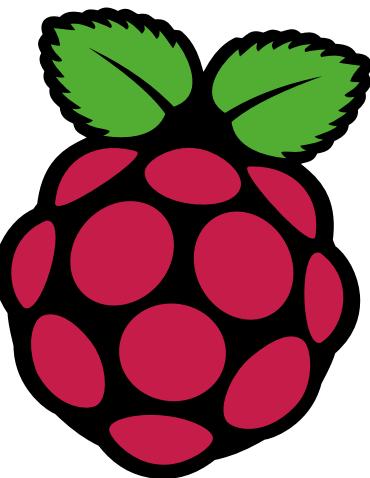




BUY IN PRINT WORLDWIDE [MAGPI.CC/STORE](http://magpi.cc/store)

The MagPi



Issue 85 | September 2019 | magpi.cc

The official Raspberry Pi magazine

RASPBERRY PI 4 YOUR NEXT DESKTOP PC

Can you do everything with a **\$55** computer?

7-DAY
TEST
DIARY

BACK TO
SCHOOL

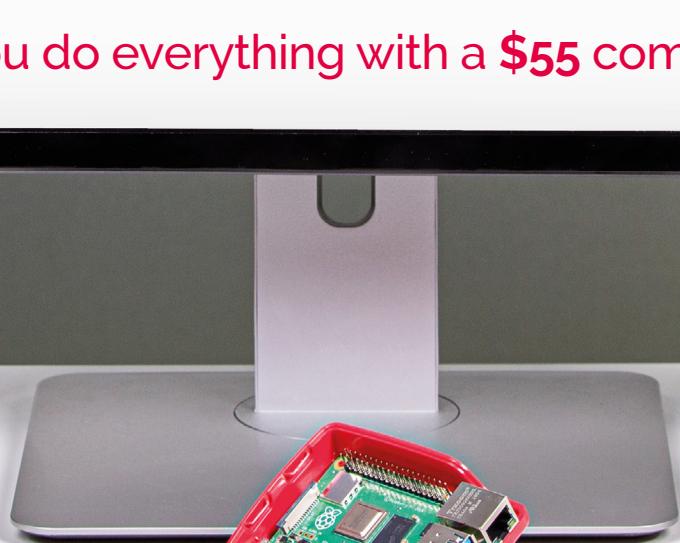
Use Raspberry Pi to
learn & teach coding

PLUS!

LEARN WEB DEVELOPMENT

MAKE AMAZING 8-BIT SPRITES

ROCKYBORG ROBOT REVIEWED!



**SUPER-FAST
NAS DRIVE**

Build a gigabit file server
with Raspberry Pi 4



£5.99

40 PAGES OF TUTORIALS

GLOBAL DELIVERY
magpi.cc/store



3 ISSUES FOR £5



- 📞 Subscribe by phone: **01293 312193**
- 👉 Subscribe online: **magpi.cc/subscribe**

✉ Email: **magpi@subscriptionhelpline.co.uk**

WELCOME to *The MagPi* 85

“Why do you go away? So that you can come back. So that you can see the place you came from with new eyes and extra colours,” said a wise – and sadly no longer with us – man named Terry Pratchett.

I’ve been thinking of that idea with Raspberry Pi 4. The new computer is taking us into different parts of the computing world.

Few things show just how different this year is than our Raspberry Pi 4 Desktop PC feature (page 22). Raspberry Pi is all grown up.

But what is ‘the place you came from’ for Raspberry Pi? Its spiritual home is developing, making, building, creating and, above all, learning. Our Back To School feature (page 64) is packed with all the educational programmes, ideas, and projects you can build with Raspberry Pi. Whether it’s discovering coding with Scratch 3 (page 6), making games (page 56), or building a network storage drive (page 48), there’s no end of things to make in this issue.

We love computing. It’s not enough to just buy stuff and learn to tap icons on a screen. We’re makers, and movers; we need to know the ‘how’, and the ‘why’, and the ‘what we can do for you’ of computing.

Only now we can increasingly do it in one space: Raspberry Pi 4.

“Coming back... is not the same as never leaving.”

Lucy Hattersley Editor



**LUCY
HATTERSLEY**

EDITOR
Editor of *The MagPi*. Can get MiniGo working on Raspberry Pi 4 for a few minutes before it crashes. Fixes old computers without self-electrocution (so far). Runs in the rain out of sheer stubbornness.

magpi.cc



**GET A
RASPBERRY
ZERO W KIT
WITH A SUBSCRIPTION!
PAGE 20**

Contents

► Issue 85 ► September 2019

Cover Feature

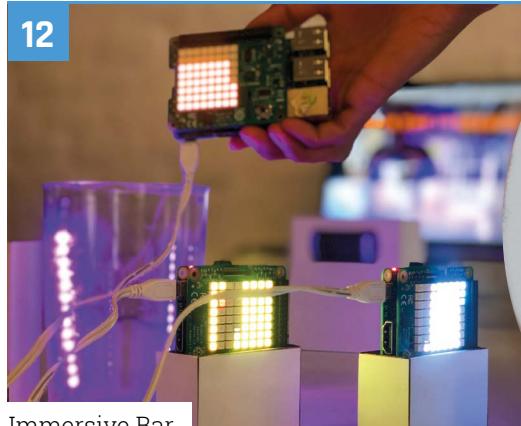
22 Raspberry Pi 4 Desktop test

Regulars

- 06 World of Raspberry Pi
- 92 Your letters
- 97 Next Month
- 98 The Final Word

Project Showcases

- 08 Pi Fighter
- 12 Immersive Bar
- 14 MyPi
- 18 Bird Feeder Monitor



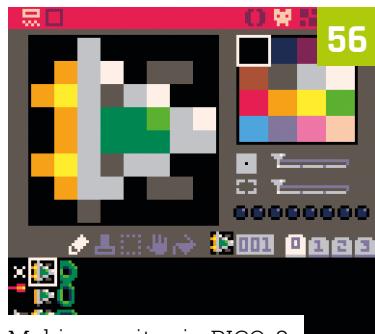
Immersive Bar



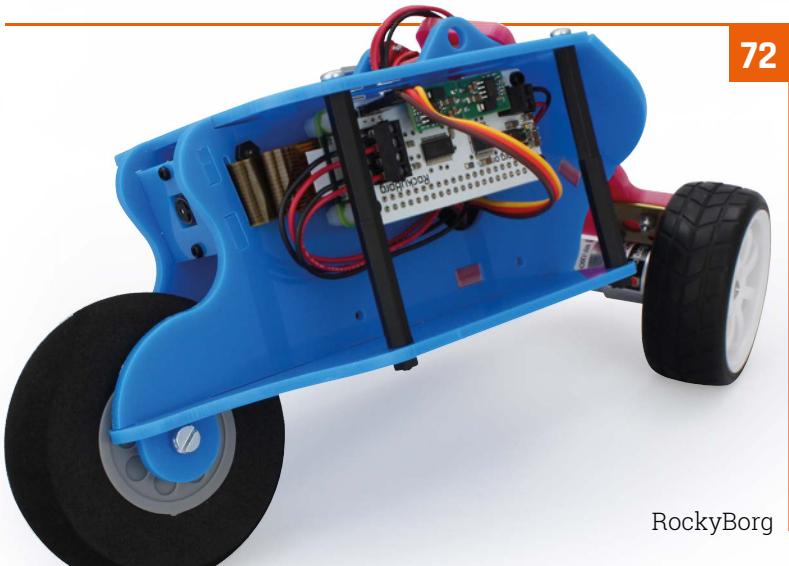
DISCLAIMER: Some of the tools and techniques shown in The MagPi magazine are dangerous unless used with skill, experience, and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. Children should be supervised. Raspberry Pi (Trading) Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in The MagPi magazine. Laws and regulations covering many of the topics in The MagPi magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in The MagPi magazine may go beyond. It is your responsibility to understand the manufacturer's limits.



Rocket Rescue – Pi Bakery



Making sprites in PICO-8



RockyBorg



Denise Leonard interview

**WIN
ONE OF
TEN**

**RASPBERRY
PI 4 CASES**

**95****Tutorials**

- 32** Get started with Mathematica
- 36** Build a low-cost robot part 2
- 40** Rocket Rescue – Pi Bakery
- 48** Build your own NAS
- 52** GUI combo boxes and lists
- 56** Making sprites in PICO-8
- 60** GPIO music box

The Big Feature

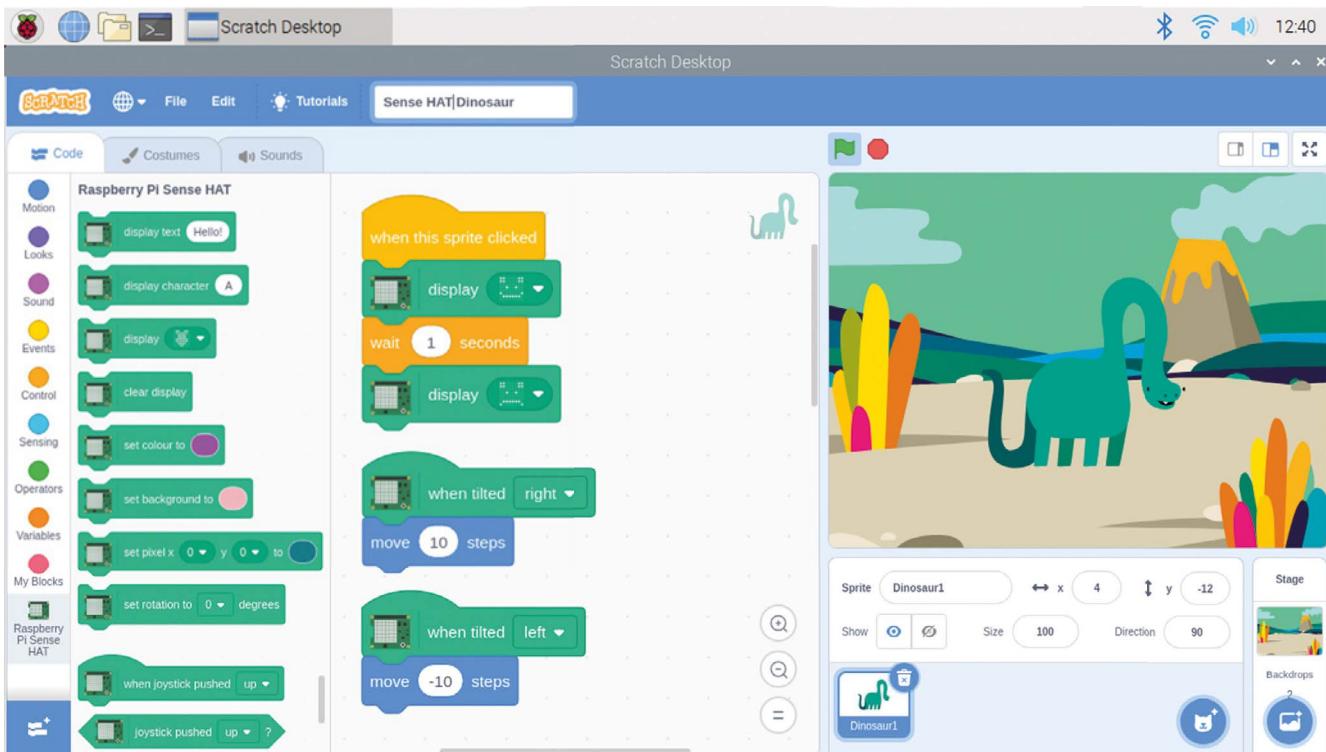
Back to school

Reviews

- 72** RockyBorg
- 74** Pi2Go Mk 2
- 76** LibreELEC on Raspberry Pi 4
- 78** NanoSound DAC 2 Pro
- 80** Top 10 media players
- 82** Learn web design

Community

- 84** Denise Leonard interview
- 86** This Month in Raspberry Pi
- 90** Events



Scratch 3 comes to Raspbian

Scratch 3 is now available on the Raspberry Pi desktop.

What's new in this updated version?, asks **Rosie Hattersley**

▲ Scratch 3 works with Raspbian Buster and adds more characters, backgrounds, and sounds

Scratch is an entry-level programming language that enables novices to take excited first steps into the world of coding and physical computing. The good news this month is that you can now install and use Scratch 3 in Raspbian.

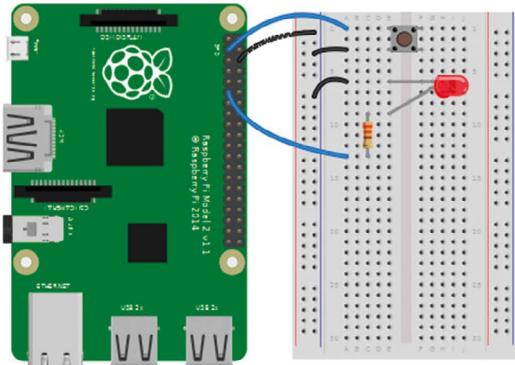
If you encountered physical computing via a Raspberry Pi or at school in the last eight years, chances are it was via Scratch. It's a gateway to eventual Raspberry Pi geekery – but a good one!

The visual nature of Scratch makes it ideal for creating games, animations, and storytelling. So, having native support for the very latest version of Scratch developed specifically to work with

Raspberry Pi is excellent news, and is likely to lead to even more Raspberry Pi users and converts.

Existing Scratch coders have nothing to fear: your existing code and creations will continue to work – but upgrading means you'll have lots of options that weren't there before. To get everyone intrigued, there are lots of brand new Code Club tutorials, project ideas, and video how-tos to explore, as well as an expanded Creative Computing Curriculum from Harvard.

The new version of Scratch lets users create even more imaginatively, with new scenarios and backgrounds, characters, and sounds. Extension support is an integral part of Scratch 3. For



```

when button 2 is pressed
turn LED 17 on

```

```

when button 2 is released
turn LED 17 off

```

▲ Add games buttons and control LEDs directly using the new extension in Scratch 3 for Raspbian

example, using Google Translate, you can share your Scratch-made stories and games with users who speak other languages.

Physical computing

You can program the GPIO pins and the Sense HAT directly, too. The Simple Electronics extension is a beginner-friendly add-on for interacting with the GPIO pins that provides a simpler way of using electronic components, currently buttons and LEDs.

So, if you want to add a new physical controller to your game, it's now a lot easier. To get the very best from Scratch 3 for Raspbian, you'll want to treat yourself to a lovely new Raspberry 4 with at least 2GB of on-board RAM (what an excuse, eh?). Scratch 3 will run on earlier Raspberry Pi models, but may take umbrage if you're trying to run other programs at the same time.

Scratch 3 is free and can be installed from the desktop. You must be running Raspbian Buster. Open up the applications menu, click on Preferences > Recommended Software, and then select Scratch 3 and click on OK.

Look out for a detailed report from the first ever UK Scratch Conference, hosted by Raspberry Pi, next issue. We will also have Scratch 3 tutorials for you. **M**



Raspberry Pi 3B+ gets a fresh outlook

Versatile enclosure system for Raspberry Pi 3B+

The USC-RPI Universal Case System securely holds the Raspberry Pi 3B+/& official 7" touchscreen with lots of room for additional electronics.

The extended UCS range is also available in 2 colours, 4 sizes & 2 heights with optional wall, desk & DIN rail mounting options.



For additional information call 0845 881 2222 or visit
phoenixcontact.co.uk/UCS



One part workout, one part game. Defeat your adversary as you punch

Pi Fighter

Gamifying boxing with a special punchbag that allows you to fight...

Luke Skywalker? **Rob Zwetsloot** starts a training montage to check it out



Richard Kirby

MAKER

A test manager for the London Underground by day, and Raspberry Pi tinkerer by night. Richard also runs the London Raspberry Pi meet-ups.

magpi.cc/rvxxXT

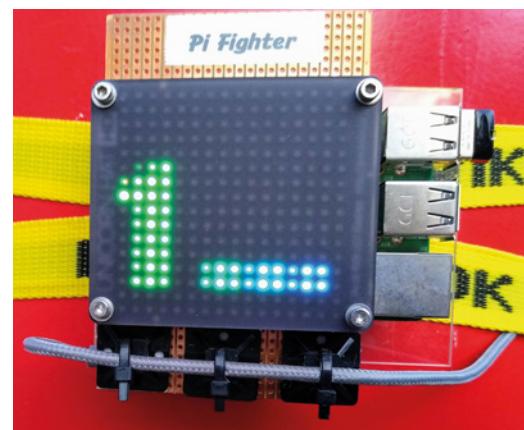
Did you know that the original version of Street Fighter had a variant where you could punch the buttons to get Ryu to attack? The harder you smacked the kick button, the more damage it would do. These apparently wore out very quickly, which is why watching Street Fighter tournaments these days is akin to watching someone playing the piano. Albeit with six buttons and a joystick.

What if you could bring this back? And combine it with other arcade classics and staples? Meet Richard Kirby's Pi Fighter.

A new challenger!

"Pi Fighter is essentially a real-world old-school fighting video game," Richard tells us. "The player chooses an opponent and challenges them to a sparring match. Each player has a certain number of health points that decrement each time the other player lands an attack. Instead of clicking a joystick or mouse button, the player hits a heavy bag. The strength of the hit is measured by an accelerometer. [A Raspberry] Pi translates the acceleration of the heavy bag (measured in G) into

"The player hits a heavy bag. The strength of the hit is measured by an accelerometer"



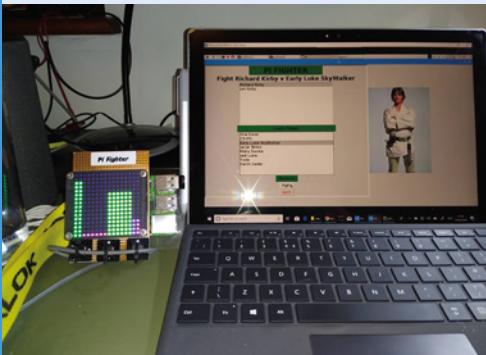
▲ 3... 2... 1... Fight!



Quick FACTS

- ▶ Pi Fighter only uses three electronic components
- ▶ Richard warns against the tricky Jedi fighters
- ▶ Raspberry Pint can be found at CodeNode in London
- ▶ The lack of moving parts made a Raspberry Pi perfect for the job
- ▶ A heavy bag is recommended over a wall bag

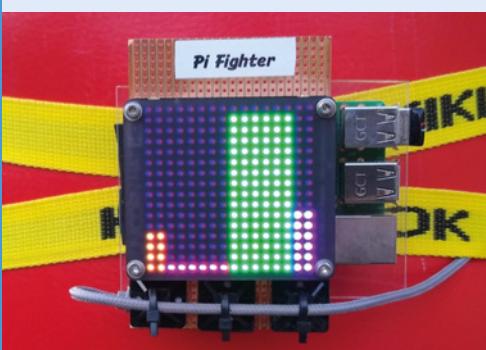
Go for broke!



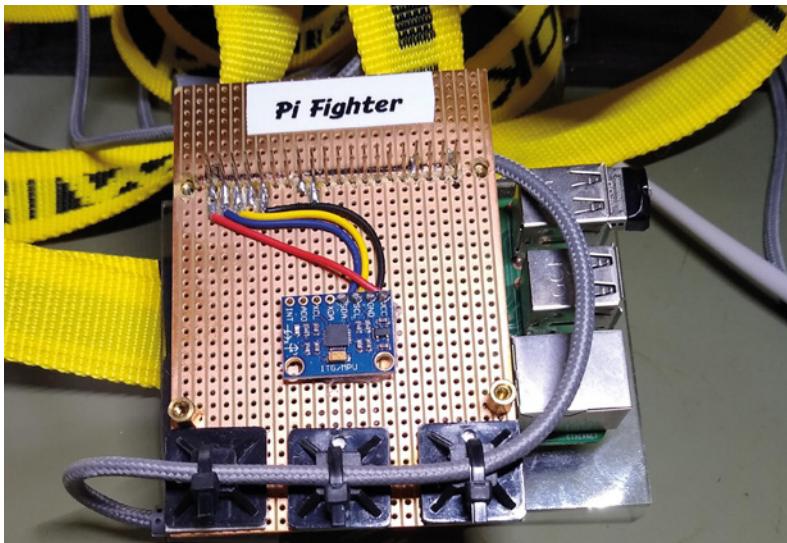
01 Step up to the bag. Your opponent has been selected and you need to defeat them before they can drain your HP.



02 Hit the bag hard and fast. The stronger your punch, the more damage your opponent receives. They do fight back, although at least they don't actually hit you.



03 Once you've defeated your foe, your HP recovers a bit. No time to rest, though: your next opponent awaits. Do you have what it takes to fight Darth Vader?



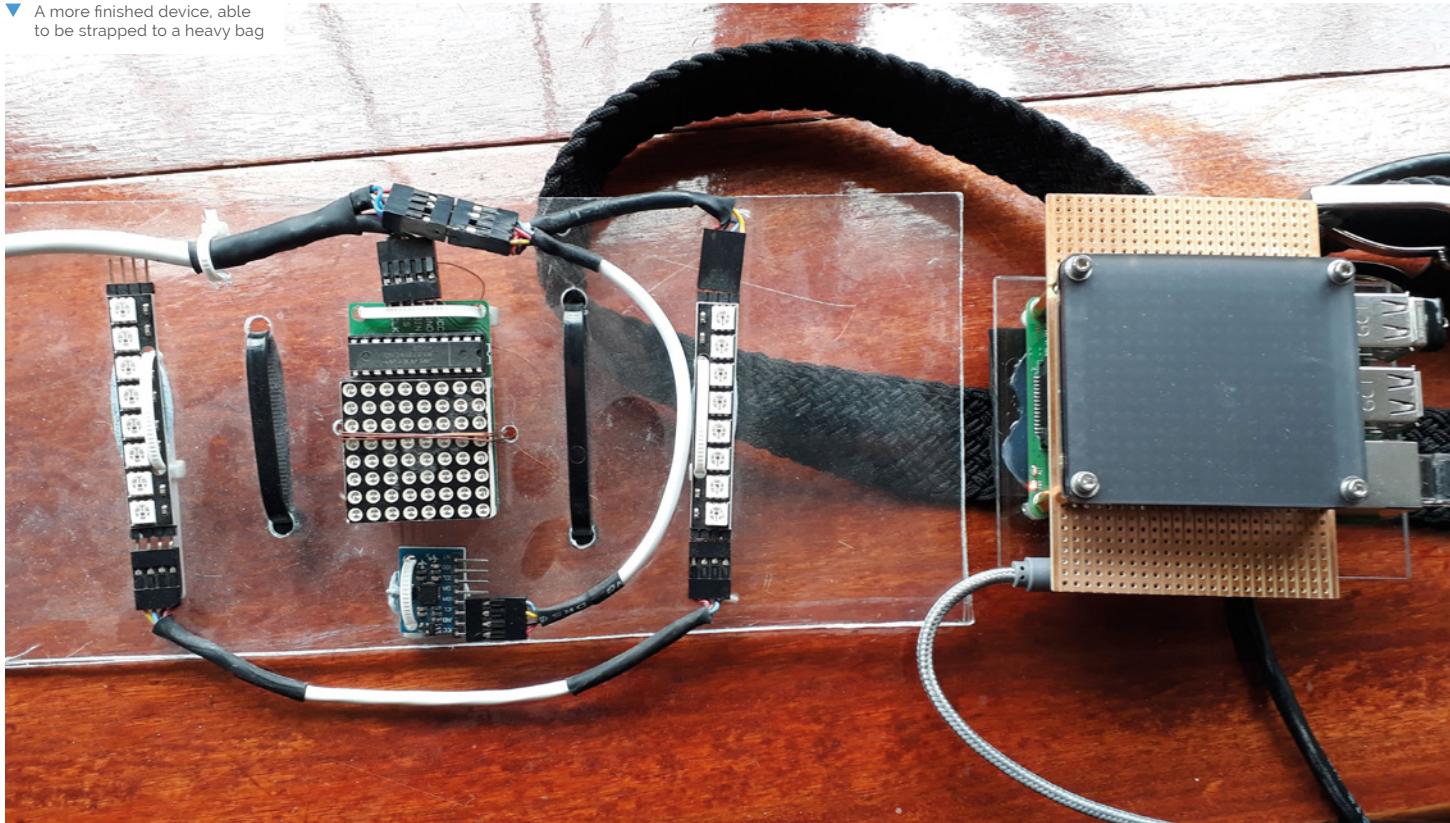
▲ The setup is remarkably simple. It's all in the code

the number of health points to decrement from the opponent. [Raspberry] Pi runs your opponent, which attacks you – you don't actually get hit, but your health points decrement each time they attack.”

It's a remarkably simple idea, and it started off as just an app that used a smartphone's accelerometer. Translating that to a Raspberry Pi is just a case of adding an accelerometer of its own.

“I realised it could be used to measure the overall strength of a punch, but it was hard to know how that would translate into an actual punch, hence the idea to use a heavy bag,” Richard explains. “This appealed to me as I studied karate and always enjoyed hitting a heavy bag. It is always difficult to gauge your own strength, so I thought it would be useful to actually measure the force. The project ended up

▼ A more finished device, able to be strapped to a heavy bag



“I thought it would be useful to actually measure the force”

consuming a good amount of time, as you would expect when you are learning.”

Finish them?

While Pi Fighter is already used at events, Richard says, “It needs a bit of tuning and coding to get everything right [...]. It could be a never-ending project for me. You can always fix things and make the software more robust, the user interface more usable, etc. It isn’t mass rollout ready, but I have never had it fail at a key moment such as presenting at a Raspberry Jam or Raspberry Pint. It (mostly) gets better every time I put some effort into it.”

If you find yourself at Raspberry Pint in London (@raspberrypint on Twitter), make sure to do a bit of a warm-up first – you might find yourself head-to-head in a boxing match with a Jedi. Here’s hoping they don’t know Teräs Käsi. ■



▲ Not many makers get to enjoy punching their own project



MAKER

Aditi Vijay,
Laurin Kraan,
Minatsu Homma

Aditi, Laurin, and Minatsu are students at Copenhagen Institute of Interaction Design (CIID). Their previous backgrounds were in graphic design, graphical user interfaces, and digital consulting.

magpi.cc/TjuzKV

Immersive Bar

Pour Sense HAT 'ingredients' to mix virtual cocktails in this fascinating blend of the physical and digital worlds. **Phil King** is shaken and stirred

Mixed reality (MR) is the merging of the real and virtual worlds to produce new environments where physical and digital objects co-exist and interact in real-time. And what better way to illustrate the concept than by mixing virtual cocktails with ingredients represented by Raspberry Pi boards equipped with Sense HATs?

This was the idea of three students at Copenhagen Institute of Interaction Design (CIID) when tasked, during an 'Immersive Experiences' course, with creating a joyful mixed reality experience which offers a recurring element of surprise.

"Emerging technologies such as mixed reality are increasingly finding their way into everyday life," explains Aditi Vijay. During a brainstorming session, she and fellow students Laurin Kraan and Minatsu Homma came up with the concept for the Immersive Bar.

In this interactive game, the player chooses from three ingredients, represented by SenseHAT-equipped Raspberry Pi boards, and pours them into a cocktail shaker. "The ingredients are unknown, but by trial and error it is possible to find out what



▲ Three Sense HATs and Raspberry Pi boards represent the cocktail 'ingredients' to be poured

they are," says Laurin. "In addition, the colours of the Sense HATs give a further indication of what the ingredients are."

Pour me a drink

When an ingredient is being poured, the Sense HAT's accelerometer detects the angle, while its LED matrix display enhances the visual effect with moving particles. An LED strip – controlled by a fourth Raspberry Pi – in the shaker visualises the liquid level. Once the shaker is full, a laptop computer – running Unity – mixes the cocktail and 'drinks' it before giving a rating. The whole setup is recorded by a camera and displayed on a screen.

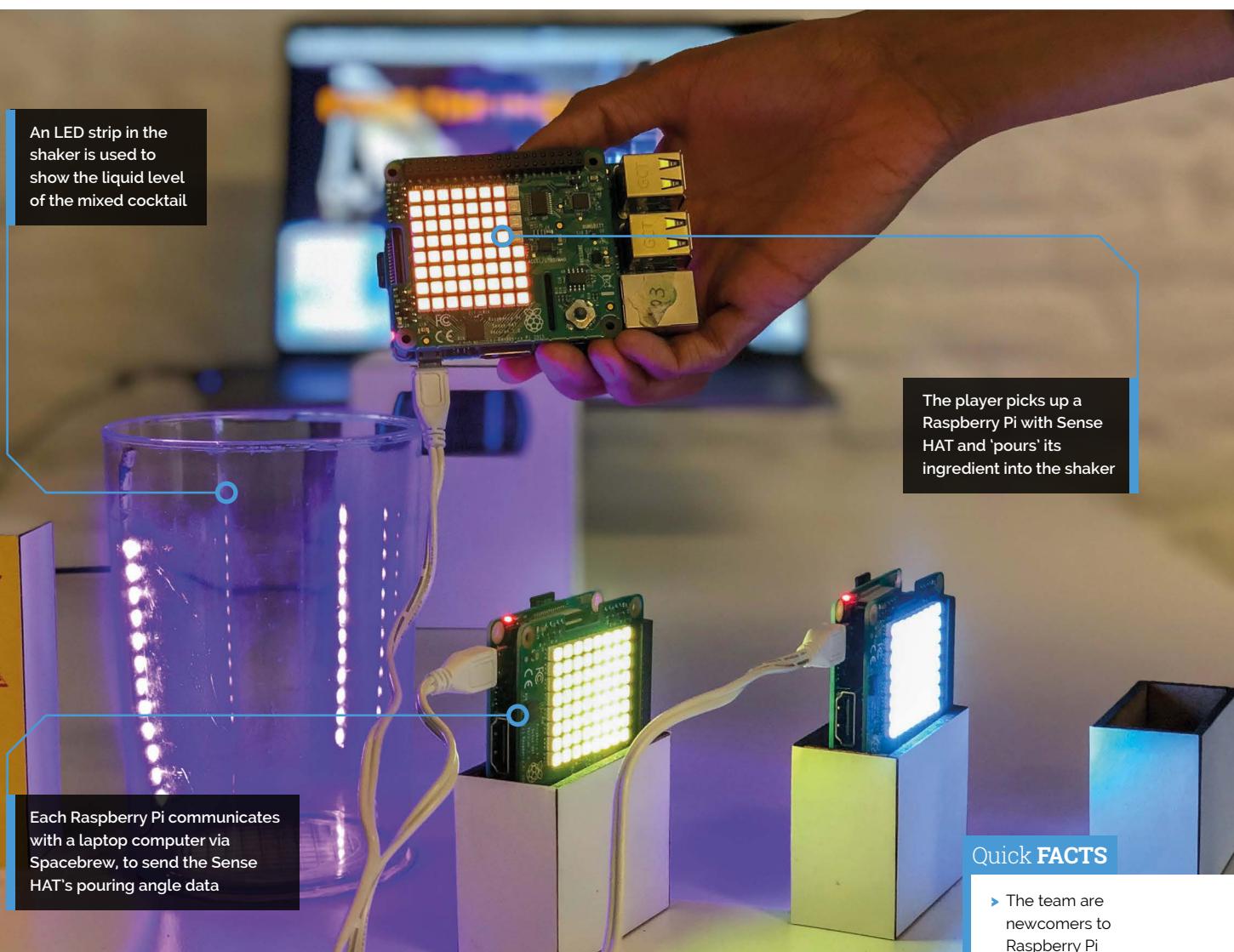
"In total, seven different drinks can be mixed," reveals Laurin, "from pure vodka to exotic cocktails such as the Astronaut." The rating is based on two factors – 1. The combination of the ingredients: while some of them complement each other very well, others lead to a nauseous creation. 2. The balance of the ingredients: is the ratio between alcoholic and non-alcoholic ingredients well balanced?"

The project took the trio just five days to create. "The most difficult part was to turn our concept into a prototype within such a short time," recalls Minatsu. "We iterated fast and explored different digital and physical feedback to create a smooth

Drink: Astronaut Score: 7/10



▲ The cocktail is rated by the computer and a score awarded, which is shown on the screen



and all-round experience. For example, we initially wanted to use a water pump to raise the liquid level inside the shaker while virtually pouring the ingredients. It turned out that the pumps we had were not strong enough and we had to switch to an LED strip for the liquid level indication.”

Student demonstration

When demonstrated to other students, faculty, and guests at CIID, the Immersive Bar project was very well received and soon had people queueing up to play. You can watch a video of the cocktail-mixing fun at magpi.cc/MZbAXC.

“The merging boundaries between the physical and digital worlds left the audience amazed,” says Laurin. “The demonstration soon turned into a competitive experience, in which the players tried to figure out how to achieve a better score than their friends.” ■

“The merging boundaries between the physical and digital worlds left the audience amazed **”**



Quick FACTS

- ▶ The team are newcomers to Raspberry Pi
- ▶ The project took five days to design and create
- ▶ Hardware items communicate wirelessly via Spacebrew
- ▶ Seven different drinks can be mixed
- ▶ The ingredients are a mystery to the player

◀ The project was a big hit with fellow students and guests when demonstrated at CIID

MyPi

MyPi, a custom-built Raspberry Pi-controlled retro games pad, takes its cue from iconic 1980s and 1990s gaming devices and now comes as a kit, as **Rosie Hattersley** discovers



Jesse Lewis

MAKER
It was love at first sight for Jesse – of Chandler, Arizona – and electronic gaming: after first seeing the Commodore VIC-20, he spent six months saving up his paper round earnings for one before going on to a career in electronic controls engineering.

makemypi.com

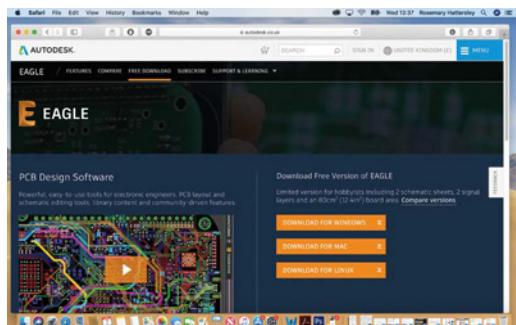
► At first, Jesse designed his own circuit boards using Autodesk EAGLE

A life-long love of games, electronics, and programming, plus some time freed up after completing a degree in IT, led retired electronics specialist Jesse Lewis to begin experimenting with mobile gaming possibilities for his Raspberry Pi. As a teenager, he had written his own games for the Commodore VIC-20.

The introduction by his son to retro gaming using RetroPie, along with a chance encounter with a Raspberry Pi in a charity shop, sparked Jesse's imagination about how to create a non-wired, handheld games console with a screen large enough to play comfortably. "The clarity wasn't bad, but everything was so tiny. I wanted a gaming unit that would not require me to wear my reading glasses to play games," explains Jesse.

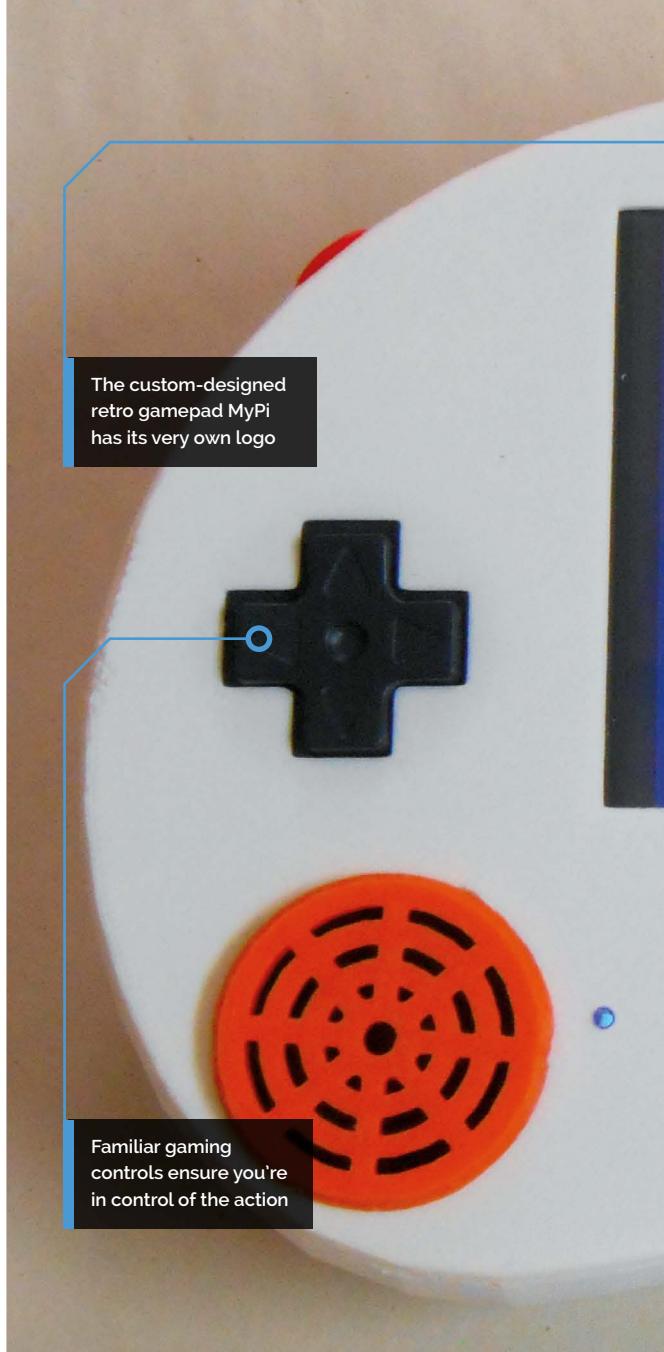
Building a handheld

At first, Jesse tried a 7-inch Android tablet screen, but this proved too large and unwieldy for his preferred mode of gaming: in a recliner or relaxing on the sofa. Instead, he chose a 5-inch screen as the basis of what would become the MyPi handheld gaming console.



The custom-designed retro gamepad MyPi has its very own logo

Familiar gaming controls ensure you're in control of the action



He decided he wanted stereo sound, to have as few wires as possible, and to emulate a gamepad shape and experience. Given his electronics background, Jesse initially used Autodesk EAGLE to design and print several designs of circuit board.

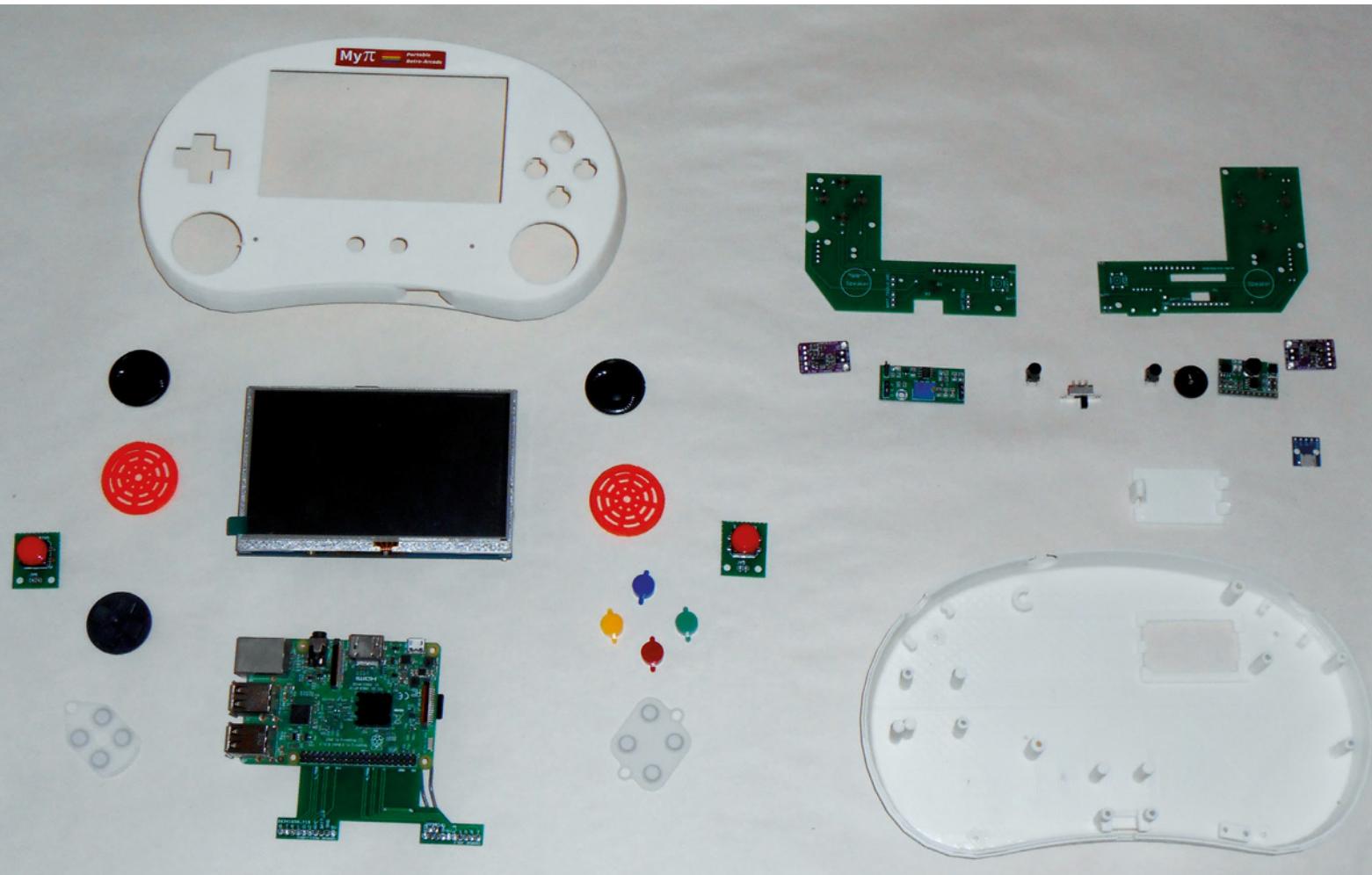
Components such as an amplifier were bought off the shelf, at which point Jesse realised it made more sense to use inexpensive modules and components with large, easier-to-solder battery management packs, rather than continuing to experiment with his own circuit boards.

Power-wise, the obvious choice was a Pimoroni LiPo SHIM, since this is specifically designed for compact mobile devices. It offers 1.5 amps of power, along with a low battery warning so the MyPi can be safely switched off and any potential damage to the microSD card avoided. However, Jesse found the LiPo SHIM was only effective when

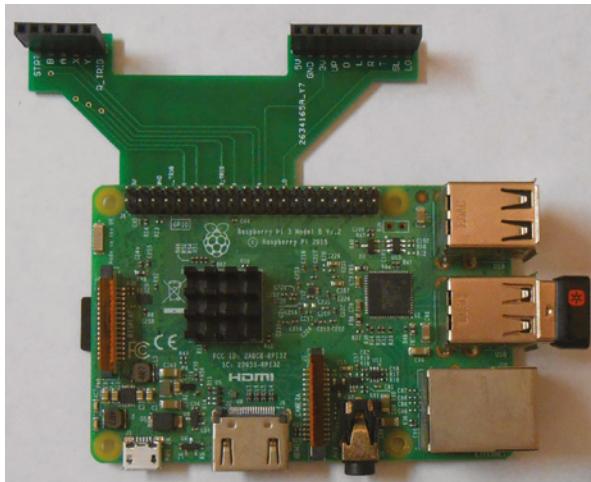


Quick FACTS

- ▶ Jesse was an avid arcade gamer in his youth
- ▶ His son bought him his first Raspberry Pi
- ▶ Playing Wolfenstein 3D catapulted Jesse into games programming
- ▶ His next handheld will be based on an Atari 5200, without the hated joystick!
- ▶ MyPi is available from Tindie



- ▲ Off-the-shelf components sourced online proved easier and more cost-effective
- ▶ MyPi was originally built around Raspberry Pi 3B

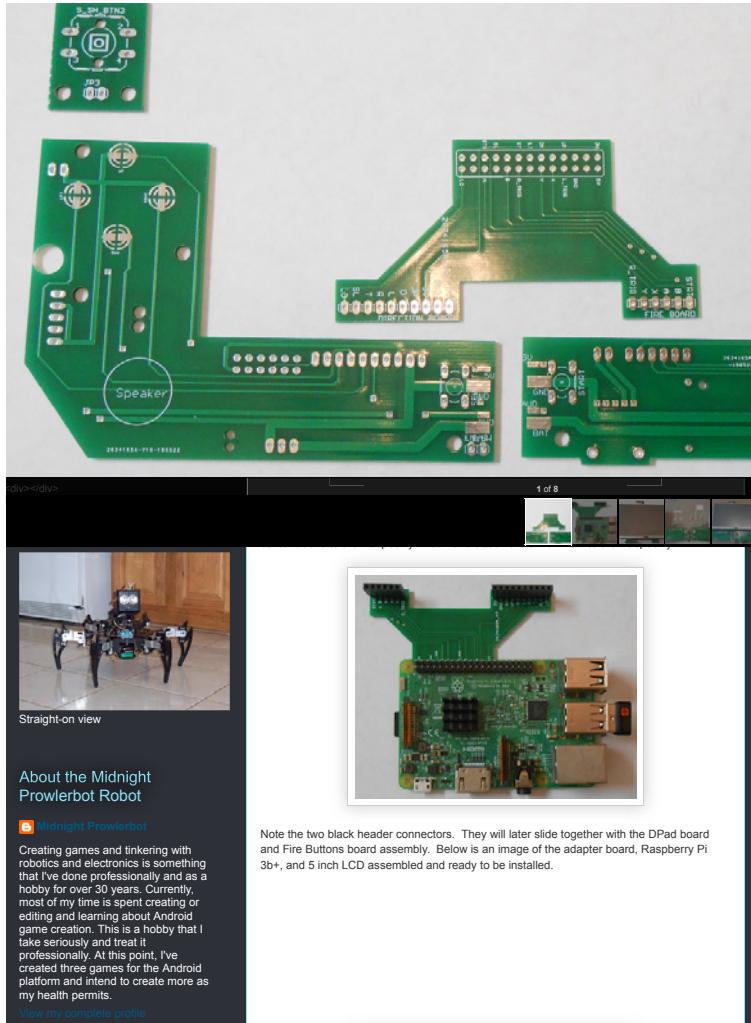


the MyPi was being powered via USB. On battery power alone it lasted just ten minutes – not long enough for a lengthy mobile gaming session. Instead he had to substitute a 2 amp load-sharing module and add a switch to power off the MyPi when its battery began to run low.

The final stages of the build were to add ventilation and covers for the speakers, after which the MyPi was all set for games.

Get your own MyPi

MyPi is available as a self-assembly kit for Raspberry Pi, with the drivers and software for RetroPie supplied on the accompanying microSD card. RetroPie can be used alongside an existing Raspberry Pi install, as well as being available as a discrete install. The Steelseries Stratus XL Bluetooth controller and drivers for the touchscreen are also preconfigured on the card.



I wanted a gaming unit that would not require me to wear my reading glasses to play games

If you want to 3D-print it, the MyPi case design is available on YouMagine (magpi.cc/Oaeugf), while the schematic file and stickers can be found on GitHub (magpi.cc/cCtByf).

For those who prefer to source and assemble everything for themselves, Jesse warns that configuring RetroPie can be challenging. The zip file for MyPi isn't on GitHub due to the whopping file size. However, says Jesse, "All of the software used can be installed and configured by anyone if they chose to spend the time learning about RetroPie and the LCD driver software."

▲ The MyPi kit comes with an old-school comprehensive instruction manual

Getting gaming



01 Selecting appropriate trigger buttons was a lengthy process, not least due to variations in height when fixed to the MyPi's circuit board. After examining the innards of several gamepads, Jesse chose carbonised rubber buttons.



02 The MyPi has a 5-inch screen and gamepad buttons plus a 2 amp SHIM with load-sharing. Jesse added a physical on/off switch to power down when the battery begins to deplete, thereby preventing damage to Raspberry Pi's microSD card and OS.



03 Jesse got his friend Scott Fillerup to 3D-print the gamepad case they designed together using SketchUp. Printing the case took several attempts due to the precision involved and different tolerances.



Bird Feeder Monitor V2.0



Stephen
B Kirby

Retired civil servant Stephen B Kirby worked in the medical industry for 30 years and he particularly enjoyed the roles that involved computer programming, saying it scratched a creative itch.

magpi.cc/oeWTca

MAKER

Catching the early bird is much easier when Raspberry Pi computers are involved, as **David Crookes** discovers

Aside from tinkering with computers, Stephen B Kirby loves to watch birds perching in his garden, but he can't always be around to see them. To get around that problem, he devised a bird feeder monitored by a capacitive touch sensor and a camera. As Raspberry Pi projects go, it has proven to be a real feather in his cap.

The idea was born from an earlier, similar project which made use of an Arduino Yún. As one of the first non-shield WiFi Arduinos, Stephen says it was perfect for the task of counting and timing the visits of the birds at his feeder. "It performed the job well for a long period of time," he tells us.

Stephen wanted to go further, however. "I decided to use more than one Raspberry Pi computer to create an interactive network of devices to perform specific monitoring tasks," he says. "I wanted this project to detect the birds on each perch and use an integrated Raspberry Pi

camera for photographs. Most importantly, I added a Raspberry Pi MQTT broker to control, monitor, and store the data for the project."

A good egg

MQTT (mqtt.org) is a lightweight messaging protocol for Internet of Things deployments, and one of its key benefits is an ability to scale. "MQTT is versatile and it can handle multiple bird feeders, cameras, or other Internet of Things devices because it has an ability to communicate with a wide variety of hardware," Stephen explains. Using MQTT also enabled him to move away from storing data in a spreadsheet on Google Drive.

"The MQTT server [...] requests and stores data from the Bird Feeder Monitor, controls its operation, and activates the monitor at dawn and shuts it down at dusk. It also controls the timing interval for requesting data, and it monitors and records the current weather conditions."



“I initially thought I would witness some sort of routine spikes of feeding”

When a bird lands on one of the feeder’s perches, it activates a capacitive touch sensor connected to Raspberry Pi Zero W via a CAP1188 breakout board. The data is then sent to and stored on a MQTT server on a Raspberry Pi 3B+, starting a timer to determine how long the event lasts.

At this point, a MQTT message is published by the Bird Feeder Monitor’s MQTT server telling the Camera Module (connected to a third Raspberry Pi) to take a photo. A note of the number and time of the perches is made along with the local temperature, humidity, cloud cover, wind speed, wind gust, and wind direction. The photos of every feathered friend are stored on the Raspberry Pi and they can be managed remotely.

Pecking order

So far, the results have been surprising. “I initially thought I would witness some sort of routine spikes



To ensure it doesn’t get wet, the camera is placed inside Stephen’s garage, peeking at the bird feeder through a window

of feeding,” Stephen says. “For example, I thought we would see a large spike in the morning and another spike before dusk, but we discovered that birds are hungry all day long. Secondly, birds treat my six-perch bird feeder like a buffet. They fly on to one perch and then proceed to hop from one perch to the next. Many of the birds make a point to circumnavigate the entire feeder, but it means the bird counts are inaccurate.”

Even so, it’s something Stephen intends to work on. He’s also hoping Cornell University, which publishes the Merlin bird-identifying app (merlin.allaboutbirds.org), will create an open-source Linux version. “I’d love to be able to auto-identify the birds,” he says. “Maybe that will happen someday.” **M**

Quick FACTS

- ▶ Three Raspberry Pi devices are utilised
- ▶ Any plastic or wooden-perched bird feeder is suitable
- ▶ MQTT communications are key to the project
- ▶ Current weather conditions are monitored via DarkSky
- ▶ The software only works with a Raspberry Pi Camera Module

SUBSCRIBE TODAY FROM ONLY £5

SAVE UP TO **35%**



Subscriber Benefits

- **FREE Delivery**
Get it fast and for FREE
- **Exclusive Offers**
Great gifts, offers, and discounts
- **Great Savings**
Save up to 35% compared to stores

Rolling Monthly Subscription

- Low monthly cost (from £5)
- Cancel at any time
- Free delivery to your door
- Available worldwide

Subscribe for 12 Months

£55 (UK) £90 (USA & Rest of World)
£80 (EU)

Free Raspberry Pi Zero W Kit with 12 Month upfront subscription only (no Raspberry Pi Zero Kit with Rolling Monthly Subscription)

📞 **Subscribe by phone: 01293 312193**
👉 **Subscribe online: magpi.cc/subscribe**

✉️ Email: magpi@subscriptionhelpline.co.uk

JOIN FOR 12 MONTHS AND GET A

FREE Raspberry Pi Zero W Starter Kit

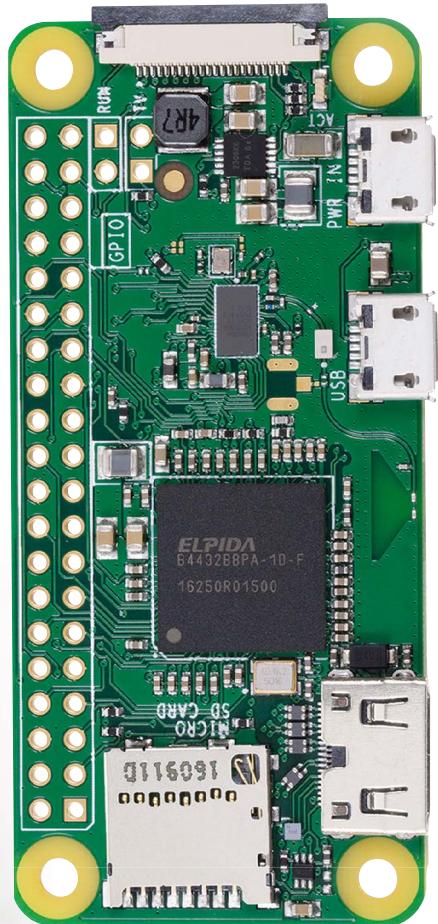
WITH YOUR SUBSCRIPTION

**Subscribe in print
for 12 months today
and you'll receive:**

- ▶ Raspberry Pi Zero W
- ▶ Raspberry Pi Zero W case with three covers
- ▶ USB and HDMI converter cables
- ▶ Camera Module connector

Offer subject to change or withdrawal at any time

**WORTH
£20**



Buy now: magpi.cc/subscribe



**SUBSCRIBE
on app stores**

From £2.29

Available on the
App Store

GET IT ON
Google Play



RASPBERRY PI 4

YOUR NEXT DESKTOP PC

PJ Evans puts a Raspberry Pi 4 to the ultimate test: a full desktop replacement

When Raspberry Pi 4 was launched earlier in 2019, the significant improvements in processor speed, data throughput, and graphics handling lead to an interesting change of direction for this once-humble small computer. Although impressive that you can run a ‘full’ Linux operating system on a \$35 device, a lot of people were just using it to get Scratch or Python IDLE up and running. Many people were skipping the graphical side altogether, and using smaller models, such as Raspberry Pi Zero, for projects previously covered by Arduino and other microcontrollers.

Raspberry Pi 4 was different. Tellingly, the Raspberry Pi Foundation released a new all-in-one kit and named it the ‘Desktop Starter Kit’. For the first time truly in Raspberry Pi history, it was considered powerful enough to be used as a daily computer without any significant compromise. Challenge accepted. We asked PJ Evans to spend a week using a Raspberry Pi 4 as his only machine. Here’s what happened.

Day 1 | Monday

Decisions, decisions



▼ The Desktop Starter Kit contains everything you need to get started



Our new favourite single-board computer comes in a selection of RAM sizes: 1GB, 2GB, or 4GB. With a difference of £20 between the 1GB and 4GB versions, it made sense to go right for the top specification. That's the version included in the official Desktop Kit (magpi.cc/hDYcvr) that I went out and bought for £105 (inc. VAT) at the official Raspberry Pi store – it normally retails for \$120 plus local taxes. My last laptop was £1900. I'm not suggesting that the two can be reasonably compared in terms of performance, but £1795 minus the cost of a monitor is a difference worth remarking upon.

Back at the office, I inspected the contents. For your money you get a 4GB version of Raspberry Pi 4, thoughtfully already installed in the new official case; the official keyboard and mouse; the new USB-C power supply; a 16GB microSD card preloaded with the Raspbian Buster operating system; and a copy of *The Official Raspberry Pi Beginner's Guide* 252-page book. It's very well packaged and presented, with little plastic waste. The book is the icing on the cake if you are looking at this set for a young person's first computer, short-circuiting the 'now what do I do?' stage. What pleased me

in particular was the inclusion of two micro-HDMI cables in the kit, allowing me to set up a dual-screen system without delay.

First tests

I set up my new workstation next to my existing laptop, with two 1080p monitors that only had DVI connectors, so I had to get a couple of £2 adapters (magpi.cc/NwxNFB) and an additional cable to get sound out of the audio jack of my Raspberry Pi. Time for an initial test-drive. Booting up into Raspbian Buster was quick, about ten seconds, and connection to WiFi easy. There's no doubting the feel of the speed improvements. Yes, I've read all the benchmark tests (magpi.cc/benchmarks), but I wanted to know how that translates to user experience. This new kit does not disappoint.

Raspbian has matured impressively as an OS. For my daily desktop scenario, the jewel in the crown is Chromium: having such a capable web browser is what makes this whole experiment feasible. Others have upped their game, too: Firefox has come a long way, and many other browsers are now available, such as Vivaldi (vivaldi.com). A check of some of my most visited sites showed Chromium to be just as capable as Chrome on my regular machine. Unsurprisingly, it wasn't as snappy and I hit a few bumps, but we'll get to that.

A day of impressions

I'm no expert when it comes to GPUs, but I was impressed with the dual-monitor support. The setup worked first time and didn't seem to have any detrimental effect on the machine's performance. I was expecting slow window drawing or things getting 'stuck', but this wasn't the case.

By the end of the first day, I was getting used to the keyboard and mouse too. They are a nice mixture of being both functional and aesthetically pleasing. The keyboard comes with a three-port hub, so you can connect the mouse if you wish. It does not have the build quality and precision of my daily wireless keyboard and trackpad, but for a fraction of the price I was surprised how much you got for your money. By the end of the week I'd grown quite fond of it.





Day 2 | Tuesday

Back to basics

There's a much shorter version of this feature that reads 'Install Raspbian, use web apps, it'll be great'. Seriously, if you're OK with the cloud, you could now go for a long swim in the Google App waters and have nothing more to worry about. Gmail, Calendar, Drive, Docs, Spreadsheet, and the list goes on (see gsuite.google.com). A full suite of business software awaits you, and the same can be said for Office365 (office.com). I successfully used both and although they do hit the CPU hard, both were functional and certainly the most friction-free way of doing day-to-day work. However, many of us would prefer to use open-source software, have privacy concerns, or do not wish to rely on a good network connection, so let's look at the alternatives.

When it comes to the standard suite of office applications – word processing, spreadsheets, and presentation software – Raspbian users are already set up and good to go with the LibreOffice suite. Six applications, all for free and, in their current incarnations, more than fit for purpose. Previous Raspberry Pi models have struggled to run these admittedly large apps at speed, but Raspberry Pi 4 has no such issues. In fact, this feature has been written solely using LibreOffice Writer and although it's noticeable that you're not on the world's fastest computer, it is more than adequate for even complex word processing and layout.

Users are already set up and good to go with the LibreOffice suite

Beyond installed software

I'm going to need email, calendars, and contacts. I ideally want to be able to access these anywhere. My current email provider uses IMAP, so that gave me a choice of clients to look at. The Claws email client is provided with Raspbian. Setup went smoothly but although fast, I found it lacking in features and had an old-school appearance that



reminded me of Eudora from the 1990s. Luckily, an old favourite, Thunderbird, was available to install (thunderbird.net). This long-term Mozilla project is still very popular. Installation and configuration was straightforward. I couldn't say the same for calendaring. Lightning, the calendar plug-in for Thunderbird, refused to talk to my remote iCloud calendar. Some research showed that Apple, although supporting the CalDAV protocol, doesn't play nice and several clients have issues.

In the end, my platform of choice for email, calendars, and contacts was Evolution (magpi.cc/cXJvYJ). This attractive, fast, and actively maintained app provided everything I needed. It's not installed by default, but can be swiftly installed using 'Add/Remove Software' under 'Preferences' in the Raspbian menu. If I was happy with a local calendar and contact list, that would be that, but I'd prefer to have access on my phone too, wherever I am. Currently, I use iCloud to do this, but is there a workable open-source alternative where I own the data?

▲ The desktop setup in all its dual-monitor glory

How to install?

Most of the software featured can be installed using 'Add/Remove Programs' in Raspbian's desktop menu, or APT on the command line.





Day 3 | Wednesday

Into the cloud

In for a penny, I thought, and decided to look for an alternative to my current cloud provider that would support all the common protocols and allow me to keep control over my data. After a few false starts, two packages provided workable solutions.

Nextcloud (nextcloud.com), a fork of popular open-source solution ownCloud, provides file storage, calendaring, contacts, note taking, and task management. Its well-supported plug-in system allows you to extend those capabilities further. Raspberry Pi is normally seen as a server platform for Nextcloud, but now we're looking to be the client too. To my surprise, I was able to get the Nextcloud server running easily on my Raspberry Pi 4 using Docker, a system for 'containerising' complex software by wrapping it in a cut-down operating system. I now had a locally running web service that offered network access to calendars, contacts, and more. Evolution connected first time, supporting tasks as well. Nextcloud offers apps for iOS and Android, allowing you to sync calendars, contacts, and files across platforms.

■ I now had a locally running web service that offered network access to calendars, contacts, and more ■

Cloud saviour

Nextcloud's Files feature is particularly worthy of note. This can replace services such as Dropbox in your workflow by acting as a central file repository. Your Raspbian desktop can connect using WebDAV, a file networking protocol, but if you're after proper file sync, there is a Nextcloud Desktop application available. Unfortunately, it's not available from repositories yet, but instructions are on Nextcloud's forums on how to get the source code and compile it yourself. I now have a folder in my home directory that is always in sync with the server. Best of all, if I go to the server's built-in website, I can share files and create one-off links that I can give to others.

The only hitch is where to run the server. For testing, I was running it locally (and it's worth pointing out that my Raspberry Pi was quite happy running my desktop, Docker, and the NextCloudPi (magpi.cc/spLCMp) server image: near-zero CPU while idling and only 2GB of memory in use), but that means outside of my local network I would have no access. In the long-term, I'm going to put Nextcloud on a dedicated Raspberry Pi and configure port-forwarding on my router. Alternatively, I may set up a virtual server 'in the cloud' so I can run it from there. A complete file and data sharing solution and I own all the data.

Syncthing (syncthing.net) is a simpler way of getting files about without worrying about servers. Just install Syncthing on every machine with which you want to share files. On each machine, you can select which directories you want to share and with whom. It's very easy to set up, detecting other Syncthing instances on the local network and quickly linking them up if you so wish. It's also highly configurable, with features such as selective syncing and bandwidth throttling. Syncthing is available for a wide range of platforms, so this is a great way to collaborate with Windows and macOS users. However, there are currently no apps for mobile devices available, so access to files 'on the go' is restricted.

Honorary mention: Rclone (rclone.org) – an open-source command-line tool for syncing directories with a number of third-party services such as Dropbox and Google Drive.

Scorchio!

The third day of testing gave me an unexpected problem. I felt there was something wrong: Raspbian felt sluggish and was struggling to run smoothly. I then noticed the dreaded thermometer icon appearing in the top-right of the desktop. The CPU was at 85 °C! I quickly removed the case cover, which had an immediate positive effect. Unfortunately, the heatwave the UK was experiencing was too much for the little computer. A fan was quickly deployed, bringing the temperature down to a calmer 45 °C. Consider a heat-sink, Pimoroni Fan SHIM (magpi.cc/qZYBWd) or Raspberry Pi PoE HAT (magpi.cc/poe).





Day 4 | Thursday

Photos and video

Now for something a bit trickier. Raspberry Pi 4 comes with a dedicated GPU, the Broadcom VideoCore VI, making it by far the most powerful Raspberry Pi in terms of graphics performance. This is great news, but is it good enough to cover the demands of photo and video work? Surely the larger PC cousins of Raspberry Pi will have the edge?

The first ‘real world’ problem to be solved was how to get photos onto Raspberry Pi in the first place. I have an iPhone, and iOS is notoriously uncooperative with things that are not Apple. I could fumble around with forwarding photos via email or using Rclone with Dropbox, but I wanted something far more elegant like the workflow I was used to: take the photo and it appears on your computer. Thankfully, Nextcloud came to the rescue again. The iOS and Android apps can both automatically sync photos back to the server as they are taken. The Nextcloud Desktop app in turn syncs the new files to the local file system. All tests worked perfectly.

First attempt in learning

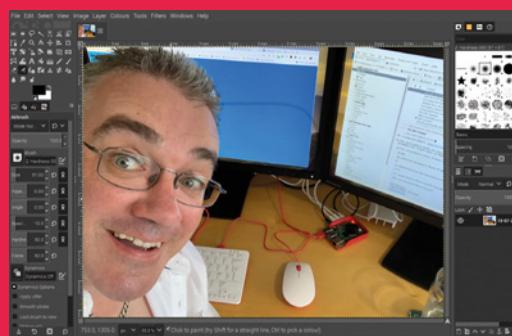
At this point, I hit the first failure of this experiment. The people’s choice for Linux photo management software appeared to be digiKam (digikam.org), and the features and screenshots did look impressive. Despite several attempts, however, the software flatly refused to run, getting stuck when starting up. This may be due to my using Raspbian Buster, or some other

issue I was unable to identify. Instead I settled on Shotwell (magpi.cc/uLYpzW), which not only worked first time but provided an intuitive and simple interface. Shotwell provides all the basic functionality you need to manage a photo library and provides the fundamentals for making small touch-ups and corrections to images, such as auto-enhance, cropping, rotation, and colour adjustments.

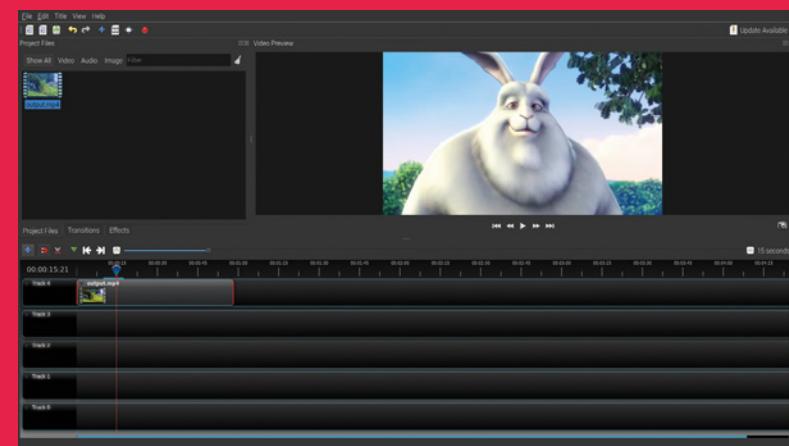
For more comprehensive photo editing, GIMP (gimp.org) is probably the most popular choice in the open-source community: a completely free, full-featured smorgasbord of image editing tools matched only by professional packages like Adobe Photoshop. This is a beast of an application and has struggled on previous Raspberry Pi models. It’s still not the fastest experience, and would quickly become frustrating if you were working in it all day every day, but for the occasional bit of work it is perfectly usable. If GIMP’s complexity looks like too much for you, check out Mirage instead.

Video editing was always going to be a big ask for such a low-cost platform. I installed OpenShot (openshot.org), a non-linear multitrack video editing platform and, sure enough, although it worked, it did struggle with the sample 720p MP4 video I imported. Raspberry Pi has dedicated hardware to handle the H.265 (HEVC) video codec only, so anything else has to be processed solely by software.

▼ For video editing, OpenShot works fine but struggles with MP4 video files



▲ The GIMP image editor is a sophisticated application and runs well enough on Raspberry Pi 4



Day 5 | Friday

Fun

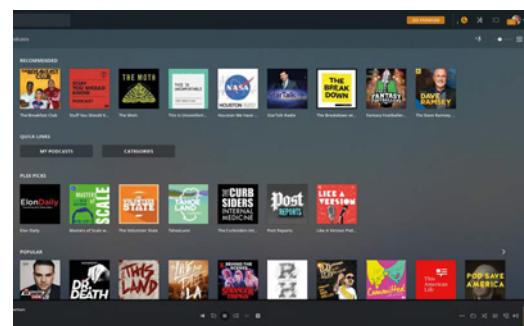
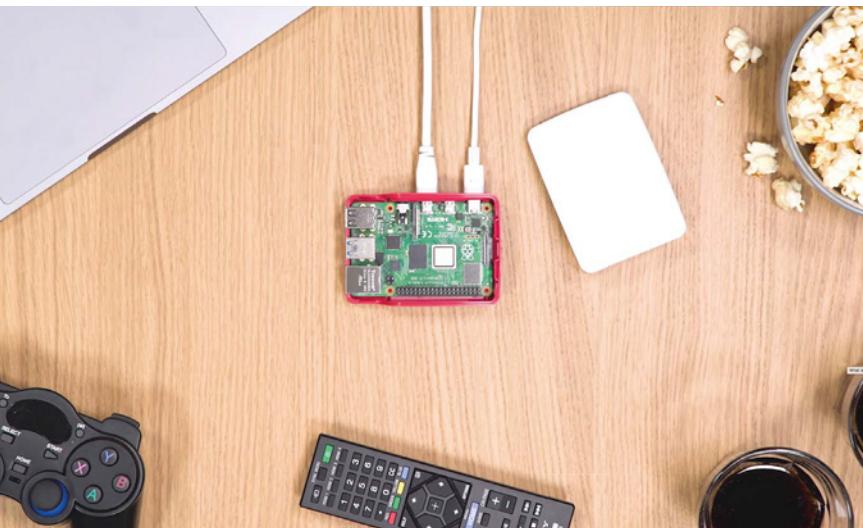
Which OS?

We've focused on Raspbian here, but you may consider Ubuntu MATE, CentOS, or openSUSE. Get multiple microSD cards and try them all!

So, the working week is almost done and it's time to relax. How can my little desktop cope with gaming? It's fair to assume I won't be playing Fortnite or F1 2019 at 60fps on Raspberry Pi any time soon, although the ubiquitous Minecraft works very well. Luckily for me, I enjoy retro gaming and Raspberry Pi is more than capable of running various emulators for 1980s and 1990s game consoles. RetroPie (retropie.org.uk) is a wonderful platform that brings multiple emulators together in a common interface, greatly simplifying installation and configuration (Note: At the time of writing, support was still being finalised for Raspberry Pi 4). If you're after something a bit more recent, there are many casual games out there, including ports of early PC classics like Doom. You can certainly be entertained by your Raspberry Pi desktop, but don't expect a VR headset experience.

How about movies, music, and podcasts instead? Raspberry Pi's potential as an affordable media streaming platform has been much vaunted. If you want a full media centre experience, you can choose between different implementations of the Kodi platform (kodi.tv), such as OSMC (osmc.tv), LibreELEC (libreelec.tv), and more. These typically come as full disk images

▼ Watching YouTube videos proved a bit hit and miss



▲ Plex is a good choice for streaming videos, music, and podcasts from your networked-attached storage

and are not intended to be run from the desktop. Although Kodi can be installed on the desktop just like digiKam, I struggled to get it to run. For these kinds of applications, I would seriously consider using multiple microSD cards. One of the joys of Raspberry Pi is you can easily swap out cards and, presto, your desktop machine is now an arcade or a media centre.

“ You can certainly be entertained by your Raspberry Pi desktop ”

Watching YouTube videos was hit and miss. Cinema mode worked well, but full-screen struggled and we suffered a few crashes. I had much better results with Plex streaming from my local server. A 720p video played flawlessly, although some tearing was in evidence. If you want to access other streaming services such as Netflix, Kodi, and friends are the way to go. Chromium does not support the DRM (digital rights management) required by Netflix.



Day 6 & 7 | Weekend Conclusions

There is no doubt, Raspberry Pi is now a capable desktop computer. No, you're not going to edit the next Pixar movie on it or explore the worlds of Elite Dangerous (but you could play the original!). A sense of perspective is what's needed. For the user who needs email, web access, and word processing or spreadsheet work, the price point is unbeatable. Even more advanced uses such as photo editing are certainly possible. If you're looking for a first computer for a young member of the family, then the price point (if they break it, it's not the end of the world), plus the possibilities afforded by the GPIO, make Raspberry Pi 4 well worth considering as their daily machine. **M**



LEARN | CODE | MAKE

OUT NOW IN PRINT
ONLY £3.99
from
store.rpipress.cc



From the makers of the official Raspberry Pi magazine

GET THEM DIGITALLY:



2nd
Edition

FULLY UPDATED FOR RASPBERRY PI 4

THE OFFICIAL
Raspberry Pi
Beginner's Guide
How to use your new computer

Fully
Updated for
**Raspberry
Pi 4**

The official Raspberry Pi Beginner's Guide

2nd Edition

by Gareth Halfacree



by Gareth Halfacree

THE OFFICIAL Raspberry Pi Beginner's Guide

**The only guide you need to
get started with Raspberry Pi**

Inside:

- Learn how to set up your Raspberry Pi, install an operating system, and start using it
- Follow step-by-step guides to code your own animations and games, using both the Scratch and Python languages
- Create amazing projects by connecting electronic components to Raspberry Pi's GPIO pins

Plus much, much more!

£10 with **FREE worldwide delivery**



Buy online: magpi.cc/BGbook

Getting started With Mathematica



MAKER

Ben Nuttall

Ben is the creator of GPIO Zero and piwheels, and is the Raspberry Pi Foundation's resident Python expert.

@ben_nuttall

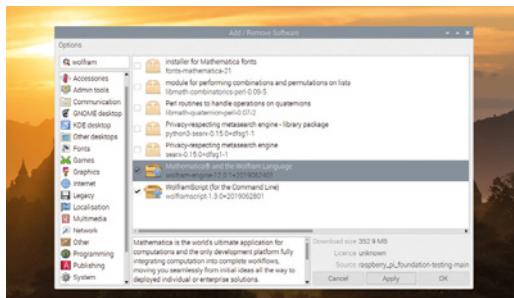
Mathematica is a computational programming tool used in science, maths, computing, and engineering. It is proprietary software that you can use for free on Raspberry Pi and comes bundled with Raspbian. Mathematica is generally used for coding projects at university level and above.

With this tutorial you will learn how to install Mathematica on Raspberry Pi, how to launch it and run commands, how to run scripts, create plots, and access the GPIO pins and the Camera Module.

01 Installing Mathematica

On the Raspbian desktop, click the top-left raspberry icon and choose Preferences > Add / Remove Software. Search for ‘wolfram’ and tick both ‘Mathematica and the Wolfram Language’ and ‘WolframScript (for the Command Line)’. Click Apply, enter your password, and click OK.

Close the Add / Remove Software window and choose Applications > Programming > Mathematica. You’ll see a splash screen with the Mathematica logo. Once it has loaded, you’ll see two windows. These are the Wolfram information



▲ Wolfram Mathematica is available for free on Raspberry Pi. Install it using the Add / Remove Software interface

You'll Need

- ▶ Raspberry Pi
- ▶ Raspbian OS
- ▶ Wolfram Mathematica

dialog and the Mathematica notebook. The link in the Wolfram information dialog will open in the web browser on the Raspberry Pi, provided you’re connected to the internet.

02 Programming in Mathematica

Click inside the Mathematica notebook window and enter:

```
Print["Hello world"]
```

Press **SHIFT+ENTER**; it will run the command and print ‘Hello world’ to the screen like this:

```
In[1]:= Print["Hello World"]
Hello World
```

You can perform mathematical calculations, for example (don’t enter the ‘In[n]:=’ part):

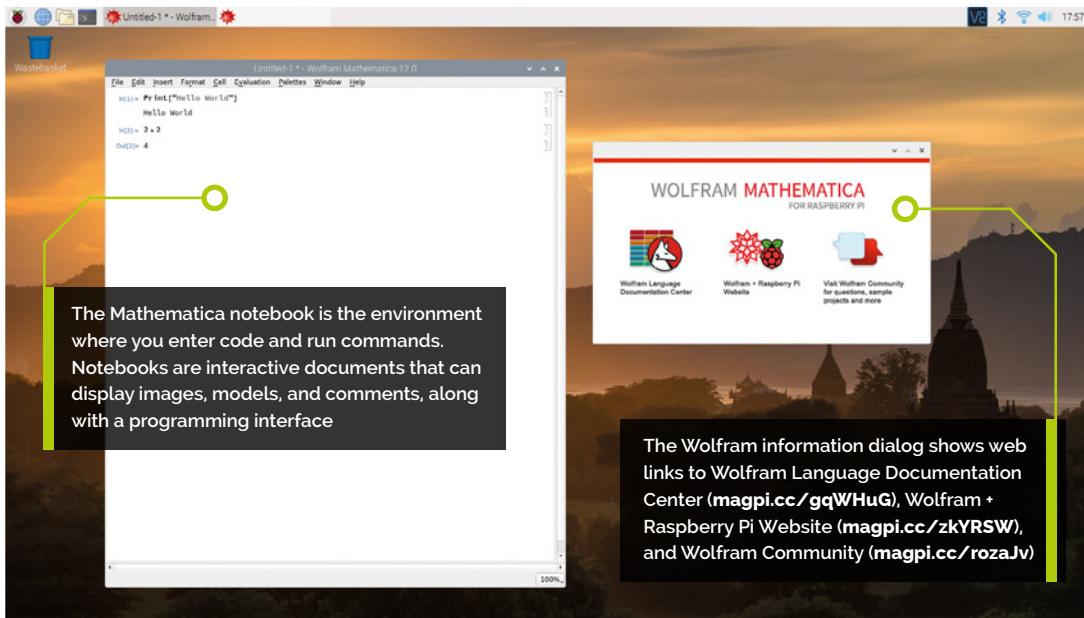
```
In[2]:= 2 + 2
Out[2]= 4

In[3]:= 16254 / 32
Out[3]= 8127 / 16

In[4]:= 1024 * 32
Out[4]= 32768
```

03 Notebook editing

You can revisit a previously entered command by clicking it or moving the edit cursor with the keyboard. You can then delete, edit, or add something and press **SHIFT+ENTER** to execute the new command in its place.



You can save a notebook and come back to it later, send it to a friend, post it online, or even hand it in as your homework! Just go to File > Save As in the notebook window.

When you open up a saved notebook, all the previous entries will be shown, including inputs and outputs. You can execute each cell again with **SHIFT+ENTER**, or all at once by selecting Evaluation > Evaluate Notebook from the menu.

04 Variables

You can store the results of calculations in variables:

```
radius = 5;
diameter = 2 * radius;
circumference = 2 * Pi * radius;
area = Pi * radius^2;
```

The semicolon at the end of each line suppresses the output being printed. Note the use of the built-in symbol **Pi** which contains a symbolic value of pi. This means that if you pass it into an equation the reference to the true value of pi is preserved, not converted to decimal and rounded:

```
In[19]:= Pi
Out[19]: π

In[20]:= tau = 2 * Pi
Out[20]: 2 π
```

To get the decimal representation of a symbolic value, use the **N** function:

■ You can revisit a previously entered command by clicking it or moving the edit cursor with the keyboard **■**

```
In[5]:= N[Pi]
Out[5]: 3.14159
```

The default number of significant figures given is six, but more can be given by specifying the number in the second argument:

```
In[6]:= N[Pi, 10]
Out[6]: 3.141592654
```

Note that this is the number of figures, not decimal places; so the 3 is included in the count, leaving nine decimal places.

05 Lists and range

You can store collections of data in a list:

```
nums = {1, 2, 3, 5, 8}
people = {"Alice", "Bob", "Charlotte", "David"}
```

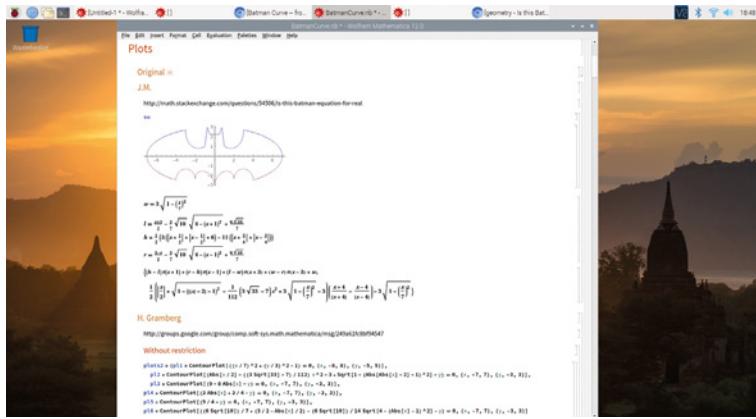
The **Range** function can be used to produce a list of numbers:

```
Range[5] (*The numbers 1 to 5*)
Range[2, 5] (*The numbers 2 to 5*)
Range[2, 5, 2] (*The numbers 2 to 5, in steps
of 2*)
```

Top Tip

Command line

You can also access the Wolfram Language from the command line by entering **wolfram** in the Terminal, or double-clicking the Wolfram Desktop icon. You'll find the command-line interface faster, although it lacks the graphical interface's interactivity and pretty printing.



▲ A Mathematica notebook displaying the 'Batman' equation. It displays the graph, along with the mathematical notation and the Wolfram Language code used to perform the calculation.

The **Table** function is a method of generating the values of a list with a function:

```
Table[i ^ 2, {i, 10}] (*Squares of the numbers
1 to 10*)
Table[i ^ 2, {i, 5, 10}] (*Squares of the
numbers 5 to 10*)
Table[i ^ 2, {i, nums}] (*Squares of the items
in the list nums*)
```

7 You can access the GPIO pins from Mathematica using the DeviceWrite and DeviceRead functions **7**

You can run a loop a number of times, or over the items in a list, with **Do**:

```
Do[Print["Hello"], {10}] (*Print "Hello" 10
times*)
Do[Print[i], {i, 5}] (*Print the numbers 1
to 5*)
Do[Print[i], {i, 3, 5}] (*Print the numbers
3 to 5*)
Do[Print[i], {i, 1, 5, 2}] (*Print the
numbers 1 to 5, in steps of 2*)
Do[Print[i ^ 2], {i, nums}] (*Print the
square of each item in the list nums*)
```

06 Function help

You can get usage help for a function by preceding the function name with a question mark **?** and pressing **SHIFT+ENTER**:

?Sqrt

You can also search for functions by entering part of the function name to find matches. Just start with a **?** and add an asterisk ***** on the end as a wildcard:

?Device*

You can use multiple wildcards:

?*Close*

07 Running scripts with Wolfram

You can write a program, save it as a normal file (usually with a **.m** or **.wl** file extension), and execute the script from the command line by adding the **-script** flag.

To run **test.m**:

```
wolfram -script test.m
```

08 List operations

You can apply an operation or function to all items in a list:

```
In[21]:= 2 * {1, 2, 3, 4, 5}
Out[21]: {2, 4, 6, 8, 10}

In[22]:= {1, 2, 3, 4, 5} ^ 2
Out[22]: {1, 4, 9, 16, 25}

In[23]:= Sqrt[{1, 2, 3, 4, 5}]
Out[23]: {1, Sqrt[2], Sqrt[3], 2, Sqrt[5]}}
```

In the last example, the square roots of 1 and 4 were given exactly, as they yield an integer value; however, the square roots of 2, 3, and 5, which are irrational, are given symbolically.

09 Matrices

One of the most useful additional components of a mathematical programming language is the ability to do matrix (**magpi.cc/EWdZms**) operations. Of course, Mathematica has these available.

To create a matrix, first enter the values as a list of lists, making sure the dimensions are rectangular, i.e. $n \times m$ where n and m are integers:

```
m = {{1, 2}, {3, 4}, {5, 6}};
```

You can view this list as a matrix by typing:

```
m // MatrixForm
```

You can perform matrix operations, such as dot product ([magpi.cc/uXkcPX](#)):

```
m = {{1, 2}, {3, 4}, {5, 6}};
m2 = {{10, 20, 30}, {40, 50, 60}};
m . m2 // MatrixForm
```

10 Plotting

You can plot interesting things using Mathematica. Plot an echidnahedron ([magpi.cc/oYemXi](#)) with the following command:

```
Graphics3D[{Opacity[.8],
Glow[RGBColor[1,0,0]],
EdgeForm[White], Lighting -> None,
PolyhedronData["Ecidnahedron", "Faces"]}]
```

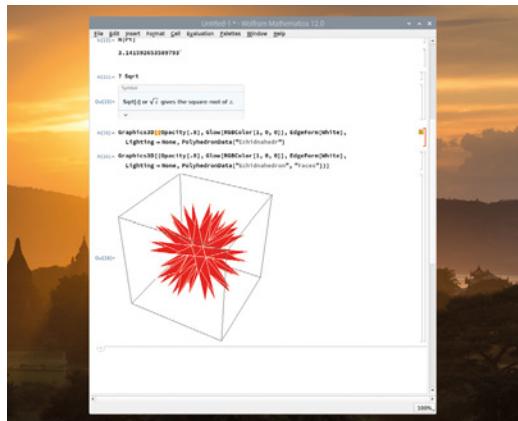
11 GPIO

You can access Raspberry Pi's GPIO pins from Mathematica using the `DeviceWrite` and `DeviceRead` functions.

The following command turns on GPIO pin 14 (using BCM pin numbering):

```
DeviceWrite["GPIO", 14 -> 1]
```

The following command turns pin 14 off:



▲ You can quickly plot interesting mathematical shapes like this echidnahedron ([magpi.cc/oYemXi](#))

```
DeviceWrite["GPIO", 14 -> 0]
```

You can also read the status of a GPIO input device (to check if a button is pressed, for example) with `DeviceRead`, in a similar way:

```
button = DeviceRead["GPIO", 14]
```

The variable `button` should now contain 0 for off or 1 for on. Read more about GPIO in general on the GPIO usage page ([magpi.cc/HVNsAN](#)).

12 Camera

You can take pictures with the Raspberry Pi Camera Module using the `DeviceRead` function. First, attach your camera as per the setup guide ([magpi.cc/NPxFqo](#)). To take a still picture with the camera, type the following command:

```
img = DeviceRead["Raspicam"]
```

Then, to save the image as a file, use `Export` and supply the save path and the variable containing the image:

```
Export["/home/pi/img.jpg", img]
```

Finally, for a bit of fun, run the **BatmanCurve.nb** notebook below in Mathematica.

BatmanCurve.nb

DOWNLOAD THE FULL CODE:

► Language: **Wolfram Language**

magpi.cc/EyYHtr

```
001. Plot[{With[{w = 3*Sqrt[1 - (x/7)^2]},
002.   l = (6/7)*Sqrt[10] + (3 + x)/2 - (3/7)*Sqrt[10]*Sqrt[4 - (x + 1)^2],
003.   h = (1/2)*(3*(Abs[x - 1/2] + Abs[x + 1/2] + 6) - 11*(Abs[x - 3/4] + Abs[x + 3/4])),
004.   r = (6/7)*Sqrt[10] + (3 - x)/2 - (3/7)*Sqrt[10]*Sqrt[4 - (x - 1)^2}],
005.   w + (1 - w)*UnitStep[x + 3] + (h - 1)*UnitStep[x + 1] + (r - h)*UnitStep[x - 1] + (w - r)*UnitStep[x - 3]], (1/2)*(3*Sqrt[1 - (x/7)^2] + Sqrt[1 - (Abs[Abs[x] - 2] - 1)^2] + Abs[x/2] - ((3*Sqrt[33] - 7)/112)*x^2 - 3*((x + 4)/Abs[x + 4] - (x - 4)/Abs[x - 4]) - 3*Sqrt[1 - (x/7)^2]}, {x, -7, 7}, AspectRatio -> Automatic]
```

Part 02

Build a low-cost wheeled robot



**Danny
Staple**

MAKER

Danny makes robots with his kids as Orionrobots on YouTube, and is the author of *Learn Robotics Programming*.

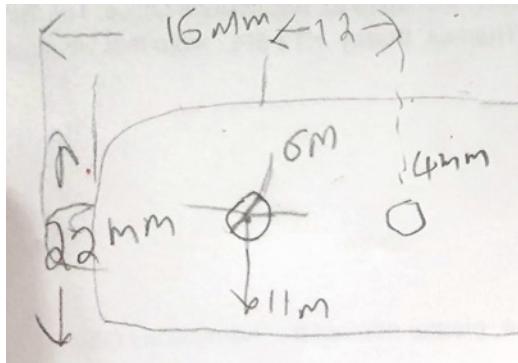
orionrobots.co.uk

You'll Need

- ▶ Plastic lunchbox – minimum 90 mm wide x 120 mm long x 60 mm high
- ▶ 2 x Gear motors with wheels magpi.cc/nhoAbj
- ▶ Battery box – 6 x AA magpi.cc/DPFaEJ
- ▶ 5V 3A UBEC magpi.cc/hZohEg
- ▶ Ball castor magpi.cc/bfALSE
- ▶ L298N module magpi.cc/EDKvrL
- ▶ Nylon screw and standoff kit magpi.cc/XaRfgp

01 Getting familiar with components

Examine the motors. Sticking out of the sides are the axles. On one side, close to the axle, is a small knob which can help lock the motor into position. The top and bottom of the motor is flat. The



▲ Engineering sketches are essential in robotics. The sketch can be rough with dimensions and notes. Sketch lots! This is the side with motors.

motors should have connection wires on them, on the opposite side to the knob.

The motor controller has a large metal heat-sink sticking up above the board.

Raspberry Pi GPIO pins should be present and go in face up so they can be used.

A lunchbox is made of thin rigid plastic. It probably has rounded corners. Take measurements using the flat edges.

02 Planning and test fitting

Test-fit the parts to check the lunchbox is big enough. Make a simple plan for where parts could go.

Put the motors in the box, as shown in **Figure 1**, with their axles to the rear. Fit your Raspberry Pi Zero in the front space and the motor controller between the motors. The UBEC (which is a DC voltage regulator) sits loosely in the box.

The battery box goes in the top – inside if it can; outside if it interferes with the heat-sink.

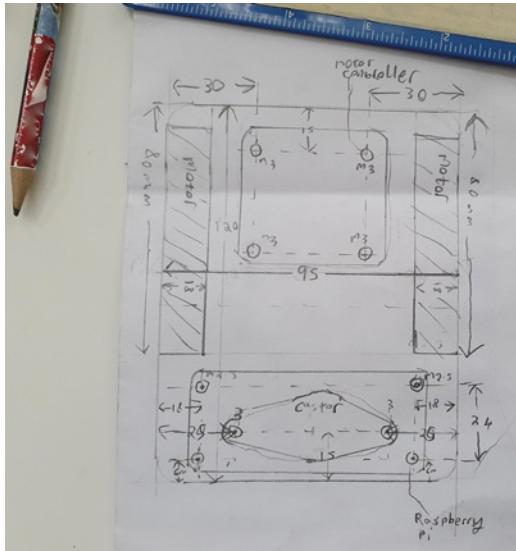
The wheels would go on the motors and the castor at the front under Raspberry Pi. Take a photo and remove it all.

03 Sketches for the motors and castor

Sketches enable you to place holes and components before making any cuts. See orthographic drawing tips at magpi.cc/tZKdQV.

Find or start a sketch of the motors. Write in dimensions for their size, axle diameter, and location, and the positioning knob on the side.

Sketch a side of the lunchbox, with holes to put motors flush with the inside bottom of the



▲ This sketch is of the bottom of a lunchbox with holes and dimensions. Yours will be slightly different. It's on the back of an envelope

lunchbox. Include dimensions to show relative positions of holes for the axle and knob.

Sketch the bottom of the lunchbox and measure the two mounting holes for the castor to go in the front middle.

You should have two lunchbox sketches with carefully checked dimensions.

■ Sketches enable you to place holes and components before making any cuts ■

04 Measuring for the electronics

The test-fit photo (**Figure 2**) is a rough guide on positioning things in the box, but sketches with dimensions have detail on where components should go.

Raspberry Pi has drawings with dimensions at magpi.cc/HdffIN. Add mounting holes for it to your lunchbox bottom sketch near the front, with dimensions for diameter and position.

Measure then sketch the motor controller's position and holes for the controller on the lunchbox bottom sketch.

Carefully check the bottom sketch. It should have holes for the castor and both electronic boards.

Sketch the top of the lunchbox, where the AA battery box goes.

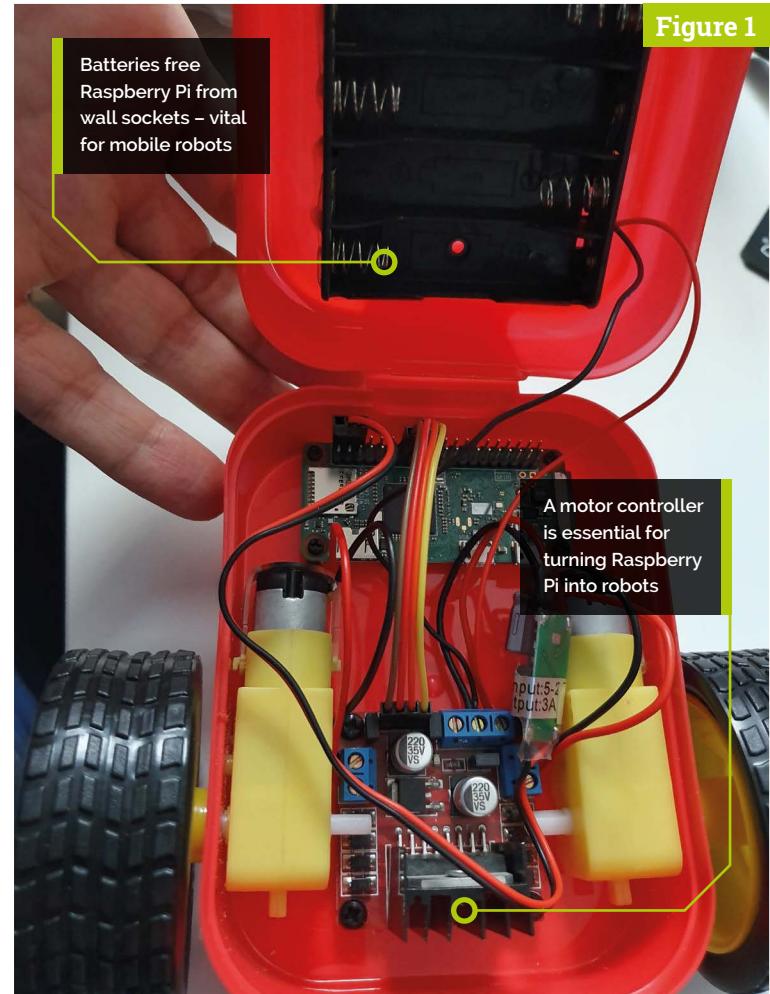


Figure 1



▲ **Figure 2** A fundamental step in robot creation is test-fitting where items might go! Use the real components or drawings of them

Top Tip



Use a vice/clamp

For your safety and a good finish, never be tempted to skip using a vice or clamp for drilling, soldering, or cutting.

04 Don't tighten nuts past finger-tight, as this can damage the bolts or the parts held by them

05 Drill the holes

Double-check sketch measurements. Use a fine marker pen to measure and make crosses as hole guides on the lunchbox.

Clamp the box firmly with a vice, ensuring the surface to drill is facing upwards. *Do not* try to hold the box with hands when drilling it!

When making holes, the part may flex, crack, or spin. The drill bit can ‘wander’ from its target. To reduce this, start with the smallest bit to make a pilot hole. Follow through with progressively wider drill bits to the desired diameters. You’ll need to reposition the lunchbox to drill different areas.



▲ Next to the axle, a motor has a knob (highlighted) to help align it. Include these in sketches as these will need holes too

It's a good idea to tug gently on the wires to check they are screwed into the terminal securely.

06 Fit the motors and castor

Line up the motor axles and knobs with their holes, then push the axles through. If any holes feel tight, drill them out a little more. Make a hot-glue line between each motor and the lunchbox to hold it, being careful to avoid getting glue on the axle.

Bolt the castor in place using M3 nylon bolts and nuts with the threads facing outward. Start with two opposite corners if it has four bolts. These bolts go under Raspberry Pi; to avoid short circuits, they must not be metal bolts.

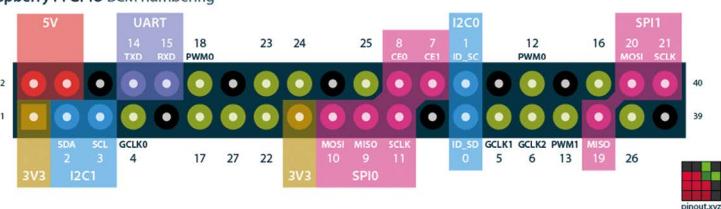
07 Wiring the motors

The motor board may be harder to access when bolted down, so it makes sense to make the motor connections at the sides first.

Start by loosening, but *not* wholly unscrewing the connection terminals. Look at the circuit. Push the wires for each motor into the two terminals for that side, and then gently but firmly screw down the terminal so it holds the wire.

▼ The website pinout.xyz is an excellent reference for Raspberry Pi GPIO pins. Use this to help when wiring the robot. Black pins are ground

Raspberry Pi GPIO BCM numbering



08 Fit Raspberry Pi and motor controller

Line up the motor controller with its holes. It may need to slot under the motor axles. Push M2.5 bolts through two opposite corners to line it up, and loosely put nuts on these. Put in the other two corners and tighten them all up.

Don't tighten nuts past finger-tight, as this can damage the bolts or the parts held by them.

For Raspberry Pi, put M2.5 standoffs thread first into the four holes and tighten nuts to them. Place your Raspberry Pi Zero W over the standoffs. Use the opposite corners method again. Leave these quite loose so your Raspberry Pi can be taken out.

09 Fitting the battery box

Using the sketch made of the top of the lunchbox, line up the battery box – on the inside if there is space; on the outside if it would stop the lunchbox closing. An extra hole is needed for cables if it's outside.

Screws could be used, but flat countersunk types are needed, making hot glue an easier option.

Make lines of hot-melt glue between the battery box and lunchbox around the sides of the battery box. There should be no glue inside the battery box, and it should be held firmly once the glue has set.

10 Wiring the batteries

The battery box connections and UBEC are wired in together. The UBEC has two sets of

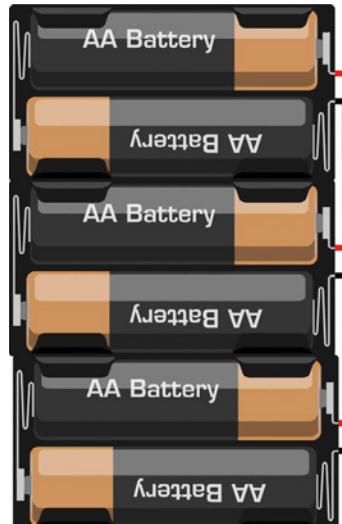
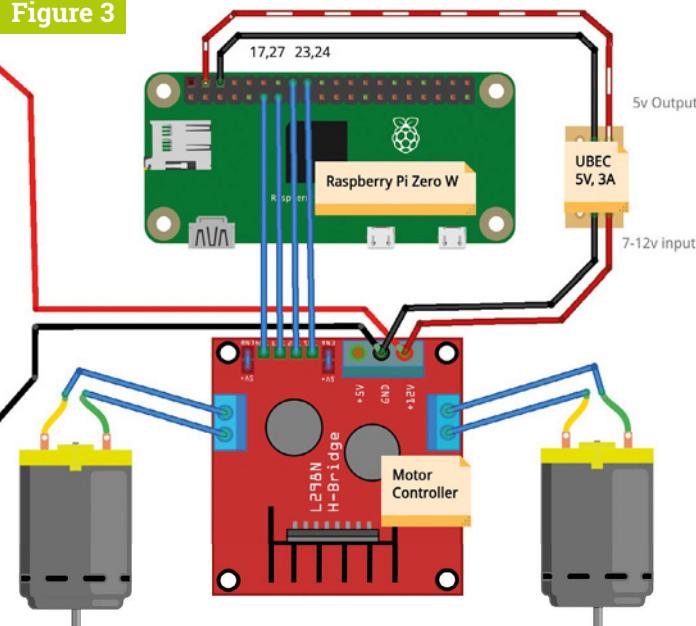


Figure 3



▲ **Figure 3** This robot has motors, batteries, Raspberry Pi, UBEC, and motor controller. Plenty of GPIO pins left for sensors, too.

connections. The input end has thicker wires with stripped ends, and the output connector has a three-pin push-fit connector.

Wire the battery positive (red) and UBEC input positive (red) together into the motor controller +12V or Vin terminal. It helps to twist the two bare ends together before pushing into the terminal and screwing down.

Wire the battery negative (black) and UBEC input negative (black) together into the motor controller GND terminal.

If you want to add a switch, put it between the batteries and the UBEC / motor controller.

11 Wiring Raspberry Pi

Use the circuit diagram (**Figure 3**) with Raspberry Pi GPIO pinout as guidance for this step. Physical pin 1 is closest to the microSD slot on Raspberry Pi Zero W boards.

The UBEC three-pin push-fit connects to GPIO so that the empty slot and red wire go to the 5V pins, with the black wire going to the ground pin.

Use female-to-female jumper wires to connect the Raspberry Pi to the motor controller pins. Connect Raspberry Pi GPIO17 and 27 pins to motor controller IN4 and IN3 pins.

Connect Raspberry Pi GPIO23 and 24 pins to motor controller IN2 and IN1 pins.

12 Finishing the robot

To finish assembly of the robot, push the wheels onto the axles.

Line up the wheels so the flat parts of the axles line up with the flat edges in the axle holes. They can require quite a bit of force, so push on the motor and the wheel to deliver it. Do not try to push the wheels on without supporting the motor.

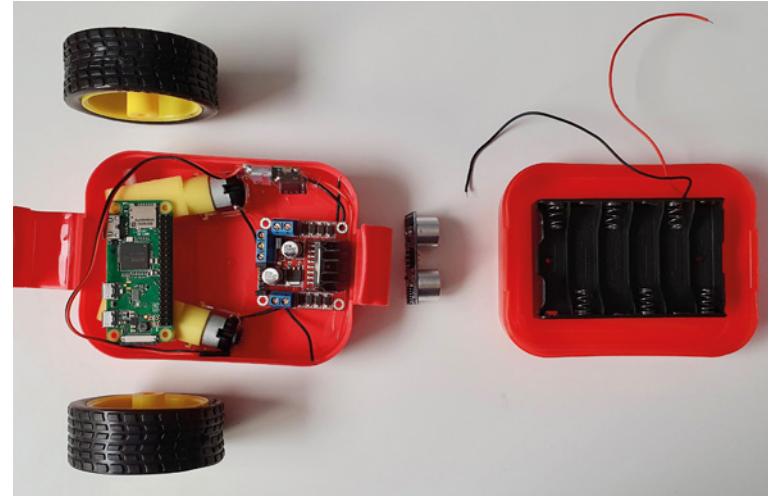
You can now also place the lid on top of the robot. If the lunchbox isn't quite tall enough, the lid might rest on top – do not force it down.

This robot is built, wired, and waiting for code!

Top Tip

Part sketches on the internet

Searching the web for part dimension pictures provides useful sketching start points. Print them and draw more on them, or use them for reference when making other sketches.



Part 03

The Squeeze: Rocket Rescue



**Mike
Cook**

Veteran magazine author from the old days, writer of the Body Build series, plus co-author of *Raspberry Pi for Dummies*, *Raspberry Pi Projects*, and *Raspberry Pi Projects for Dummies*.

magpi.cc/TPaUff



Warning!
High voltage

The dynamo torches in this project can produce high voltage, so be careful.

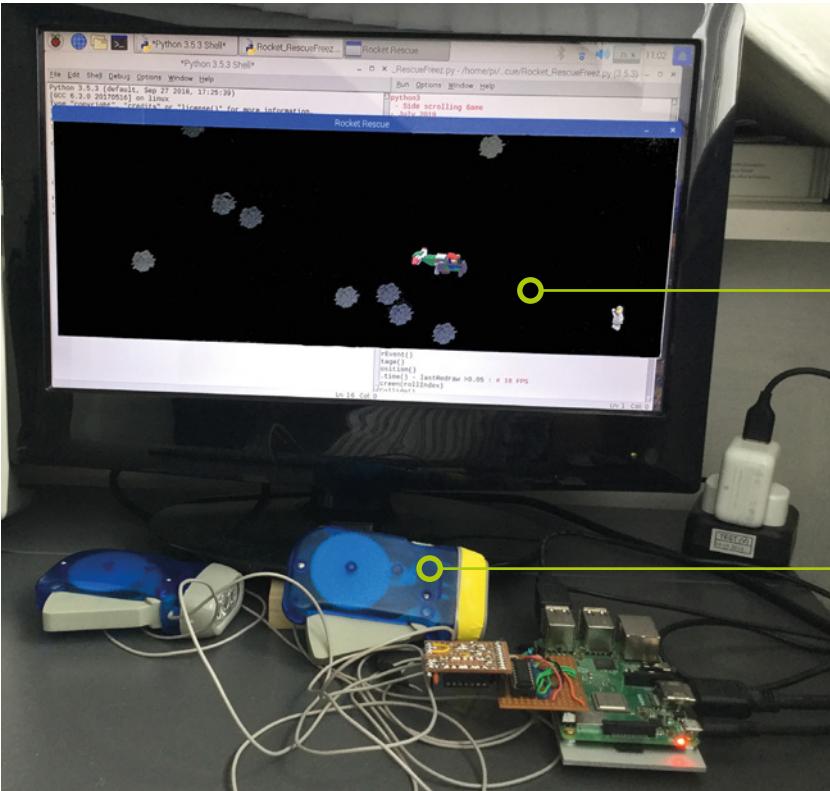
You'll Need

- ▶ A/D converter from *The MagPi* #68
magpi.cc/68
- ▶ Squeeze controller from *The MagPi* #83
magpi.cc/83

In the last of these three articles looking at using a dynamo torch as a computer interface, we see how we can write a classic 8-bit-like game, 'Rocket Rescue'. In it, you have to dodge the moving asteroid field to rescue a stranded astronaut. Once you have them, you must get them back to the other end of the screen. Control of the spacecraft's movement is done with the two squeeze dynamo torches, using sound from an actual NASA Apollo mission. It is not as easy as you think, but with a bit of practice it can be done.

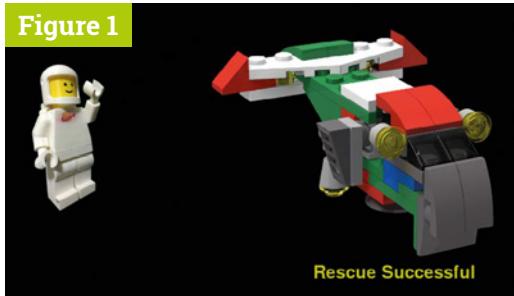
01 Movement

This game uses the two dynamo torches to control the horizontal and vertical speed of a spacecraft. As each controller can only output one voltage, we have to use that voltage to indicate both positive and negative movement. To do this, we have the concept of drift; this is the speed of movement that will be made if the controllers are outputting a zero voltage. So, to maintain a fixed position, the controller needs to be pumped to a mid-point level. Anything else will result in a positive or negative movement.



Spaceship about to rescue the stranded astronaut

Squeeze controllers control spaceship movement



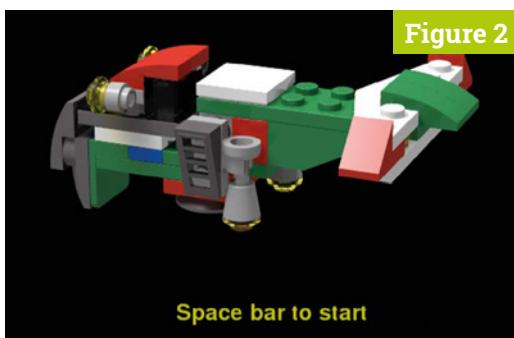
▲ Figure 1 The astronaut has been successfully rescued

02 The hazards

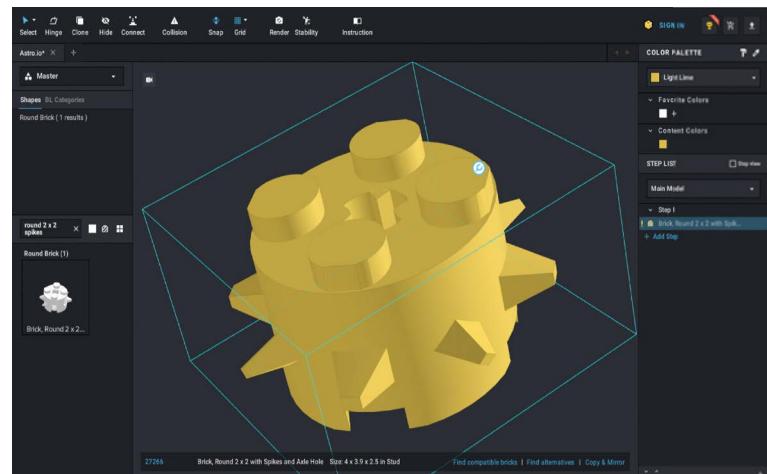
The hazards are the asteroids drifting around the screen. If your ship crashes into one, it will be destroyed and your mission fails. Each one is assigned a random velocity, either positive or negative in each direction. This means they can drift in all directions. Once they drift off the screen completely, they will appear on the opposite side. You can change the range of speed and the number of asteroids with a simple edit of the code, but the defaults will give you a good challenge. You could even make them static for younger players.

03 Space rescue

The astronaut is at the lower right-hand corner of the screen; they do not move. Touching them with the front of the spaceship will cause docking and the astronaut will be transferred into the ship. The ship will then turn round, and you must guide it back to the left-hand end of the screen. That constitutes a successful mission, and there will be a screen showing if the mission was successful or not, followed by the animated splash screen asking you to press the **SPACE** bar starting off a new run, see **Figures 1** and **2**.



▲ Figure 2 A still from the splash screen, waiting to get the go-ahead for a new mission



▲ The asteroid brick before the speckled black silver colour is applied

04 Graphics

Getting good-quality graphics is essential for a polished-looking game. Here we used the Lego virtual construction kit, Studio, to create images of the spacecraft and asteroids. We first looked at this in *Lego Boost* articles in the *The MagPi* #48 to #50. You can get the .io model files needed in the GitHub repo (magpi.cc/dhaAam). The asteroids are a single brick coloured black with speckles; we rotated it to an interesting position and then rendered it, before reducing it to a 50x44 pixel image.

" The key part of any game like this is the ability to detect a collision between two objects **"**

05 The playing spaceship

The spaceship was rendered as a sequence of 640x420 pixel images with a rotation of 20° per frame, and is used for the splash screen. In order to prevent the playing spaceship looking a bit two-dimensional, we rendered a sequence of twelve images where the roll of the model changed by 5° at a time. This changed the perspective a bit, and the appropriate image was chosen for display depending on how far up the screen it was. The ship's destruction was also rendered as a sequence of 14 images, with the Lego parts flying apart and then disappearing.

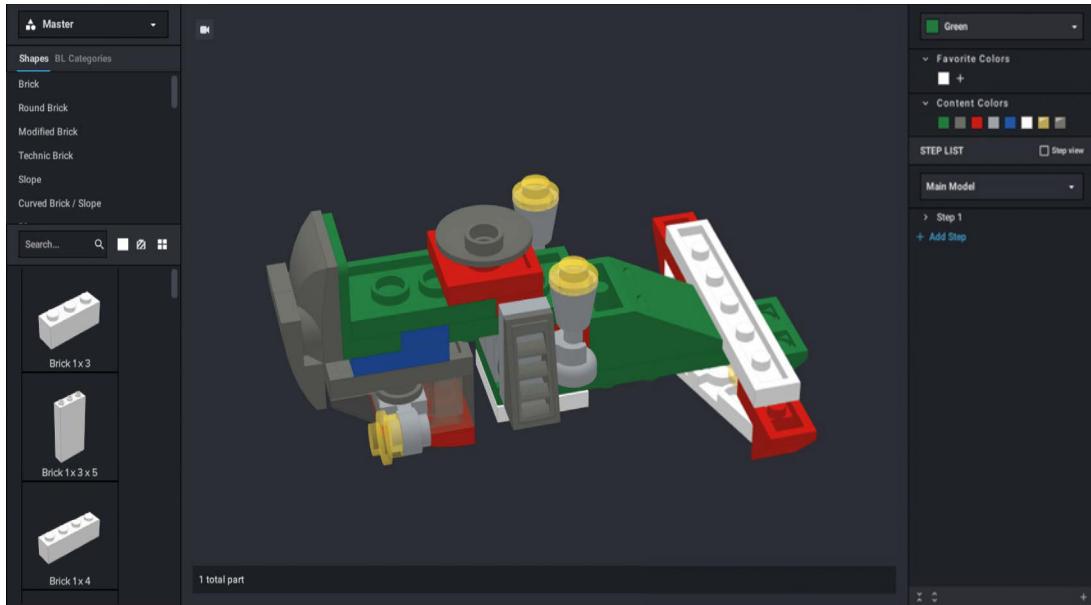
06 Detecting a collision

The key part of any game like this is the ability to detect a collision between two objects. Fortunately, Pygame makes it simple to do by detecting overlapping rectangles. However, while the images do have a bounding rectangle, they are on a transparent background and cover an

Top Tip

Playing

Be consistent about which hand controls vertical and horizontal movement.



► Building the spaceship in Studio

Top Tip



Hardware

Never connect things to the GPIO pins when the Raspberry Pi is powered up.

area bigger than the visible graphic – so if you simply used that, collisions would be detected when objects passed close to each other but didn't touch. The solution is to have a collision rectangle that encloses, as closely as possible, just the image. **Figure 3** shows a greatly enlarged image of an asteroid with its collision rectangle.

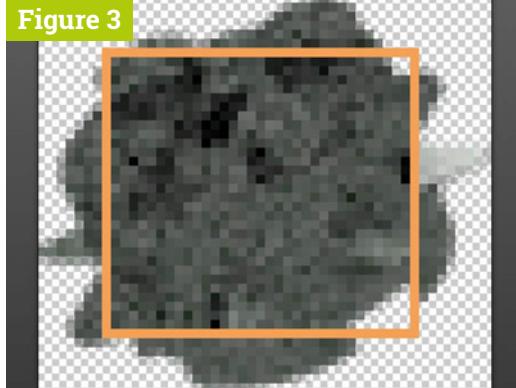
07 Collision rectangle

The spaceship's collision rectangle is a bit more complex because of the roll applied to it. It consists of two rectangles, one unchanging covering the front section, and a rear one that changes size according to the degree of roll, because you get to see more or less of the tail fin. For those wondering why a spaceship has a tail fin, it is for manoeuvring in the atmosphere when it lands back on Earth. This means we don't just need one collision rectangle, but a list of twelve. **Figure 4** (overleaf) shows the game being played with the collision rectangles being displayed, for testing purposes.

08 The sound

'In space no one can hear you scream,' went the old movie poster, but this poses a bit of a problem if you want to make the physics accurate: no explosion sound. So, we downloaded the NASA sound files of the highlights of the Apollo 9 mission. The phrase 'what are you doing throwing everything overboard?' might be what they would say at mission control if they were looking at the

Figure 3



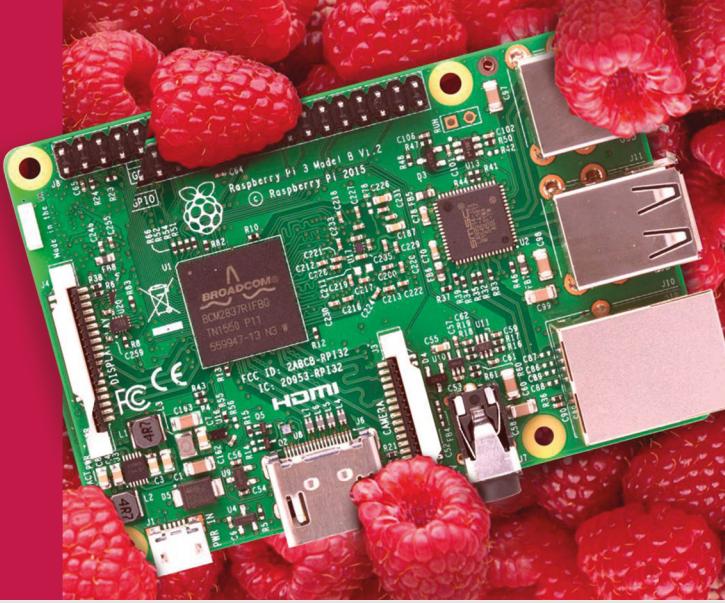
▲ **Figure 3** A highly enlarged view of the asteroid graphic showing the collision rectangle used

spacecraft on radar and saw lots of fragments. Similarly, 'understand hard dock' is used for picking up the astronaut, and 'mission confirmed' is used as the message about a successful outcome.

09 The program

The code is shown in the `Rocket_Rescue.py` listing; that, along with the graphics and sound files, can be found on our GitHub page. The main function controls the overall logic flow of the game. Note that the program is constantly reading the controllers and calculating the position of the various objects. But these are only displayed every 0.05 seconds, giving a frame rate of about 18 per second. At first, we had the screen updated every time something was moved by a pixel, but this caused really sluggish movement as there were more screen refresh calls than was needed.

American Raspberry Pi Shop



- Displays
- Cases
- Project Kits
- Add-on Boards
- HATs
- Arcade
- Cameras
- Cables and Connectors
- Sensors
- Swag
- Power Options
- GPIO and Prototyping

Partner and official reseller for top Pi brands:



and many others!

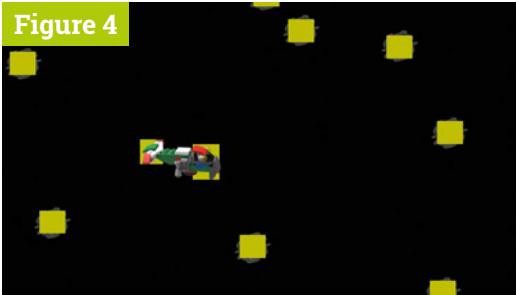
Price, service, design,
and logistics support for
VOLUME PROJECTS



USA **PiShop.us**

Canada **BuyaPi.ca**



Figure 4

▲ Figure 4 Playing the game during testing, with the collision rectangles showing in yellow

10 Customising

By uncommenting line 56, you can get the asteroids to remain stationary; this is ideal for very young players or to develop your control skills. Similarly, line 88 sets the number of asteroids you have to cope with. If you want to see the collision rectangles while you play, then uncomment lines 162, 163, and 170. The first list in line 17 controls how sensitive each controller is; make the numbers smaller to get a bigger reading from each one. This, combined with the drift variable in line 70, controls the overall movement or speed of the game.

11 Alternative controllers

Prolonged use of these controllers can lead to muscle pain, at least from the old decrepit folks here at the Bakery. So, during the development of the game, we used the four-potentiometer box we made for the Drum Sequencer project in *The MagPi* #68. This made it easier to play as all you had to do was to set the position of two of the four potentiometers to get a constant velocity in the required direction. However, this was not as much fun as using the dynamo torches.

In conclusion

There are lots of different things you can do to customise this game. You can record the time of each rescue and show a top ten list of the times. You could have a number of static asteroids in a preset pattern, or even many preset patterns that change with the game. You could increment the number of asteroids after each successful run and return them to a starting number when you failed. In this way you get a ‘level’ number that you could compete to see who can get the highest. Whatever you do, do have fun. ■

Rocket_Rescue.py

► Language: Python

```

001. #!/usr/bin/env python3
002. # Rocket Rescue - Side scrolling Game
003. # By Mike Cook - July 2019
004.
005. import os, pygame, sys, csv, time, spidev, random
006.
007. pygame.init()      # initialise graphics interface
008. pygame.mixer.quit()
009. pygame.mixer.init(frequency=22050, size=-16, channels=2,
010. buffer=512)
011. os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'
012. pygame.display.set_caption("Rocket Rescue")
013. pygame.event.set_allowed(None)
014. screen = pygame.display.set_mode([1200,420],0,32)
015. textHeight= 36;font =pygame.font.Font(None, textHeight)
016. backCol = (0,0,0) ; start = False ; random.seed()
017. scale = [4600, 4600] ; nAv = 10 # samples to average
018. avPoint = [0.0,0.0] ; average = [0.0,0.0]
019. speed =[0.0,0.0]; rollIndex=0; rPhase = 0 #rescue phase
020. p1 = [0] * nAv ; p2 = [0] * nAv ; runningAv = [p1,p2]
021.
022. def main():
023.     global start
024.     init() ; setUpPrams() # set up variables
025.     while True:
026.         time.sleep(1.5)
027.         intro.play() # intro music
028.         while not start: # until space bar is pressed
029.             splashScreen()
030.             checkForEvent()
031.         start = False ; intro.play() # intro music
032.         lastRedraw = time.time()
033.         while not done : # for the duration of the mission
034.             checkForEvent()
035.             readVoltage()
036.             updatePosition()
037.             if time.time() - lastRedraw >0.05 : # 18 FPS
038.                 drawScreen(rollIndex)
039.                 checkCollide()
040.                 lastRedraw = time.time()
041.             missionResult() ; time.sleep(2)
042.             setUpPrams() # set up next run
043.
044. def setUpPrams():
045.     global manRect,astLst,place,astroSpeed,done
046.     global colRect,rPhase,mission,astroPosX,astroPosY
047.     done = False ; rPhase = 0 ; mission = False
048.     manRect.x = 1100 ; manRect.y = 320
049.     place[0] = 0 ; place[1] = 0
050.     for i in range (0,nA):
051.         astLst[i].x = random.randint(160,890)
052.         astLst[i].y = random.randint(150,400)
053.         astroPosX[i] = float(astLst[i].x)
054.         astroPosY[i] = float(astLst[i].y)
055.         astroSpeed[i] = ((0.5-random.random())/100.0,
```

**DOWNLOAD
THE FULL CODE:**



magpi.cc/dhaAam

```

((0.5-random.random())/100.0))
056.     #astroSpeed[i] = (0.0,0.0) # for no movement
057.     colRect[i].x = astLst[i].x +7 # collision Rect
058.     colRect[i].y = astLst[i].y +6
059.
060. def init():
061.     global spi,ship,place,lastPos,minLim,maxLim,drift
062.     global shipRectF,sRecRc,astroPosX,astroPosY
063.     global rectC,astro,man,manRect,astLst,astroSpeed
064.     global nA,breakUpR,breakUpRect
065.     global colRect,shipR,splash,sNum,manSplash,breakUp
066.     global cheers,confirmed,dock,intro,bell,crash
067.     sNum = 0 ; place = [400.0,230.0] # X/Y position
068.     lastPos = [-20,-20] ; minLim = [-10,-10] # X/Y
069.     maxLim = [1080, 352] # limits on ship placement
070.     drift = [ -0.05, 0.05 ] # no input movement
071.     cheers = pygame.mixer.Sound("sounds/end.ogg")
072.     confirmed = pygame.mixer.Sound("sounds/done.ogg")
073.     dock = pygame.mixer.Sound("sounds/hardDock.ogg")
074.     intro = pygame.mixer.Sound("sounds/Intro.ogg")
075.     bell = pygame.mixer.Sound("sounds/BellToll.wav")
076.     crash = pygame.mixer.Sound("sounds/overboard.ogg")
077.     spi = spidev.SpiDev() ; spi.open(0,0)
078.     spi.max_speed_hz=1000000
079.     splash = [pygame.image.load("images/Splash_"++
str(i)+"_.png").convert_alpha() for i in range(0,18) ]
080.     manSplash = pygame.image.load(
"images/manSplash.png").convert_alpha()
081.     shipRectF = pygame.Rect(82,18,37,48)
082.     ship = [pygame.image.load("images/SpaceCargo_"++
str(i)+"_.png").convert_alpha() for i in range(0,12) ]
083.     shipR = [pygame.transform.flip(
ship[i],True,False) for i in range(0,12) ]
084.     breakUp = [pygame.image.load("images/Cargo_"++
str(i)+"_.png").convert_alpha() for i in range(0,14) ]
085.     breakUpR = [pygame.transform.flip(
breakUp[i],True,False) for i in range(0,14)]
086.     breakUpRect = breakUpR[0].get_rect()
087.     astro = pygame.image.load(
"images/Astro.png").convert_alpha()
088.     nA = 10 # number of asteroids
089.     astLst = [astro.get_rect() for i in range(nA)]
090.     colRect =
astro.get_rect().inflate(-14,-12) for i in range(nA)]
091.     astroPosX = [0.0] * nA ; astroPosY = [0.0] * nA
092.     astroSpeed = [(0.005,0.005)] * nA
093.     man = pygame.image.load(
"images/Sman.png").convert_alpha()
094.     manRect = man.get_rect()
095.     rectC =[ (10,4,30,53),(11,6,31,45),(9,7,34,42),
(9,11,32,34),(9,11,32,30),(11,13,30,26),
(12,14,30,21),(12,17,31,17),(14,15,23,18),
(14,13,25,25),(15,10,17,29),(15,9,20,32) ]
096.
097.
098.
099.     sRecRc=[
pygame.Rect(rectC[i]) for i in range(0,12)]
100.
101. def getRoll(y):
102.     r = int(12*(290-y)/290)
103.     r = constrain(r,0,11)
104.     return r
105.
106. def constrain(val,min_val,max_val):
107.     return min(max_val,max(min_val,val))
108.
109. def checkCollide():
110.     global rPhase, done, mission
111.     if rPhase:
112.         if place[0] <= minLim[0]:
113.             done = True ; mission = True # mission
success
114.     else:
115.         if shipRectF.collidrect(manRect):
116.             dock.play() ; rPhase = 1
117.         if shipRectF.collidelist(colRect) >
-1 or shipRectR.collidelist(colRect) > -1:
118.             crash.play() # sound for collision
119.             distractSeq() ; time.sleep(3.0)
120.             done = True ; mission = False # mission fail
121.
122. def splashScreen():
123.     global sNum
124.     pygame.draw.rect(screen,backCol,(0,0,1200,420),0)
125.     screen.blit(splash[sNum],(280,-40 ))
126.     drawWords("Space bar to start",513,369)
127.     pygame.display.update() ; sNum += 1
128.     if sNum >=18:
129.         sNum = 0
130.     time.sleep(0.1)
131.
132. def missionResult(): # display failed / success
133.     global sNum
134.     pygame.draw.rect(screen,backCol,(0,0,1200,420),0)
135.     screen.blit(splash[sNum],(280,-40 ))
136.     if mission: # mission True = success
137.         confirmed.play() # sound for success
138.         screen.blit(manSplash,(130,100 ))
139.         drawWords("Rescue Successful",513,369)
140.         pygame.display.update()
141.         time.sleep(1.0)
142.         cheers.play() # applause
143.     else:
144.         bell.play() # sound for success
145.         drawWords("Rescue Failed",513,369)
146.         pygame.display.update() ; time.sleep(2)
147.
148. def distractSeq():
149.     for j in range(0,14):
150.         pygame.draw.rect(
screen,backCol, (0,0,1200,420),0)
151.         for i in range(0,nA):
152.             screen.blit(astro,(astLst[i].x,astLst[i].y))
153.             if rPhase:

```

Rocket_Rescue.py (continued)

► Language: Python

```

154.     screen.blit(breakUpR[j],(place[0],place[1]))
155. else:
156.     screen.blit(breakUp[j],(place[0],place[1]))
157. pygame.display.update() ; time.sleep(0.1)
158.
159. def drawScreen(index):
160.     global shipRectF, shipRectR
161.     pygame.draw.rect(screen,backCol,(0,0,1200,420),0)
162. #pygame.draw.rect(screen,(192,192,0),shipRectF,0)
163. #pygame.draw.rect(screen,(192,192,0),shipRectR,0)
164. if rPhase:
165.     screen.blit(shipR[index],(place[0],place[1]) )
166. else:
167.     screen.blit(ship[index],(place[0],place[1]) )
168. for i in range(0,nA):
169.     screen.blit(astro,(astLst[i].x,astLst[i].y) )
170. #pygame.draw.rect(screen,(192,192,0),
171. colRect[i],0)
172. if not rPhase:
173.     screen.blit(man,(manRect.x,manRect.y))
174. pygame.display.update()
175. def drawWords(words,x,y) :
176.     textSurface = pygame.Surface((14,textHeight))
177.     textRect = textSurface.get_rect()
178.     textRect.left = x ; textRect.top = y
179.     pygame.draw.rect(screen,(102,204,255) ,(
180.         x,y,14,textHeight-10),0)
181.     textSurface = font.render(words, True, (
192,192,0),backCol)
182.     screen.blit(textSurface, textRect)
183. def updatePosition():
184.     global place, update, lastPos, speed, astroPosX
185.     global astroPosY,colRect,rollIndex,rPhase,shipRectR
186.     for i in range(0,2):
187.         if i==0:
188.             place[i] += speed[i] + drift[i]
189.         else:
190.             place[i] -= speed[i] - drift[i]
191.         place[i] = constrain(place[i],minLim[i],maxLim[i])
192.         if lastPos[i] != int(place[i]):
193.             update = True ; lastPos[i] = int(place[i])
194.             rollIndex = getRoll(place[1])
195.         if update:
196.             shipRectR=sRecRc[rollIndex]
197.             if rPhase: # rescue phase 0 = inbound / 1 = back
198.                 shipRectF.x = place[0]+12;
199.                 shipRectF.y=place[1]+18
200.                 shipRectR.x = place[0] + 118 -
201. rectC[rollIndex][2]
202.                 shipRectR.y = place[1] + rectC[rollIndex][1]
203.             else:
204.                 shipRectF.x = place[0]+82;
205.                 shipRectF.y=place[1]+18
206.                 shipRectR.x = place[0] + rectC[rollIndex][0]
207.                 shipRectR.y = place[1] + rectC[rollIndex][1]
208. for i in range(0,2):
209.     for i in range(0,nA):
210.         xp = astroPosX[i] ; yp = astroPosY[i]
211.         astroPosX[i] += astroSpeed[i][0]
212.         astroPosY[i] += astroSpeed[i][1]
213.         if int(xp) != int(astroPosX[i]) or int(yp) !=
214. int(astroPosY[i]):
215.             colRect[i].x = astLst[i].x + 7
216.             colRect[i].y = astLst[i].y + 6
217.             astLst[i].x = int(astroPosX[i])
218.             astLst[i].y = int(astroPosY[i])
219.             if astLst[i].x > 1250: #X past right side
220.                 astroPosX[i] = -48.0 ; astLst[i].x = -48
221.             if astLst[i].y > 420: #Y past the bottom
222.                 astroPosY[i] = -44.0 ; astLst[i].y = -44
223.             if astLst[i].x < -50: #X past left side
224.                 astroPosX[i] = 1250.0 ; astLst[i].x = 1250
225.             if astLst[i].y < -44: #Y over the top
226.                 astroPosY[i] = 420.0 ; astLst[i].y = 420
227. def readVoltage():
228.     global average, avPoint, speed, runningAv
229.     for i in range(0,2):
230.         adc = spi.xfer2([1,(8+i)<<4,0]) # request
231.         reading = (adc[1] & 3)<<8 | adc[2] # join bytes
232.         runningAv[i][int(avPoint[i])] = reading
233.         avPoint[i]+=1
234.         if avPoint[i] >= nAv:
235.             avPoint[i] = 0
236.             average[i] = 0
237.             for j in range(0,nAv): # new running average
238.                 average[i] += runningAv[i][j]
239.             average[i] = average[i] / nAv
240.             speed[i] = average[i] / scale[i]
241. def terminate(): # close down the program
242.     print("Closing down please wait")
243.     pygame.mixer.quit()
244.     pygame.quit() # close pygame
245.     os._exit(1)
246. def checkForEvent(): # see if we need to quit
247.     global start, done
248.     event = pygame.event.poll()
249.     if event.type == pygame.QUIT :
250.         terminate()
251.     if event.type == pygame.KEYDOWN :
252.         if event.key == pygame.K_ESCAPE :
253.             terminate()
254.         if event.key == pygame.K_RETURN :
255.             done = True
256.         if event.key == pygame.K_SPACE :
257.             start = True
258.     # Main program logic:
259.     if __name__ == '__main__':
260.         main()
261.

```

File Edit Tools Window Help

Welcome

Code

Design

Configure

Analyze

Debug

Help

Open Documents

Controller.cpp

Controller.h

```

72 void Controller::ProcessFollowLeftWall()
73 {
74     MotorLeft = BaseSpeed - PIDLeft;
75     MotorRight = BaseSpeed + PIDLeft;
76 }
77
78 void Controller::ProcessFollowRightWall()
79 {
80     MotorLeft = BaseSpeed + PIDRight;
81     MotorRight = BaseSpeed - PIDRight;
82 }
83
84 void Controller::ProcessDrive()
85 {
86     MotorLeft = MotorRight = static_cast<double>(DriveStraightSpeed);
87 }
88
89 void Controller::ProcessTurn()
90 {
91     if (SensorLeft > SensorRight)
92     {
93         MotorLeft = static_cast<double>(TurnSpeed);
94         MotorRight = 0;
95     }
96     else
97     {
98         MotorLeft = 0;
99         MotorRight = static_cast<double>(TurnSpeed);
100    }
101 }
102
103 bool Controller::TransitionFromNullToDrive()
104 {
105     return true;
106 }
107
108 /* Change state only if robot has not seen any wall TRANSITION_TO_DRIVE_TIMEOUT_SEC seconds
109 */
110 bool Controller::TransitionToDrive()
111 {
112     if (!CanSeeFrontWall() && !CanSeeLeftWall() && !CanSeeRightWall())
113     {
114         if (!

```

Issues Alarms Search R... Application Compile General Connect...

Code

File Edit Tools Window Help

Welcome

Code

Design

Configure

Analyze

Debug

Help

Object

mainWindow

centralwidget

widget_content

overview

widget

cd

Temperature

Humidity

Pressure

Dark mode Off

Issues Alarms Search R... Application Compile General Connect...

Design

File Edit Tools Window Help

Welcome

Code

Design

Configure

Analyze

Debug

Help

System configurator

Mov... MoverDe... GPIOServer

GPIO Pins

Name	Model	Nr	InternalValue	Value	Unit
MotorLeft	GPIOPinIO_G...	8			
MotorRight	GPIOPinIO_G...	7			
PushButton	GPIOPinIO_G...	9			
LightButton	GPIOPinIO_G...	10			

Signals

Name	Model	Input	InternalValue	Value	Unit
NumberOfNodes	CDPSignal...		0	0	count
MotorLeftPWM	CDPSignal...		0	0	
MotorRightPWM	CDPSignal...		0	0	
IsOn	CDPSignal...				

Parameters

Name	Value	Unit	Type	Min	Max
SignalTimeout	1	s	double	0	0
	0		double	0	0

States

Name	Description
Null	The IOServer is in the initial state.
Online	The IOServer is Online.
Offline	The IOServer is Offline.

Properties

Name	Type	Value	Description	SaveOnCh...	LogEv...
------	------	-------	-------------	-------------	----------

Issues Alarms Search R... Application Compile General Connect...

Configure

File Edit Tools Window Help

Welcome

Code

Design

Configure

Analyze

Debug

Help

Untitled

Move...ront 100,000

Move...Left 50,000

Move...ight 0,000

0,000 50,000 100,000

05.02.2018 14:19:25.000 05.02.2018 14:19:30.000

Issues Alarms Search R... Application Compile General Connect...

Analyze

Now free for home projects

A professional control system development tool

CDP Studio is a development platform for industrial control systems, now coming with a free version for non-commercial use. The system can run on a Raspberry Pi, supports C++, open source libraries and has a large feature toolbox including GPIO, I2C and MQTT. Its built in GUI design tool and features lets you code less and do more.

Free download on www.cdpstudio.com

CDP Technologies AS
Nedre Strandgate 29
P.O. Box 144
NO-6001 Ålesund, Norway

Tel: +47 990 80 900
info@cdptech.com
www.cdpstudio.com





Build your own NAS



MAKER

**PJ
Evans**

PJ is a writer, developer, Milton Keynes Raspberry Jam wrangler, and he checks his backups regularly, just like you do.

@mrpjevans

Got a lot of digital stuff? Make it available anywhere in your home with your own Raspberry Pi network-attached storage

If you've got a lot of files like photos, music, or movies, chances are they are sitting on a hard drive somewhere. Getting access to those files and making sure they are protected from drive failure can be challenging without an expensive network-attached storage (NAS) solution. These file- and media-serving black boxes can punch a hole in your bank account, particularly the professional versions aimed at businesses. Now, thanks to the improved throughput of Raspberry Pi 4 with USB 3.0 and Gigabit Ethernet, you can build a fully featured NAS for a fraction of the cost.

01 Is NAS for you?

So what is a NAS, anyway? A well-implemented, network-attached storage device is typically a headless device (no keyboard or

monitor) providing access to large amounts of data from anywhere on your network. It must also offer availability and resilience for your data. That means it should protect against system failures that cause significant downtime, and make sure no data is lost as a result of those failures. The files themselves should be available with appropriate security measures over desired protocols. In simpler terms, it's a box on to which you dump all your movies, photos, music, and other stuff so you can get to it wherever and whenever.

02 Self-storage

The most important decision you'll make is how much storage you'll need. The design of Raspberry Pi means using external USB disks. Rotary drives give us lower cost and higher capacity than SSDs. Raspberry Pi 4 offers USB 3.0, so make



▲ USB 3.0 and full Gigabit Ethernet are essential to building a good Raspberry Pi NAS

sure you get external USB drives that take advantage of that extra speed. To provide a layer of protection, you'll need to double the number of drives to make sure your data is safer. We decided on 1 terabyte of storage, meaning two 1TB external drives. For reliable power we added a powered USB 3.0 hub.

03 Prepare the OS

Download Raspbian Buster Lite (magpi.cc/raspbian) and burn it to a microSD card. Once booted, make sure SSH has been enabled by running `sudo raspi-config` and selecting Interfacing Options > SSH. If you wish, configure WiFi at this point, but for a decent NAS you'll ideally be using the lovely full-speed Gigabit Ethernet port. Finally, change your password and, under Network Options, change the Hostname (the NAS's network name) if you wish. We imaginatively changed ours to 'nas', so the network address is 'nas.local'. Finally, make sure everything is up-to-date with `sudo apt update && sudo apt -y upgrade`, then reboot.

04 Add your storage

Using the powered USB 3.0 hub connected to your Raspberry Pi, plug in all your USB disk drives. Give the system a few seconds to 'see' the disks, then enter the following:

```
lsblk
```

This command tells you about devices connected to the system. The one starting 'mmcblk0' is the microSD card containing Raspbian. If you have two USB disks installed and working, you should also see 'sda' and 'sdb' (Storage Device A and Storage Device B). If you have more drives, it will continue up the alphabet.

05 Prepare the drives

Next, we need to partition the drives so Raspbian can understand how to store data on them. To do this we use fdisk.

```
sudo fdisk /dev/sda
```

When prompted for a command, enter 'n' for new partition. (If you get an error that a partition already exists, use 'd' to delete it – this will lose any data on the disk!)

Then enter 'p' (for primary partition). You'll be asked a series of questions about sectors. Don't panic. Just keep pressing **ENTER** (accepting the defaults) until 'Created a new partition' appears. Now type 'w' (to write the changes to the disk).

fdisk will now exit. You need to repeat the process for the second drive by entering:

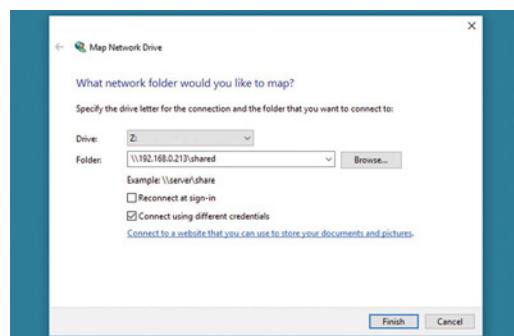
```
sudo fdisk /dev/sdb
```

...repeating the process as before.

"With RAID-1, or mirroring, anything written to one disk is automatically written to the other"

06 Create the RAID

RAID (redundant array of inexpensive disks) is a method for protecting data by duplicating it over multiple disks. There are many different forms, but we're using one of the simplest: RAID-1, or mirroring. Anything written to one disk is automatically written to the other. Should a disk fail, your NAS keeps running and you don't lose anything. Replace the failed disk as soon as possible and the array is 'rebuilt'.



▲ To access the NAS in Windows 10, in File Explorer, click This PC, then 'Map Network Drive'. Enter \\ip-address-of-nas\\shared

You'll Need

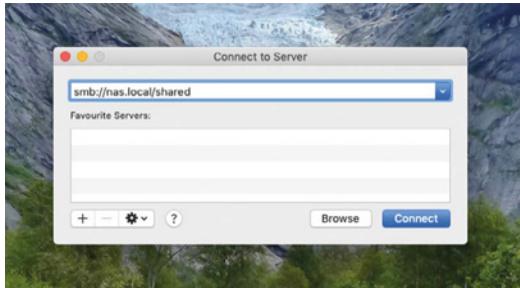
- ▶ 2 x External USB drives (minimum)
e.g. magpi.cc/AMyZWb
- ▶ USB 3.0 powered hub
e.g. magpi.cc/GwFcwX
- ▶ Gigabit Ethernet (recommended)
- ▶ UPS (optional)

Top Tip

Raspbian is not the only fruit

OpenMediaVault is a dedicated NAS server image with 100% web-based configuration. Although it does not support RAID for USB drives, it does have a host of features.

► In macOS, access your shared files by opening Finder and clicking 'Network'. After a few seconds, you should see the host name of your NAS appear



First, install the software RAID manager, mdadm:

```
sudo apt install mdadm
```

Now instruct mdadm to create the RAID-1 array:

```
sudo mdadm --create --verbose /dev/md0
--level=mirror --raid-devices=2 /dev/sda1 /dev/sdb1
```

Once done, the user 'pi' can access the Samba share from Windows, macOS, or other Raspberry Pi devices ■■■

07 Mount the drive

Raspbian will now see both physical disks as a single device. You can format and mount the new virtual drive:

```
sudo mkdir -p /mnt/raid1
sudo mkfs.ext4 /dev/md0
sudo mount /dev/md0 /mnt/raid1/
ls -l /mnt/raid1/
```

You should see one item: 'lost+found'. The RAID-1 system is operational. Next, make sure that the drive is mounted whenever you boot.

```
sudo nano /etc/fstab
```

Add the line:

```
/dev/md0 /mnt/raid1/ ext4 defaults,noatime 0 1
```

Quit (**CTRL+X**, followed by **Y**), then run the following so the RAID array starts up correctly on boot:

```
sudo mdadm --detail --scan | sudo tee -a
/etc/mdadm/mdadm.conf
```

Reboot and you should have **/mnt/raid1** ready to go.

08 Do the Samba!

Now to share some files on the network using the popular protocol, SMB/CIFS. The Raspbian version of this has the slightly more friendly name of Samba, but it is not installed by default. Run the following:

```
sudo apt install samba samba-common-bin
```

If you are asked any questions, just select the default answer. Now let's make a directory and allow all users access:

```
sudo mkdir /mnt/raid1/shared
sudo chmod -R 777 /mnt/raid1/shared
```

Tell Samba to share the directory on the network by editing the config file:

```
sudo nano /etc/samba/smb.conf
```

At the bottom, add the following:

```
[shared]
path=/mnt/raid1/shared
writeable=Yes
create mask=0777
directory mask=0777
public=no
```

Save (**CTRL+X**, followed by **Y**), then restart Samba:

```
sudo systemctl restart smbd
```

09 Granting access

To give a user access to the shared files, we need to run a special command to set a Samba password. So, to grant access to the current user, 'pi':

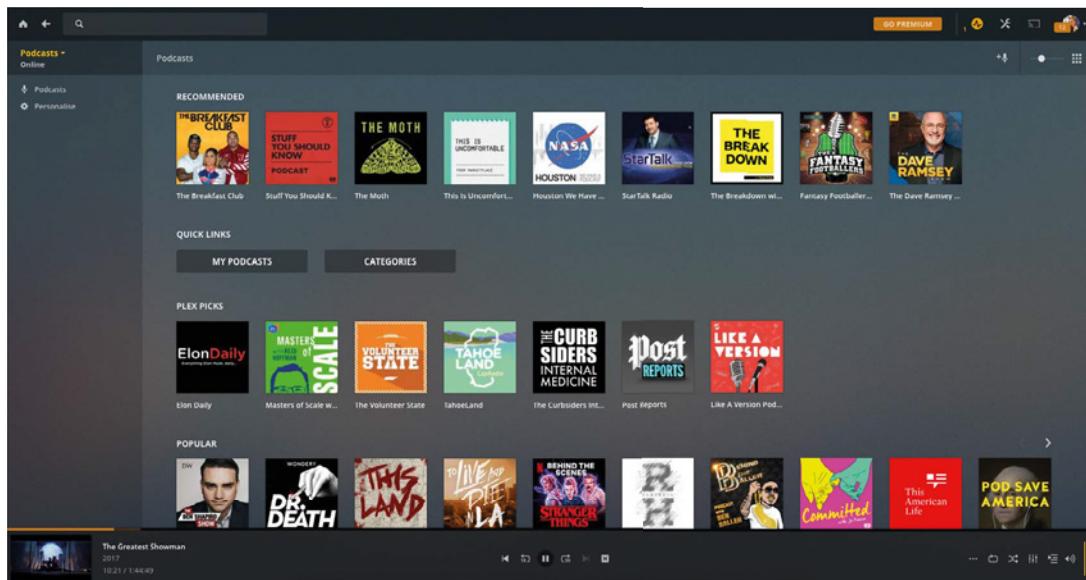
```
sudo smbpasswd -a pi
```

You'll be asked to choose a password (it doesn't have to be the same as your Raspberry Pi password). Once done, the user 'pi' can access the Samba share from Windows, macOS, or other Raspberry Pi devices, with the ability to read and write files.

To create additional users:

```
sudo adduser username
sudo smbpasswd -a username
```

...where 'username' is your choice of username.



NAS servers have many uses, such as acting as a huge file store for media servers like Plex

10 Create home directories

If you want to create file shares that are private to individual users, just create their own directory on the RAID array:

```
mkdir /mnt/raid1/shared/username
sudo chown -R username /mnt/raid1/shared/
username
sudo chmod -R 700 /mnt/raid1/shared/username
```

Again, replace `username` with the user you want. Now only that user can access that directory.

Alternatively, you can create additional entries in `smb.conf` for multiple shares.

11 Backup, backup, backup

RAID is not a backup system. It provides a certain level of data redundancy, but will not be of any help if you accidentally delete a file. If a drive does fail, your system will be in a ‘degraded’ state, meaning that data is at risk until the drive is replaced. If the second drive fails, disaster.

The ideal solution is to use a cloud provider such as Google or Dropbox to back everything up. Utilities such as Rclone (rclone.org) can sync entire directory structures onto many different providers’ storage. Set this up and create a regular cron job to make sure your data survives.

12 Don’t interrupt!

You can polish off this project with an uninterruptible power supply (UPS). Sudden power cuts can spell disaster for Linux-based systems due

to the way they handle files in memory. This battery backup safely keeps your Raspberry Pi and hub running in the event of a power cut. Many UPSes can communicate their status to your Raspberry Pi over USB, so a safe shutdown can be triggered.

13 Add more features

Our NAS can now create file shares, the most basic of capabilities. Professional NAS software often offers additional protocols such as Apple AFS, FTP, and many others. Most of these can also be implemented on a Raspberry Pi NAS. For the more adventurous user, Docker is an excellent way of making your NAS perform multiple functions without getting into a configuration nightmare. Why not set up a DLNA streaming server or run multiple databases? If you’ve enabled SSH, you’ve already got SFTP available; just connect using your favourite FTP client using `/mnt/raid1/shared` as the starting point.

14 Conclusions

NAS can be expensive. For a much more affordable way to store loads of files to share with friends or family, Raspberry Pi 4 is ideal. It can’t compete with Intel-based systems in terms of speed or features, but if you have some external USB disks lying around, it’s a very affordable way to not only serve your data, but protect it as well.

Thanks to Alex Ellis and Emmet Young for their excellent blog posts on RAID (magpi.cc/qzPmjo) and Samba (magpi.cc/HoivNg).

Top Tip

Keep spares

The comparatively low costs of this project mean spares can be kept to hand, allowing quick swap-outs of the disks and the Raspberry Pi.

Part 05

Combo boxes with C and GTK



MAKER
Simon Long

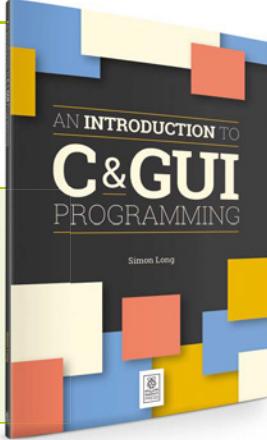
Simon Long is a software engineer working for Raspberry Pi, responsible for the Raspberry Pi Desktop on both Raspbian and Debian.

rpf.io

An Introduction to C & GUI Programming

For further tutorials on how to start coding in C and creating GUIs with GTK, take a look at our new book, *An Introduction to C & GUI Programming*. Its 156 pages are packed with all the information you need to get started – no previous experience of C or GTK is required!

magpi.cc/GUIbook



Create combo boxes for user input and associate list stores with them

Widgets like spin buttons, radio buttons, and check buttons are useful to allow a user to make selections between small numbers of options, but sometimes we need to offer a larger number of options. A combo box is a good way of offering the user a selection of choices without taking up huge amounts of space; GTK offers two types of combo box.

Text combo boxes

The simpler kind of combo box is the GtkComboBoxText, which only allows each option to be plain text. Here's how to create one:

```
GtkWidget *comb = gtk_combo_box_text_new ();

gtk_combo_box_text_append_text (
GTK_COMBO_BOX_TEXT (comb), "Option 1");
gtk_combo_box_text_append_text (
GTK_COMBO_BOX_TEXT (comb), "Option 2");
gtk_combo_box_text_append_text (
GTK_COMBO_BOX_TEXT (comb), "Option 3");
gtk_combo_box_set_active (
GTK_COMBO_BOX (comb), 0)
```

The `gtk_combo_box_text_new` function creates an empty combo box, and entries can be added to it (in the order they should appear, from top to bottom) using calls to `gtk_combo_box_text_append_text`.

The final function call shown above, `gtk_combo_box_set_active`, sets the initial value of the combo box – in this case, setting it to 0 chooses the first option added to the box, ‘Option 1’. Note that this is a function for the widget GtkComboBox, rather than for GtkComboBoxText – because GtkComboBox is a parent of GtkComboBoxText, you can use the functions of the parent on the child with an appropriate cast (**Figure 1**).

To read the currently selected value, use:

```
int sel = gtk_combo_box_get_active (
GTK_COMBO_BOX (comb));
```

This returns the index of the currently selected item – so 0 for ‘Option 1’, 1 for ‘Option 2’ and so on. Alternatively,

```
char *selected =
gtk_combo_box_text_get_active_text (
GTK_COMBO_BOX_TEXT (comb));
```

...can be used to return the actual text string which is selected. (Note that this function only works on text combo boxes; it can't be used on the full combo box described in the next section.)

The signal generated by a combo box when the value is changed by the user is called `changed`, so a suitable callback like:

```
void combo_changed (
GtkWidget *wid, gpointer ptr)
{
    int sel = gtk_combo_box_get_active (
GTK_COMBO_BOX (wid));
    char *selected =
```

```
gtk_combo_box_text_get_active_text (
    GTK_COMBO_BOX_TEXT (wid));
    printf ("The value of the combo is %d %s\n",
    sel, selected);
}
```

...can be connected using:

```
g_signal_connect (comb, "changed",
    G_CALLBACK (combo_changed), NULL);
```

The GtkComboBoxText is a useful control for simple cases, but the full GtkComboBox gives a lot more flexibility. Like some other widgets for free-form data display, it is used as a front end on a database structure called a *list store*.

Each cell of the list store data table can store a text string, an integer, or even an image

List stores

A list store can be thought of as a table of data with multiple rows and columns. Each cell of the table can store a text string, an integer, or even an image.

Once a list store is created, a combo box can be associated with it and one column of the list store can be used to provide the options for the box. The code for this is a little longer and more complex than that for a GtkComboBoxText above, but adds the potential for much more sophisticated handling of combo boxes. Here's the code to do that:

```
int pos = 0;

GtkListStore *ls = gtk_list_store_new (
    1, G_TYPE_STRING);
    gtk_list_store_insert_with_values (
    ls, NULL, pos++, 0, "Option 1", -1);
    gtk_list_store_insert_with_values (
    ls, NULL, pos++, 0, "Option 2", -1);
    gtk_list_store_insert_with_values (
    ls, NULL, pos++, 0, "Option 3", -1);

GtkWidget *comb =
    gtk_combo_box_new_with_model (
    GTK_TREE_MODEL (ls));

GtkCellRenderer *rend =
    gtk_cell_renderer_text_new ();
```

```
gtk_cell_layout_pack_start (
    GTK_CELL_LAYOUT (comb), rend, FALSE);
    gtk_cell_layout_add_attribute (
    GTK_CELL_LAYOUT (comb), rend, "text", 0);
```

First, we create a GtkListStore:

```
GtkListStore *ls = gtk_list_store_new (1,
    G_TYPE_STRING);
```

The `gtk_list_store_new` function takes a list of arguments – the first is the number of columns in the list store, and this is followed by a list of the types of data stored in each column. In this example we are creating a store with a single column, and that column will hold a text string.

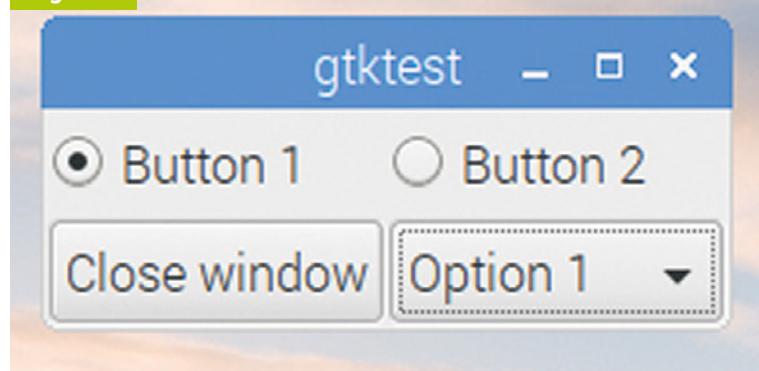
We then add some entries to the list store:

```
gtk_list_store_insert_with_values (ls, NULL,
    pos++, 0, "Option 1", -1);
    gtk_list_store_insert_with_values (ls, NULL,
    pos++, 0, "Option 2", -1);
    gtk_list_store_insert_with_values (ls, NULL,
    pos++, 0, "Option 3", -1);
```

Each call to `gtk_list_store_insert_with_values` takes the list store as an argument, followed by a `NULL` pointer. (In some cases, this would be a pointer to what is called an *iterator*.)

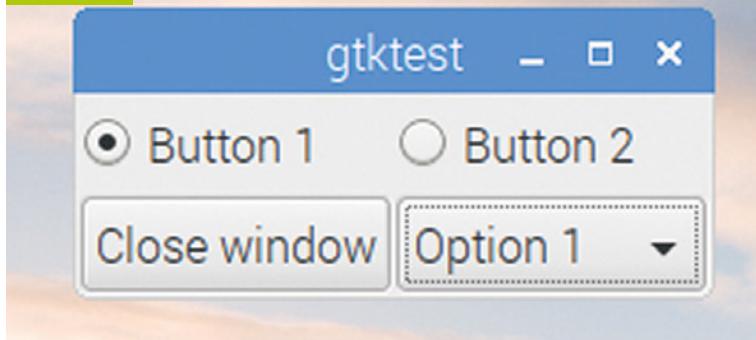
These are followed by the index within the list store where the new entry is to be placed – effectively the number of the row to be used for the data – and then a list of paired values. The first value of each pair is the column where the data is to be stored, and the second is the data itself; the list of pairs is terminated with a `-1` for the column value. In this case, we are only adding a single text string item to each row, in column 0.

Figure 1



▲ A GtkComboBoxText at the bottom-right of the window

Figure 2



▲ A GtkComboBox instead of the GtkComboBoxText – don't worry; they are supposed to look the same!

We then associate a combo box with the list store:

```
GtkWidget *comb =
gtk_combo_box_new_with_model (
GTK_TREE_MODEL (ls));
```

This creates a new combo box and tells GTK that the data for this combo box comes from the `ls` list store we previously created. The data used by a combo box is of type GtkTreeModel, so we cast our list store to this type. Now we set which column of the list store should be shown in the combo box.

```
GtkCellRenderer *rend =
gtk_cell_renderer_text_new ();
```

■ The data used by a combo box is of type GtkTreeModel, so we cast our list store to this type ■

First we create a GtkCellRenderer – this is a code object which is used to create a graphical representation of data in a table. In this case, it is a text renderer, which is used to display text strings.

```
gtk_cell_layout_pack_start (
GTK_CELL_LAYOUT (comb), rend, FALSE);
```

We then add this renderer to the cell layout of the combo box – this means that the renderer will be called when the combo box wants to display some data. (The final parameter, set to `FALSE`, is whether or not the data is expanded to fill free space in the layout; this has no effect on a combo box.)

```
gtk_cell_layout_add_attribute (
GTK_CELL_LAYOUT (comb), rend, "text", 0);
```

Finally, we set the ‘text’ attribute of the cell renderer to the data from the supplied column number of the list store – this causes the renderer

to display the data in column 0 as text in the combo box.

Try building and running the code, and satisfy yourself that it does the same thing as the previous example with GtkComboBoxText (**Figure 2**).

Some of the code may look a bit like black magic, and in truth it is a complicated way to put some text in a combo box; cell renderers make more sense in some other circumstances, which we will look at next month. But they are a necessary complication to using list stores to provide data for combo boxes, and that can be very useful, because we can process the data in a list store and have the combo box automatically reflect that processing.

For example, in the code above, we've put the three text strings into the list store in alphabetical order, but in the real world we can't guarantee that data will come in a nice tidy order. With a list store, it is easy to sort the data, by replacing the line:

```
GtkWidget *comb =
gtk_combo_box_new_with_model (
GTK_TREE_MODEL (ls));
```

...with:

```
GtkTreeModelSort *sorted =
GTK_TREE_MODEL_SORT (
gtk_tree_model_sort_new_with_model (
GTK_TREE_MODEL (ls)));

gtk_treeSortable_set_sort_column_id (
GTK_TREE_SORTABLE (sorted), 0,
GTK_SORT_ASCENDING);

GtkWidget *comb =
gtk_combo_box_new_with_model (
GTK_TREE_MODEL (sorted));
```

In this case, we create a GtkTreeModelSort, and initialise it with the data from our original list store. We then sort the data alphabetically by setting the sort column of the GtkTreeModelSort to 0 (because column 0 contains the data we want to sort on) and specifying ascending (from A to Z) sort order. Finally, we use the sorted model as the data source for our combo box, rather than the original unsorted model.

Similar functions exist to allow you to filter the rows in a list store, and you can combine sorts and filters to easily customise what is shown in a combo box based on settings elsewhere in an application – this is particularly useful when, for example, using a cascading series of combo boxes, each of which restricts the options in a subsequent box. ■

Wireframe

Join us as we lift the lid
on video games



Visit wfmag.cc to learn more

Part 03

Make 8-bit graphics in PICO-8



MAKER
Dan
Lambton-Howard

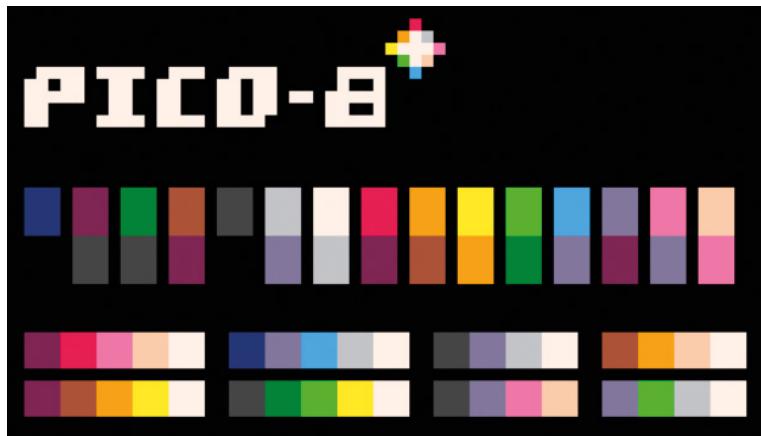
Dan is an independent game designer based in Newcastle upon Tyne, where he is lucky enough to make games for his PhD.

@danhowardgames

With a refreshingly simple set of tools, PICO-8 is the perfect place to plot pixels. Sprites – 2D images composed with pixels – have been a mainstay of game development since, well, forever, and have seen a recent resurgence due to the rise of pixel-art indie titles like Celeste, Spelunky, and Stardew Valley. We'll be taking a look at how to create effective 8-bit sprites for our space shooter, how to use a sprite sheet for animation, some basic background 'parallax' scrolling, and some simple space explosions for good measure.

01 Tools of the trade

If you've been following this tutorial series from the start, you'll already be acquainted with PICO-8's small, but mighty, sprite editor. Load up your game, then switch to the editor to look more closely at what we'll be working with (**Figure 1**).



▲ **Figure 2** PICO-8's distinctive 16-colour palette offers a surprisingly versatile range of shades

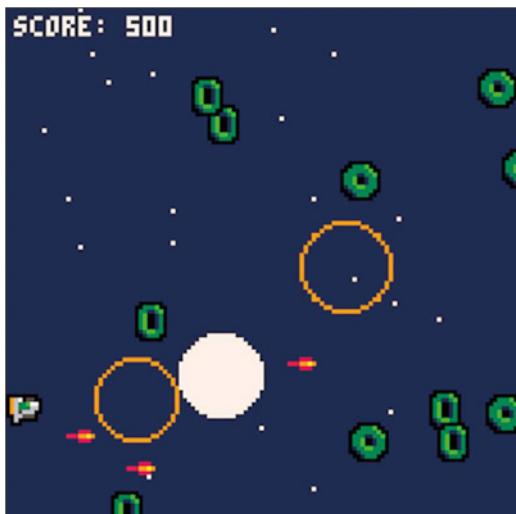
The big box on the top-left is your sprite window where you can plot pixels. The tool set below allows you to draw, copy, select, pan, and fill. And to the right you have your colour selector, brush size, and zoom sliders, as well as sprite flags. What more could you need?

02 A pixel artist's palette

Part of PICO-8's popularity is its striking 8-bit palette. Creator Joseph 'Zep' White spent a long time choosing a set of 16 complementary colours that offer a wide range of shades and tones. See the palette image (**Figure 2**) for a breakdown of how they can be combined. But a classic approach is to choose a primary colour and use a lighter complementary colour to show highlighting, and a darker shade for shadow. To demonstrate, we've highlighted the grey (colour 6) of our space fighter, with white (7) on the tips. You can take this approach with all of your sprites.

03 Sprites, sheets, and animation

So how do we animate? The easiest way is to draw a new sprite for each animation frame and store them in a sprite sheet. Then, in runtime, we swap through these different sprites to make an object appear to come alive. The sprite sheet, shown at the bottom of the screen (see **Figure 1**), indexes each sprite on the sheet with a number. Let's start with the bad guys. Next to your original enemy sprite, draw a few more to show it at various stages of jiggling menacingly. Don't worry too much about smoothness as we can always edit them later.



▲ It's hard to tell on a static image, but these stars use parallax scrolling to give a sense of speed and depth

04 They live!

To implement the animation, we will need to add a few things to our code. First of all, in the `create_enemy()` function we need to add a new table to our enemies that stores all the sprite indexes for their gruesome animation. Add `enemy.sprites={2,18,2,34}`. We will be moving through this table from left to right at set intervals. To keep track of this we will need a timer: add `enemy.animtimer=0` to the function as well. Now, we need to actually tell it to change the enemy sprites along with the timer.

“The inclusion of the speed variable adds a little bit of flavour”

05 It's all in the timing

In the main `_draw()` function, find the enemy loop and at the start of it add `enemy.animtimer+=1` to increment each frame. Below this, add `enemy.sprite = enemy.sprites[fir(enemy.animtimer/5-enemy.speed*3)%#enemy.sprites+1]`, which looks complicated but really just compares the animation timer to the number of sprites in our sprites table and moves us along one. The inclusion of the speed variable adds a little bit of flavour that makes faster-moving enemies animate faster. Run your game and check it out in action.

06 Flickering fire

We can repeat this process for the player's ship, too. Draw another sprite which shows the

rocket engines flaring or flickering next to the original player sprite. Even just a couple of pixels different between frames is enough to make a sprite come alive. In `_init()` where we declare our player table, you'll need to add another animation timer and table of sprites, just as we did for the enemy. We'll also increment this timer in the draw function and add the line `player.sprite = player.sprites[player.animtimer%#player.sprites+1]` below this to make a fast flicker.

You'll Need

- ▶ PICO-8
magpi.cc/pico8
- ▶ Raspberry Pi
- ▶ Keyboard and mouse

07 Soaring through space

That's made our ship look a little better, but it still looks static. Let's animate it banking left or right when it moves. Draw a sprite of the ship banking left and one banking right. You can copy and paste your original to act as a starting point. Then copy these sprites and animate the tail flicker

Our sprite sheet shows the player's ship flickering and banking left and right, whilst the enemies jiggle menacingly

From left to right: draw, stamp, select, pan, and fill. A simple set of tools for some tasty pixel art

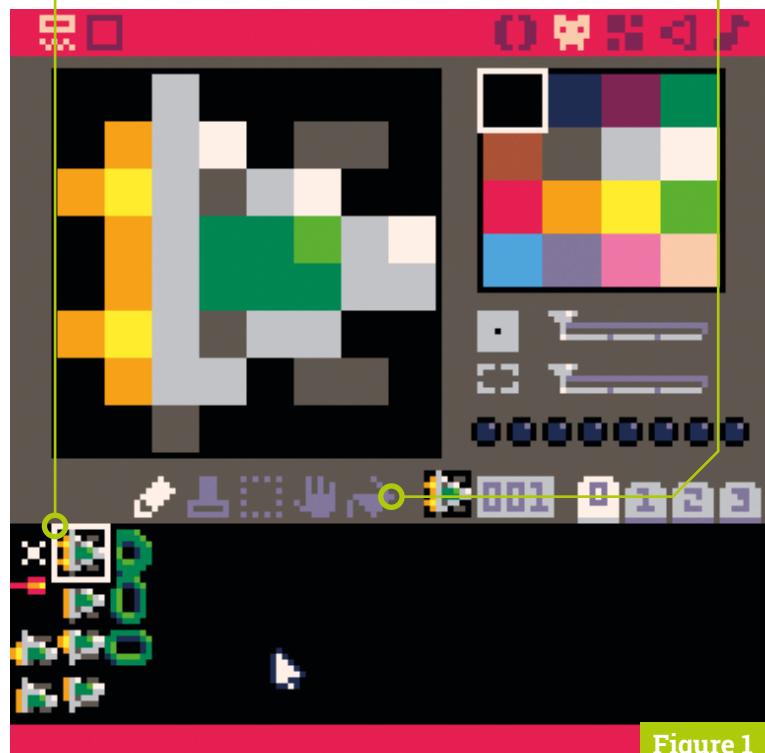
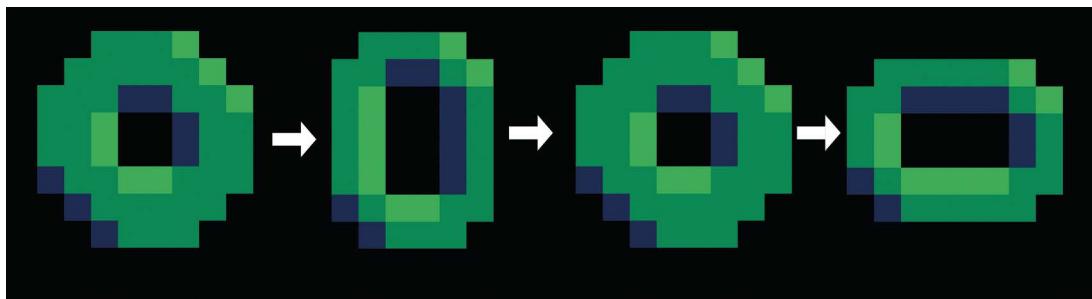


Figure 1

▶ Create a new sprite for each frame in an animation to make your enemies come alive!



Top Tip



Spotlight on sprites

PICO-8's `spr()` function is very flexible and can flip sprites horizontally and vertically, change their height and width, and more.

for each. Now we add a conditional to our `_update()` function that will swap our sprites table depending on if the up or down direction keys are pressed. Now our little space fighter will soar majestically through space.

08 Things that go boom

Currently our lasers are a puny red rectangle, but we can do better than that. Create a deadly-looking laser sprite in index 16 and replace our previous `rect()` function call with `spr(16, laser.x-5, laser.y)`. That's a clear improvement, but something is still missing: you guessed it, explosions. To create dynamic-looking impacts, we will be writing two functions: one to create them, and the other to draw them as flash of circles. Very nice. We will also declare a new explosions table in `_init()` and write a `for` loop to handle drawing.

09 Pyrotechnics

Our `create_explosion` function creates, you guessed it, explosions. Much in the same way as we've created enemies and lasers previously. Our `draw_explosion` function draws a circle depending on what stage the explosion's timer is at, then deletes it when it reaches 4. To see it work, add `create_explosion(enemy.x, enemy.y, rnd(4)+8)` just

▼ Indie hits such as Stardew Valley are masterclasses in modern pixel art



after we delete enemies upon collision with a laser. Using a random number for explosion radius gives us a little more flavour by varying the size of the explosions slightly. You should add a big explosion when the player is destroyed, too.

10 Parallax to the max

Parallax scrolling is an easy and effective way of adding depth and movement to a background by scrolling things at different speeds. Let's make a starfield to give the feeling our plucky starship pilot is in hyperdrive. Create a new stars table in `_init()` and populate it with stars using a `for` loop. Next, at the start of `_draw()`, add `rectfill(0,0,128,128,1)` to colour the screen deep space blue, and another `for` loop that draws each star as a single pixel, moves it from right to left, and resets it when it goes off screen. Warp factor 4!

11 A simple shader

'Shader' is a term used to describe a graphical treatment given to a rendering of a sprite or other asset. They are used extensively in game development, often to achieve a specific aesthetic style. Now we have a background, our sprites don't stand out as well as they did. Let's write a shader function `outline_sprite()` that draws a sprite offset in eight directions in black, using `pal()` to reset the palette, then the original colour sprite on top. Now, replace the player and enemy `spr()` calls, and see how a simple shader can make them pop!

12 Next steps: sound

Our game looks good. We have both time-based and movement-based animation. We have lasers and explosions. We have a scrolling starfield and a simple shader so our sprites stand out. However, our game will always feel lifeless without sound. So, next issue we will be making some spacey SFX to bring our shooter to life, and we'll be composing some 8-bit chiptunes. See you there! 

part3code.p8

► Language: Lua

DOWNLOAD
THE FULL CODE:



magpi.cc/fB0ksD

```

001. --new code reference for space shooter
002. --see full project in github for full context
003.
004. --within _init()
005. player.sprites={1,17} --player sprite table
006. player.animtimer=0 --player animation timer
007. explosions={} --explosions table
008. stars = {} -- background stars table
009. for i=0,24 do -- populate starts table with 24
   stars
010.   add(stars,{
011.     x=rnd(128),
012.     y=rnd(128),
013.     speed=rnd(10)+1
014.   })
015. end
016.
017. --within _update()
018. if btn(2) and not btn(3) then --banking left
019.   player.sprites = {33,49}
020. elseif btn(3) and not btn(2)then -- banking right
021.   player.sprites = {32,48}
022. else -- flying straight
023.   player.sprites = {1,17}
024. end
025.
026. --within _draw()
027. for enemy in all(enemies) do
028.   enemy.animtimer+=1 -- increment enemy
      animation timer
029.   --assign sprite to be drawn depending on enemy
      speed
030.   enemy.sprite = enemy.sprites[flr(
      enemy.animtimer/5-enemy.speed*3)%#enemy.sprites+1]
031.   outline_spr(enemy.sprite,enemy.x,enemy.y)
032. end
033.
034.
035. rectfill(0,0,128,128,1) --draw background
036. for star in all(stars)do
037.   star.x -= star.speed --move star left
038.   pset(star.x,star.y,7) --draw star as white dot
039.   if star.x < 0 then --if off screen then reset
040.     star.x = 128
041.     star.y=rnd(128)
042.   end
043. end
044.
045.
046. --within create_enemy()
047. enemy.sprites={2,18,2,34} --sprite set
048. enemy.animtimer=0 -- timer for animations
049.
050. --new functions
051. function create_explosion(x,y,radius)
052.   local explosion={
053.     x=x,
054.     y=y,
055.     radius=radius,
056.     timer=0,
057.   }
058.   add(explosions,explosion)
059. end
060.
061. --draw explosions
062. function draw_explosion(explosion)
063.   if explosion.timer<2 then -- white filled circle
064.     circfill(explosion.x,explosion.y,explosion.
      radius,7)
065.   elseif explosion.timer<4 then -- red filled circle
066.     circfill(explosion.x,explosion.y,
      explosion.radius,8)
067.   elseif explosion.timer<5 then -- orange circle
068.     circ(explosion.x,explosion.y,explosion.radius,9)
069.   del(explosions,explosion) -- delete
070.   return
071. end
072. explosion.timer+=1
073. end
074.
075. function outline_spr(sprite,x,y)
076.   for i=1,15 do --set all colours to black
077.     pal(i,0)
078.   end
079.   for xoffset=-1,1 do --draw sprites offset by
080.     for yoffset=-1,1 do --1 pixel in each direction
081.       spr(sprite,x+xoffset,y+yoffset)
082.     end
083.   end
084.   pal() --reset palette back to normal
085.   spr(sprite,x,y) --draw main sprite
086. end

```

GPIO music box

Create a customisable music machine that you control at the touch of a button



**Marc
Scott**

Marc is a Senior Learning Manager at the Raspberry Pi Foundation.

raspberrypi.org

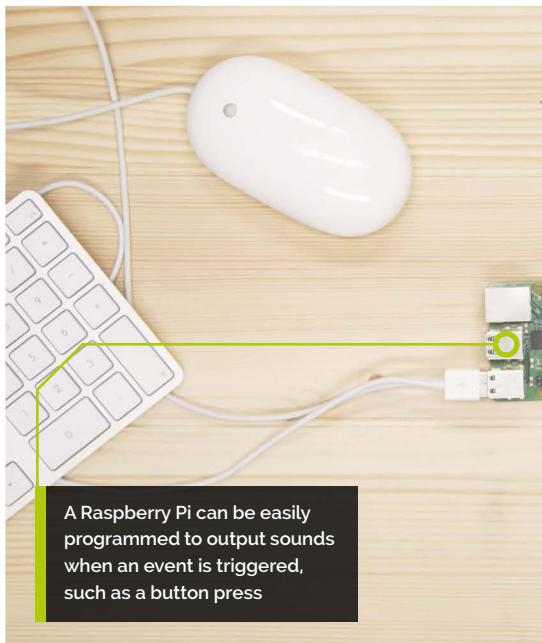
MAKER

In this project, we'll build a button-controlled 'music box' hooked up to a Raspberry Pi's GPIO pins that plays different sounds when different buttons are pressed. Not only does it help teach about using push-buttons and other inputs via GPIO Zero, it's also a good way to learn about playing music or other audio with Python. Let's get started.

01 Set up your project

You will need some sample sounds for this project. There are lots of sound files in Raspbian, but it can be a bit difficult to play them using Python. However, you can convert the sound files to a different file format that you can use in Python more easily.

First, in your home directory (`/home/pi`) create a directory called **gpio-music-box** by right-clicking and selecting New Folder. You will use the new directory to store all your files for the project.



02 Copy the sample sounds

Create a folder called **samples** in your **gpio-music-box** directory.

There are lots of sample sounds stored in the `/usr/share/sonic-pi/samples` directory. In this step, you will copy these sounds into the **gpio-music-box/samples** directory.

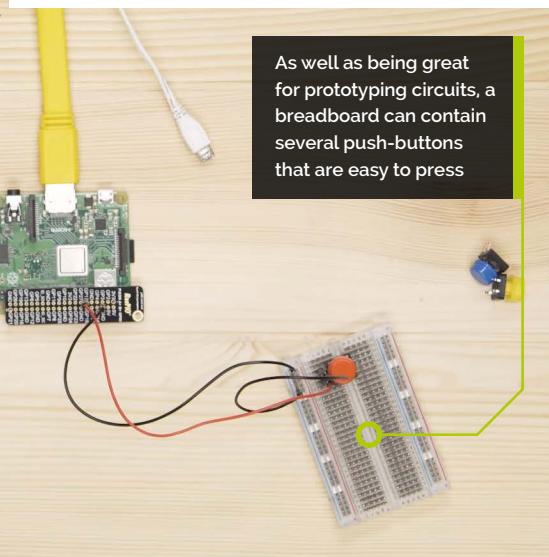
Click on the icon in the top-left corner of your screen to open a Terminal window. Type the following command to copy all the files from one directory to the other:

```
cp -r /usr/share/sonic-pi/samples/*
~/gpio-music-box/samples/.
```

When you have done that, you should be able to see all the FLAC (.flac) sound files in the **samples** directory.

03 Convert the sound files

To play the sound files using Python, you need to convert the files from FLAC to WAV format. In a Terminal, move to your **samples** directory:



You'll Need

- ▶ Push-buttons
- ▶ 220 Ω resistors
- ▶ Jumper cables
- ▶ Breadboard
- ▶ Speakers or headphones

```
cd ~/gpio-music-box/samples
```

Then enter the following commands. This will convert all the FLAC files to the WAV format and then delete the old files.

```
for f in *.flac; do ffmpeg -i "$f" "${f%.flac}.wav"; done
rm *.flac
```

It will take a minute or two, depending on the Raspberry Pi model that you are using. You should now be able to see all the new .wav files in the **samples** directory.

04 Play sounds

Next, you will start to write your Python code. You can use any text editor or IDE to do this — Thonny is always a good choice; you can find it in the Raspbian desktop applications menu (click the top-left raspberry icon) under the Programming category.

To start to create the instruments of your music box, you need to test whether Python can play some of the samples that you have copied. First, import and initialise the Pygame module for playing sound files:

```
import pygame
pygame.init()
```

Save this file in your **gpio-music-box** directory as **musicbox.py**. Choose four sound files that you want to use for your project, for example:

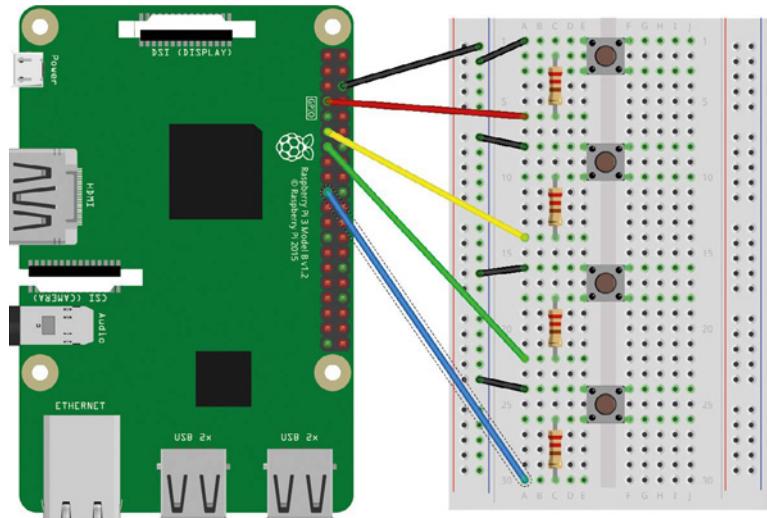
```
drum_tom_mid_hard.wav
drum_cymbal_hard.wav
drum_snare_hard.wav
drum_cowbell.wav
```

Then, create a Python object that links to one of these sound files. Give the file its own unique name. For example:

```
drum = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_tom_mid_hard.wav")
```

Create named objects for your remaining three sounds: cymbal, cowbell, and snare.

Save and run your code. Then, in the shell pane in the Thonny editor, use **.play()** commands to play the sounds. For example:



▲ Figure 1 Here's the way we've wired up our music box

listing1.py

DOWNLOAD THE FULL CODE:

► Language: Python 3

magpi.cc/musicbox

```
001. import pygame
002. from gpiozero import Button
003.
004. pygame.init()
005.
006. drum = pygame.mixer.Sound("/home/pi/gpio-music-box/
    samples/drum_tom_mid_hard.wav")
007. cymbal = pygame.mixer.Sound("/home/pi/gpio-music-box/
    samples/drum_cymbal_hard.wav")
008. snare = pygame.mixer.Sound("/home/pi/gpio-music-box/
    samples/drum_snare_hard.wav")
009. bell = pygame.mixer.Sound("/home/pi/gpio-music-box/
    samples/drum_cowbell.wav")
010.
011. btn_drum = Button(4)
```

drum.play()

If you don't hear any sound, check that your speakers or headphones are working, properly connected, and that the volume is turned up.

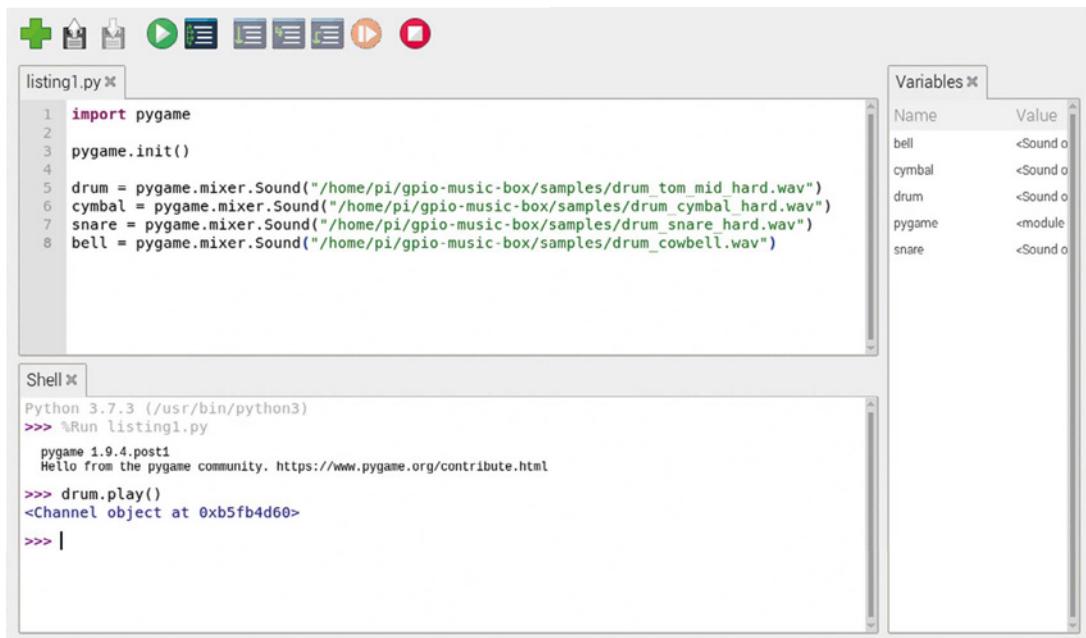
05 Connect your buttons

You will need four buttons, wired to GPIO pins on the Raspberry Pi. You'll need to have one side each wired to a different, programmable GPIO pin (not 5V, 3.3V, or GND), and all of them will also need to end at GND, with a resistor somewhere in the circuit.

Top Tip

Video tutorial

Check out a shorter, video version of this tutorial on YouTube here: magpi.cc/ivNfZv.



▶ Try playing your sounds using `.play()` commands in Thonny's shell pane

Place the four buttons into your breadboard. Wire each button to a different numbered GPIO pin. You can choose any GPIO pins you like, but you will need to remember their BCM numbers, which you can find on [pinout.xyz](#). Otherwise, just refer to **Figure 1**.

■ You will need four buttons, wired to GPIO pins on the Raspberry Pi ■

Top Tip

More music.
More buttons

Have more buttons spare and more music you want to use? Create a sample machine or even a fancy keyboard!

06 Play sounds at the press of a button

We're going to use the GPIO Zero Python library to control the buttons. When a specific button is pressed, the program should call a function such as `drum.play()`.

However, when you use an event (such as a button press) to call a function, you don't use brackets `()`. This is because the program must only call the function when the button is pressed, rather than straight away. So, in this case, you just use `drum.play`.

First, set up one of your buttons. Remember to use the numbers for the GPIO pins that you have used – refer to **listing1.py** for how this should look. Remember, you need to import the correct part of GPIO Zero and then define the relevant button. To play the sound when the button is pressed, just add this line of code to the bottom of your file:

```
btn_drum.when_pressed = drum.play
```

Run the program and press the button. If you don't hear the sound playing, then check the wiring of

your button. Now, add code to make the remaining three buttons play their sounds. This should end up looking something like **listing2.py**.

07 Improve your script

The code that you have written should work without any problems. However, it's generally a good idea to make your code a bit cleaner once you have a prototype that works.

The next steps are entirely optional. If you're happy with your script, then just leave it as it is. If you want to make your script a bit cleaner, you can have a go at storing your button objects and sounds in a dictionary, instead of having to create eight different objects.

Have a look at the steps below to learn about creating basic dictionaries and looping over them.

08 Guide to dictionaries

A dictionary is a type of data structure in Python. It contains a series of 'key : value' pairs. Here is a very simple example:

```
band = {'john' : 'rhythm guitar', 'paul' : 'bass guitar', 'george' : 'lead guitar', 'ringo' : 'bass guitar'}
```

The dictionary has a name, in this case `band`, and the data in it is surrounded by curly brackets `{}`. Within the dictionary are the 'key : value' pairs. In this case the keys are the names of the

band members; the values are the names of the instruments they play. Keys and values have colons between them (:), and each pair is separated by a comma (,). You can also write dictionaries so that each ‘key : value’ pair is written on a new line.

```
band = {
    'john' : 'rhythm guitar',
    'paul' : 'bass guitar',
    'george' : 'lead guitar',
    'ringo' : 'bass guitar'
}
```

To look up a particular value in a dictionary, you can use its key. So, for instance, if you wanted to find out what instrument ringo plays, you could type:

```
band['ringo']
```

09 Creating a sound dictionary

First, create a dictionary that uses the Buttons as keys and the Sounds as values.

```
button_sounds =
{Button(4): pygame.mixer.Sound("/home/pi/
gpio-music-box/samples/drum_tom_mid_hard.wav"),
 Button(17): pygame.mixer.Sound("/home/pi/
gpio-music-box/samples/drum_cymbal_hard.wav"),
 Button(27): pygame.mixer.Sound("/home/pi/
gpio-music-box/samples/drum_snare_hard.wav"),
 Button(10): pygame.mixer.Sound("/home/pi/
gpio-music-box/samples/drum_cowbell.wav")}
```

You can now use a `for` loop over the dictionary to tell the program to play the sound when the button is pressed:

```
for button, sound in button_sounds.items():
    button.when_pressed = sound.play
```

The final code can be found in **listing3.py**. Have fun with your new musical Raspberry Pi! ☺



▲ There are many Python IDEs and text editors, but Thonny is easy to use and already in Raspbian

listing2.py

► Language: **Python 3**

```
001. import pygame
002. from gpiozero import Button
003.
004. pygame.init()
005.
006. drum = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_tom_mid_hard.wav")
007. cymbal = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_cymbal_hard.wav")
008. snare = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_snare_hard.wav")
009. bell = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_cowbell.wav")
010.
011. btn_drum = Button(4)
012. btn_cymbal = Button(17)
013. btn_snare = Button(27)
014. btn_bell = Button(10)
015.
016. btn_drum.when_pressed = drum.play
017. btn_cymbal.when_pressed = cymbal.play
018. btn_snare.when_pressed = snare.play
019. btn_bell.when_pressed = bell.play
```

listing3.py

► Language: **Python 3**

```
001. import pygame
002. from gpiozero import Button
003.
004. pygame.init()
005.
006. button_sounds = {Button(4): pygame.mixer.Sound("/home/pi/
gpio-music-box/samples/drum_tom_mid_hard.wav"),
007.                   Button(17): pygame.mixer.Sound("/home/
pi/gpio-music-box/samples/drum_cymbal_hard.wav"),
008.                   Button(27): pygame.mixer.Sound("/home/
pi/gpio-music-box/samples/drum_snare_hard.wav"),
009.                   Button(10): pygame.mixer.Sound("/home/
pi/gpio-music-box/samples/drum_cowbell.wav")}
010.
011. for button, sound in button_sounds.items():
012.     button.when_pressed = sound.play
```

BACK TO SCHOOL

Learn and teach computing
with Raspberry Pi.

By Rosie Hattersley

Raspberry Pi was designed to help teachers teach, and students learn, computer science. It's a superb device for picking up computing skills. Ever since the first Raspberry Pi launched, back in 2012, it's become a vital tool for anybody involved with programming.

Only 10,000 units of the original single-board computer were expected to sell, each designed to go to a student to help them pick up computing skills. Several million sales later and Raspberry Pi sits at the heart of a giant charity with the aim of "putting the power of computing and digital making into the hands of people all over the world."

This organisation is packed with people making incredible projects and running all sorts of schemes, events, and activities around the globe. It's combined with other organisations like Code Club and teamed up with the ESA (European Space Agency) to produce exciting programs for teachers and students alike.

Here are just some of the things you can do to teach and learn computer science with Raspberry Pi.



66

Isaac Computer Science

Discover a brand-new scheme designed to boost students' confidence in A-level Computer Science

68

Picademy Bytes

Bite-sized sessions for learning computer science and digital making skills

70

Astro Pi

Get students directly involved in what's happening in space by running code aboard the ISS

Isaac Computer Science

This brand-new site is designed to boost students' confidence in A-level Computer Science

When Raspberry Pi first began being used in education it was largely aimed at primary school pupils due to the simplicity of coding with Scratch. Tom Bennett says in his book, *Teacher Proof*, "Raspberry Pi offers a mechanism for kids to explore and improve in a designated, clear area – computer design".

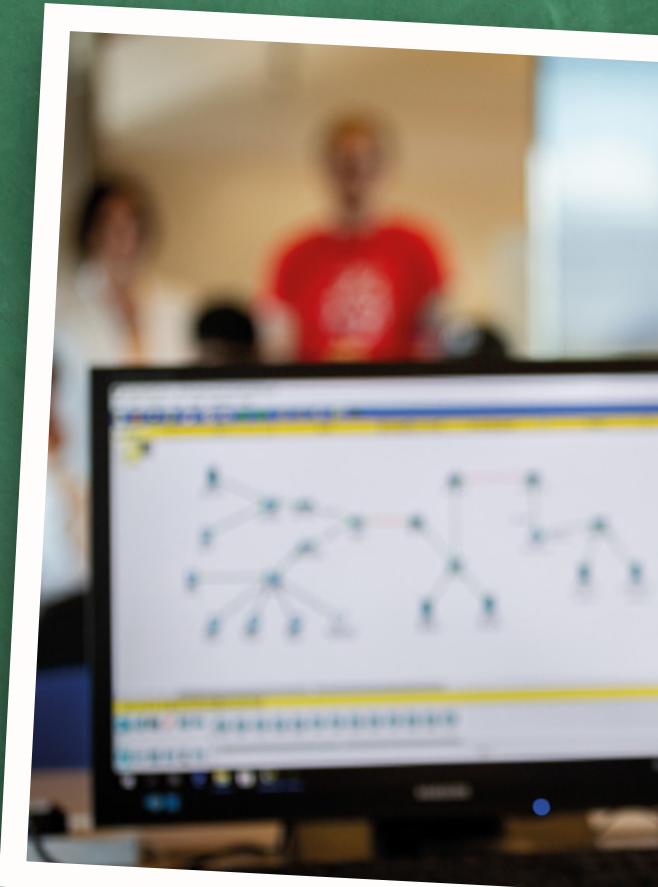
Perhaps unsurprisingly, Raspberry Pi has also been enthusiastically embraced by secondary school and A-level students.

This summer, the Department for Education launched the Isaac Computer Science platform (magpi.cc/GhToSo) to bolster older students' computing knowledge. However, there's no need to be an A-level Computer Science student to use it. As with the CPD training offered by the NCCE (National Centre for Computing Education: teachcomputing.org), Isaac Computer Science aims to spread computer knowledge across subject disciplines for both teachers and students.

The searchable list of topics spans theory and programming modules covering the likes of algorithms, databases, networking, and security. Some are specifically aimed at easing the transition from GCSE to A-level.

Sign up for Isaac Computer Science

Head to isaaccomputerscience.org to sign up. Either give your DOB or tick to signify you meet the 13-years-old minimum age threshold. If you're a teacher or student, you can elect to get notifications of new group assignment and details of related events in the UK. Non-students and non-teachers can still use the site.



▲ A-level Computer Science students can use Isaac Computer Science to consolidate their understanding of all elements of the course and get tailor-made help

Isaac Computer Science's online dashboard gives an overview of assignments set and lets you complete related course modules covering theory topics, programming challenges, and practice, with a range of Parson's puzzles in which students must place blocks of code in the correct order so it runs successfully.

"Aims to spread computer knowledge across subject disciplines"

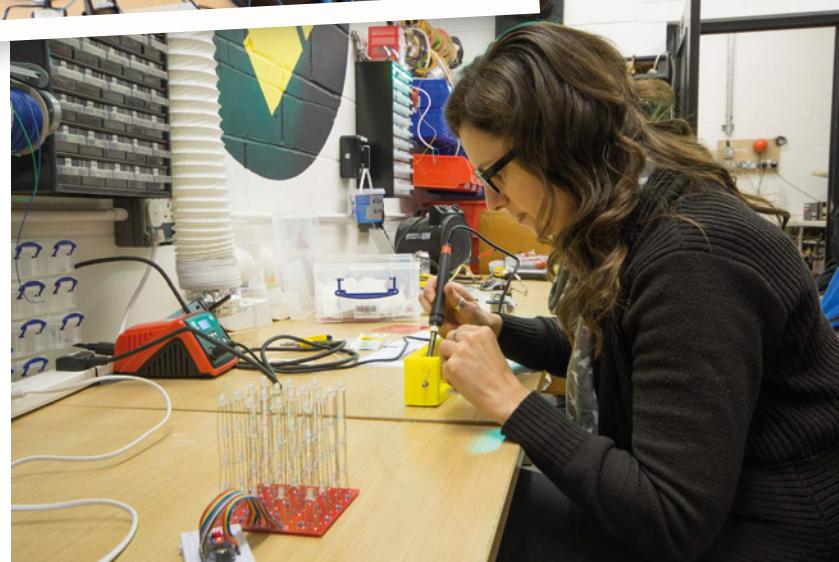
If you're an A-level Computer Science student or teacher, you can select between AQA and OCR exam boards so the course context and revision materials are tailored to the course content. Choose either syllabus view or suggested teaching. There are free training resources tailored to teachers and students, plus suggested questions to tackle to get warmed up. Hints along the way provide plenty of encouragement and links to useful reminders of mathematical and logistical rules.





► Students can take Isaac Computer Science modules alongside both the OCR and AQA and Computer Science syllabuses – ideal when it comes to revision

▼ Create a space where makers can build things and tinker with projects



HackSpace magazine

Enjoy using Raspberry Pi, Arduino, and electronics? HackSpace magazine is packed with projects you and your students can work on. Download issues free from hsmag.cc.



Build a makerspace

Makerspaces are inspiring hubs of creativity and hands-on learning where people bounce ideas off each other and perhaps collaborate on projects. If you and your students fancy the idea of setting up your own tinkering and digital making space, there are plenty of tips and resources at magpi.cc/VswMQT.

This free course – led by Raspberry Pi’s Director of Educator Support, Carrie Anne Philbin – is specifically designed for use in schools, libraries, and community spaces and can help you find partners to aid you with funding and kitting it out. It’s packed with practical tips and best practice ideas from other makerspace enthusiasts around the world.



▲ Picademy Bytes training sessions take place all over the UK

Picademy Bytes

Picademy offers Raspberry Pi computer training for educators and volunteers. Picademy Bytes are new bite-sized sessions



One of the issues with cramming our schools with increasing amounts of tech is the onus it places on teachers and support staff to learn new digital skills.

Picademy addresses this issue with tailored teacher training around the country. So, as part of your own professional development, you get trained in how to use the very tools students are most excited about.

Picademy Bytes is a new scheme designed to meet the needs of educators who are unable to attend the two-day sessions. The 60- to 90-minute

Hello World

Hello World is a Raspberry Pi magazine specifically aimed at schools. Published five times a year, it is jam-packed with teaching and learning resources and lesson ideas. Subscribe for free at helloworld.cc.



Picademy Bytes training sessions will be delivered by Community Trainers at different locations across the UK.

“Five years ago, the Raspberry Pi Foundation recognised a need for free, high-quality CPD (continuing professional development) for educators,” says Dan Elwick, Learning Manager. “In response, we started running Picademy, a two-day training event.” Those who finish Picademy become a Raspberry Pi Certified Educator.

Picademy is ideal if you’re keen to get involved with coding but have different subject specialisms.

Being off-site, you can focus on training without the usual school day distractions, then come back to the classroom primed to inspire some independent learning.

Each Picademy Bytes session is open to all teachers and educators and no previous experience is required. ‘An introduction to physical computing’ activities cover setting up a Raspberry Pi and using Scratch or Python on it to make LEDs flash. There are also ideas for linking this to elements of the National Curriculum.

#picademy

Raspberry Pi and Toronto Public Library

Linda Horne @lindaphorne · 20h

#picademy I made my LEDs go blinky-b-



Photos **Videos**

linky!

zizes your uptime, speeds up
ness.

▲ Picademy and Picademy Bytes sessions both get you to use a Raspberry Pi to make LEDs flash

"Give teachers the confidence to use physical computing and teach computer science"

The intention is that this will give teachers the confidence to use physical computing and teach computer science concepts.

The first Picademy Bytes sessions took place at Staffordshire University this summer. Attendees – mainly secondary school teachers – praised the mix of theory and practical sessions including plenty of ideas they could use in the classroom.

Future Picademy Bytes sessions will be posted on the Computing at School website: magpi.cc/DqgBha.

Back-to-school Bootcamp

A regular training regime builds confidence in your ability, as well as giving you the push to go further. That's the thinking behind Raspberry Pi's 'Bootcamp' programme. The online resources at rpf.io/bootcamp include everything from the basics of Python coding to creating your own GUIs (graphical user interfaces) and using object-oriented programming skills. There are block-based learning tools to help primary school teachers and more traditional programming for older students.

Once you've nailed the basics, more advanced tutorials take you through creating your own algorithms. As well as equipping you for the school year, completing modules earns you credits towards the NCCE's computer science educator programme. Sign up for CPD sessions at teachcomputing.org.

National Centre for Computing Education

BETA This is a new service – your [feedback](#) will help us to improve it.

Helping you teach computing

Discover training, resources and guidance to help you teach computing with confidence.

[Get started!](#)

National Centre for Computing Education

BETA This is a new service – your [feedback](#) will help us to improve it.

Courses

Please note that for face-to-face courses you will be taken to STEM Learning to see further details and book.

Online courses are hosted by FutureLearn. If you go through our site to book these courses will come with a free upgrade. You will need a separate FutureLearn account to sign up for them, please use the same email address as you used for your account on this site so we can track your progress.

After you complete the course, go to your dashboard to track your progress towards certification.

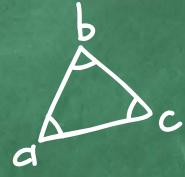
Diagnostic Tool

Need help finding a course that's suitable for you? Use our diagnostic tool to identify the areas of subject knowledge where you need most support.

[Use the diagnostic tool](#)

Find a course:

▲ Boost your own computer science knowledge by completing modules in the NCCE computer science educator programme



Astro Pi

The Astro Pi Challenge gives students the opportunity to become a space programmer!

With the 50th anniversary of the Apollo Moon landing, there's no better time to focus on all things space-related.

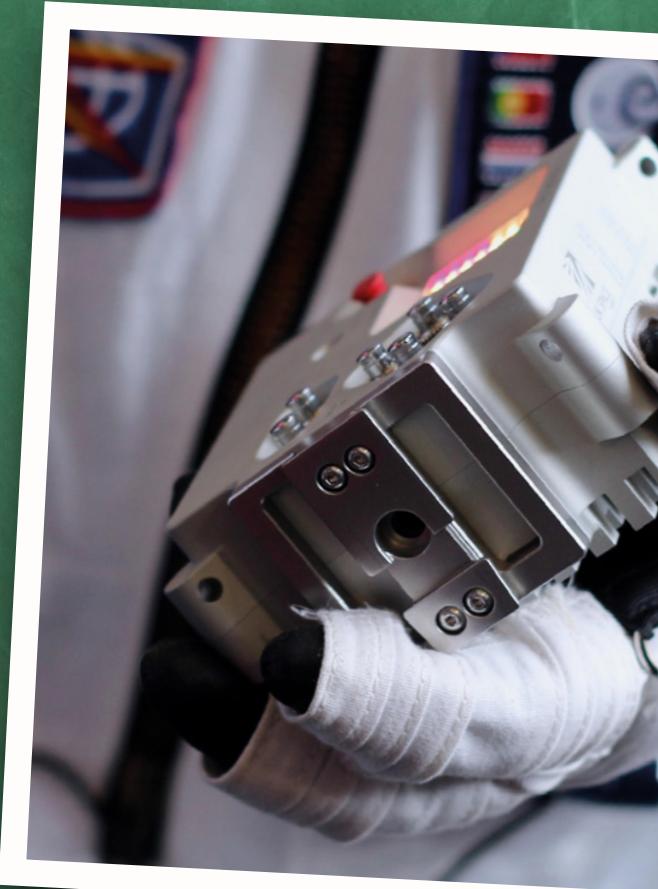
The European Astro Pi Challenge, run by the European Space Agency in collaboration with the Raspberry Pi Foundation, is a fantastic way to get kids using Raspberry Pi in a real – but out of this world – setting. There are two missions:

Mission Zero (rpf.io/mo) challenges teams of young people up to 14-years-old to write code that runs on the two Raspberry Pi computers in orbit on the International Space Station, displaying a message to ISS astronauts along with the ISS's current ambient temperature. Successful space coders will get a unique certificate showing the exact position of the ISS when their code ran.

Mission Space Lab runs over an eight-month period and asks teams of young people up to 19-years-old to design an experiment that will run on the ISS, and to analyse the findings. This year the Astro Pi devices in orbit will be upgraded to Raspbian Buster, which supports machine learning libraries like TensorFlow. Last year, two CoderDojo teams successfully used their experiment time to

Physical Computing with Raspberry Pi and Python

The free Physical Computing course run by FutureLearn (magpi.cc/QahqhC) is an ideal introduction to creating systems you and your students can use to control objects around you. Use Python to control circuits, LEDs, and buttons, then write a game applying your skills.



▲ An Astro Pi module being held by an ISS astronaut
Image credit: Alasdair Allan @aallan - Babilim Light Industries

"Write code that runs on the two Raspberry Pi devices in orbit on the ISS"

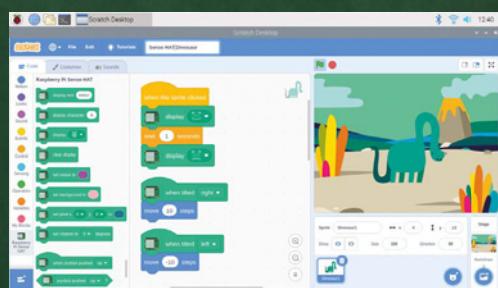
look for evidence of wildfires and even captured a photo of the Soyuz rocket launch. Quite a coup!

Schools, Code Clubs, and CoderDojos that want a shot at being selected must sign up and submit their initial idea by 25 October. They then have until 14 February 2020 to write and test the code for the experiment. Selected teams are announced at the end of February, with experiments being deployed throughout April. Analysis of experiment findings takes place in May and the winning Mission Space Lab experiments are announced in June, along with some rather special prizes.

Both missions launch on 12 September. For more information on how to enter, visit astro-pi.org.



Teaching Programming in Primary Schools



Scratch provides a gentle introduction to computer programming for younger pupils and primary teachers with little-to-no experience of computer science. This free introductory online course (magpi.cc/HhXCAT) shows how Scratch's art-, music-, and games-based approach makes Raspberry Pi programming fun and guides you through any potential obstacles you and your pupils may encounter.



► Luca Parmitano on board the ISS with an Astro Pi unit
Image credit: ESA

▼ Running a Code Club is great fun, for adults as well as pupils



Prepare to Run a Code Club

Code Clubs are aimed at 9- to 13-year-olds, so they're perfect for schools. They are designed to spark curiosity and inspire creativity. You don't need coding knowledge yourself, but you and your pupils can learn as you go using the step-by-step project guides at magpi.cc/XENsJj. You may also like to invite volunteers to help you run your Code Club. Older pupils can make great mentors, but there's also a volunteer call-out option when you register your school for Code Club at codeclub.org.

RockyBorg

SPECS

DIMENSIONS:
16×24×13 cm
(W×L×H)

WHEELS:
70 cm front,
65 cm rear

SPEED:
Approx. 1m/s

BATTERY:
Up to four
hours run time
from eight AAA
rechargeable
batteries

COMPATIBILITY:
Reversible
upper chassis
plate to
accommodate
Raspberry Pi 2,
3, 3B, 3B+, and
all Raspberry Pi
Zero models

► PiBorg ► magpi.cc/dfcqRi ► £99 / \$135

PiBorg offers three-wheels of madcap mayhem with its latest robot kit. By **Lucy Hattersley**

RockyBorg is a slightly madcap racing robot from PiBorg, the geniuses behind Formula Pi (Raspberry Pi's premier AI racing competition). This latest creation follows in the tyre tracks of MonsterBorg and DiddyBorg, but with a significant reduction in price and complexity, and one less wheel. Indeed. This is the first three-wheeler we've tested at Raspberry Pi towers. And we're entirely smitten.

Most of the robots we encounter have steerable front wheels, or a tank-like differential steering (where the speed or direction of the left and right motors are adjusted to turn). RockyBorg does things differently. Two 180 rpm motors on the rear provide forward momentum, while a servo tilts the whole body of the robot (including an affixed front wheel) to change direction. The result is

► Dual DC motors provide forward propulsion, while a servo motor tilts the body to provide steering

a trike that tilts into corners. Everybody we've shown RockyBorg to has loved it.

The on-board Camera Module (not supplied) tilts with the body, leading to some great race footage. This YouTube video shows RockyBorg in all its epic action (magpi.cc/wceWXb).

RockyBorg is also good value at just £99. This includes the acrylic parts, servo, and two motors, plus the new custom RockyBorg motor controller. You also get the on/off switch and battery compartment (but need to supply your own AAA batteries).

The supplied motor controller is similar to the PiBorg's ZeroBorg, with support for the two direct motors and a single servo.

You need to bring your own Raspberry Pi to the party, plus a Camera Module if you want to add vision to RockyBorg. It's not a complete kit, but you can use it with most Raspberry Pi boards.

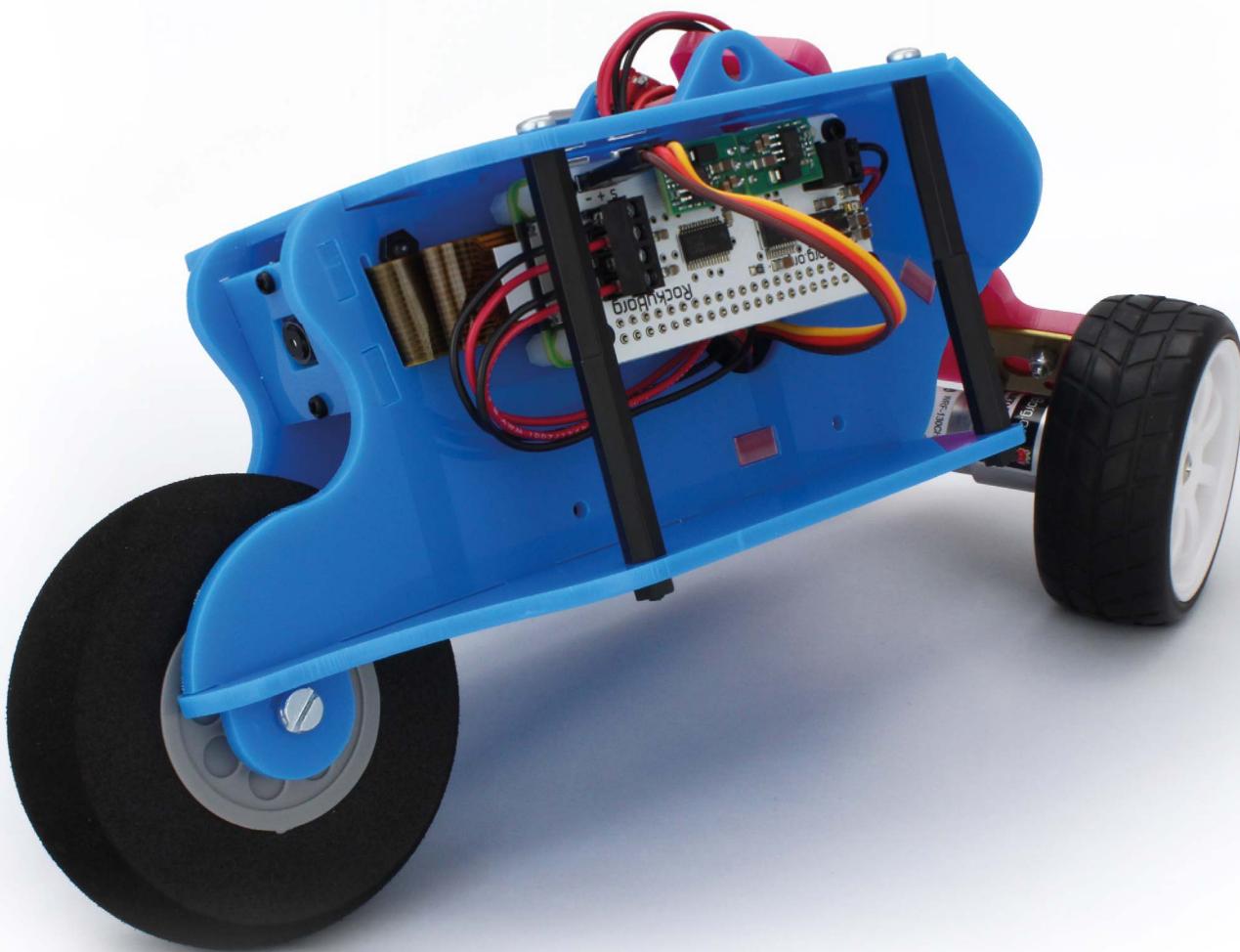
Ours was built with Raspberry Pi Zero WH, but we've also tested one with Raspberry Pi 3A+. All recent models of Raspberry Pi can be mounted to the side, and PiBorg is developing an add-on for Raspberry Pi 4 that adapts the ports on the newer board and adds a 3A power supply.

► It's not a complete kit, but you can use it with most Raspberry Pi boards ►

Building the RockyBorg

RockyBorg is a clever design. It features two vertical plates around the front wheel, and a top and bottom plate (both made of acrylic). These clip together to box in the battery compartment, keeping it secure and adding stability to the centre of the robot.



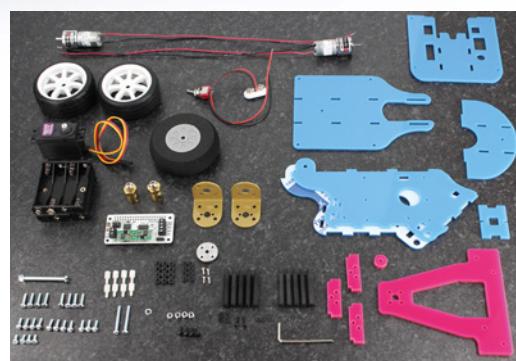


Your Raspberry Pi board sits attached to the left plate, and the RockyBorg motor controller fits on top of Raspberry Pi. The wires from the motors feed through the back plate, through a hole in the top, and out another hole in the left.

It is a fiddly thing. Take a look at the RockyBorg build instructions in order to see what's in store (magpi.cc/pTiCmO). It's not a particularly difficult build, but be sure to follow the instructions carefully. There's a lot of looping of wires and cables that needs to be done with precision.

Once the robot is built, the wheels must be calibrated. PiBorg has a calibration guide (magpi.cc/vGBtCQ) which enables you to test each motor is working correctly and set the default return position for the robot.

Then it's on to the software installation (magpi.cc/mOmMXC), which pulls code from PiBorg's GitHub repo (magpi.cc/hGtKhg). The code enables remote control of RockyBorg with a Bluetooth controller (we tested it with a PS4 gamepad). And much fun is to be found jamming



the RockyBorg around. RockyBorg also responds to a web interface, which provides camera feedback.

From there, you move on to the API interface. It's still a work in progress, and we've recently seen OpenCV support added, so hopefully you'll be able to use it to test out self-driving in the future. How easy this will be with the tilting camera will be interesting, though. Our experience with remote control and Python testing was fabulous. ■

All the parts included with the RockyBorg build. You need to supply your own Raspberry Pi board and Camera Module

Verdict

We're letting some of the fiddly build complexity slide because we love playing with the end result so much. And, it's great value! RockyBorg is our favourite RC robot to date.

9/10

SPECS**MOTOR**

CONTROLLER:
DRV8833

**BUILT-IN
SENSORS:**

2 × TCRT5000 IR
obstacle,
4 × SFH3710 light,
2 × TCRT5000 IR
line follower,
2 × optical wheel

LIGHTS:

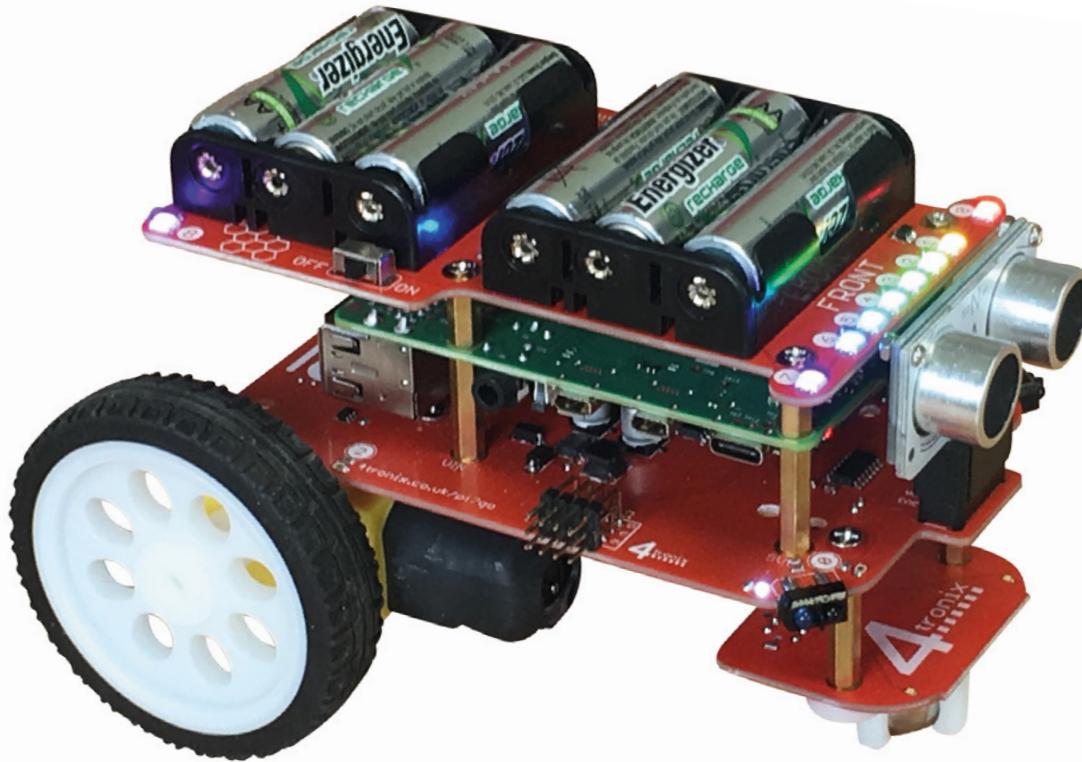
10 × SK6812-3535
smart RGB LEDs

POWER:

6 × AA (or 7 × AA
or 2 × 18650)

EXPANSION:

2 × Breakout
board slots,
4 × SVG servo
connectors,
4WD option



Pi2Go Mk2

The robot's chassis comprises the main and battery boards, with Raspberry Pi in the middle, secured with hex pillars

Car manufacturers regularly launch redesigned versions of popular models, so why not robot makers? 4tronix's Pi2Go Mk2 is a souped-up reimaging of its original two-wheeled flagship robot that was launched back in 2015.

Plentiful improvements include new, thinner wheels with high-grip tyres and optical encoders, a better motor controller that offers higher speeds and near-instant braking, twin breakout edge connectors for add-ons, and even a four-wheel drive option.

Assembly line

As with the original model, it comes in kit form with no soldering required. Assembly takes around an hour, using an online step-by-step guide (magpi.cc/YHZWiZ) and the supplied screwdriver and spanner, although the variety of different-

sized screws and hex pillars is a little bewildering, particularly as we had four mysterious tiny screws left at the end.

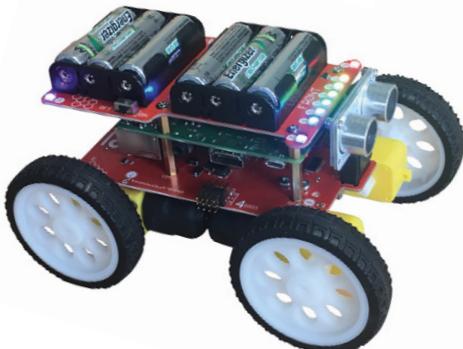
In the standard kit, you get everything needed bar a 40-pin Raspberry Pi (A, A+, B, or B+), microSD card, and AA batteries (rechargeables recommended). A 6 × AA battery board is included as standard, although you may well want to swap this out for a 7 × AA version (or 2 × 18650) to power a Raspberry Pi 4. Either way, the battery board – which sits on the top of the robot – also features ten individually addressable RGB LED blinkenlights.

Located under the Raspberry Pi, the main board is based on the reliable DRV8833 dual H-bridge motor driver. The board is packed with features, including four analogue light sensors (one on each corner), two infrared obstacle sensors on the front corners, and two breakout slots. Located on

► 4tronix ► magpi.cc/NyXcDx ► £72 / \$87

4tronix's flagship robot gets a major redesign and extra features. By **Phil King**

4WD pack



If you prefer four wheels, a 4WD pack is available (for £11.40) to transform the Pi2Go into a four-wheel drive vehicle. Removing the castor board, the extra wheels and motors are secured with mounts and hex pillars. The motors are then wired into two spare sockets at the front of the main board. The only downside is that the extra wheels obscure the front IR obstacle sensors, while the castor board's line sensors are also lost.



the front and side, these slots are used to add a supplied ultrasonic distance sensor and four-digit, seven-segment display – by default, for showing your Raspberry Pi's IP address to help you access it via SSH or VNC.

The breakout slots are also compatible with Pimoroni's Breakout Garden boards, so you could add all sorts of extra functionality. In addition, the main board features a 12-pin connector – wired to 5V power, ground, and GPIO pins – to add up to four servos.

“The robot's chassis is held together with sturdy hex pillars which double as power routers”

With Raspberry Pi sandwiched in the middle, the robot's chassis is held together with sturdy hex pillars which double as power routers, also powering the two line-follower sensors on the bottom of the small castor board. This eliminates the need for fiddly, messy wiring.

Special mention must go to the new frame-like metal motor mounts which clamp the standard yellow motors securely to the underside of the main board, where there are two optical sensors to gauge (via plastic encoder wheels) the rotational speed/position of each wheel at 20 pulses per second.

Start running

With everything assembled, a flick of the switch powers everything up, including your Raspberry Pi. After SSHing in and ensuring SPI and I^C are enabled, the software is installed with a trio of Terminal commands (magpi.cc/DCgioK). The Python library comes with several examples to showcase the various features of the Pi2Go Mk2, such as LEDs, ultrasonic sensor, and IP display. The motor test code enables you to control the robot's movement with key presses – it's pretty nippy at top speed and can turn quickly on the spot. The step test program allows more precise, methodical control of movement, making use of the rotary encoders, as does the line-following code.

You'll need to combine bits of example code for autonomous movement and suchlike. The robot may also be programmed using ScratchGPIO. **M**

▲ The new wheels are thinner and grippier and feature encoders used with optical sensors on underside of the main board

Verdict

A major improvement on the original robot, which was already very good, the Pi2Go Mk2 is absolutely packed with features and sensors, plus options for expanding its functionality. Non-solder assembly is simple and it's great value.

9 /10

LibreELEC 9.1 Beta

SPECS

KODI VERSION:
18.3 'Leia'

COMPATIBILITY:
All Raspberry Pi versions

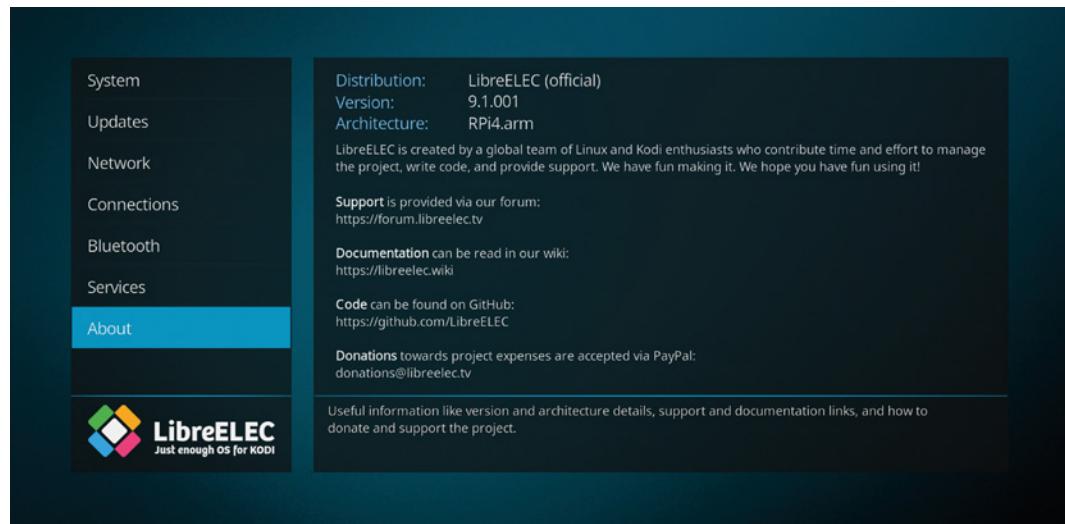
INSTALLATION:
Graphical wizard or burnable image

IMAGE SIZE:
129MB

► 4K works fine for now, but will get better

► LibreELEC ► libreelec.tv ► Free

LibreELEC comes to Raspberry Pi 4. **Rob Zwetsloot** tests it out to see if it's ready for prime time yet



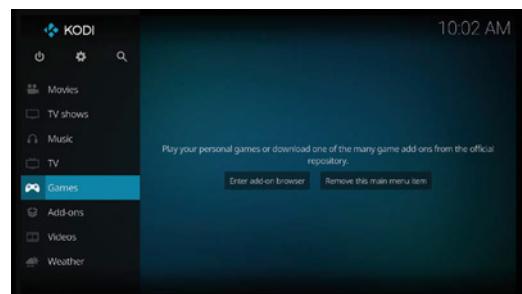
You could easily argue that Raspberry Pi was already the perfect media PC – why get excited for a Raspberry Pi 4-compatible version of LibreELEC/Kodi, then? One of the many improvements to Raspberry Pi 4 was the inclusion of hardware H.265 decoding – also known as HEVC. This means it can support 10-bit colour, as well as the 4K output of the new Raspberry Pi 4.

And we can say right off the bat that with H.265 encoded media, it's noticeable immediately. Colour tearing and glitches are completely gone, with media running at full speed without it seeming to struggle. 4K works fine as well – not great, just fine – at 30Hz.

Size matters?

That's not the only reason we decided to do a quick re-review of LibreELEC, though – there's also the question of what version of Raspberry Pi 4 do you need. Could you save \$20 and get the 1GB version over the 4GB, or is there a difference?

Yes, there is a difference, but no it's not worth the \$20 (for this purpose) in our opinion. After



▲ The video game part has a way to go

some extensive testing with the three versions, playback was the exact same, even with the most demanding bit rates. The UI, however, was noticeably just a little bit more responsive on the higher gigabyte models. Just the slightest bit, though – nothing to splurge on. At the very least, the 1GB model is a bit better than previous Raspberry Pi computers, in our experience.

So, to recap: you don't even need to buy the best version of Raspberry Pi 4 to get the major benefits of the hardware redesign. That's a huge win. ■

Verdict

LibreELEC 9.x still needs some work upstream. However, the beta is well worth jumping on board if you have a Raspberry Pi 4 and the need to use the extra power.

9/10



CoderDojo

Can I start a CoderDojo club in my local area?



CoderDojo is a global network of free, volunteer-led, project-based programming clubs for children aged 7–17. Dojos are championed by individuals all around the world who are passionate about giving young people the opportunity to learn to code.

Starting a Dojo is a fun and incredibly rewarding experience

You don't need to possess technical skills to start a Dojo. The most important attribute is that you can bring people together for a shared goal.

We're ready to support you by providing:

- **Learning resources and guides**
- **A free event management system**
- **Certificate templates, digital badges, and giveaways**

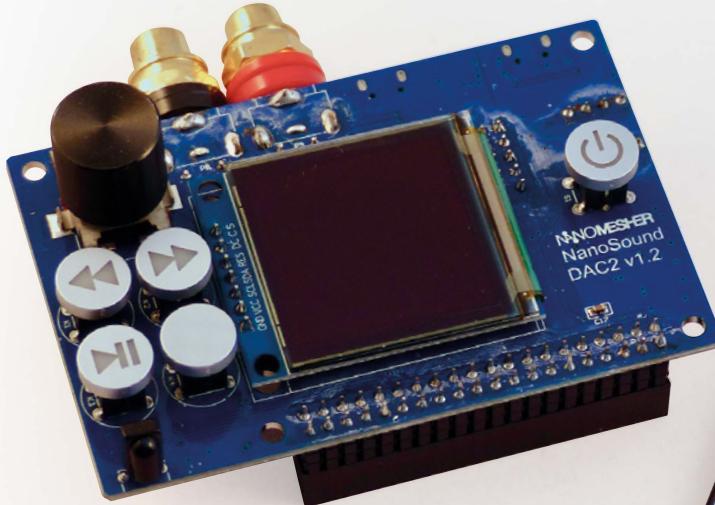


"I started a Dojo to give my kids a place to meet other children also interested in programming and making games. I get to see them making new friends, learning from one other, and they loved it. Realising how I had created such a wonderful place for children has ignited a spark in me."

- Maroes, CoderDojo NL

Start your own club. Join us at [CoderDojo.com](https://www.coderdojo.com)

SPECS

FORM FACTOR:
HAT**AUDIO OUT:**
2 x RCA/phono
sockets**DISPLAY:**
1.5-inch colour
OLED**SOFTWARE:**
Volumio or
XBMC**CONTROLS:**
5 x GPIO
buttons, plus a
rotary encoder

NanoSound DAC 2 Pro

► Nanomesher ► magpi.cc/TOGZoV ► £73 / \$88

Raspberry Pi-powered audiophile DACs are big business. Is there room for another? **PJ Evans** grabs his speakers

The original NanoSound DAC, a highly rated Kickstarter audiophile HAT, has now been superseded by the NanoSound DAC 2. It comes in three variations; we tested the Pro version.

The HAT contains the all-important DAC, but also a range of control buttons, rotary encoder for volume, OLED display, and IR remote control. There's no case, but you can either buy a 3D-printed one for £35 or print your own. Just add an amplifier and speakers.

Speed limits

We tested with a Raspberry Pi 3B+. The DAC 2 is supplied with a microSD card loaded with many audiophiles' favourite playback software, Volumio. Although the sound was very good indeed, the control interface was sluggish and crashed repeatedly. The HAT's controls worked intermittently and the screen would sometimes freeze. Frustrated, we swapped out the Raspberry Pi 3 for a 4. The result was transformative: the website was slick and the controls responsive.

Verdict

The screen is gimmicky and the software could be easier. Regardless, the audio quality and performance when used with Raspberry Pi 4 has earned NanoSound DAC 2 Pro our approval.

7 / 10



▲ You can buy or 3D-print your own box to make it ready for the living room

The little OLED screen is cute, displaying album art and playback information, but it's so small as to wonder what the point is, especially as the only way to construct playlists is by using the web app (which worked well on mobile). The playback buttons and volume control were more useful and worked as expected.

Concerns over the form factor are unimportant if the sound is good and, oh, it is very good indeed: rich, deep sound, full of detail. There's no denying the build quality and thought that's gone into making this a superior way of listening to music. ■

HackSpace

TECHNOLOGY IN YOUR HANDS

THE NEW MAGAZINE
FOR THE MODERN MAKER



ISSUE #22
OUT NOW

hsmag.cc



10 Best:

Raspberry Pi media players

Here's how to play music and video on your Raspberry Pi

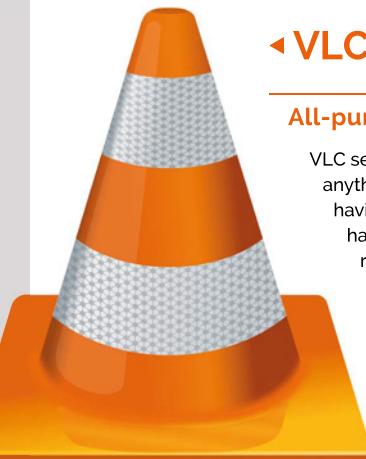
Raspberry Pi has been used to play video and music since the very first board landed in the hands of an enthusiast. Its use as a media platform is legendary, and for good reason. Want to take advantage of Raspberry Pi's media powers? Here are ten excellent ways... 

◀ VLC Media Player

All-purpose player

VLC seems to be able to play just about anything you throw at it, while still having a very small install size. It also has great streaming abilities, and can receive web streams and network streams as well. It also has a GUI.

magpi.cc/qsXYHH



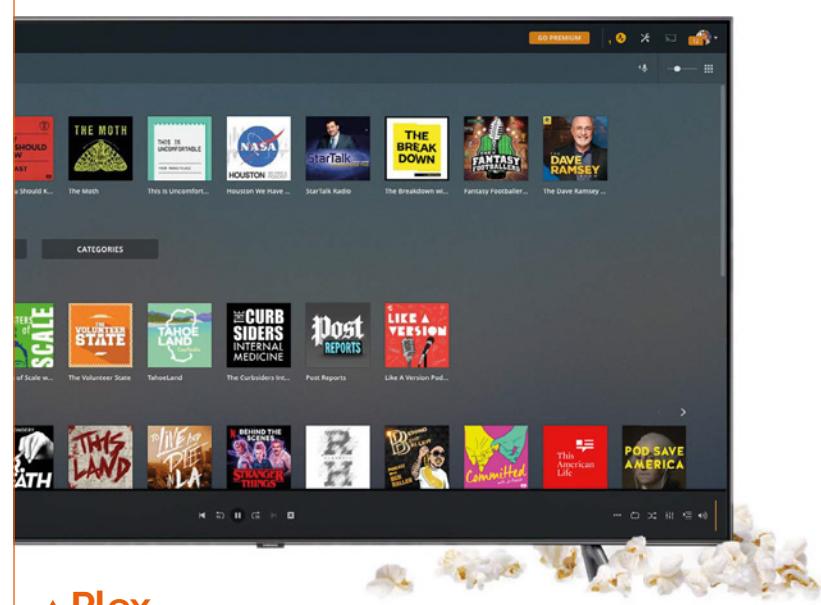
► MPlayer

The legend

MPlayer is still adored by many long-term Linux and open-source software users, and for good reason: it's great. You'll probably have to compile it from source to get a version you prefer, though.



mplayerhq.hu



▲ Plex

Ultimate media library

Need a tiny media server that will stream to all your devices that support Plex? Using Raspberry Pi you can create the ultimate streaming media server, with the tiniest footprint in both physical space and on your electricity bill.

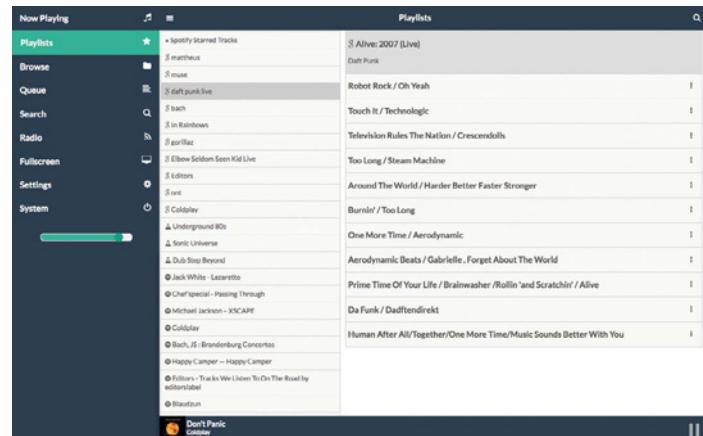
plex.tv

▼ Pi MusicBox

Music streaming software

Another excellent streaming solution for your home. As well as being able to connect to your local music collection, it connects well to online services such as Spotify and SoundCloud.

pimusicbox.com



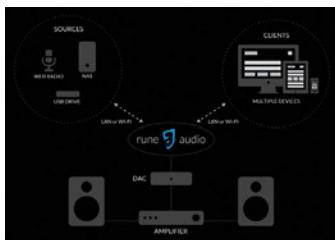


▲ OSMC

Raspberry Pi HTPC

A Kodi alternative for people who like slightly more modern user interfaces. It has a special installer that can get a Raspberry Pi microSD card set up in minutes, and it also supports streaming to other devices.

osmc.tv



▲ RuneAudio

In-home music system

RuneAudio is a complete home hi-fi system, connecting to local network storage and internet streams. It can be controlled by multiple clients if run headless, and plays music perfectly through a DAC. It's also open-source.

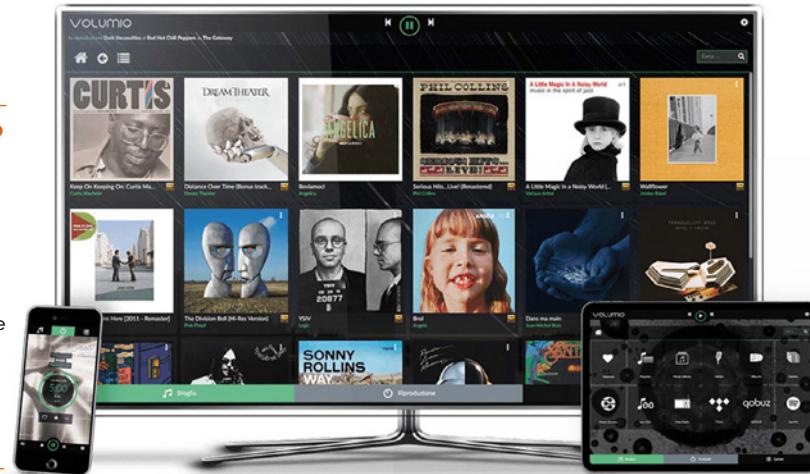
runeaudio.com

► Volumio

All-in-one audio

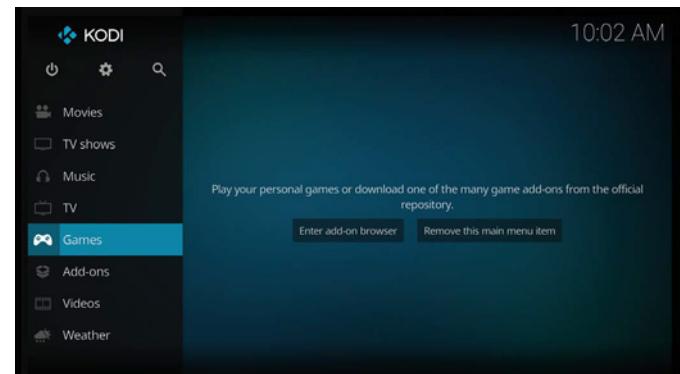
Volumio wants you to have everything connected to one device, and has created a system to allow you to do that. Not only can you store music on it, you can stream music from other places to it.

volumio.org



KODI

Kodi is software used in many home-theatre PCs and other kinds of media PCs. You can install it on Raspbian on Raspberry Pi; however, we find it works a lot better if you're using it on a dedicated media device, especially in conjunction with operating systems that make use of it.



▲ LibreELEC

The purest Kodi

Our favourite Kodi-based OS gives you the pure, familiar, highly customisable Kodi experience. It can play just about anything you can throw at it, including a lot of video games now. Read more about its performance on Raspberry Pi 4 on page 76.

libreelec.tv

OMXPlayer

Command-Line player

This is great for having media play when you turn a Raspberry Pi on, or to control it from an SSH-connected remote computer. It makes use of minimal resources, while also being hardware-accelerated for great picture quality.

magpi.cc/ABdFRJ

Learn web design with Raspberry Pi

Building webpages is a great introduction to computing and a vital skill that every coder should know. By **Lucy Hattersley**

CoderDojo: Build Your Own Website

AUTHOR Clyde Hatter
Price: £10/\$£12
magpi.cc/fZxvFO

CoderDojo is a global movement of free, open, volunteer-led coding clubs (Dojos) where young people aged 7–17 (Ninjas) can explore digital technology with the support of their fellow Ninjas and volunteer mentors.

CoderDojo has plenty of web resources, which are now available on the Raspberry

Pi projects website. These include HTML & CSS Sushi Cards (magpi.cc/sPtNJN)

CoderDojo was also behind a rather great book called *Create With <Code>: Build Your Own Website*. This colourful book is aimed at youngsters and covers a wide range of HTML and CSS, plus even some JavaScript. 

Websites

Bookmark these webpages



CSS
I have been figuring out CSS for 20 years. Every time I figure something out, I write it down. This has grown into quite a collection of work. You can find some of it here.

by Rachel Andrew

BOX SET

RACHEL ANDREW

Rachel has been designing websites for years and has one of the best collections of CSS resources and articles around. Make sure you bookmark Rachel's website.
rachelandrew.co.uk

HTML 5 DOCTOR

There are great articles, but it's the element index you'll want to

bookmark. These offer a guide to any web elements you'll come across.

html5doctor.com

HTML CHEAT SHEET

There are lots of cheat sheets out there, but we like this one because it has live previews of the code elements.

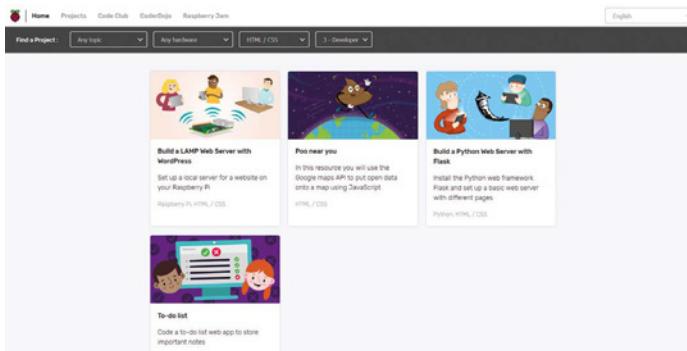
magpi.cc/UGuMBu

Raspberry Pi

AUTHOR
Raspberry Pi Foundation

Price:
Free

rpf.io/projects



Most Raspberry Pi projects have a physical element to them, but many also incorporate web technologies. These projects can help boost your understanding of web development.

So, visit the Raspberry Pi projects ‘Find a project’ page, (magpi.cc/ddWFqK) and click the ‘Any software’ drop-down menu. Change this option to ‘HTML / CSS’ and you’ll find

a range of neat web-based projects. Projects range from super-simple, such as building an online birthday card, up to more advanced, like ‘Build a Python Web Server with Flask’ or ‘Build a LAMP Web Server with WordPress’. Click the ‘Any level’ drop-down menu and choose between levels 1 and 4 to find projects that match your abilities. 

Raspberry Pi Full Stack Raspbian

AUTHOR
Peter Dalmaris

Price:
£125 / \$150

magpi.cc/reAwam

Udemy has a range of video courses designed to teach all aspects of computing. Its Full Stack Raspbian course teaches you how to install Raspbian Lite and work in headless mode to set up a stack of technology for IoT (Internet of Things) deployment. It’s a great way to learn about the technologies sitting behind the front end of HTML and CSS, and use the unique computing features offered by Raspberry Pi (such as GPIO and attaching sensors).



It covers a range of handy technologies, such as Flask, SQLite, and jQuery. While nothing is covered in complete depth, there’s enough knowledge here to get you beyond HTML and CSS and into the back end.

As with all Udemy courses, don’t be put off by the price. It’s frequently on sale (and is currently £14.29 at the time of writing). Just wait for a sale to come around. 

Free online courses

These online courses are free and highly thought of



W3SCHOOLS

This educational website has tutorials and reference files for HTML, CSS, JavaScript, and all the other web technologies you will need.

w3schools.com

CODEACADEMY

Codecademy turns up a lot in learning computer skills. Its ‘Introduction to Web Development’ course is highly thought of.

codecademy.com

HTML AND CSS FOR BEGINNERS

If Udemy’s Full Stack Raspbian seems too much, then consider its free HTML and CSS course. It’s a short six-hour course with a couple of coding exercises.

magpi.cc/MVMQxw



Denise Leonard

Denise is a teacher on special assignment to help kids in the alternative school system. We talk to her about how Raspberry Pi is helping

► Day job **Teacher** | ► Community role **Educator** | ► Twitter **@deniseleonard**

Denise Leonard isn't just a teacher – she's a 'teacher on special assignment' for some schools in San Jacinto, California.

"I am a teacher, but I am not in the classroom full-time," Denise explains. "I conduct the enrolment activities for [one of my schools], read transcripts making sure students are on track to graduate, and, most importantly, help motivate students to complete their credits and come to school."

▼ The Sense HAT is shown off at one of the many events held

Apparently not content with all this work, she also runs after-school coding classes that teach computing with a Raspberry Pi, and is also a Raspberry Pi Certified Educator, having completed a Picademy course in the US.

"Over the past three years I feel that my students have benefited greatly from the addition of my Raspberry Pi classes," Denise tells us. "While exposure to [digital making] was the goal, the most significant success has been an increase in student connectedness to school. Motivated by the projects and materials available to them as part of the Raspberry Pi course, I witnessed increased desire by many of my students to attend school each day. The more often they attend school, the more connected they feel, which [has] positively affected their credit production, attendance, and behaviour."

What kind of classes do you teach?

Currently, I am teaching computer science sections to both the alternative school and the independent study school students. I teach both an



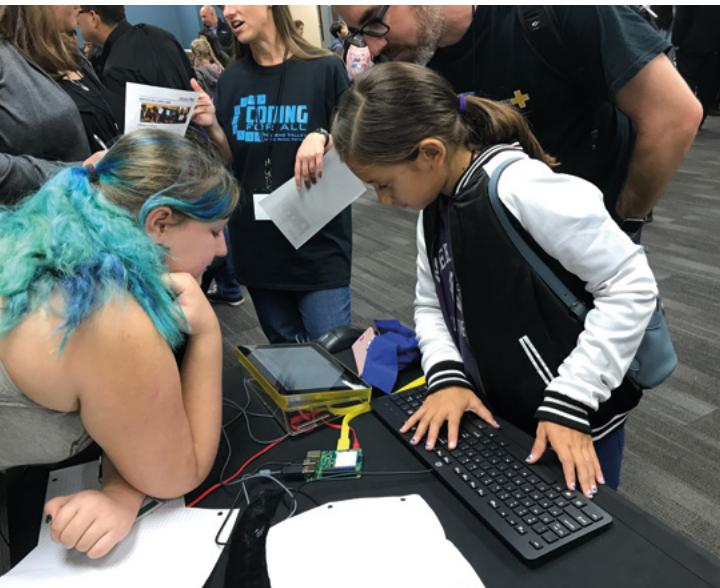
▲ Playing with simple circuits at the Riverside Community Summit

advanced and beginner course using Raspberry Pi computers. I also am a big makerspace advocate. I spend time helping to ensure our Library Media Technician has the Raspberry Pi Projects website available for students who want to code using Raspberry Pi computers during makerspace time.

What happens in your computing classes?

Instructional experiences include construction and use of various equipment in the computer innovation classroom, access to the makerspace, practice in physical computing with Raspberry Pi and other microcontrollers, along with coding music with Sonic Pi.

All students research and complete engaging activities, which are all personalised





▲ Kids show off their projects at a CSforALL event

with their specific learning needs in mind. This ability to personalise and allow student choice in creating projects is largely due to the Raspberry Pi Foundation's tremendous wealth of school-friendly resources. They are accessible and easily adaptable, even for educators and students with little to no programming experience.

When did you learn about Raspberry Pi?

I believe I first heard of Raspberry Pi in 2012 during a discussion with my husband, an IT professional, who had read about them and was excited at how much they could do for so little cost. We proceeded to talk through ideas and dream up ways that such a powerful, inexpensive computer could be used as an introduction to programming for those that didn't have access to much bigger and more expensive devices. It wasn't long before we had one in our home and we

began to look at some of the tools that it made available.

The scripting language Python, which comes pre-installed with [Raspberry] Pi's Raspbian operating system and the Scratch application made programming easy to visualise and learn. It seemed that in a very short time there were lots of new articles about Raspberry Pi, with new websites and magazines available all the time. Even as our own enthusiasm grew, I was excited to see the adoption of [Raspberry] Pi and how it was introducing so many to the world of programming: boys and girls of all ages, all over the world. ■

"I am beyond grateful to San Jacinto Unified School District for supporting Mountain View High School and Mountain Heights Academy in the pursuit of educating students in computer coding and related activities and careers, with an emphasis on Raspberry Pi computers and the Python computer language."

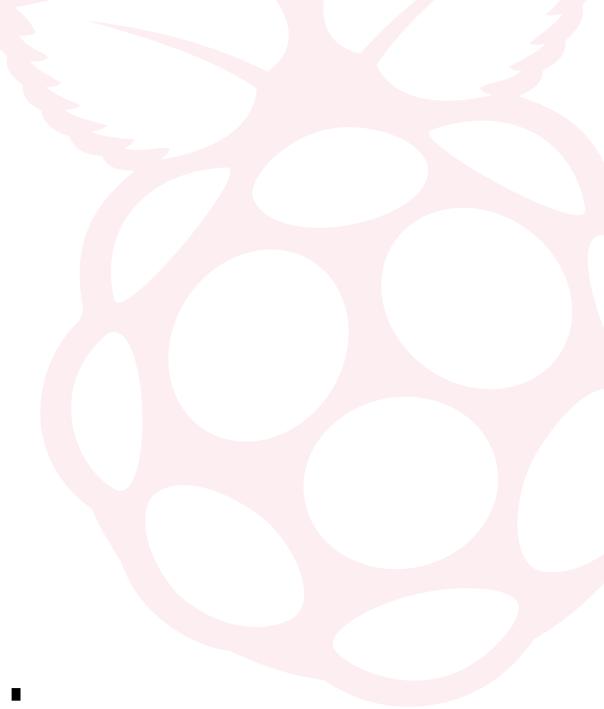
m All students research and complete engaging activities, which are all personalised ■

Aquaponics with Raspberry Pi

"At Mountain View High School / Mountain View Academy, I am working with students to combine our love of computer science and aquaponics. We have tanks with both edible and non-edible fish in our indoors and outdoor aquaponics centres where students care for the aquatic animals, test the water with commercial water chemistry methods, and ensure that there is a proper balance between bacteria and nutrients in the water. Students learn plant classification, daily measuring light with illuminance light meters, charting the light spectrum as well as the plant growth, water PH, ammonia, nitrates, nitrate levels, and air temperature.

"Students use Raspberry Pi computers to monitor temperature in the fish aquaria. A Python script running on [Raspberry] Pi sounds an alarm if the water temperature gets too hot or too cold for the fish and sends students and/or teacher a text message indicating the problem. Students have also written code that makes time-lapse videos of plants growing using Raspberry Pi cameras.

"This upcoming school year our goal is to bring Raspberry Pi computers to the outside aquaponics area, allowing students to monitor the plants and fish outdoors."



This Month in Raspberry Pi

MagPi Monday

Amazing projects direct from our Twitter

Every Monday we ask the question: have you made something with a Raspberry Pi over the weekend? Every Monday, our followers send us amazing photos and videos of the things they've made. Here is a small fraction of them. Follow along at the hashtag **#MagPiMonday**.

01. This astronaut seems pretty pleased to be up so high, whatever the altitude limit was!
02. We'll use this opportunity to remind you that you can read all about using Raspberry Pi 4 as a desktop PC earlier in the issue
03. This music box looks great, and has a very special use as well
04. We're not sure you can actually buy this at your local Porsche dealership, but we wish we could
05. Something a bit different, but nonetheless very cool
06. For a first project, we're impressed by the large variety of components!
07. Taking photos of excellent birds? A very worthy use of a Raspberry Pi camera



Bill Harvey
@PiCodeandChips

01

Replies to [@TheMagPi](#)

I organised a Hight Altitude Balloon workshop Thursday / Friday (not quite the weekend but school has finished). Big success. Had to limit the height due to the weather but we did manage a successful flight.



11:13 AM · Jul 29, 2019 · [Twitter for iPhone](#)

02 Ade @Stressed_Local
Replying to @TheMagPi
Nothing flash here..just a desktop replacement 🙌😊

03 Alan O'Donohoe @teknoteacher
Replying to @TheMagPi
At @PrestonRJam, Garry demonstrated a music box he built for a relative suffering with dementia.

0:23 1.5K views

04 Ciprian Manea @ovipic
Replying to @TheMagPi
new red Porsche #PiCar #unPi

1:04 PM · Aug 5, 2019 · Tweetbot for iOS

05 Geir Nøklebye @geirnoklebye
Replying to @TheMagPi
Not so much of a picture, but running OpenSim regions on mono 6 on Raspberry Pi 4.
dayturn.com/viewer/index.php
Maptile of some of the regions

06 Martin Parker @Mr_MartinParker
Replying to @TheMagPi
As it works great I used my CameraPi rather than working on it

07 Martin Parker @Mr_MartinParker · Aug 3
Pictures today taken by my CameraPi project at Lotherton Hall. #RaspberryPi zero w upcycled in an old 110 camera #trueCompactCamera
[Show this thread](#)

08 karim kiswarday @kiswarday
Replying to @TheMagPi
My first raspberry pi project 😊

0:17 1.9K views

10:46 AM · Aug 12, 2019 · Twitter for Android

Video star

What Raspberry Pi and *The MagPi* videos might you have missed this month?

We write a lot here on *The MagPi* – you'd be surprised how many words go into each issue! Sometimes, though, we ditch our keyboard and get in front of a camera to make some great videos for people in the Raspberry Pi community. Here are some of our videos from the past month... **M**

01. Five ways to cool your Raspberry Pi 4

Want to keep your brand new Raspberry Pi 4 cool? We have five excellent ways that you can try out – although we feel perhaps one of them is the clear winner.

magpi.cc/cdSHF

02. We asked our engineers your Raspberry Pi 4 questions...

An hour of questions and answers from the engineering team at Raspberry Pi, including Eben Upton, Gordon Hollingworth, and James Adams. Well worth a watch!

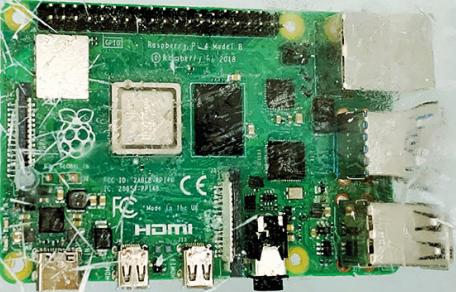
magpi.cc/bKLNGV

03. Build a Raspberry Pi music box ft. Dr Sally Le Page

We covered the full tutorial on how to build this sound machine on page 60, but you can always follow along to the video version hosted by Sally.

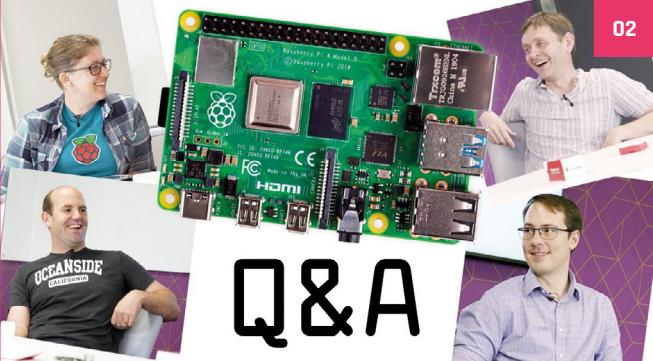
magpi.cc/ivNfZv

5 WAYS TO COOL



RASPBERRY PI 4

01



Q & A

02

BUILD A RASPBERRY PI MUSIC BOX



03

Crowdfund this!

Raspberry Pi projects you can crowdfund this month

CROWDFUNDING A PROJECT?

If you've launched a Raspberry Pi-related project, let us know!

magpi@raspberrypi.org

NovaPi NP01



From the description: "Using the very large and fast Virtex-5, NP01 is currently the most powerful FPGA [HAT] available for direct interfacing to [Raspberry Pi] using the 40-pin GPIO connector – it also works as a standalone FPGA device for independent applications." Sounds pretty excellent, then!

► [kck.st/2ZppESb](https://www.kickstarter.com/projects/270730332/novapi-np01-the-most-powerful-fpga-hat-for-raspbe)

Best of the rest!

Here are some other great things this month

DUNGEONS & DRAGONS MAP SCREEN

This is something we've been looking at making ourselves – it is technically extremely easy, after all, and you can use software like Roll20 to control what's displayed on the screen remotely. Perhaps a future tutorial?

► magpi.cc/aFDFQv

BABY LOGGING

It's a dirty job and someone has to do it. Apparently that's now a Raspberry Pi 2. The simple buttons allow these parents to log what they've done with their newborn. It should make for some interesting graphs.

► magpi.cc/nNSpQs

PiCoolFAN 4

Advanced Active Cooling System & Hardware Real Time Clock

The wealth of features in a cooling HAT

Automatic Active Cooling with PWM FAN speed regulation

DS1307 Hardware RTC with coin battery 1220/25

3 System LEDs indicators (Cold, FAN, HOT)

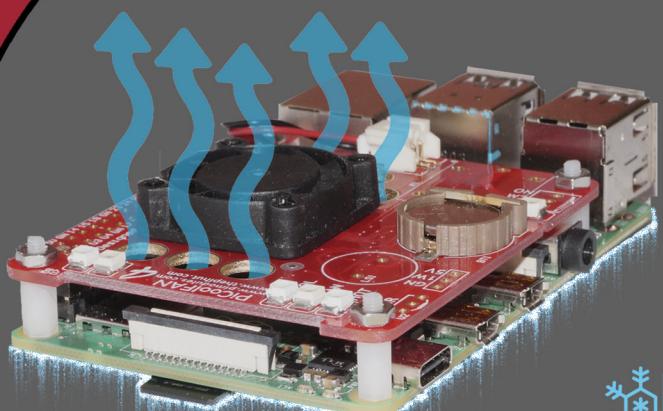
No installation is needed, not using any additional GPIO

Additional Temperature Sensor Placed exactly above CPU

5V/3V 1-wire interface with ESD protection and 2 x User LEDs

Optionally: Infra-Red Receiver, High Current Relay and Buzzer

Full user programmable and CPU core temperature monitor via I2C



AIR CIRCULATION
TECHNOLOGY



Intelligent modules
for your **Raspberry Pi**

ORDER NOW
www.pimodules.com



Raspberry Jam Event Calendar

Find out what community-organised Raspberry Pi-themed events are happening near you...

01. Desert Hack

- Friday 30 August to Sunday 1 September
- 📍 Birla Institute of Technology and Science, Pilani, India
- ▶ magpi.cc/kaprsa

A 54-hour hackathon for entrepreneurs in the field of computing to help brainstorm and create products.

02. KremsPi

- Saturday 7 September to Sunday 8 September
 - 📍 Danube University Krems, Krems, Austria
 - ▶ magpi.cc/QCrPup
- A debut Raspberry Jam at the Danube University Krems, for those who want to learn more about Raspberry Pi.

03. Coding and Physical Computing for Kids

- Saturday 14 September
- 📍 Long Beach Public Library, Long Beach, NY, USA
- ▶ magpi.cc/WpeDgW

A coding and physical computing event where kids and parents can get hands-on practice with Raspberry Pi.

04. Raspberry Jam South Mountain Community College

- Saturday 14 September
- 📍 South Mountain Community College, Phoenix, AZ, USA
- ▶ magpi.cc/dHrjFr

Students can receive hands-on experience on creating things with a Raspberry Pi computer.

05. Seattle Raspberry Jam

- Wednesday 18 September
- 📍 Bellevue Library, Bellevue, WA, USA
- ▶ magpi.cc/NvsBzz

Seattle Raspberry Jam is a free monthly Raspberry Pi club/event/meet-up for beginners to experts.

06. 8th Bognor Regis Raspberry Jam

- Saturday 21 September
- 📍 Bognor Regis Library, Bognor Regis, UK
- ▶ magpi.cc/pYisun

This event is ideal if you have a Raspberry Pi or a micro:bit and are not sure what to do with it.

07. Jamming in Marlborough

- Sunday 22 September
- 📍 Marlborough Town Hall, Marlborough, UK
- ▶ magpi.cc/PpCKPF

Encouraging skills development and expertise in digital systems/computing and embedded software for all age groups.

08. Raspberry Jam Gateshead

- Saturday 28 September
- 📍 Gateshead Central Library, Gateshead, UK
- ▶ magpi.cc/fCvvmB

Try out Raspberry Pi, tinker with fun projects, and see some Raspberry Pi-powered projects in action.

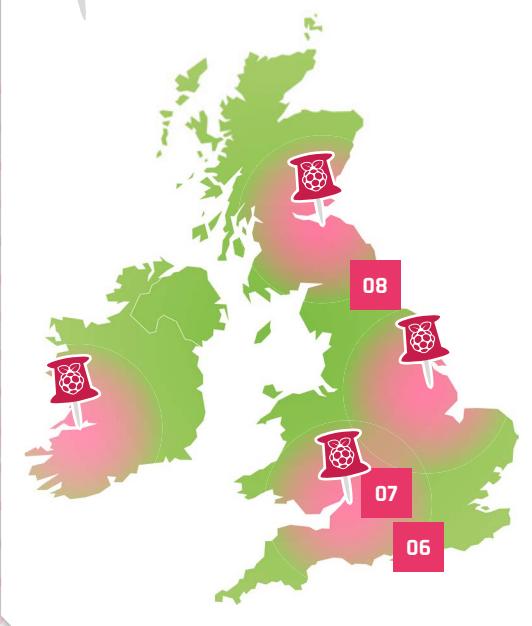
FULL CALENDAR

Get a full list of upcoming events for September and beyond here:

rpf.io/jam



We've highlighted some of the areas in need of a Jam! Can you help out?



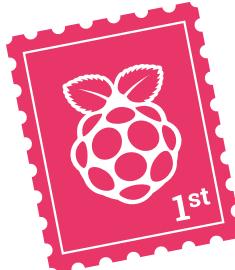
Raspberry Jam advice: Promotion

"I use Twitter and Facebook. I've found it useful to get the message directly to schools who use Twitter, so the IT specialist can pass the message on to parents and children. I think old-fashioned posters put up around the area work well too."

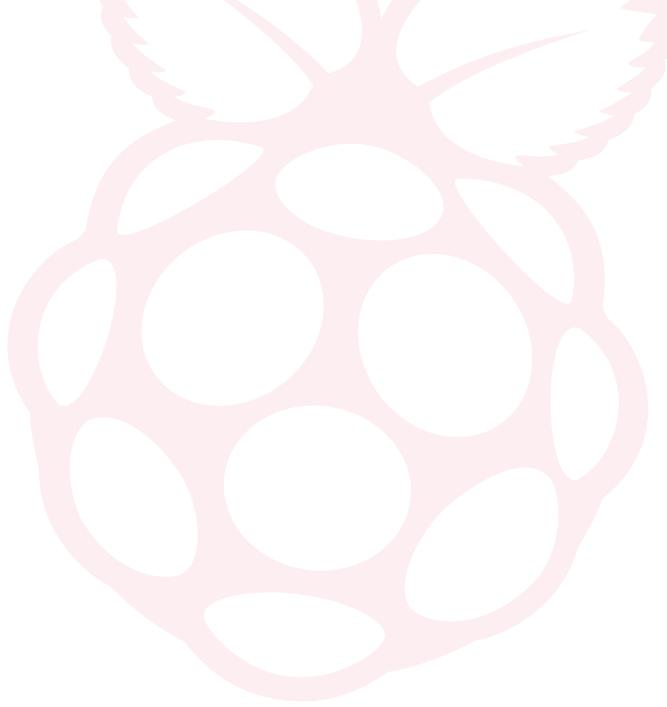
Anne Carlill – York Raspberry Jam

Every Raspberry Jam is entitled to apply for a Jam starter kit, which includes magazine issues, printed worksheets, stickers, flyers, and more. Get the book here: magpi.cc/2q9DHfQ





Your Letters



Der MagPi

Today I bought *The MagPi* magazine at a local store. [...] Unfortunately, I realised that there should be a CD that came with the magazine. I found the place where it should be but it wasn't there, and the CD package was ripped – probably somebody stole the CD. Is it possible for me get this disc with the data of the e-books? That would be very nice!

Falko via email

We assume you're talking about the German version and, unfortunately, we're unable to assist as we don't distribute it. Contact the German distributor at: magpi.de. The German version translates our articles and republishes them via CHIP, who also publish it in France and the Netherlands, so if you like the magazine but English isn't your preferred language, track them down!

Contact us!

- Twitter @TheMagPi
- Facebook magpi.cc/facebook
- Email magpi@raspberrypi.org
- Online raspberrypi.org/forums

Advanced SSH

Of course I use SSH and your article was a nice read, but I have a problem. I have several Raspberry Pi [devices] in my network, all running their jobs and doing great work – well, I think they are.

I live in a place where the power does disappear and have used short-life UPS which covers them mostly, but too often they run down and off goes everything. So far [there hasn't been any data] corruption, but I am concerned.

So if I am around, I log into each [Raspberry] Pi and shut it down – a couple have a press-and-hold button that a Python script detects and shuts it down – but if I am not there, I can't do that.

I have looked and failed to find a way to automate the process except on a [Raspberry] Pi by Pi basis. What I would love to do is use one of the systems to monitor the power; when time's up, remotely log into each [Raspberry] Pi and shut them down safely. I thought of using SSH, but have failed to discover how.

Any ideas, pointers, or have I missed something so obvious?

Jonathan via email

This is not something we've quite tried ourselves; however, sending out SSH commands via Python is something that exists – there's a good thread on it on Stack Overflow: magpi.cc/UicjXk.

With this, you can automate all your Raspberry Pi devices being shut down one by one via a master Raspberry Pi of some kind. With that, you can easily send a shutdown command (we like to use `sudo shutdown -h now`, although there are shorter methods).

Now the trick becomes to keep an eye on your UPS with the master Raspberry Pi. We've seen some HATs that exist which have some power control capabilities, usually for enterprise, but you can probably find one that you could use. Otherwise, check out this page on a Raspberry Pi UPS monitor – it might be a quicker method for you: magpi.cc/qhQuTZ.

We'll probably cover controlling other Raspberry Pi devices with Python and SSH in a future issue, but hopefully these links will help you for now – otherwise, check out the Raspberry Pi Forums, as they're all very smart there: raspberrypi.org/forums.

Back issues in bulk

Is there a place where I can buy all issues from the oldest to the current edition in a single, bulk purchase, instead of going through and adding each to my cart individually on your website?

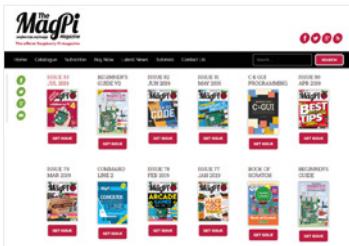
Cohl via Facebook

Unfortunately, due to the way magazines work, we don't keep a full catalogue of all our past issues that you can purchase. The only back issues available can be found on our store (store.rpipress.cc), and it usually only goes back a year or two, with many holes.

Also, a lot of issues of *The MagPi* were never printed. Binders were produced of the first 30 issues; however, there are several issues

from 2015 that were only released as PDFs.

Speaking of which, all the PDFs of our back issues are available for free online, and you are more than welcome to print them off for your own library, as they are supplied under a Creative Commons licence. You can find all the catalogue of back issues here: magpi.cc/issues.



▲ Make sure to check out our catalogue page for any old issues you wish to read as a PDF

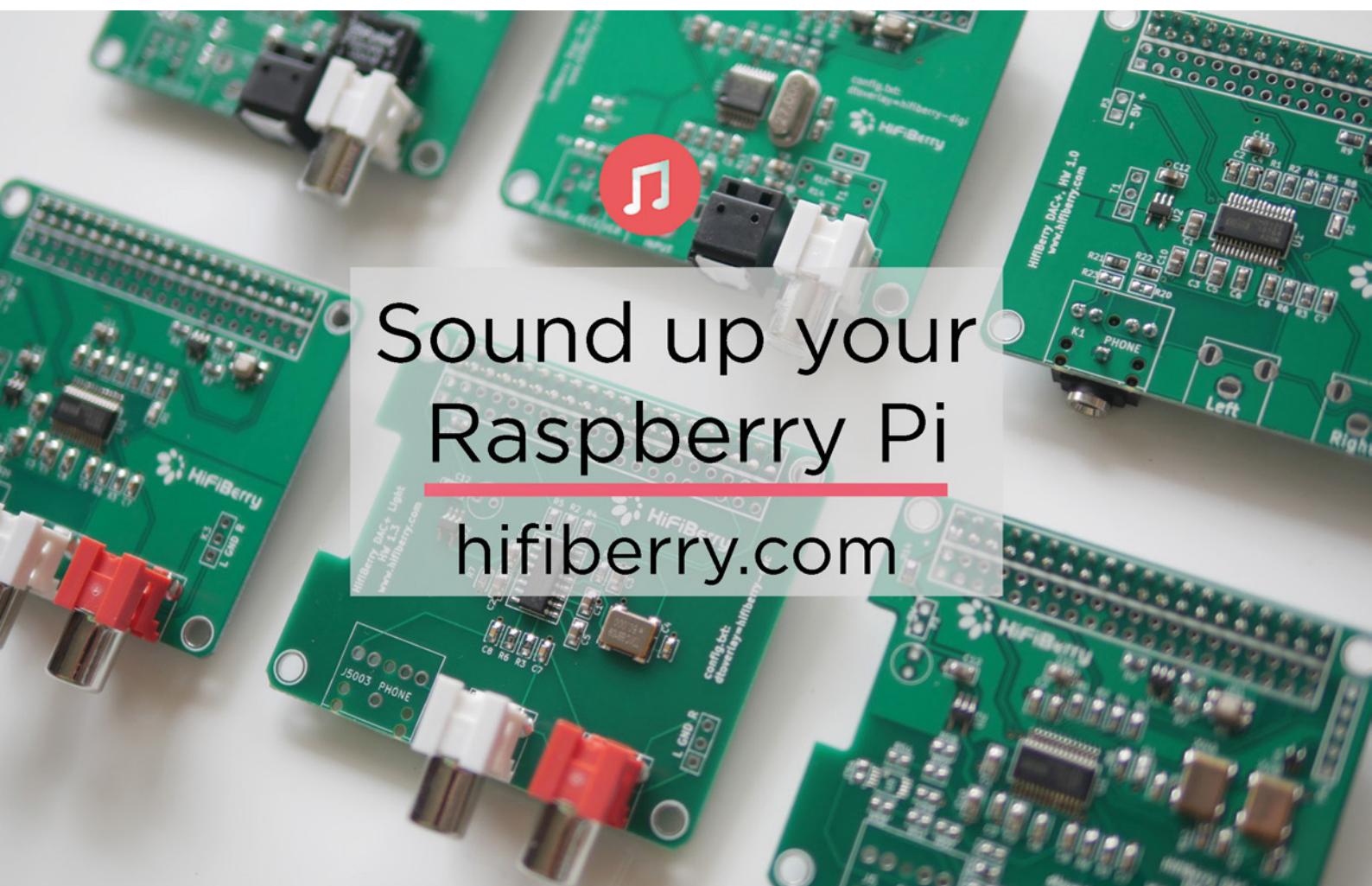
Get the code

I'm working my way through *The MagPi* book about games with Python. It keeps referring to downloads of Python code; any idea where these might be on your site? I keep looking and can't find them.

Pete via email

All of our code lives on our GitHub page (github.com/themagpimag). You can find code from the Essentials books, and a zip file for each magazine issue with the code from tutorials. For the Python book, each snippet of code has a shortlink to the relevant place on GitHub – you can find them at the top of each block of code!

Sound up your
Raspberry Pi
hifiberry.com



(Hello World)

THE MAGAZINE FOR COMPUTING AND
DIGITAL MAKING EDUCATORS

SUBSCRIBE
FREE
IN PRINT AND DIGITAL

FIND US ONLINE:

helloworld.cc

 @HelloWorld_Edu

 fb.com/HelloWorldEduMag



WIN

ONE OF TEN



RASPBERRY PI 4 THREE-PIECE CASES

This great-looking, simple case fully covers a Raspberry Pi 4 while still having gaps to allow for ventilation.

“With a modern design, OKdo’s stylish black ABS plastic case provides protection whilst allowing access to all ports”

– OKdo

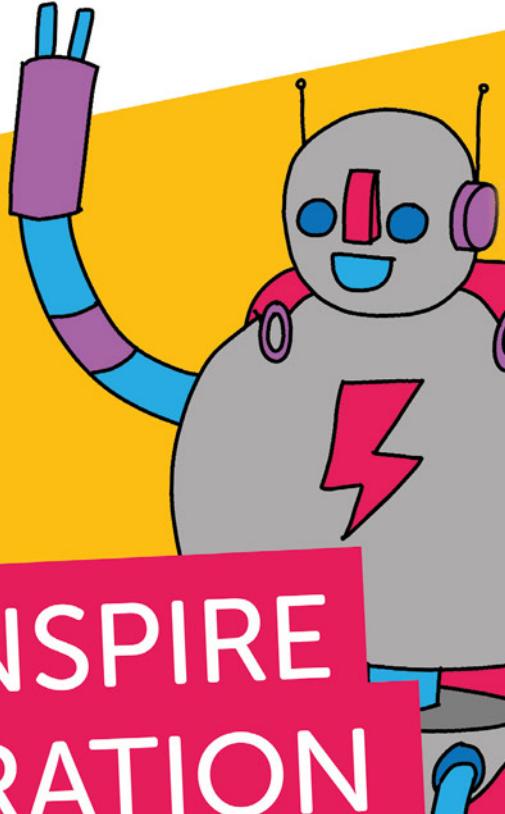
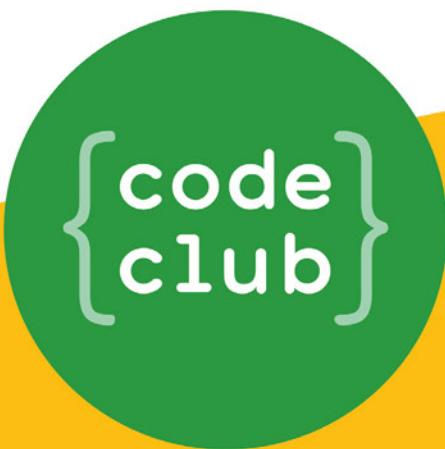
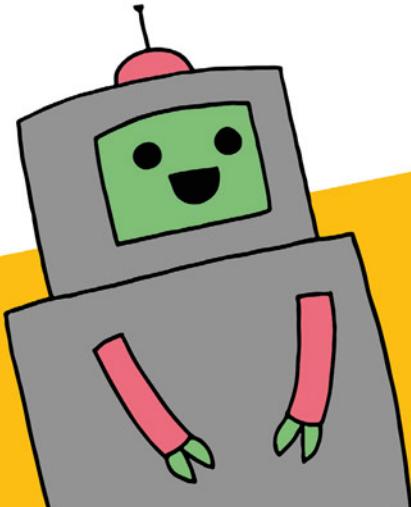


In association with **OKdo™**

Head here to enter: magpi.cc/win | **Learn more:** magpi.cc/sEiUsp

Terms & Conditions

Competition opens on **28 August 2019** and closes on **26 September 2019**. Prize is offered to participants worldwide aged 13 or over, except employees of the Raspberry Pi Foundation, the prize supplier, their families, or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram or Facebook.



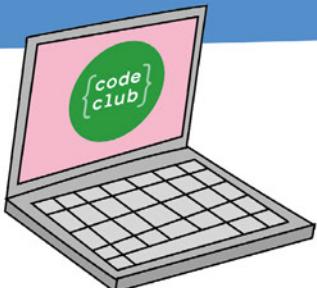
CAN YOU HELP INSPIRE THE NEXT GENERATION OF CODERS?



Code Club is a network of volunteers and educators who run free coding clubs for young people aged 9-13.

We're always looking for people with coding skills to volunteer to run a club at their local school, library, or community centre.

You can team up with friends or colleagues, you will be supported by someone from the venue, and we provide all the materials you'll need to help children get excited about digital making.



To find out more, join us at
www.codeclubworld.org



NEXT MONTH | *MagPi*

HALLOWEEN PROJECTS

Summon a petrifying project from
the depths of your makerspace



THE MAGPI #86
ON SALE 26 SEPTEMBER

Plus!

Learn to code
with Scratch 3

Build an interactive map

Make a glow stick
measurer

Create music and
SFX for games

DON'T MISS OUT!
magpi.cc/subscribe

TWITTER @TheMagPi

FACEBOOK fb.com/MagPiMagazine

EMAIL magpi@raspberrypi.org

EDITORIAL

Editor

Lucy Hattersley
lucy@raspberrypi.org

Features Editor

Rob Zwetsloot
rob.zwetsloot@raspberrypi.org

Sub Editors

Phil King and Nicola King

ADVERTISING

Charlotte Milligan
charlotte.milligan@raspberrypi.org
+44 (0)7725 368887

DESIGN

criticalmedia.co.uk

Head of Design

Lee Allen

Designers

Sam Ribbits, Harriet Knight

Illustrator

Sam Alder

CONTRIBUTORS

Mike Cook, David Crookes, Dan Elwick, PJ Evans, Tom Hadfield, Rosemary Hattersley, Daniel Lambton-Howard, Simon Long, Ben Nuttall, Marc Scott, Laura Sach, Danny Staple

PUBLISHING

Publishing Director
Russell Barnes
russell@raspberrypi.org

Director of Communications
Liz Upton

CEO
Eben Upton

DISTRIBUTION

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

SUBSCRIPTIONS

Unit 6 The Enterprise Centre
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE
+44 (0)1293 312193
magpi.cc/subscribe
magpi@subscriptionhelpline.co.uk



This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

The MagPi magazine is published by Raspberry Pi (Trading) Ltd, Maurice Wilkes Building, St John's Innovation Park, Cowley Road, Cambridge, CB4 0DS. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products, or services referred to or advertised in the magazine. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0).



ISSN: 2051-9982



Fixing a hole

For **Lucy Hattersley**, learning is a lesson all in itself

The very first program I remember was for the ZX81. It was a platformer I typed out from a printed sheet. The RUN moment produced a 0 for a character.

Our valiant heroine (let's call her 'Zero') had to jump perilously through a moving hole to get to the platform above. There was no moving left, or right; Zero had to wait for the exact time to jump.

Zero has metaphorically prepped me for a rocky relationship with coding for the rest of my life.

“There was the general impression that creative types with dyed red hair should not do technical subjects”

School code

I loved programming at school. I remember rooms full of BBC Micro computers all linked to a single tape drive, and a frustrated teacher shouting to a void of children who all had to type LOAD and then press RETURN at the same time.

There was the circuit board that never worked. The Z80 machine code that never worked. The turtle robot that went the wrong way. Like Christina Aguilera, it “made us work a little bit harder”.

Then there was the general impression that creative types with dyed red hair should not do technical subjects and, instead, be gently (but firmly) nudged to Art and English. I sucked at Art and aced at English. So, writing it was.

Guess what I ended up writing about for a living?

Crash courses

Many years later, it was MITx that taught me to code with any degree of competence. I first came across

MIT lectures on iTunes U; eventually it all migrated to MITx, along with interactive lessons and forums. I joined in courses on Computational Thinking, Data Science, Computer Science and Programming Using Python, plus Software Construction in Java. Everything I could lay my hands on (and all for free). I'm still doing MITx courses, but have briefly diverted to Ohio State to learn Calculus. It turns out you can only go so far at MIT with English and a smile.

I've been writing articles for *The MagPi* ever since the magazine became a part of Raspberry Pi itself. These days I commission most of what you read, responding to a range of super-talented coders, makers, and writers. And of course, the amazing projects that you all make with Raspberry Pi.

Still, I like to think I contribute. My Python programming and object-orientated programming (OOP) back in issues 53 (magpi.cc/53) and 54 (magpi.cc/54) helped a lot of folks. I still get compliments on it – teachers telling me they finally understand OOP. Among the very best was when a teacher recently asked me if they could use it for their students (of course you can). Since then, I've been writing tutorials on Google Coral / AIY and AI.

If I can bash the hole in the computer science barrier a little wider, making it easier for people to follow, that's enough.

I hope you enjoy *The MagPi*. I really do. It's a very important magazine. ☺

Lucy Hattersley

AUTHOR

Lucy is a coder. And a writer. Though not necessarily in that order. She edits *The MagPi* magazine and submitted this article to herself after deadline in memory of Douglas Adams.

@lucyhattersley

— Uni  & Unique —



UniPiCase for Raspberry Pi 4

Perfect option for your DIY or high-volume commercial projects



Fits all kinds of HATs

Ideal PoE HAT case



Use with HAT or Pi alone

Simple and professional



Rapid, tool-free assembly

GPIO cable pass-through



Two wall mount options

Well ventilated



www.UniPiCase.com



Designed and manufactured in North America

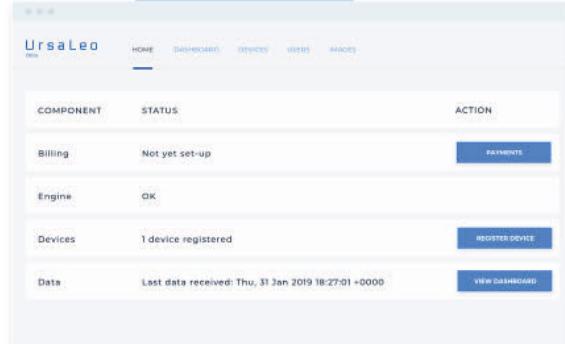
Use your Pi to collect sensor data to the Google Cloud



Ursa Leo

Download our Raspbian package to turn your Pi into a Google cloud gateway. Display data on dashboards, store and download it, use it to drive emails, texts and other alerts.

Enter code **magpi1**
at account creation to receive
a free Pi debug board*



- LEDs indicate BLE, internet and cloud connectivity
- Console interface
- Safe power down switch

ursaleo.com/raspbian

*Offer available for North America and Europe only

