodhi@fedora ~]$ sudo inxi -m
Memory:
  RAM: total: 11.61 GiB used: 7.79 GiB (67.1%)
  Array-1: capacity: 16 GiB slots: 2 EC: None
  Device-1: ChannelA-DIMM0 type: DDR3 size: 4 GiB speed: 1600 MT/s
  Device-2: ChannelB-DIMM0 type: DDR3 size: 8 GiB speed: 1600 MT/s
[bodhi@fedora ~]$ sudo inxi -m --extra 3
Memory:
  RAM: total: 11.61 GiB used: 7.81 GiB (67.3%)
  Array-1: capacity: 16 GiB slots: 2 EC: None max-module-size: 8 GiB
    note: est.
  Device-1: ChannelA-DIMM0 info: single-bank type: DDR3 detail: synchronous
    size: 4 GiB speed: 1600 MT/s volts: N/A width (bits): data: 64 total: 64
    manufacturer: Kingston part-no: ACR16D3LS1KNG/4G serial: 25052268
  Device-2: ChannelB-DIMM0 info: single-bank type: DDR3 detail: synchronous
    size: 8 GiB speed: 1600 MT/s volts: N/A width (bits): data: 64 total: 64
    manufacturer: Kingston part-no: 99U5428-082.A00LF serial: 17441052
[bodhi@fedora ~]$
```

While inxi can probe most components without root access, it'll prompt you when something requires elevated privileges.

QR CODE GENERATOR

Qrencode

Version: 4.1.1 **Web:** https://fukuchi.org/works/qrencode

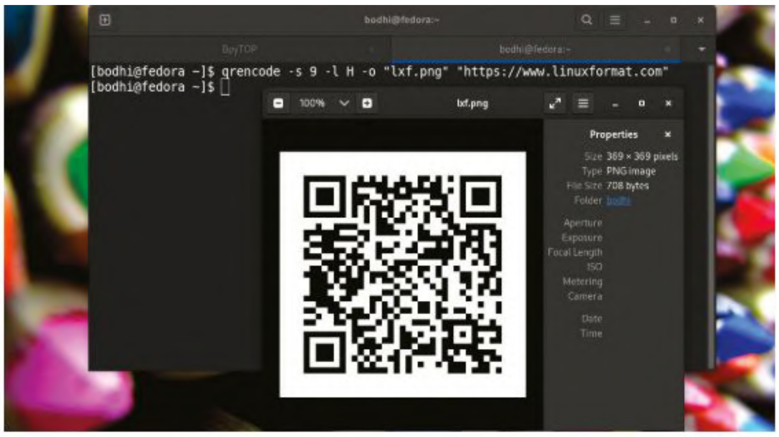
Most of us use QR codes on a daily basis (*do I?–Ed*), and more often than not for making payments. However, these nifty codes can do a lot more. You can, for instance, use QR codes to take users to a website, share your visiting cards and more.

Qrencode is a Linux utility that helps you generate custom QR codes without much effort. It's available in the official repos of most distros so you can just grab it with either `sudo apt install qrencode` on Ubuntu or `sudo dnf install qrencode` on Fedora.

Qrencode is a CLI utility that takes a few arguments and spits out the appropriate QR code. Take, for instance, this command:

```
$ qrencode -s 9 -l H -o "lxf.png" "https://www.linuxformat.com"
```

This command will generate a QR code that'll take people to the *Linux Format* website as soon as they scan it. The QR code will be named **lxf.png** and saved in the current directory from where we run *qrencode*. The `-s 9` specifies the image will have a pixel size of 9, as opposed to the default value of 3.



```
[bodhi@fedora ~]$ qrencode -s 9 -l H -o "lxf.png" "https://www.linuxformat.com"
[bodhi@fedora ~]$
```

The `-l` option specifies the error correction level, and `H` asks us to ensure the image has a high error-correction level. The higher the error correction level, the more damage the QR code can sustain before it becomes unreadable.

To generate a QR code with our Wi-Fi details, use:

```
$ qrencode -s 9 -l H -o "wif.png" "WIFI:S:<SSID>;T:WPA2;P:<password>;;"
```

Simply replace `<SSID>` and `<password>` with the name and credential of your Wi-Fi network, your Wi-Fi name and password respectively, and you're good to go.

As soon as anyone scans this code from their phone, it will ask them to connect to your Wi-Fi network automatically. Always remember to scan the QR code and verify that it does what you've designed it for, before you share it or print it.

Remember that increasing the error tolerance also increases the size of the image. We advise using the option wisely.

TEXT EXPANDER

Espanso

Version: 2.1.5-beta
Web: https://espanso.org

Expanso is a powerful text expansion tool that can help you type faster by replacing your typed text with its longer equivalent. It works by analysing your typed text and compares to substitutions in its configuration file, and then make appropriate replacements when there's a match.

To get started on a Debian-based distro like Ubuntu, first download the precompiled binary:

```
$ wget https://github.com/federico-terzi/espanso/releases/download/v2.1.5-beta/espanso-debian-x11-amd64.deb
```

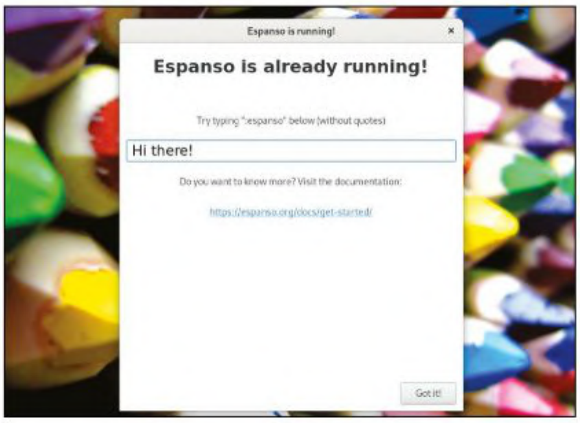
And then install it with:

```
$ sudo apt install ./espanso-debian-x11-amd64.deb
```

On all other distros, the project recommends using the official AppImage:

```
$ wget -O Espanso.AppImage 'https://github.com/federico-terzi/espanso/releases/download/v2.1.5-beta/Espanso-X11.AppImage'
```

Now make it executable with `chmod +x Espanso.AppImage`, before registering it with `sudo ./Espanso.AppImage env-path register`.



Espanso is already running!

Try typing ":espanso" below (without quotes)

Hi there!

Do you want to know more? Visit the documentation:

https://espanso.org/docs/get-started/

Espanso has some wonderful online documentation, and we suggest you peruse it to get the most from the program.

Before you can use it, first register *Espanso* with Systemd: `$ espanso service register`. You can now start *Espanso* with `espanso start`. While Wayland support is marked as experimental, the program worked without any issues on our Wayland-equipped Fedora 36 installation.

Once *Espanso* is running, all you need to do is type `:` followed by the trigger word to get it to replace it with the appropriate text. By default the tool ships with `:espanso`, which automatically expands to `Hi there!:`. You can use the program to define your own triggers, or save yourself some time and instead download pre-cooked *Espanso* packages, which include a series of pre-defined triggers. Visit the *Espanso* Hub at **https://hub.espanso.org** to peruse through the list of pre-defined trigger words contributed by the *Espanso* community.

SUDOKU PUZZLE

nudoku

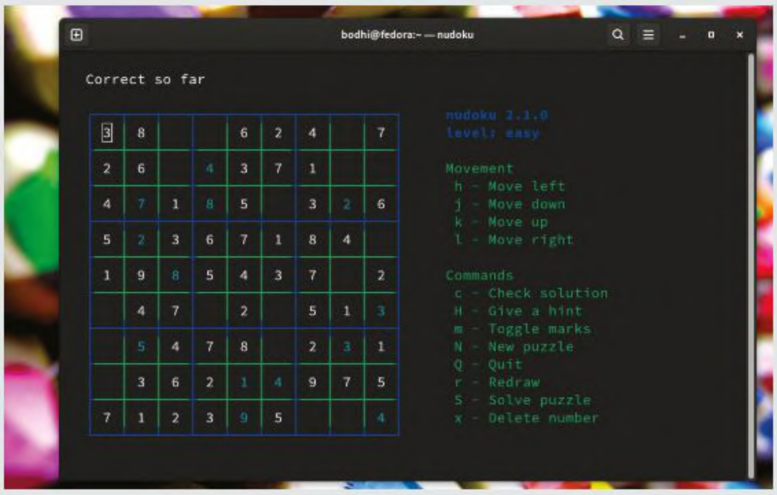
Version: 2.1.0
Web: https://github.com/jubalh/nudoku

Nudoku is a ncurses-based *Sudoku* game, which challenges you to place numbers one to nine within grids of various sizes. You'll find it in the official repos of most distros and most users can install it with either `sudo apt install nudoku` on Ubuntu or `sudo dnf install nudoku` on Fedora.

Once installed you fire it up with the `nudoku` command. You can use the arrow keys to move between cells and use the numeric keys to enter a number in a blank cell. Besides these, the game also has a number of shortcut keys.

For instance, you can press the C key after entering a number to validate your answer. In the same vein, pressing H prints a hint by putting a number in one of the empty cells in the puzzle, and you can solve the puzzle in its entirety with the S key.

By default, *nudoku* initiates an Easy puzzle. To change the difficulty level you'll have to use the `-d` option when launching the game. For instance, `nudoku -d hard` will present a difficult puzzle to solve. The `-d` option accepts easy, normal, and hard values.



```
Correct so far

3 8     6 2 4   7    nudoku 2.1.0
2 6   4 3 7 1        level: easy
4 7 1 8 5   3 2 6
5 2 3 6 7 1 8 4      Movement
1 9 8 5 4 3 7   2     h - Move left
  4 7   2   5 1 3     j - Move down
  5 4 7 8   2 3 1     k - Move up
  3 6 2 1 4 9 7 5     l - Move right
7 1 2 3 9 5     4
                     Commands
                      c - Check solution
                      H - Give a hint
                      m - Toggle marks
                      N - New puzzle
                      Q - Quit
                      r - Redraw
                      S - Solve puzzle
                      x - Delete number
```

If you resize the terminal mid-game, use the r key to redraw the puzzle to match the new dimensions of the window.

If you want to solve them offline, you can also instruct *nudoku* to generate PDF files containing one or more puzzles.

For instance:

```
$ nudoku -p puzzle.pdf -n 12 -d normal
```

This command will print a **puzzle.pdf** file that contains 12 *Sudoku* puzzles of normal difficulty over three pages.

Similarly, you can also generate a PNG image of a single *Sudoku* puzzle of your desired difficulty level.

```
$ nudoku -i sudoku.png -d easy
```

This command will generate a PNG image of an easy *Sudoku* puzzle. Read through *nudoku's* man page to familiarise yourself with the available options.

PHYSICS SIMULATOR

Powder Toy

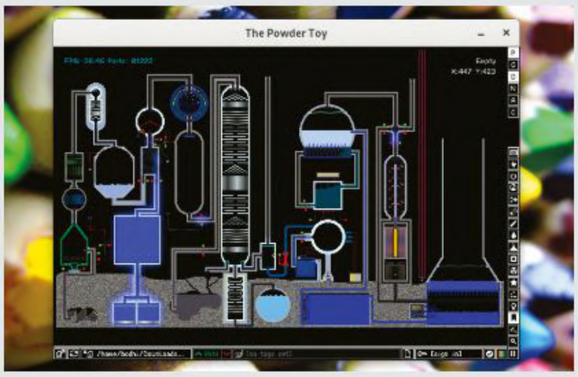
Version: 96.2
Web: https://powdertoy.co.uk

The Powder Toy is a free physics sandbox game, whose description will never be able to do justice to its awesomeness. You can use *The Powder Toy* to simulate interaction between gravity, air, pressure, heat and various substances.

Installing the program is fairly simple. Just grab and extract its compressed binary, and you're good to go:

```
$ wget -c https://powdertoy.co.uk/Download/
powder-lin64.zip
$ unzip powder-lin64.zip -d powder
$ cd powder
# ./powder
```

The Powder Toy's interface looks rather busy for a game. On the right side, you have a toolbar that enables you to select various substance types. There are solids, liquids, gases, explosives, elements and more. The actual list of materials for the select substance type, appears at the bottom. For instance, under liquid you'll get everything from carbonated water and salt water to liquid nitrogen and diesel. It'll also tell you how to make a particular substance explode.



It might not look like it, but once you get the hang of it, The Powder Toy can be addictive.

You can use *The Powder Toy* to create walls, or containers, then fill them with materials and watch them interact. You can for instance, simulate the hydrogen fusion at the core of stars by adding immense heat and pressure.

The game gives you the option to pause and resume your simulations, so you can carry out small tweaks and then observe how they affect your creations.

You can also load one of the many existing simulations, created by *The Powder Toy* community. And there are some impressive constructions out there, including complex machines, bombs and intricate laboratories.

To use the downloaded .cps files, first register the game with your distro using the Ctrl+i keyboard shortcut. Then you can simply double-click the .cps files to load them in *The Powder Toy*.

SEARCH TOOL

Cerebro

Version: 0.5.0
Web: https://cerebroapp.com

Cerebro is a comprehensive search program for Linux that enables users to search files, launch programs, look up places on maps, perform translations and a lot more.

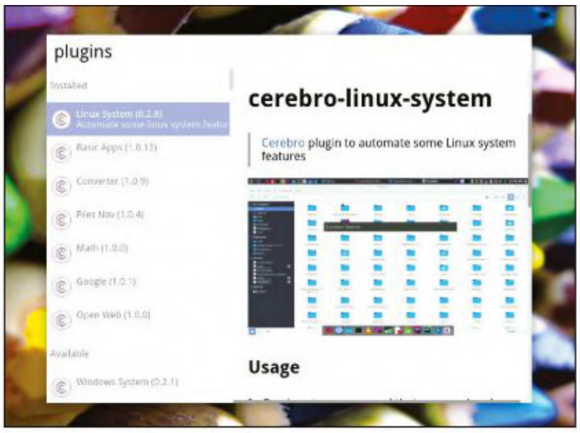
You can install the tool using its AppImage:

```
$ wget -O cerebro.appimage 'https://github.com/cerebroapp/cerebro/releases/download/v0.5.0/Cerebro-0.5.0.AppImage'
$ chmod +x cerebro.appimage
```

On first launch, *Cerebro* will ask if you'd like to integrate it with your system. Doing so will integrate the program in your applications menu. Your other option is to use the appimage to bring it up.

Once *Cerebro* is up and running, you can use the Ctrl+Space keyboard shortcut to bring up the *Cerebro* Search option.

To get started, type `settings` in *Cerebro* Search to bring up the *Cerebro* Settings window. From here you can change the keyboard shortcut to bring up the search bar, switch to a dark theme, choose the country to fetch more relevant results, and more.



While Cerebro is pretty handy straight out of the box, its set of plugins make it extremely useful.

In addition to files, you can use *Cerebro* for a variety of tasks. For instance, you can search Google or perform calculations, or convert between units. You can further extend *Cerebro's* functionality by adding additional plugins. Bring up *Cerebro* and type `Plugins` to bring up a list of installed and available options. You can scroll through the list on the left and read a description of any plugin before using the Install button to add it to your *Cerebro* installation.

Some of the interesting plugins are the Stackoverflow plugin that can help you search the popular Q&A website for developers. Similarly, there's the Wiki plugin that enables you to search Wikipedia from *Cerebro* itself. Then there's the Translate plugin that uses Google Translate to work its magic and convert strings between languages.

FLATPAK PERMISSIONS MANAGER

Flatseal

Version: 1.8.0 **Web:** https://github.com/tchx84/flatseal

Flatseal is a GUI utility that enables you to review and tweak the permissions given to all Flatpak programs that you use.

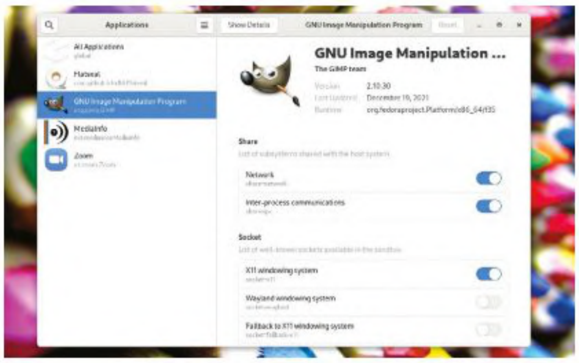
Flatpak programs use two different permissions models. Static permissions are those that are set by the developers while building the program, while Dynamic permissions are granted by the users when they run the application.

Both of these permissions can be modified by the users. However, you'll have to get down to the terminal to tweak them. With *Flatseal* you're able to tweak the permissions, but from the comforts of a graphical tool.

Unsurprisingly, *Flatseal* is available as a Flatpak. You can install it with:

```
$ flatpak install flathub com.github.tchx84.Flatseal
$ flatpak run com.github.tchx84.Flatseal
```

Flatseal has an intuitive interface that displays all the Flatpaks you use, along with all its supported permissions. The permissions are divided into groups such as share, device, filesystem and more, each of which has a handful of relevant permissions.



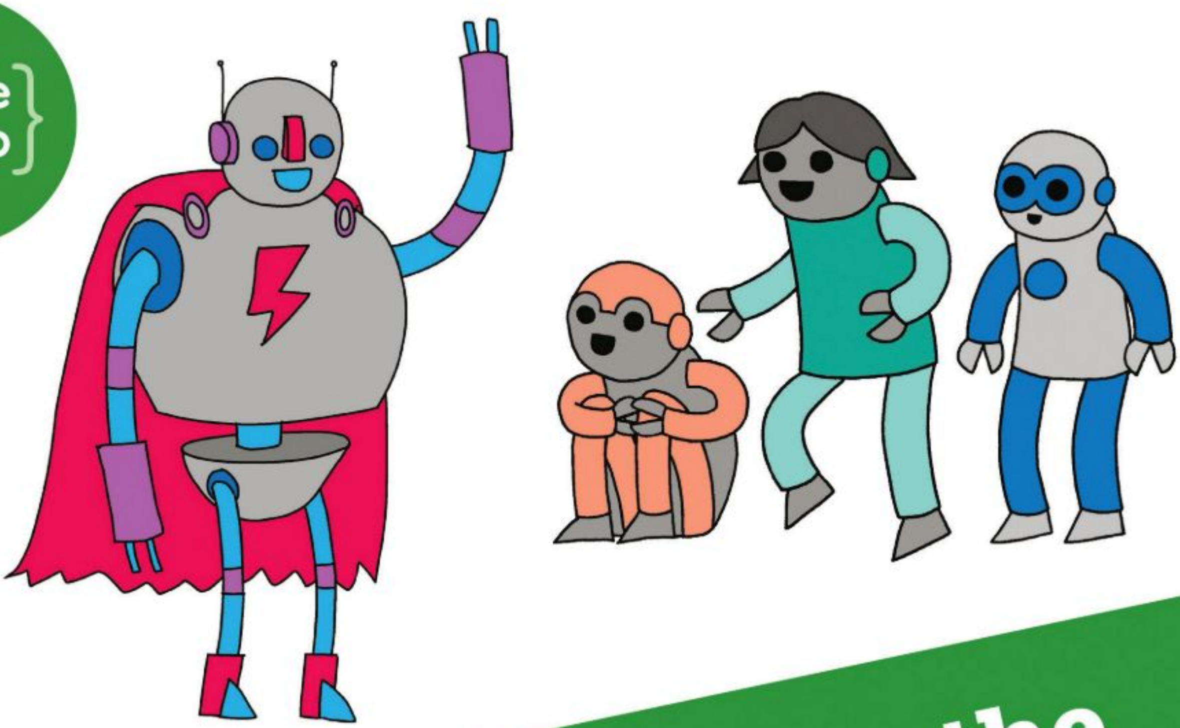
While most Flatpaks ship with appropriate settings, you can use Flatseal to modify their behaviour by rescinding permissions to certain resources.

For instance, you can use the program to control whether the application should have access to the network, and whether it has permission to open inside of an X11 or Wayland window. In terms of hardware you can control whether the application can play sounds and access your microphone, or take advantage of GPU acceleration, and a lot more.

Enabling or disabling a permission is as simple as toggling the relevant permission. Remember that not all programs have access to every permission. For instance, an image-management program has no business interacting with the speakers and microphone on the computer. These inapplicable permissions will be greyed out in Flatseal's interface.

Finally If a Flatpak app starts misbehaving after you've tweaked its permissions, you can use the Reset button to revert it back to its factory permissions. So experiment with confidence! LXF

{code club}



Can you help inspire the next generation of coders?

Code Club is a nationwide network of volunteer-led after school clubs for children aged 9-11.

We're always looking for people with coding skills to volunteer to run a club at their local primary school, library or community centre for an hour a week.

You can team up with colleagues, a teacher will be there to support you and we provide all the materials you'll need to help get children excited about digital making.

There are loads of ways to get involved!

So to find out more, join us at **www.codeclub.org.uk**

How to develop multi-threaded code

Mihalis Tsoukalos explains all you need to know about Rust and concurrency so you can assemble efficient multi-threaded tools.

OUR EXPERT

Mihalis Tsoukalos is a systems engineer and a technical writer. Find him online at @mactsouk.

This month's instalment of our ongoing Rust series will cover concurrent programming. The difficulty with concurrency can be summed up as "Shared Mutable State is the root of all problems". However, Rust takes a different approach towards concurrency. Instead of dealing with the mutable part, Rust deals with the shared part. Put simply, the Rust compiler will let you know about any errors in the "shared" part; therefore, if your Rust code compiles you have no such errors. As a result, it's okay to have shared memory among threads in Rust.

So, it's in concurrent programming that you might appreciate all the Rust rules regarding borrowing, ownership and mutability. To get us going we'll present a simple concurrent Rust program that's far from perfect.

Basic concurrency

Rust starts a new thread using the `thread::spawn()` function that's part of the `std::thread` module. The core code of **simple.rs** is the following:

```
for _ in 0..5 {
  thread::spawn( || {
    println!("Hi there!!");
    thread::sleep(Duration::from_millis(1000));
  });
}
```

Two things are happening here. First, there's a for loop used for calling `thread::spawn()` five times and therefore creating five threads. Second, there's a call to `thread::sleep()` for delaying the termination of the thread. Threads usually don't just print messages, but perform actual work that takes time. Let's now run **simple.rs**, which generates the following output:

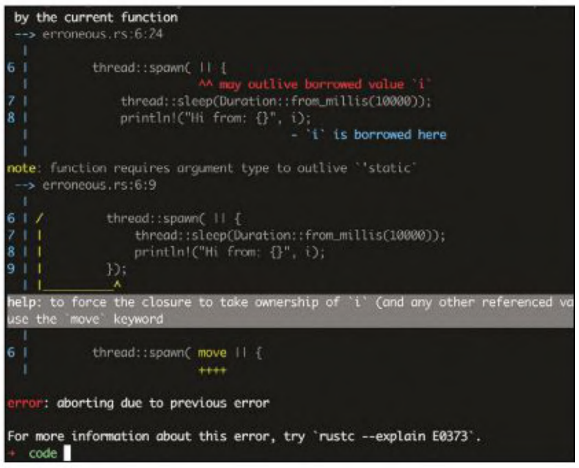
```
Start.
End.
```

First, bear in mind that there are two `println!()` statements before and after the for loop, which is the reason for the two lines of output. But where are the messages from the five threads? It looks like **simple.rs** doesn't behave as expected. The messages from the threads aren't printed on screen, mainly because the `main()` function (in reality the thread used for executing the `main()` function) doesn't wait for the five threads to finish – we can add an additional `thread::sleep()` call after the for loop for the `main()` thread to wait, but we can't depend on such arbitrary parameters because we still don't know how long it's going to take the threads to finish and whether the sleep duration of the `main()` thread is going to give enough time to them.

So, in our case, the `main()` function ends first, which means the entire program ends without giving the threads enough time to do their jobs. Additionally, the for loop in **simple.rs** ignores the value of the loop parameter. But what happens if we want to print the value of the loop parameter? Read on!

QUICK TIP
Get the code for this tutorial from the Linux Format archive: www.linuxformat.com/archives?issue=292.

You can learn more about Rust at www.rust-lang.org.



```
by the current function
  --> erroneous.rs:6:24
   |
6  |         thread::spawn( || {
   |                        ^^ may outlive borrowed value `i`
7  |             thread::sleep(Duration::from_millis(10000));
8  |             println!("Hi from: {}", i);
   |                                     - `i` is borrowed here
   |
note: function requires argument type to outlive `'static`
  --> erroneous.rs:6:9
   |
6  | /         thread::spawn( || {
7  | |             thread::sleep(Duration::from_millis(10000));
8  | |             println!("Hi from: {}", i);
9  | |         });
   | |__________^
help: to force the closure to take ownership of `i` (and any other referenced va
use the `move` keyword
   |
6  |         thread::spawn( move || {
   |                        ++++

error: aborting due to previous error

For more information about this error, try `rustc --explain E0373`.
→ code
```

The error message in erroneous.rs and a hint about the solution produced by the Rust compiler when trying to compile it.

An erroneous situation

Let's create an erroneous situation related to Rust and concurrency, so you can avoid making the same mistake. We're going to change the code of **simple.rs** in order for the program to print the value of the loop parameter – the new source file is called **erroneous.rs**.

The screenshot (*above*) shows part of the code of **erroneous.rs** as well as the output of the Rust compiler when trying to compile it. The compiler output tells us that to pass the ownership of the **i** variable (which is used in the loop), to a thread we need to use the move keyword when calling `thread::spawn()`.

We're now going to correct the errors of both

simple.rs and **erroneous.rs**, which means that the compiler is not going to complain about the ownership of **i** and that the threads of the new version, which is called **correct.rs**, are going to have enough time to print their output.

So, we have to use the move keyword when calling `thread::spawn()` for the error to be corrected. The move keyword used in `thread:spawn()` transfers the ownership of a variable from the main thread to the thread we're about to spawn. However, even if we do that, we still don't get any output from the program, just as with **simple.rs**. To correct this logical error, we're going to use a join handle. The full implementation of **correct.rs** can be seen in the screenshot (*right*).

So, what does a join handle do and how can it help us? Join handles are used for telling `main()` to wait for the associated threads to finish. This is implemented with the `join()` call, which blocks until all associated threads have finished. In our case, we have five threads and a vector for storing their join handles – a join handle is the return value of the `thread::spawn()` function, which we can ignore if we want to. Running **correct.rs** produces the following output:

```
./correct
Start.
Hi there: 4
Hi there: 1
Hi there: 3
Hi there: 0
Hi there: 2
End.
```

Bear in mind that if we run **correct.rs** multiple times, there's a very good chance that we'll obtain a different output each time. This occurs because thread execution isn't controlled by the user but rather by the Linux operating system and its scheduler. Therefore, we shouldn't make any assumptions about the order of execution of our threads. This is a important detail that we should always remember.

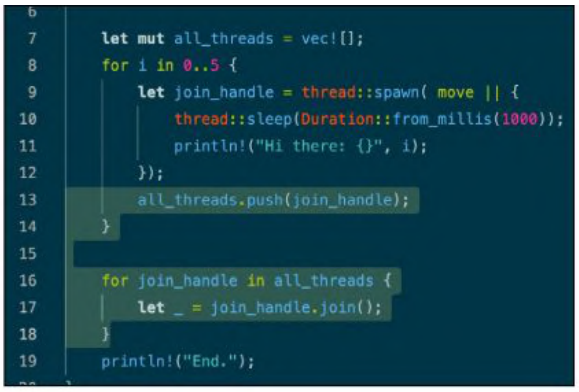
Thread errors

It'll be helpful to know how to handle errors generated inside a thread, other than the thread of the `main()` function. Errors are mainly down to external situations unrelated to the program's code. The next example, saved as **errors.rs**, is going to try to delete a file that doesn't exist, which is a common error condition. Look at the following code example from **errors.rs**:

```
fn delete_file(filepath: &'static str) -> thread::Result<()>
{
  thread::spawn( move || {
    fs::remove_file(filepath).expect("Unable to
remove");
  }).join()
}
```

Here, the `delete_file()` function creates a thread that attempts to delete a file, which is given as an argument to the function. The good thing here is that the return value of `fs::remove_file()` is the return value of the thread and then the return value of the `delete_file()` function. This gives `main()` the opportunity to examine the return value of `delete_file()` and act accordingly.

The screenshot (*bottom of page 92*) shows the Rust code of **errors.rs** as well as its output. The `main()` function uses pattern matching to determine whether



```
    let mut all_threads = vec![];
    for i in 0..5 {
        let join_handle = thread::spawn( move || {
            thread::sleep(Duration::from_millis(1000));
            println!("Hi there: {}", i);
        });
        all_threads.push(join_handle);
    }

    for join_handle in all_threads {
        let _ = join_handle.join();
    }
    println!("End.");
```

Correct.rs uses both the variable of the for loop in its threads and waits for all threads to finish before the end of the main() function.

there was an error or everything went as expected.

Panic? Oh, go on then

Bad things can happen to both good people and good threads. This section is going to discuss the panicking of threads in Rust. Unfortunately, discussing what to do when bad things happen to good people is beyond the scope of this magazine. A panic situation is mostly related to conditions that have to do with the running Rust code. Rust has the `thread::panicking()` function found in the **std::thread** module that can help you determine whether the current thread is unwinding because of a panic or not.

The technique is presented in **panicking.rs**. The most important part of **panicking.rs** is the following:

```
impl Drop for Filename {
  fn drop(&mut self) {
    if thread::panicking() {
      println!("thread::panicking()!");
    } else {
      println!("No thread::panicking()!");
    }
  }
}
```

Let us now discuss about the use of Drop and how it helps our purpose. The Drop trait gets called when an instance of a variable that implements it goes out of scope – in our case the Filename structure. Inside the `drop()` function implementation we check whether the thread that used that variable has panicked or not using

QUICK TIP

You can learn more about the std::sync::mpsc module at https://doc.rust-lang.org/std/sync/mpsc/ and about std::thread at https://doc.rust-lang.org/std/thread/fn.spawn.html. You can find more information about the Rayon library at https://github.com/rayon-rs/rayon. Finally, the Rust documentation site is at www.rust-lang.org/learn.

» MUTEXES EXPLAINED

A mutex is a mutually exclusive flag that enables a single thread to do something critical, in most cases changing the value of a shared variable, while the other threads are waiting for the same mutex to become available in order to get it and do their jobs on the shared variable. Therefore, a mutex allows only a single thread to have access to a piece of data at a given time. If multiple threads want to access the same data, they'll have to wait until they get the relevant mutex. In other words, any thread that wants to access the shared data needs to acquire the lock to that mutex.

Once the thread is done working with the shared variable or variables, it should unlock the mutex and make it available to other threads. However, Rust does the unlocking automatically so there's no need to call an unlock function. The disadvantage of mutexes is that you need to write extra code to use them and that if you forget to unlock a mutex, your program will most likely going to misbehave or crash – again, this is not a problem with Rust.

»

QUICK TIP
Concurrency is a way of structuring your components so that they can be executed independently when possible. In contrast, parallelism is the simultaneous execution of multiple entities. Put simply, concurrency is better than parallelism.

`thread::panicking()`. After that, the error of the thread is passed to the main() function as before. We try to delete a file, which this time is given as a command line argument to **panicking.rs**. Running **panicking.rs** twice produces the following output:

```
$ ./panicking /tmp/file
No thread::panicking()!
File deleted!
$ ./panicking /tmp/file
thread::panicking()!
Failed to delete file!
```

The first time we execute **panicking.rs**, the file that's about to be deleted already exists. Therefore, **panicking.rs** is executed without any issues and the return value of `thread::panicking()` is false. However, the second time **panicking.rs** panics, which means that `thread::panicking()` returns true and the relevant code is executed. Although we're just printing a simple message here, the use of the Drop trait in combination with `thread::panicking()` is a chance to perform cleanup operations such as closing open files and network and database connections, as well as writing logging information regarding the panic situation.

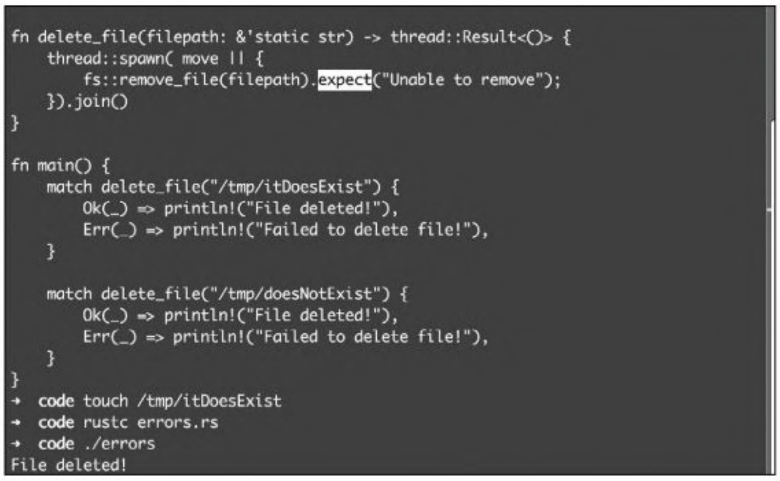
Message channels

Channels in Rust are used for providing asynchronous communication among threads. You can consider a Rust channel as a queue for safely sending Rust values – not raw data – among Rust threads. Bear in mind that channels are unidirectional, which makes channels look like Unix pipes with a consumer (Receiver) and a producer (Sender) part. The following code excerpt from **singleChannel.rs** illustrates the most important channel concepts:

```
let (t, r) = mpsc::channel();
t.send(val).unwrap();
let received = r.recv().unwrap();
```

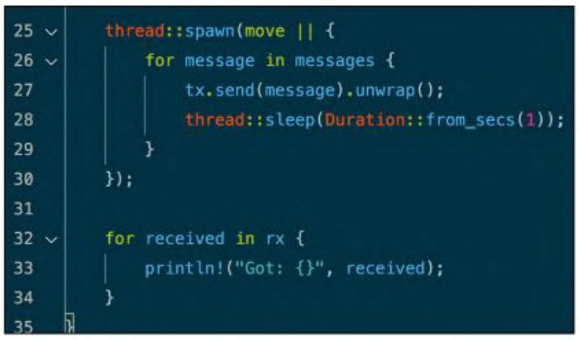
The `std::sync::mspsc::channel()` function is used for creating a new channel – the function returns a tuple that contains a Sender and a Receiver, respectively. The `send()` function is used for sending a value to a channels whereas `recv()` is used for reading an existing value from a channel. It's important to remember that there's no need for calling `join()` to synchronise threads when using channels since `recv()` blocks while waiting to receive a value or until a channel closes.

The screenshot (*above*) shows part of the Rust code of **singleChannel.rs**. The program creates two channels.



```
fn delete_file(filepath: &'static str) -> thread::Result<()> {
    thread::spawn( move || {
        fs::remove_file(filepath).expect("Unable to remove");
    }).join()
}

fn main() {
    match delete_file("/tmp/itDoesExist") {
        Ok(_) => println!("File deleted!"),
        Err(_) => println!("Failed to delete file!"),
    }

    match delete_file("/tmp/doesNotExist") {
        Ok(_) => println!("File deleted!"),
        Err(_) => println!("Failed to delete file!"),
    }
}
→  code touch /tmp/itDoesExist
→  code rustc errors.rs
→  code ./errors
File deleted!
```

Part of the code as well as the output of errors.rs that shows how to handle errors created in threads.



```
    thread::spawn(move || {
        for message in messages {
            tx.send(message).unwrap();
            thread::sleep(Duration::from_secs(1));
        }
    });

    for received in rx {
        println!("Got: {}", received);
    }
```

The use of channels for sending and receiving messages through the code of singleChannel.rs, which creates and uses two channels.

The first one is used for sending a single message using `send()` and getting it back using `recv()`, and the second channel is used for sending multiple messages that are consumed using an iterator over the channel receiver.

Another important point is that a channel is considered to be closed if either the transmitter or receiver part is dropped. When the sending of data ends (goes out of scope), the channel is considered as closed, which ends the final for loop. Rust sends an appropriate signal to denote that no more values are coming to the channel.

Channels are versatile. They enable Rust threads to exchange information and data to flow among threads and create pipelines. MSPC stands from Multi-Producer, Single Consumer, but we've only used a single producer in **singleChannel.rs**. However, we're going to continue discussing channels by presenting a technique that enables the exchange of messages among threads using multiple sender threads.

Multiple sender

There's a technique that enables multiple threads to exchange multiple messages using a single channel. So, we're going to have multiple threads (Producers) that send messages to a channel and a single thread (Consumer) that reads those messages and prints them on screen – in our case, the exchanged data is going to be numbers of the Fibonacci sequence. The interesting part of **channels.rs** is as follows:

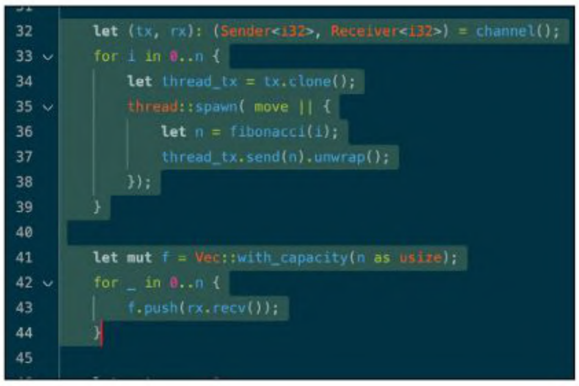
```
for i in 0..n {
        let thread_tx = tx.clone();
...
```

The `tx.clone()` call clones the sender in order to send values from multiple threads – this is the trick for creating multiple senders for a channel. The purpose of **channels.rs** is to calculate and add numbers of the Fibonacci sequence. The screenshot (above right) shows the implementation of `main()` as found in **channels.rs**. It's important to remember that there's no need for calling `join()` to synchronise threads when using a channel since `recv()` blocks while waiting to receive the next value. In our case, all incoming data is stored in a Vec structure and read from there.

Running **channels.rs** produces the following output:

```
$ ./channels 10
Going to create 10 threads!
0 2 5 1 1 3 8 13 21 34
Sum = 88
```

Nothing is stopping us from creating larger pipelines with threads that exchange messages using channels –



```
    let (tx, rx): (Sender<i32>, Receiver<i32>) = channel();
    for i in 0..n {
        let thread_tx = tx.clone();
        thread::spawn( move || {
            let n = fibonacci(i);
            thread_tx.send(n).unwrap();
        });
    }

    let mut f = Vec::with_capacity(n as usize);
    for _ in 0..n {
        f.push(rx.recv());
    }
```

The implementation of the main() function as found in channels.rs, demonstrating the use of multiple senders in Rust channels.

just try to keep the design clean and simple because things can get complex when using multiple threads. Next, we'll discuss the use of mutexes in concurrency.

Using a Mutex

As mentioned earlier, a Mutex is a flag that enables code to operate exclusively on some shared variables or other shared resources. The use of mutexes is briefly illustrated in the following code snippet:

```
let sum = sum.clone();
let mut total = sum.lock().unwrap();
```

The first statement clones the shared variable, which is defined elsewhere, whereas the second statement obtains the mutex for that shared variable.

The screenshot (*below right*) shows part of the Rust code of `main()` as found in **mutex.rs**. The mutex is embedded in an Arc (atomically reference counting) data type that's thread safe. The reason for doing so is that an Arc converts every type into one that can be safely shared among multiple threads. The place where the total variable is defined is where a thread takes control of the mutex. After that, we increment the total variable according to our wills in order to make changes to the value of the sum variable. At any given time, only a single thread can access and modify the sum variable and have its mutex through the total variable, which makes modifying its value thread-safe. The good thing is that Rust automatically unlocks the mutex once the shared variable goes out of scope, which means that there's no need for calling a separate function to unlock it. At the end of the program, the `main()` thread gains access to the mutex and prints the value of sum.

Much like with **channels.rs**, **mutex.rs** computes the summary of the first **n** Fibonacci numbers, where **n** is given as a command line argument to the program. However, each thread adds its result to the mutex variable instead of temporarily putting it into a vector in order to compute the desired sum later on.

Running **mutex.rs** produces the following output:

```
$ ./mutex 10
Going to create 10 threads!
Summary: 88
```

A rayon light

The Rayon library (**https://docs.rs/rayon/latest/rayon**) takes sequential code and converts it into safe parallel code on its own. As Rayon is an external library, we're going to need to create a *cargo* project. So, we create a new *cargo* project as follows:

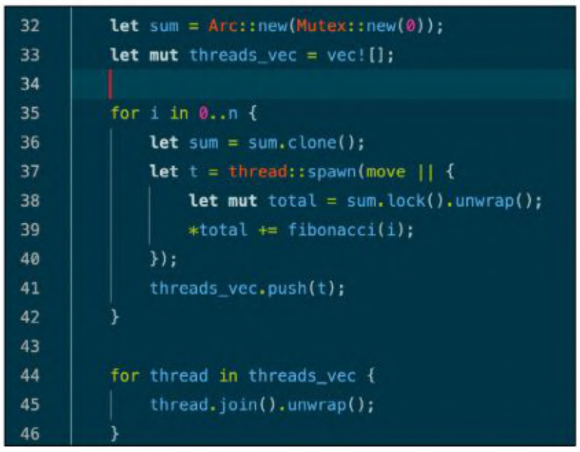
```
$ cargo new fibo
$ cd fibo
$ rm -rf .git .gitignore # we don't need Git at this point
```

The **Cargo.toml** file should include the `rayon = "1.5"` line in the `[dependencies]` section. Once again, we're going to calculate numbers of the Fibonacci sequence. The important part of the implementation of `main()` as found in **src/main.rs** is going to be the following:

```
let total: i32 = v
  .par_iter()
  .map(|x| fibonacci(*x))
  .sum();
```

The only thing that you need to do to make **src/main.rs** to use the Rayon library, apart from `use` and `extern` statements at the beginning of the file, is to use `par_iter()` instead of just `iter()`! The Rayon library is going to deal with all the details so there's no need for calling `std::thread::spawn()`! .However, we're going to need to create a vector and populate it with the desired range of numbers in order for Rayon to be able to iterate over the desired range of numbers.

The final instalment in our Rust series is going to be on network programming. Until then, keep practising Rust and make your code work concurrently! LXF



```
    let sum = Arc::new(Mutex::new(0));
    let mut threads_vec = vec![];

    for i in 0..n {
        let sum = sum.clone();
        let t = thread::spawn(move || {
            let mut total = sum.lock().unwrap();
            *total += fibonacci(i);
        });
        threads_vec.push(t);
    }

    for thread in threads_vec {
        thread.join().unwrap();
    }
```

Here's the Rust code of mutex.rs that illustrates the use of mutexes for safely sharing variables among multiple threads.

» THE FORK-JOIN PARALLELISM

The `thread::spawn()` function that's used for creating new threads returns a JoinHandle structure that's "an owned permission to join on a thread". In other words, JoinHandle provides a way to join the child thread. Usually, you're going to use the provided JoinHandle to call the `std::thread::join()` method to make the `main()` function wait for the thread to finish. This is called fork-join parallelism because `std::thread::spawn()` creates a new thread whereas `std::thread::join()` deals with waiting for a thread to finish. The fork-join parallelism has many advantages because of its simplicity, which is its most important advantage.

Another method of carrying out parallel programming is Async/Await (Asynchronous Programming). Learn more at **https://rust-lang.github.io/async-book/03_async_await/01_chapter.html** and at **https://moodle.cs.pdx.edu/mod/page/view.php?id=491**.

PYTHON

Credit: www.pygame.org/project/133/166

Adding gamepad support to Tetris

Andrew Smith likes nothing better than fitting oddly shaped blocks into oddly shaped holes, and a gamepad makes that much easier…

OUR EXPERT

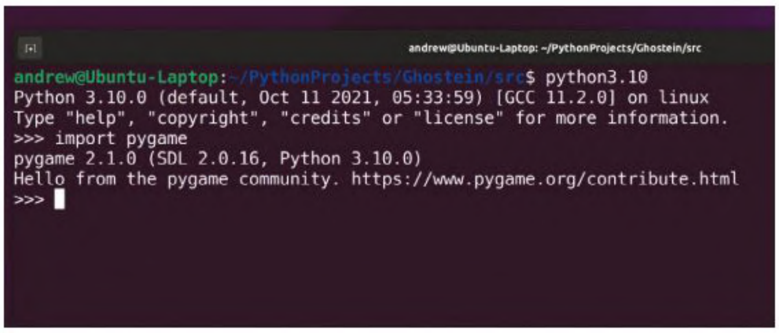
Andrew Smith is a software developer at NHS Digital, and has qualifications in software engineering and computer networking.

This month, we going to look at a *Tetris* project that's been written in Python, and then add code to support an Xbox One controller. In theory, the code should work with other PC-based compatible controllers, but they've not been tested for this tutorial and so may require some adjustments and configuration changes. Your millage may vary!

Before starting this tutorial, we advise plugging your game controller (if you have one) into your computer and making sure that it works with your distro of Linux (we're using Ubuntu). This tutorial is based on a *Tetris* project originally created by Luke Arnston – see **www.pygame.org/project/133/166**.

Tetris, a graphical puzzle game first created in 1984 by Alexey Pajitnov, it has since become available on multiple platforms and implemented using a range of programming languages. One of the most popular platforms that this video game was available on was the Nintendo Game Boy, which was released in the late 80s/early 90s. The Game Boy was a relatively cheap, hand-held device that could be easily packed away and taken and played anywhere.

Tetris involves players creating lines from shapes (Tetronimoes) that descend gradually into the game playing area. These shapes can also be manipulated (moved left to right and rotated) to fit in with other shapes to create a complete line. Once a complete line is created, the line then disappears and the player is awarded points. The game ends when new shapes generated can not descend into the game playing area. The game is deceptively simple to play, and can become maddeningly addictive.



```
andrew@Ubuntu-Laptop: ~/PythonProjects/Ghostein/src
andrew@Ubuntu-Laptop:~/PythonProjects/Ghostein/src$ python3.10
Python 3.10.0 (default, Oct 11 2021, 05:33:59) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pygame
pygame 2.1.0 (SDL 2.0.16, Python 3.10.0)
Hello from the pygame community. https://www.pygame.org/contribute.html
>>> 
```

This screenshot shows a successful installation of Python and PyGame, as an example from a previous Linux Format tutorial.

Installation and setup

We'll install and setup Python 3.10 for the benefit of this tutorial. For those readers who have Python/PyGame already installed, Python 3.8+ should be fine to use with this tutorial. Type the following to install Python 3.10 and PyGame:

```
$ sudo apt-get install python3.10
$ sudo apt-get install python3-pip
$ python3.10 -m pip install pygame
```

Check your versions of python and pygame. Next, `git clone` from repository:

```
$ git clone https://github.com/asmith1979/lxf292_tetris/
```

As an example, the whole project has been put into a folder called **PythonProjects** which was created before downloading the project. (The source code and project can also be retrieved from the *Linux Format* archive). This tutorial will focus on the source code located in the folder called **lxf292_tetris**, which if not already in that folder type `cd lxf292_tetris` to get into the folder and gain access the Python source code. You'll see two Python source code files: **main.py** and **lxf292ans.py**. The Python file **main.py** is the Python script file that you'll be editing, while **lxf292ans.py** contains the full tutorial code to be implemented.

To edit and view the source code you can either use a default text editor installed on your flavour of Linux (Ubuntu for example) or you could use something more specific such as *Notepad++*, *PyCharm* or *VS Code*. The choice is entirely up to you. In this tutorial, we'll be using *gedit* to view and edit the source files with. If and when using this method to view or edit source files, it may be helpful to open up two console windows. One can then be used for either editing or viewing source files, while the other terminal window makes it straightforward to execute the PyGame code.

Now that we've either downloaded or cloned the project to our machine, let's run the project so that we can at least have a play of *Tetris*. For some readers that may be all they want to do; however, it's a good start for all to see what the project can do.

As described earlier, there are two Python files: **main.py** and **lxf292ans.py**. The main difference between these two Python files is that **lxf292ans.py** has game controller support while **main.py** lacks this facility. In either file, the keyboard controls involve using the cursor keys to move the *Tetris* game piece left, right and down (there's no up because the pieces continually fall from the top. Keys Z and X are used to rotate the piece left or right. To run the game program with game controller support type `python3.10 lxf292ans.py`. To run the game program without game controller support type `python3.10 main.py`.

Project overview

This project uses a reasonable amount of image and sound resources. The image resources for this project are located in the **Tetris\Image** folder and the sound resources for this project are located in **Tetris\SFX**. The project uses various individual (helper) functions to help with resource loading and gameplay, and these include the following:

» load_image (used to load an image resource)
» load_sound (used to load a sound resource)
» main (used to start the game)
» Game (renders game graphics and processes inputs)
» GameOver (renders the 'game over' screen)
» Pause (processes the pause function of the game)

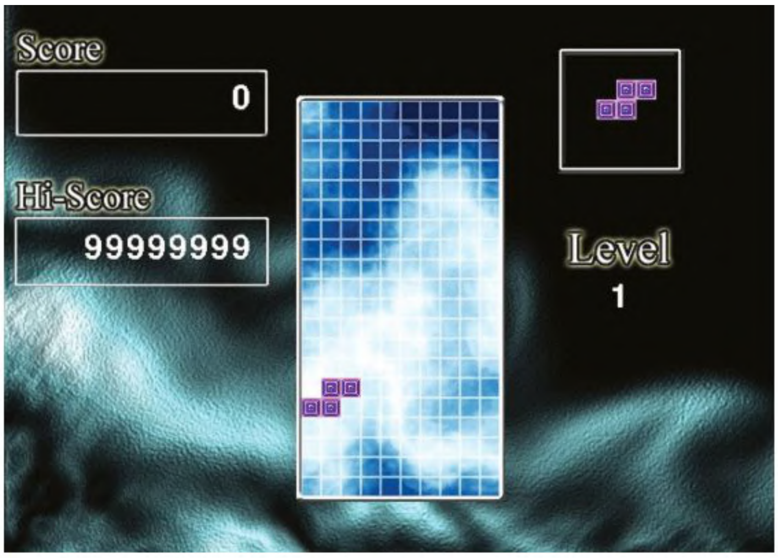
In addition to the above, there have been the following classes implemented to represent objects in the game:

» piece (represents a *Tetris* piece)
» block (represents part of a *Tetris* piece)
» grid (contains all of the dropped *Tetris* pieces)

Let's start by looking at a couple of the individual functions identified above, namely `load_image` and `main`. As the name suggests, `load_image` is used to load an image resource from the **Tetris\Image** folder:

```
def load_image(name, colorkey=None):  # Basic
function to load images
  fullname = os.path.join('Image', name)
  try:
    image = pygame.image.load(fullname)
  except (pygame.error, message):
    raise SystemExit(message)
  image = image.convert()
  if colorkey is not None:
    if colorkey is -1:
      colorkey = image.get_at((0,0))
    image.set_colorkey(colorkey, RLEACCEL)
  return image
```

As can be seen from the above code sample of `load_image`, two arguments are passed in: `name` and `colorKey`, which by default is set to None. The argument `name`, as it suggests, stores the name of the image resource to be loaded. The second argument, `colorKey`, is used to store the colour used in the top left-corner of the image. Any pixels in the image that have that specific colour value will be transparent. The reason for this when an image is created and saved, the edge of the background of the image may be sometimes saved with it. Dealing with `colorKey` takes care of this. There's also error checking implemented into the function so that if the image resource can't be found then the game program is exited.



The start of level one, and the first S-shaped playing piece is falling into place. This game of Tetris will have to go on for some time, if the player is to beat that impressive high score.

The main function featured in the source code is first called on when the program is launched:

```
def main():
  """main Function."""

  #Start Pygame
  pygame.init()
  screen = pygame.display.set_mode((640, 480))
  pygame.display.set_caption('LXF 292')
  while(1):
    if(Game(screen)):
      if(GameOver(screen)):
        pass
```

QUICK TIP

Get the code for this tutorial from the Linux Format archive: www.linuxformat.com/archives?issue=292.

» GOING PYGAME LOOPY

The PyGame loop can usually be looked on as on the main essential loop that is implemented into a PyGame program. In the case of this project, the loop has been implemented several times which usually handles both the rendering of the graphics and controlling user input whether that be from a keyboard, mouse and game controller.

Usually the loop will terminate with a boolean value such as True or False depending on how it has been implemented into the script. In regards to this project a main loop has been put into the following functions:

» Pause
» GameOver
» Game
» Main

The Pause function has implemented a control loop so that the player control what happens once the game is in a pause state. The GameOver function also has the control loop implemented to allow the user to control what happens at the end of the game. Looking at the control code in the GameOver function, it allows the player to press the Escape key on the keyboard to exit the game program and it allows the user to also to press the Enter key to continue onto another game of Tetris.

»

QUICK TIP
Before writing code to define a process, consider designing the process on a whiteboard or paper using process flow and UML (unified modelling language) design. This can also help to reduce the guesswork in getting your processes to work.

```
        else:
            return
    else:
        return
```

The PyGame library is initialised as well as the screen resolution and the main game loop that processes the program. Another useful function that's been implemented is a function called `gridCollison`. This is a function that takes in three arguments – Piece, Grid and Direction – which the function will return True if there's been a collision, or False if there hasn't been a collision. The function is called whenever the player selects a direction for the piece to travel in (left, right, or down). See an example of this below:

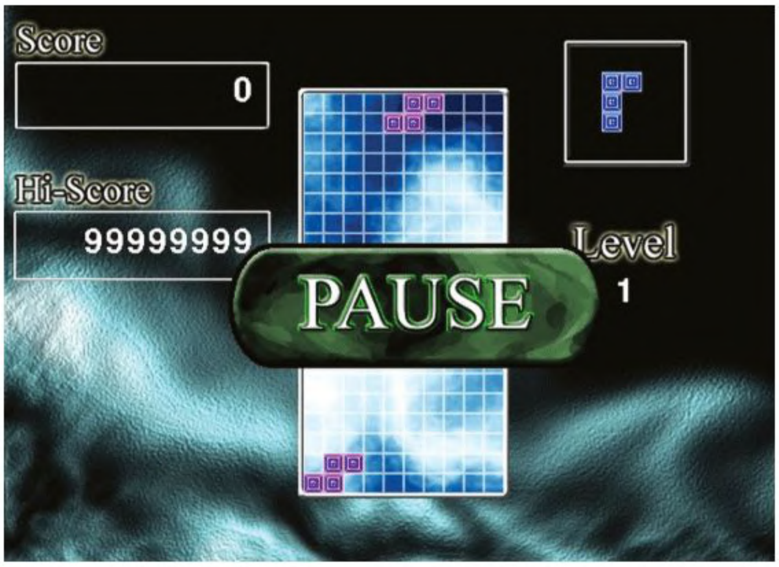
```
if event.key == K_LEFT:
  #Check grid collision if blocks are moved to the left
  if not gridcollision(currentBlock, gameGrid, 1):
    currentBlock.moveleft()
    block_moved = 1
```

As can be seen in the above code example, when the player chooses to move the piece to the left, the function `gridCollision` is called, which passes in the current block, game grid and direction value (which is 1 for left) to identify if there's been a collision. The 'not' keyword is also in the condition that calls the function. Essentially, saying `if gridColliection(..,..,..) == False:` is an alternative way to write this.

Focus on classes

Now that we've covered some of the individual functions featured in the code, we'll now look at some of the classes that have been implemented to represent some objects and concepts we have in the game program. The first class that we'll look at is a class called Block. A number of Blocks are used to make a Piece to be placed on the Grid. The following is the relationship between these class objects:

```
class block:
  """ One of the Four blocks used in a tetris piece """
  def __init__(self, x, y):
    # X and Y are 0-3, and are used to determine where
    # the block is on the 4x4 grid used with Piece
    self.xGrid = x
    self.yGrid = y
```



The ability to pause and restart the current game of Tetris enables the player can move away from the computer to do something else. That cup of tea isn't going to brew itself!

The class Block only contains a class constructor along with x- and y-position values for the piece, which values are passed in on when the object is created. See below for how a Block object is created.

```
if choice == 1:   #Red block is the horizontal bar
  self.blockimage = load_image("redblock.gif")
  self.blocks = [ block(1, 0), block(1, 1), block(1, 2), block(1,
3) ]
```

The above code was extracted from code located in the Piece class code. In the above example, a straight piece of four blocks is created. As much as the comment may say `Red block is the horizontal bar`, in the code it's creating the straight piece in a vertical fashion that can then be manipulated by the player to be horizontal instead of vertical. The main concept to understand about the Block class is that it's the simplest building block element of the game.

Falling to Pieces

The Piece class is used to represent a Tetronimoe, which could be a straight horizontal piece as previously described, a square block, an L-shaped or T-shaped piece that could fall into the *Tetris* playing area and is then manipulated by the player. Specifically, the Piece class can create any one of the following pieces:

- Horizontal Bar
- Square
- T-Shape
- S-Shape
- Z-Shape
- F-Shape

It's also worth mentioning that the Piece class contains methods to control the shape in the playing area such as rotateright and rotateleft to rotate the Tetronimoe left or right. Other movement functions included are moveLeft,moveRight and moveDown.

The class Grid contains all of the dropped pieces during gameplay. An important method to identify in this class is one called `addpiece`. The main purpose of `addpiece` is to add a piece to the game grid. Let's take a look at this method in more detail below:

```
def addpiece(self, piece):
  for i in range(0,4): #add each piece to the grid
    xSet = piece.xVal + piece.blocks[i].xGrid
    ySet = piece.yVal + piece.blocks[i].yGrid
    self.currentgrid[ySet][xSet] = piece.piecechosen
    if ( self.height < 20 - piece.yVal ):
      self.height = 20 - piece.yVal
      # DEBUG STATEMENT ^
```

After adding a piece to the grid, the height is altered so the player has less time to decide how to fit the next piece. The method is called from the Game function, on the occasion the piece collides with another piece in the game grid and on the occasion the piece hits the bottom of the grid. The example below shows a piece being added to the game grid after it has hit the bottom of the game grid.

```

....

else:  # Hit the bottom
  gameGrid.addpiece(currentBlock)
```

To add game controller support, we'll be making modifications in the Game function of the script near to where the keyboard controls are already implemented. In the Game function, there's a main PyGame while loop implemented, with a line that reads like the following code excerpt:

```
for event in pygame.event.get():
```

The game controller that was used for this tutorial was an Xbox One PC-compatible controller. By this stage of the tutorial you should be confident that the controller you want to use is already recognised by your distro of Linux and is working, in this case Ubuntu.

Note that if your distribution is unable to recognise your game controller then the following code will not work. Unfortunately, it's beyond the scope of this tutorial to provide solutions for game controller incompatibility issues.

It's gamepad time

We're first going to implement game controller button controls to rotate the *Tetris* piece as it descends on the game grid. Under the for loop identified above, type the following code:

```
if event.type == JOYBUTTONDOWN:
  if event.button == 1: # B
    currentBlock.rotateright(gameGrid)
    block_moved = 1
  if event.button == 2: # X
    currentBlock.rotateleft(gameGrid)
    block_moved = 1
```

The event type `JOYBUTTONDOWN` identifies that a button has been pressed on the game controller. On the Xbox One game controller used for this tutorial, button 1 equates to the B button to rotate the piece to the right and button 2 equates to the X button to rotate the piece to the left.

There's a chance these actions may be different depending on your type of controller, so you may need to play around to get the correct values to match up to the buttons on your game controller. Next we'll implement the directional controls for the game controller with the following:

```
if event.type == JOYHATMOTION:
  if event.value[0] == -1: # Move Left
    # print('LEFT')
    if not gridcollision(currentBlock, gameGrid, 1):
      currentBlock.moveleft()
      block_moved = 1
      currentBlock.xVelocity = 0
      currentBlock.timer = 0
  if event.value[0] == 1: # Move Right
    # print('RIGHT')
    if not gridcollision(currentBlock, gameGrid, 2):
      currentBlock.moveright()
      block_moved = 1
      currentBlock.xVelocity = 0
      currentBlock.timer = 0
  if event.value[1] == -1: # Move Down
    if not gridcollision(currentBlock, gameGrid, 3):
      currentBlock.movedown()
      block_moved = 1
      currentBlock.yVelocity = 0
```

The event type `JOYHATMOTION` covers the game controllers directional controls (left, right and down). As mentioned earlier, some of these values may be different depending the PC-compatible game controller that you're attempting to use.

The one part we've left out is connecting the video game controller to the pause functionality, which is already implemented for keyboard control. If, for example, we were to know that button number 6 corresponds to the Start/Pause button on the game controller, would you be able to work out the code to write to interface with it now? Over to you! LXF

» OPP IN A NUTSHELL

Even though, in part, this project has been built with and is controlled by individual functions, it's also used object orientated programming (OOP) techniques to control and form the building blocks for the project. For this project there have been three classes created and used, which are:

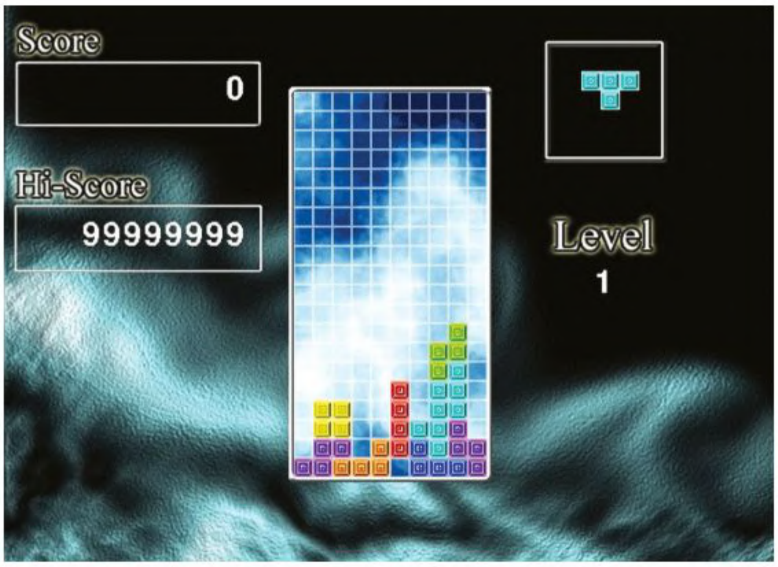
- » Block
- » Piece
- » Grid

Each of the classes contains what's known as a class constructor, which is used to initialise values on creation of an object and to perform basic initialisation operations. Furthermore, classes Piece and Grid contain methods (or functions) that are used to control the behaviour and state of the object during gameplay. In the Piece class, there have been methods implemented so that the player can control the piece object during gameplay, such as moving the piece from left to right, up and down, and rotating the object left or right.

When implementing methods as part of a class, instead of writing the function individually the 'self' keyword is used. The method can then use this to help identify itself as part of that class object. If you've never come across object orientated programming before, think of a class as a self-contained unit of data and operations.

QUICK TIP

When trying to find something in a lot of code, consider using the IDE's search facility to take you directly to what you're looking for.



Our game of Tetris is underway and the player is making slow but steady progress. Each time a line of blocks is completed, it'll disappear and player will be awarded points.

» LET US DROP INTO YOUR LIFE... Subscribe now at **http://bit.ly/LinuxFormat**

NEXT MONTH

CODE, SCRIPT AUTOMATE!

Make your Linux life easier with hacks, scripts and coding secrets to automate your world.

State of the art: 3D printers

We catch up with the latest developments in the making world and test the best enthusiast 3D printers that money can buy.

Screen cast your life

All the tips and tricks to a smooth streaming life, so you can become a top streamer using the best open source tools.

CCTV systems

We have the complete guide to building a working home CCTV system with all the kit, software and Pi you'll need.

What happened to Prolog?

Once touted as the original AI powering logic language, Prolog fell into obscurity. Will the current AI advances see its return?

Contents of future issues subject to change – the first LXF created by AI, 'cause there's no natural intelligence.

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
Big diagram 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
Paul Alcorn, Mats Tage Axelsson, Neil Bothwick, Stuart Burns, Jonas DeMuro, Kseniya Fedoruk, Karsten Günther, Matthew Hanson, Matthew Holder, Jon Masters, Nick Peers, Aaron Peters, Les Pounder, Michael Reed, David Rutland, Mayank Sharma, Shashank Sharma, Andrew Smith, 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
VMD tech Paul Newman
Content director Joe Cox
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

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

9042

Discover the story of paper
www.lovepaper.org
Scan for paper facts, activities, blogs and much more!

9000

LOVE PAPER®
www.lovepaper.org

With thanks to
NATIONAL BOOK tokens