THE OFFICIAL RASPBERRY PI MAGAZINE

The MagPi

The official Raspberry Pi magazine

Issue 54 February 2017

raspberrypi.org

MAKE A NIGHT-VISION NATURE CAMERA

Create your own Raspberry Pi-powered motion-sensing camera trap

MULTI-BOOT YOUR PI

The second part of our expert tutorial

COMMAND LINE GUIDE

Sudo apt-get good!

LEARN TO CODE WITH C

Simplify common operations on strings

HAPPY 5TH BIRTHDAY!

Join Raspberry Pi on the big day

OBJECT-ORIENTED PROGRAMMING

The follow-up to last month's bumper guide!

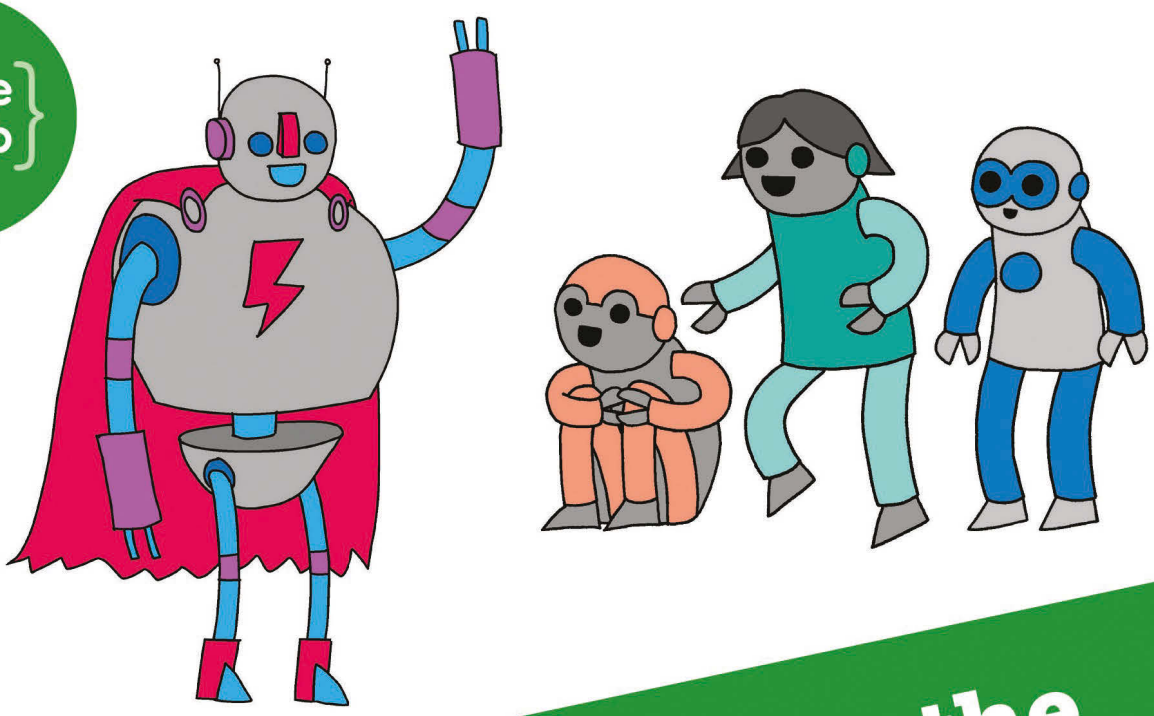
Also inside:

- **CREATE AN EMPATHETIC ROBOT? HOW THOUGHTFUL**
- OPEN SOURCE CLOUD STORAGE POWERED BY PI
- **RASPBERRY PI ROBOTICS KITS MUST BE WON!**
- PIMORONI'S NEW PAN AND TILT HAT REVIEWED

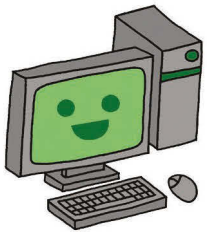
Issue 54 • Feb 2017 • £5.99

9 772051 998001 02

THE ONLY PI MAGAZINE WRITTEN BY THE RASPBERRY PI COMMUNITY



Can you help inspire the next generation of coders?



Code Club is a nationwide network of volunteer-led after school clubs for children aged 9-11.

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

You can team up with colleagues, a teacher will be there to support you and we provide all the materials you'll need to help get children excited about digital making.

There are loads of ways to get involved!

So to find out more, join us at **www.codeclub.org.uk**

This advertising space was kindly donated to Code Club by MagPi Magazine

WELCOME TO THE OFFICIAL MAGAZINE

It's a project that has captured the hearts and minds of enthusiasts throughout the Raspberry Pi community: the magic mirror. A wonderful, useful piece of home automation straight out of Tony Stark's house that you can build with a Raspberry Pi, an old monitor, and some wood.

We've wanted to do a magic mirror tutorial in the magazine for quite some time, so it was a delight that one of the driving forces behind it, Michael Teeuw, agreed to help us create the definitive guide to powering up your home.

If home automation isn't your thing, we have some great coding projects this issue, such as more from Simon Long on coding with C, the conclusion of our multi-booting guide, and the continuation of Lucy Hattersley's beginner's coding guide with a look at object-oriented programming. As for me, among other things I've reviewed a robot. I like robots; they're fun. Take a look for yourself and see if you agree…

I hope you enjoy this issue!

Rob Zwetsloot
Features Editor

THIS MONTH:

FIND US ONLINE raspberrypi.org/magpi **GET IN TOUCH** magpi@raspberrypi.org

The MagPi

Available on the App Store

GET IT ON Google Play

EDITORIAL
Managing Editor: **Russell Barnes** **russell@raspberrypi.org**
Features Editor: **Rob Zwetsloot**
News Editor: **Lucy Hattersley**
Sub Editors: **Laura Clay, Phil King, Lorna Lynch**

DESIGN
Critical Media: **criticalmedia.co.uk**
Head of Design: **Dougal Matthews**
Designers: **Lee Allen, Daiva Bumelyte, Mike Kay**
Illustrator: **Sam Alder**

PUBLISHING
For advertising & licensing:
russell@raspberrypi.org +44 (0)7904 766523
Publisher: **Liz Upton**
CEO: **Eben Upton**

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

SUBSCRIPTIONS
Select Publisher Services Ltd
PO Box 6337
Bournemouth
BH1 9EH | **+44 (0)1202 586 848**

CONTRIBUTORS
Sam Aaron, Alex Bate, Henry Budden, John Cole, Mike Cook, Gareth Halfacree, Phil King, Simon Long, Winfried Plappert, Matt Richardson, Richard Smedley & Michael Teeuw

recycle
When you have finished with this magazine, please recycle it.

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., 30 Station Road, Cambridge, CB1 2JH. 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.

Contents

Issue 54 February 2017 raspberrypi.org/magpi

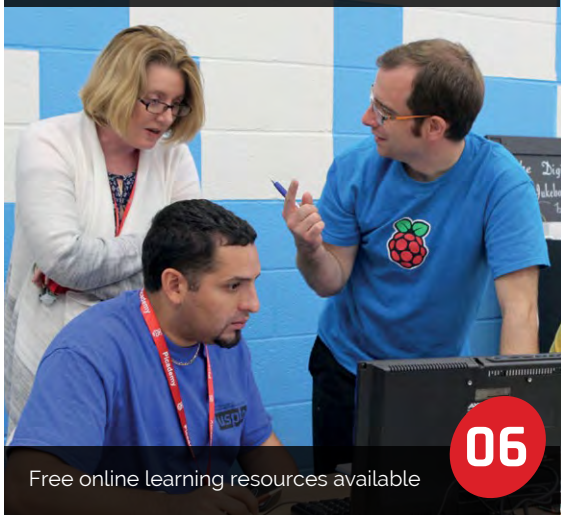
TUTORIALS

COVER FEATURE

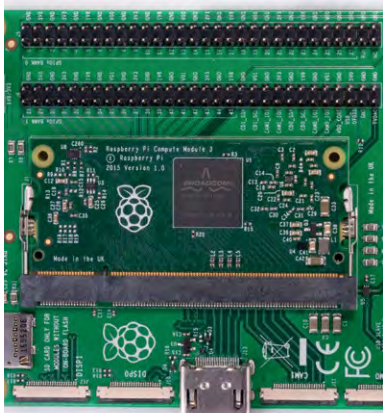
MAGIC MIRROR

IN THE NEWS

ONLINE TRAINING



COMPUTE MODULE 3



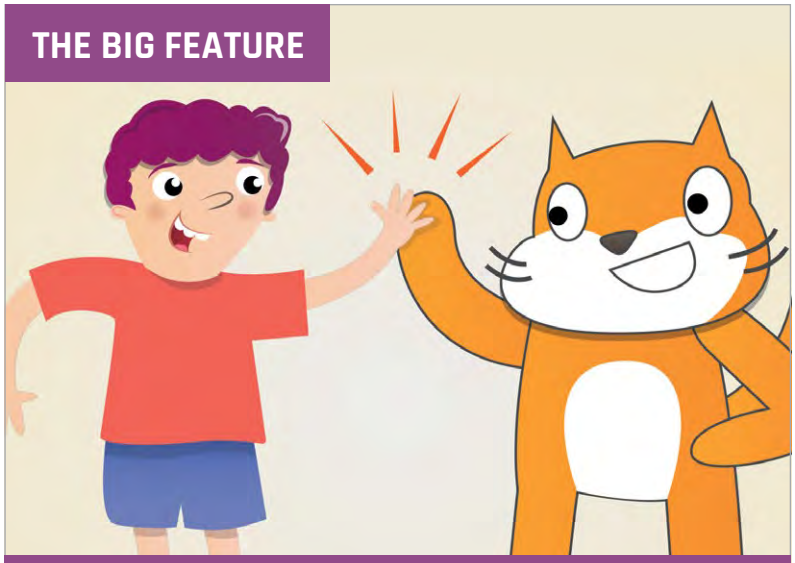
The latest developer spin of the Raspberry Pi is out now **08**

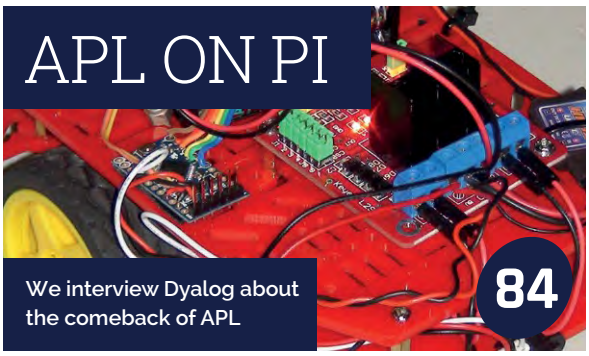
PI IS 5

Celebrate the Pi's fifth birthday this March **12**

Contents

THE BIG FEATURE





YOUR PROJECTS

REGULARS

COMMUNITY

REVIEWS

RASPBERRY PI ANNOUNCES ONLINE TRAINING

CERTIFICATION

Both courses are free. You can buy a Certificate of Achievement for this course – a personalised certificate and transcript in digital and printed formats – to prove what you've learnt (£59). A Statement of Participation (£34) is also available.

Learn to teach programming and physical computing with free Raspberry Pi courses

The Raspberry Pi Foundation has launched two new online courses designed to enable educators around the world to learn how to teach computing with a Raspberry Pi.

"We hope they will inspire a new army of enthusiastic makers around the world," says Lauren Hyams, online training manager.

The two offerings are Teaching Programming in Primary Schools, and Teaching Physical Computing with Raspberry Pi and Python.

Both courses are available via FutureLearn (**magpi.cc/2h5Sthf**). The courses are completely free. A printed certificate, to prove that learners have completed the course, is available for a small fee.

Each course is designed to last four weeks. There are around two hours of materials for learners to work through each week. "It's fine to take more time to reflect and learn at your own pace, though," Lauren tells us. "Courses begin on 20 February 2017, and you can sign up for both of them right now."

The two courses will be repeated later in 2017.

Global learning

The Raspberry Pi Foundation has trained over 540 educators in the US and the UK this year, "which we're immensely proud of," remarks Lauren.

However, this model of face-to-face training has its limitations. "We often get questions like 'Why haven't you run a Picademy near me yet?' and 'When are you coming to train us?'."

Professional educators will guide teachers around the world in teaching coding and physical computing

COURSES FOR TEACHERS

The Raspberry Pi Foundation is offering two online courses

The Raspberry PI Foundation grew frustrated at having to tell people that it didn't have plans to provide a Picademy in their region in the foreseeable future. So, it has developed these courses as a way to reach educators around the world.

Schools was designed for non-subject-specialist primary or K-5 teachers. "You don't need any prior experience of programming to take part," confirms Lauren.

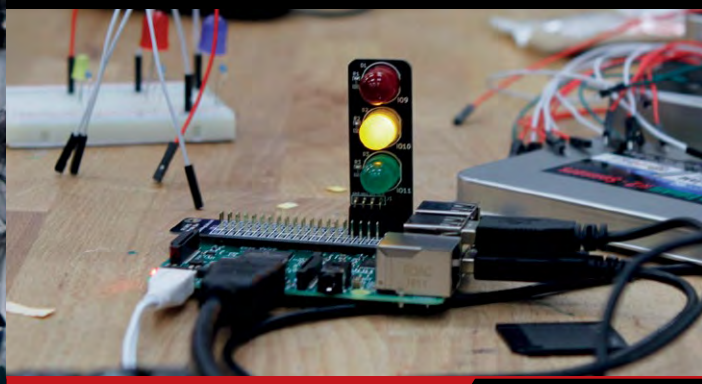
Teaching Physical Computing with Raspberry Pi and Python

> "Courses begin on 20 February 2017, and you can sign up for both of them right now"

Everybody is welcome

"This new free training supports our commitment to the White House's Computer Science For All initiative," says Lauren.

Everybody is welcome to sign up for the two courses. However, both courses are designed with particular educators in mind. Teaching Programming in Primary was designed for anyone interested in digital making. "It will be of particular use to teachers who are note subject specialists, computing teachers, and design and technology teachers who are interested in using the Raspberry Pi and Python in their classroom," explains Lauren.



TEACHING PHYSICAL COMPUTING WITH RASPBERRY PI AND PYTHON

Start Date: 20 February | **Duration: four weeks**

Time: two hours per week | **magpi.cc/2iH8GtD**

This four-week course will introduce you to physical computing, showing you how easy it is to create a system that responds to and controls the physical world, using computer programs running on the Raspberry Pi. You'll apply your newfound knowledge to a series of challenges, including controlling an LED with Python, using a button press to control a circuit, and making a button and LED game.

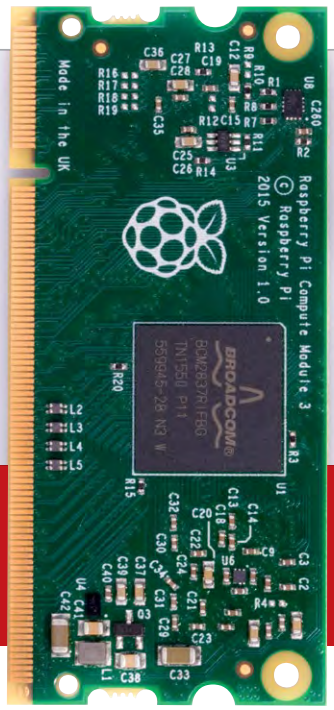
TEACHING PROGRAMMING IN PRIMARY SCHOOLS

Start Date: 20 February | **Duration: four weeks**

Time: two hours per week | **magpi.cc/2jx6Xul**

This four-week course will provide a comprehensive introduction to programming and is designed for primary or K-5 teachers who are not subject specialists. You'll have the chance to apply your understanding of the concepts through projects, both unplugged and on a computer, using Scratch as the programming language. Discover common mistakes and pitfalls, and develop strategies to fix them.

COMPUTE MODULE 3 LAUNCHED



New model based on Pi 3 is ten times more powerful

The Raspberry Pi Foundation has launched a new version of its Compute Module – a Raspberry Pi in a more flexible form factor, intended to provide an easy and cost-effective route to producing customised products. While the original Compute Module, launched in 2014, contained the guts of a first-generation Pi, the new Compute Module 3 (CM3) is based on the Pi 3, so offers a major boost in performance.

"It's got the same BCM2837 processor which can run at up to 1.2HGz, and 1GB of RAM," says Foundation COO James Adams. This means it provides twice the RAM and roughly ten times the CPU performance of the original Compute Module (CM1). Just like the Pi 3, the CM3 is able to run Windows 10 IoT Core, opening up many possibilities: "An IoT project can be ported from a Pi 3 onto a custom CM3-based system very easily."

> "Based on the Raspberry Pi 3, it offers a major boost in performance"

One issue with the CM1 was the fixed 4GB of eMMC flash storage: "Some users wanted free access to add their own flash [storage]," explains James. To solve this, two versions of the CM3 are being released: one with 4GB eMMC on board and a 'Lite' model which requires the user to add their own SD card socket or eMMC flash. "We expect this version to be quite popular."

The Compute Module 3 will be used in NEC's new range of smart, large-format displays

A quick upgrade

With a few caveats, the CM3 can be used a drop-in replacement for the CM1 since they are pin-compatible; the CM3 is 1mm taller, however, while the CPU can pull a lot more current from the VBAT power supply line and will generate far more heat under heavy load.

The Compute Module 3 and CM3 Lite are priced at $30 and $25 respectively (excluding tax and shipping), while the CM1 (which isn't being made obsolete) is $25. An updated IO breakout board (CMIO3) has also been launched, which will accept all three models.

CREATION MADE EASY

A complete kit to expand your universe.

WD PIDRIVE™ COMPUTE CENTRE

Kit Includes:

- WD PiDrive Foundation Edition 375GB
- Raspberry Pi 3
- Wireless Keyboard & Mouse
- Black Square 6"x6" Enclosure Top and Bottom
- WD PiDrive Cable
- Power Supply and USB cable
- MicroSD™ card (with preloaded software)

$109.99

PRELOADED WITH STARTER SOFTWARE

Start creating right away with our custom NOOBS OS Installer, Raspbian PIXEL, and Raspbian Lite, boot from the included microSD card and launch operating systems directly onto the USB drive.

ORGANIZE PROJECTS IN A SINGLE PLACE

The included Project Spaces software lets you create up to five work spaces to organize, access, and customize your projects in just one Raspberry Pi and hard drive setup.

EASY TO ASSEMBLE WITH EVERYTHING YOU NEED

Using a custom 6x6 enclosure to house a Raspberry Pi and WD PiDrive side by side, the WD PiDrive Compute Centre keeps your components in a tidy, attractive unit.

WDLABS™

wdlabs.wd.com/magpi54a

Western Digital, WD, the WD logo, WDLabs and the WDLabs logo are registered trademarks or trademarks of Western Digital Corporation or its affiliates in the U.S. and/or other countries. Raspberry Pi is a trademark of the Raspberry Pi Foundation. microSD is a trademark of SD-3C, LLC. All other marks are the property of their respective owners. As used for storage capacity, one gigabyte = one billion bytes. Total accessible capacity varies depending on operating environment. Product specifications subject to change without notice. Pictures shown may vary from actual product. © 2017 Western Digital Corporation or its affiliates.

KOSOVO PI WARS

Young makers in troubled country learn to build robots

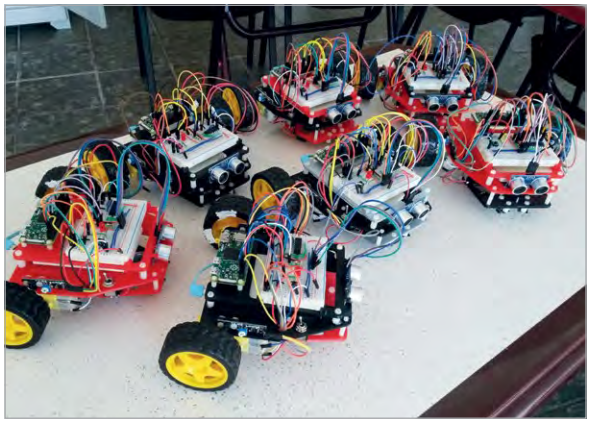
Above The Young Innovators is a robotics club based in Shtime, southern Kosovo

Many of us will know of Kosovo due to the war that took place there in 1998–9.

However, young makers in this country are putting that real war behind them by running the country's first ever Pi Wars (**magpi.cc/2iHC4QK**).

"We declared independence from Serbia just eight years ago," says Andy Moxon. Despite living in difficult times, "the country is not without Python coding, physical computing, robots, and a good number of Raspberry Pi computers."

Below The robots built by Young Innovators, ready to do battle in Pi Wars



Andy has been running an after-school club called Young Innovators (**magpi.cc/2iHAtuc**), based in the small town of Shtime in southern Kosovo.

"The club aims to bring maths and physics to life, while also teaching the students programming and robotics," explains Andy.

One area which has caused a lot of excitement in the club has been the recent introduction of an Ultimaker 2+ 3D printer (**magpi.cc/2iHC5Eg**). "Using FreeCAD, we have designed the chassis of the robots from nothing," reveals Andy. "This has been a tough but worthwhile exercise, demonstrating the wonders of 3D prototyping."

> "Big thanks must go out to the Raspberry Pi community"

"Our robots are pretty standard," he adds. A Pi Zero is powered by a thin mobile phone power bank. Each robot has two motors controlled by an L293D motor controller chip. Also included are ultrasonic distance and infrared line sensors.

"We use two additional infrared sensors to count wheel revolutions," says Andy, "having painted white stripes on our wheels using nail polish." The sensors open the robots up to some interesting autonomous challenges.

The robots took part in the country's first Pi Wars battle in December. The event was even covered by TemaTV, a regional television station.

"Big thanks must go to the Raspberry Pi community," says Andy. "I first used a Raspberry Pi just a year ago and without the dedication of excellent bloggers, we would never have been able to reach this stage."

GAME FOR A LAUGH

Pioneers programme launches by challenging teenagers to make the Foundation laugh

The Raspberry Pi Foundation has officially launched its Pioneers programme for teens with a challenge to make everyone laugh.

From electric shock buzzers to voice modulators and dancing robots, there's lots of fun to be found in Raspberry Pi.

There's a long tradition of creating pranks in electronic circles. MIT students once hacked the lights in their tower block to create a giant game of Tetris they could play outside the building.

Teenagers between the ages of twelve and 15 are being challenged to form teams and build something that'll make other teens laugh.

These projects are then shared online to earn the kudos and respect of other young makers. They'll also win money-can't-buy prizes and cool swag.

"We want to find and support teenage digital makers in the UK," says Rob Buckland, director of programmes. "The aim of Pioneers is to provide guidance, inspiration, and mentorship to teenage makers, and to the adults who mentor them."

Take a look at **magpi.cc/2iHKIP5** for more information.

Teenagers are being challenged to make everybody laugh with the Pioneers programme

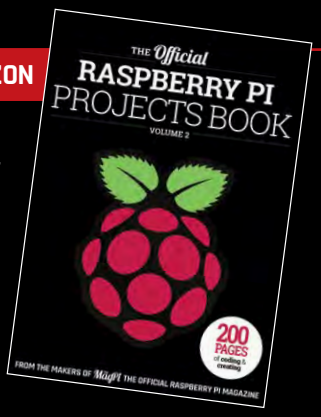
NOW TRENDING

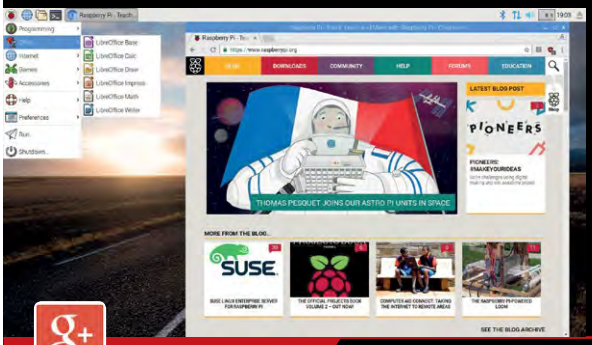
The stories we shared that flew around the world

PROJECTS 2 ON AMAZON

magpi.cc/2hrrb7r

News that *The Official Raspberry Pi Projects Book 2* had gone on sale at Amazon went wild on Facebook during December. We think it makes a great present all year round, but hope lots of you were happy to find it under a tree.





DEBIAN + PIXEL

magpi.cc/2h9miyY

News about Debian + PIXEL set Google Plus on fire last month. In case you missed the free DVD on last month's magazine, you can download the Raspbian + PIXEL ISO direct from the Raspberry Pi website (**magpi.cc/2iB5geO**).

CEMENT THWOMP

magpi.cc/2gPJiRY

Made from real cement, this incredible case has a Raspberry Pi inside. A custom mould is used to cast the concrete into the shape of a character from Super Mario 3D. A big, big hit on Twitter.

BIG BIRTHDAY WEEKEND 2017

Above right Raspberry Pi Foundation CEO Philip Colligan hosting a workshop lecture at last year's Big Birthday Party

Above left Cambridge Junction

Raspberry Pi Foundation set to throw a massive party for its fifth year

The Raspberry Pi Foundation has announced plans for its Big Birthday Weekend party in 2017.

Taking place from Saturday 4 March to Sunday 5 March, the Big Birthday Weekend a two-day event when the whole community meets up with the Raspberry Pi Foundation team.

"Our annual birthday weekend is always great fun for families, hobbyists, and educators," says Helen Drury, outreach and events manager, "as well as our team of amazing community volunteers. we're doing to support educators teaching computer science."

The party will feature talks and workshops covering multiple subjects and interests. Show-and-tell stands will showcase incredible creations, while vendors exhibit their kits in the Maker Market.

BIG BIRTHDAY WEEKEND

Raspberry Pi Birthday Weekend 2017

Cambridge Junction, Clifton Way, Cambridge, CB1 7GX
Saturday 4 March 2017: 10.30am-6.30pm
Sunday 5 March 2017: 10am-5.30pm
Tickets: £5 (16 years and over), free (under 16)
Ticket purchase available here: magpi.cc/2iHMAax

> "Goody bags and cupcakes throughout the weekend"

"This year it's going to be bigger and better than ever."

The two-day event will be taking place at Cambridge Junction, the leading arts centre in Cambridge. Tickets are £5 if you are aged 16 or over. Entrance is free for under-16s.

"Raspberry Pi and Code Club will be both turning five," reads the invitation. "We'd love for you to come and see what Raspberry Pi is all about, try some of our Code Club projects, and learn what There will also be goody bags and cupcakes throughout the weekend.

The full programme will be available shortly. Workshops will be registration-only, and you can sign up for them when they launch.

There are lots of ways you can get involved with the Big Birthday Weekend (beyond just turning up). In the meantime, if you haven't been to a Raspberry Pi party before, check out the Raspberry Pi Foundation blog from last year: **magpi.cc/2cBFJyo**.

RASPBERRY PI IN THE HOUSE

Home of the future powered by Raspberry Pi

Futurehome is a self-adjusting home automation kit with the Raspberry Pi Compute Module at its heart.

"Futurehome lets you control, automate and monitor your home from anywhere in the world,"

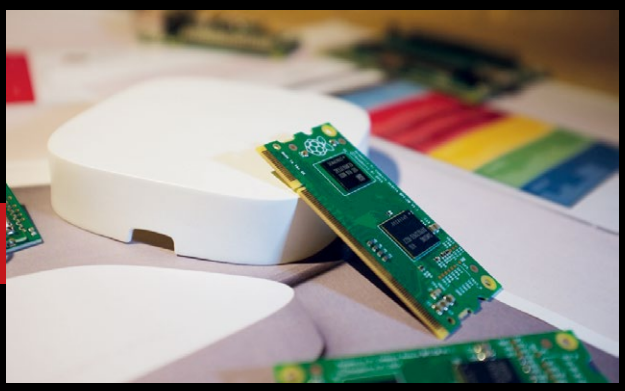
> "Automate and monitor your home from anywhere"

says Odd Eivind Evensen, product development manager.

The company sells a Smarthub (**magpi.cc/2hRxxOL**) for NOK 2499 (£233/$359). Following a successful Indiegogo campaign, Futurehome has started production. The units are reportedly selling in their thousands.

It's a good example of what entrepreneurial makers can create with a Raspberry Pi.

The Smarthub works like an internet router, but for smart devices. It "communicates through the open standards Z-Wave and EnOcean."



Above Made with the Compute Module, the Smarthub is a successful home automation product

These open standards allow for hundreds of different types of gadgets to be controlled, covering everything from lighting to heating and motion sensors.

At the heart of the device is the Smarthub containing a Raspberry Pi Compute Module. Futurehome has also created an iPhone and Android app to control the system.

"The small form factor and the power of the Raspberry Pi [Compute Module], not to mention the eMMC memory, all makes it the perfect server," the firm tells us.

TAKING THE RASPBERRY PI TO THE NEXT LEVEL

HIPSY
HOME INTRUSION PREVENTION SYSTEM

24 HOUR

GIPSY
GLOBAL INTRUSION PREVENTION SYSTEM

MANAGED CYBER-PROTECTION SERVICE

DEVELOPED BY THE INDUSTRY EXPERTS
idappcom

LINUX OPERATING SYSTEM • INDUSTRY STANDARD INTRUSION DETECTION ENGINE • SMALL WEB SERVER • DNS SERVER • NETWORK SCANNER
ENHANCED WIRELESS FUNCTIONS • CLOUD BASED MANAGEMENT VPN • IDAPPCOM RULES MANAGEMENT SYSTEM
ACCESS TO 10,000 PLUS EXPLOIT RULES • OPTIONAL MALWARE AND PHISHING PROTECTION

PROTECT YOUR FAMILY, FINANCES & FILES AT HOME & ON THE GO

SHIELD FROM EXPLOITS
Attackers working from outside your system

PROTECT FROM PHISHING
Invitations to reveal your private information

BLOCK MALWARE
Attacks that have been installed inside your system

Our products are built on the best of British small computer, using our decades of expert security experience. We provide 24 hour security management via the cloud connected Idappcom Security Operations Centre.

Taking the Raspberry Pi to the Next Level of Cyber Protection
Visit: www.ipssecurityrules.com Call: +44 (0) 203 355 6804

BUILD YOUR OWN MAGIC MIRROR

Live in the future with your own internet-connected mirror straight from science-fiction

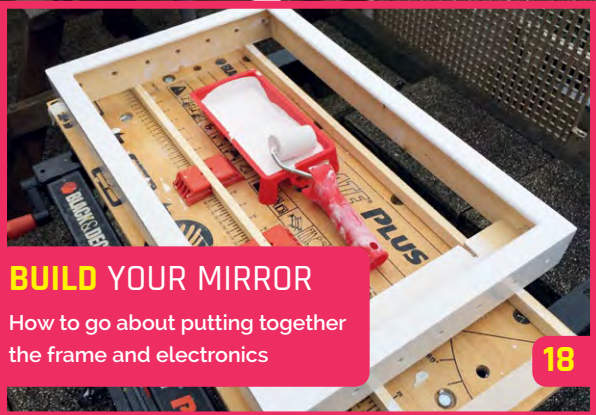
We live in a world where more and more items are becoming part of the Internet of Things. Fridges sending an email about buying some more milk. Washing machines tweeting you when they're done. An app on your phone that lets you turn on a light. A lot of this seems gimmicky, though, which is probably why the magic mirror concept has taken off so well in the maker community – getting all the information you need at a glance while checking your appearance before you leave the house. It's passive and useful. Michael Teeuw pioneered the concept on the Raspberry Pi, making it easy enough for everyone to make one. All you need to build your own is a bit of spare time and a trusty saw.

THE PROCESS STEPS TO BUILD A MAGIC MIRROR

GET THE PARTS

Everything you need to make your own mirror

16



BUILD YOUR MIRROR

How to go about putting together the frame and electronics

18

GET ALL THE SOFTWARE AND INFO YOU NEED ONLINE AT:

MAGICMIRROR.BUILDERS

THE CLEVER CODING ASPECT

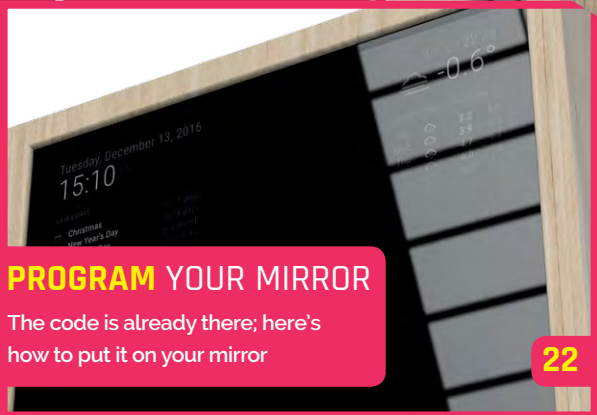
The code for the mirror is all there and ready to be installed with one line

TWO-WAY MIRROR

Used in the right way, you can make an info-laden mirror

BUILDING THE FRAME

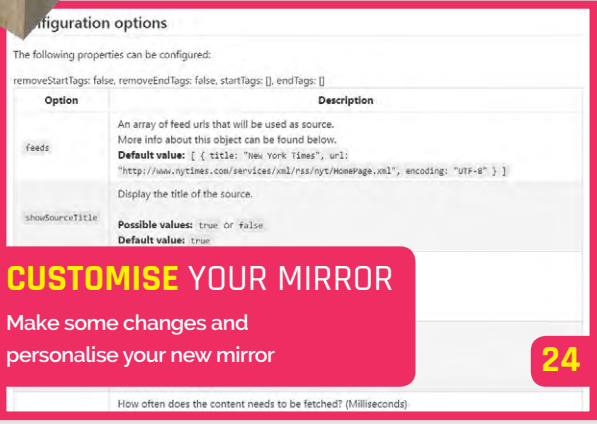
Get out your tape measure and put a pencil behind your ear



PROGRAM YOUR MIRROR

The code is already there; here's how to put it on your mirror

22



CUSTOMISE YOUR MIRROR

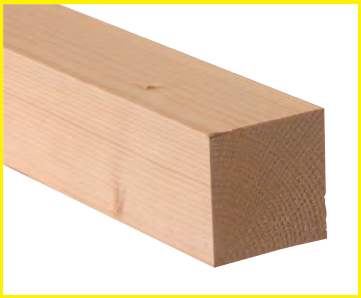
Make some changes and personalise your new mirror

24

COMPONENTS

What you need to build your mirror

BUILDING MATERIALS



WOOD FOR FRAME

Plywood is a good option for this. If you're a little more skilled at carpentry, however, you can have a look at pine or another material. Make sure it's sturdy and deep enough to contain all the electronics when built – have a look at our build steps for an idea of the size in comparison.

WOOD FOR FRONT

The fascia can be made from skirting board or moulding – make sure it's wider than the wood you plan on using for the frame so it can keep everything in place.

NAILS

We'll use these to attach the front of the frame; 15-20mm should do.

SCREWS

A small selection of wood screws to build the frame. They don't need to be huge, though: about 20mm longer than the depth of the wood should be fine.

WOOD PAINT & FILLER

You'll want to make your frame look good, so get some filler to help smooth it out, and some wood paint to finish it off. If you like the wood you've used, get some varnish instead.

WOOD GLUE

We're using this to make sure everything stays together as intended. Think of it as a backup for the screws.

TWO-WAY MIRROR

Buy one the same size as the monitor, which will also be the same size as your frame. You can get some made from acrylic.

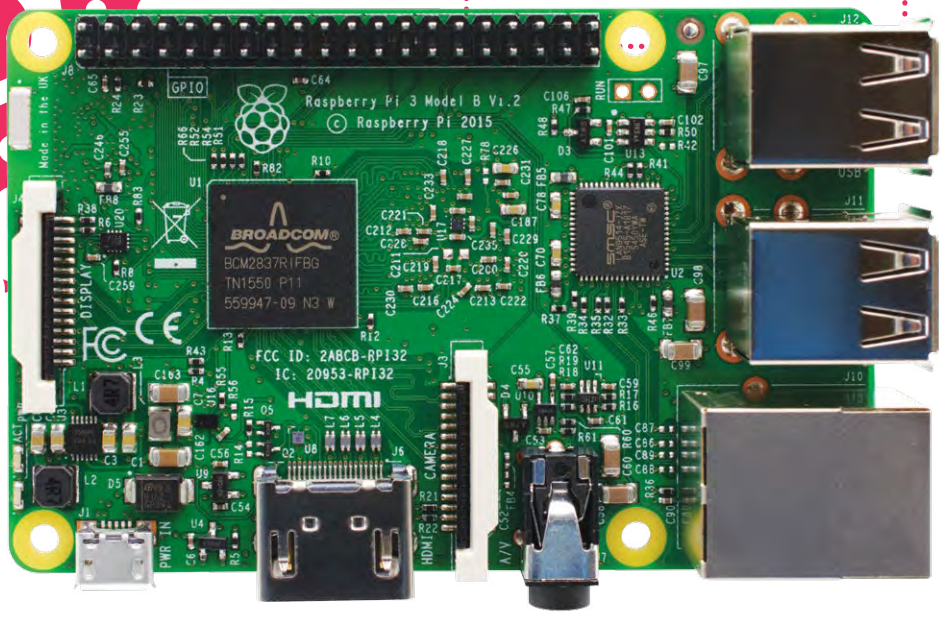
ELECTRONICS

MONITOR/TELEVISION

A large monitor or old LCD television should do the trick for this. The lighter the better, though. You may want to remove the outer casing on your monitor if you can, to save on space and weight.

RASPBERRY PI

You'll need one to power the TV, of course. A case (and a WiFi dongle if you're using a Pi 2) should help make things a little more tidy and secure.

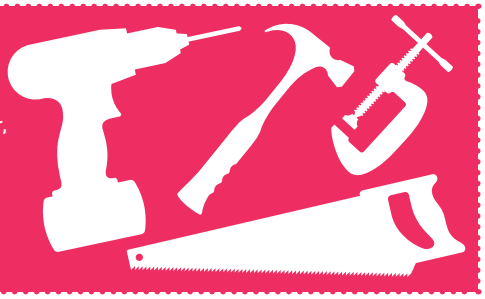


CABLES

You'll need power for the monitor and the Pi, as well as an HDMI for the picture output.

TOOLS

You'll need a saw, a hammer, a drill, some clamps, and various painting tools to do this. Research what you'll need for specific bits.



STEP 01:

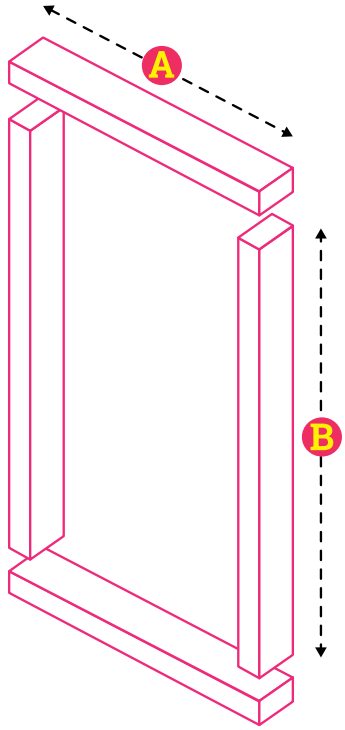
MEASURE THE MONITOR

Measure the dimensions of the monitor. It may be an idea to see how deep it is as well, to ensure your wood selection is correct; it's best to have a little space between the back of the monitor and the wall, so make sure your wood will allow for this.

STEP 02:

CUT THE WOOD

The basic frame is made up of four pieces of wood. You can view it as two side pieces and a top and bottom piece. The sides should be slightly longer as the mirror will be in portrait orientation. The sides should be the same length as the long edges of the monitor, while the top and bottom should be the length of the short sides plus the width of the wood pieces so that they can fit neatly on top as a rectangle. Make sure to not make the fit too snug for the monitor – allow for an extra millimetre or two.



A Width of monitor + depth of wood × 2

B Height of monitor

REMEMBER: MEASURE TWICE, CUT ONCE

STEP 03:

ASSEMBLE THE FRAME

As this is the main frame, it's best to have this as sturdy as possible. Screwing the wood into place with two screws will keep it nice and strong, but adding a little wood glue will make sure it stays together even better.

STEP 04:

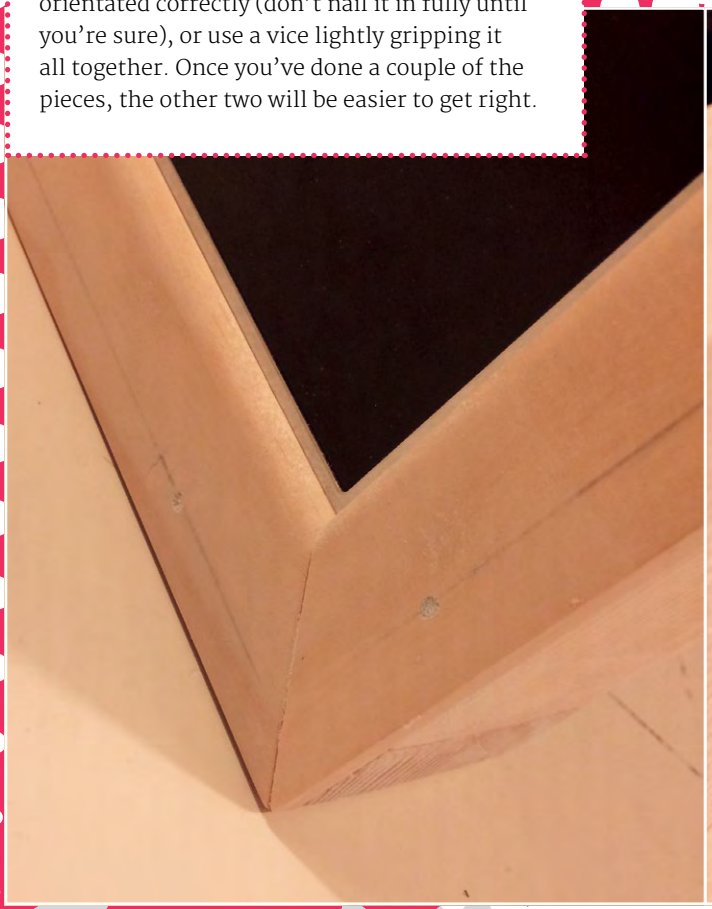
CUT THE FRONT

With the main frame done, we can now add the front of the frame. This has two functions: it covers the bezel on the monitor, making it look a bit better aesthetically, and acts as a lip to hold the monitor and mirror in place. Make sure the material is deeper than the wood for the frame to make this lip, and cut the ends at an angle for when you join them all together.

STEP 05:

ATTACH THE FRONT

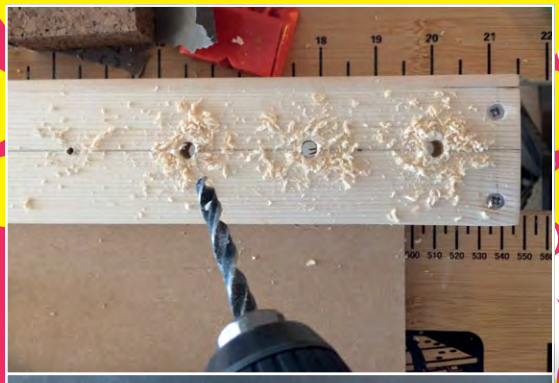
Carefully nail on the front pieces, making sure they're flush to the edge of the outside. Do one nail at each end first to make sure it is orientated correctly (don't nail it in fully until you're sure), or use a vice lightly gripping it all together. Once you've done a couple of the pieces, the other two will be easier to get right.



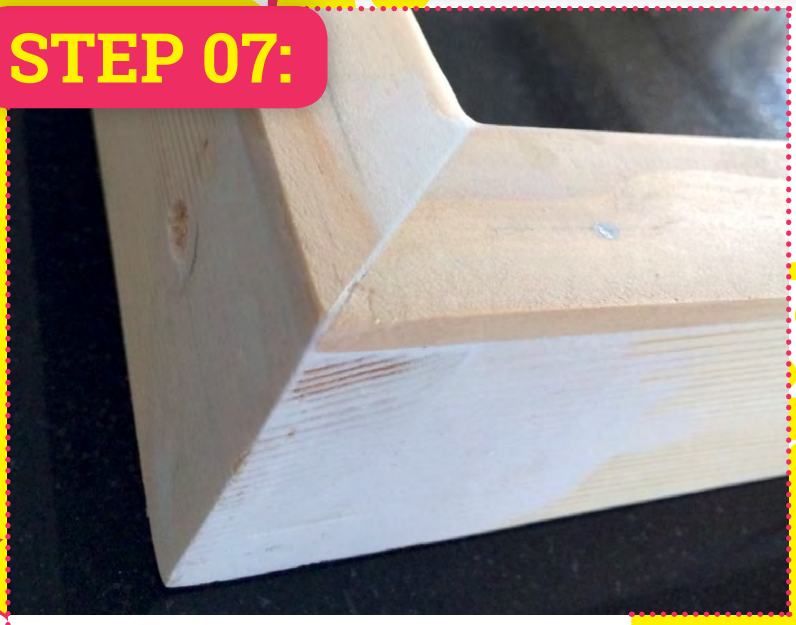
STEP 06:

FINAL PIECES

Drill a few holes through the top and bottom parts of the frame, as shown – this aids ventilation. Nothing should be getting toasty hot in there, but it's better to have some air going through. You'll also need a piece, as shown, with some slots to hang over screws in the wall. You should make a little indentation on the back of the bottom piece of wood so the power cables can run through to the monitor and Pi. Finally, create some small brackets that you can fasten to the frame to keep the monitor from falling out of the back.



STEP 07:



SANDING & SMOOTHING

Use a little filler to cover any dents, as well as the nail heads on the front of the frame. Give it a sand as well, to smooth everything off and prepare for painting.

STEP 08:

GET PAINTING

In this example, the frame has been painted white with some wood paint. Make sure to do this in a well-ventilated room.

STEP 09:

SLOT IN THE MIRROR

When the paint is dry, flip the frame over so it's resting on its front and carefully lower the mirror in. We're going to keep it in place using the overhang of the front moulding and the monitor, effectively sandwiching it between them.

STEP 10:

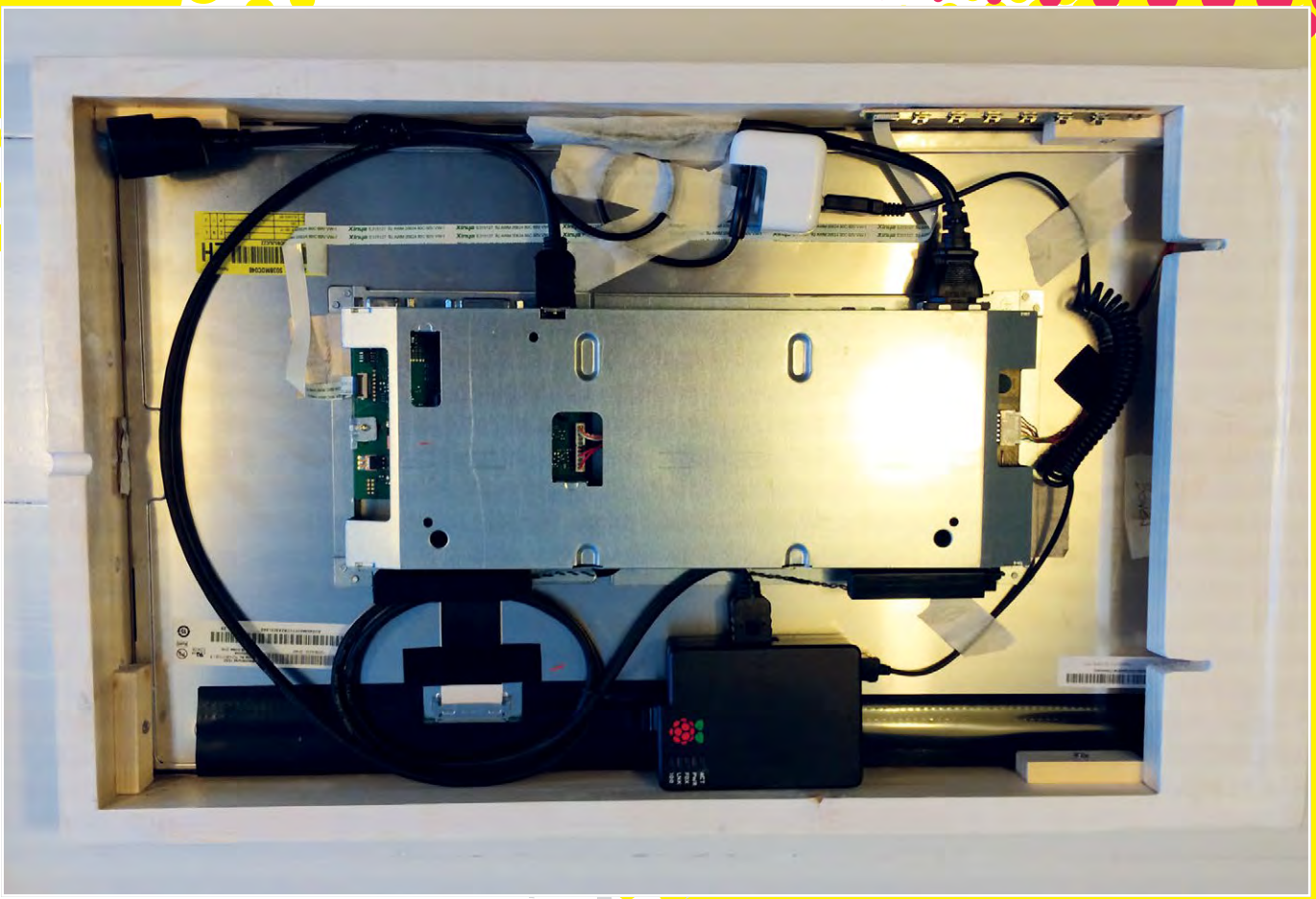
INSTALL THE ELECTRONICS

This bit is pretty easy: just put the monitor into the back, fasten it in with the little wooden brackets, and hook up the Raspberry Pi to the HDMI. Run power cables through the indentation you made.

STEP 11: EXTRA TIP!

ONE CABLE, TWO DEVICES

One thing the more advanced maker could do is combine the power cable to the TV with a plug-to-USB adapter. This way you only need one cable running through to the mirror.



PROGRAM YOUR MIRROR

Install the software to your Raspberry Pi and make your mirror truly magical

Once you've hung your mirror on the wall, or placed it wherever it's now going to live, it's time to install the software. Michael has made the process incredible easy, and all you need to do is type the following command:

```
curl -sL http://magpi.cc/MirrorInstall | bash
```

It will go through the installation process and set up some defaults... and that's it! Your mirror is ready to use.

PERSONALISATION

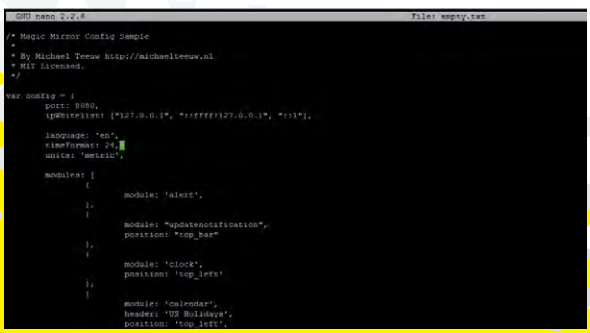
Well, almost ready: you may want to have a quick look at editing those default values to make sure it's running as you'd like it to. The settings are kept in a config file which you can create by using:

```
cp ~/MagicMirror/config/config.js.sample ~/
MagicMirror/config/config.js
```

You can now access and modify it with:

```
nano ~/MagicMirror/config/config.js
```

Here are some of the options you can modify...



The sample configuration file gets you started, but there's way more you can do

OPTION	DESCRIPTION
`port`	The port on which the MagicMirror server will run on. The default value is 8080.
`address`	The IP address of the mirror you use to connect to it.
`ipWhitelist`	The list of IPs which are allowed to connect to the mirror. The default value is `["127.0.0.1", "::ffff:127.0.0.1", "::1"]`. It is possible to specify IPs with subnet masks (`["127.0.0.1", "127.0.0.1/24"]`) or define IP ranges (`["127.0.0.1", ["192.168.0.1", "192.168.0.100"]]`).
`zoom`	This allows scaling of the mirror interface with a given zoom factor. The default value is 1.0.
`language`	The language of the interface. Possible values are en, nl, ru, fr, and so on, but the default value is en.
`timeFormat`	The style of clock to use. The accepted values are 12 and 24. The default is 24.
`units`	The units that will be used in the default weather modules. Possible values are metric or imperial. The default is metric.
`modules`	An array of active modules. There must always be an object in here.
`electronOptions`	An optional array of Electron (browser) options. This allows configuration of e.g. the browser screen size and position (defaults `.width = 800` and `.height = 600`). Kiosk mode can be enabled by setting `.kiosk = true`, `.autoHideMenuBar = false`, `.fullscreen = false`.

DATE

What's the time and today's date? Better stop admiring the reflection or you will be late for work!

CALENDAR

Is today a holiday? Is there a meeting coming up? Is this colour of tie appropriate for it?

NICE MESSAGE

Thanks mirror, you're looking pretty good yourself! My compliments to the amazing person that made you

TEMPERATURE

What's the weather going to be like today? Is it umbrella weather? Better make it back before sundown as well

NEWS

Any important news to catch up on? Is there traffic on the commute? Oh, look: *Crystal Maze* is coming back!

CUSTOMISING YOUR MIRROR

Make your mirror truly yours by adding and customising modules

As well as the pre-installed default modules, you can add third-party modules that have either been created by the community or yourself. They're easy to add: you just need to download the files and then update the configuration file to use them.

First of all, take a look on the page for the MagicMirror modules in the GitHub repo here: **magpi.cc/2iqWPUh**. You'll find a list of great modules to add, such as a Bitcoin monitor and something that displays today's XKCD comic. Pick one you like and copy the link location to it.

To install the module, first move to the **modules** folder with **cd~/MagicMirror/modules** and then download the data for it with:

```
git clone https://github.com/[author]/
[module-name]
```

...with the GitHub link pasted from the link you copied. Check out the readme for the module and see if there are any other steps to perform, otherwise open up the **config.js** file from before and add the module to the module section. You'll need to format it something like:

```
{
  module: 'module name',
  position: 'position',
  header: 'optional header',
  config: {
    extra option: 'value'
  }
},
```

Here's the full list of options to use...

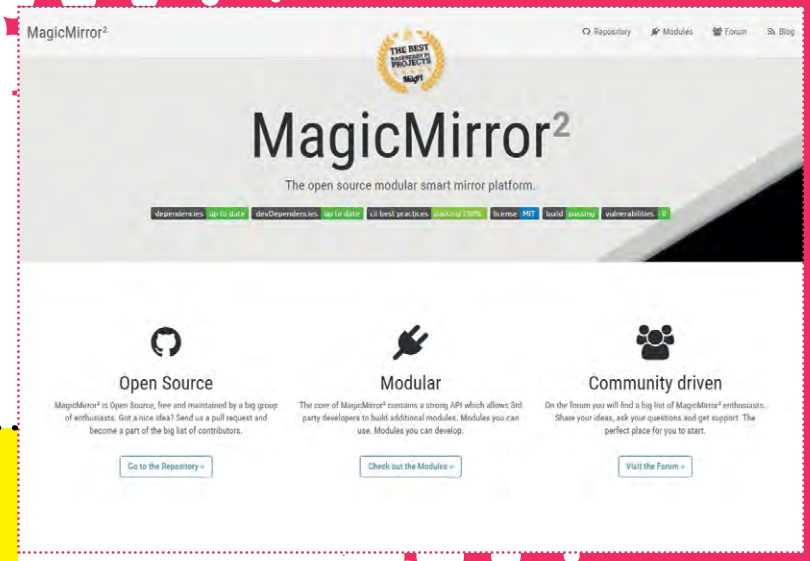
DON'T WANT TO DISMANTLE YOUR MIRROR?

Use SSH from another computer to access the Pi inside

OPTION	DESCRIPTION
module	The name of the module. This can also contain the subfolder. Valid examples include **clock**, **default/calendar**, and **modules/[module name]**.
position	The location of the module on the mirror. Possible values are: **top_bar**, **top_left**, **top_center**, **top_right**, **upper_third**, **middle_center**, **lower_third**, **bottom_left**, **bottom_center**, **bottom_right**, **bottom_bar**, **fullscreen_above**, and **fullscreen_below**. This field is optional, but most modules require it to be set. Check the documentation of the module for more information. Multiple modules with the same position will be arranged based on the order in the configuration file. Some trial and error is required.
classes	Additional classes which are passed to the module. The field is optional.
header	To display a header text above the module, add the header property. This field is optional.
disabled	Set disabled to true to skip creating the module. This field is optional.
config	An object with the module configuration properties. Check the documentation of the module for more information. This field is optional, unless the module requires extra configuration.

SUPPORT & COMMUNITY

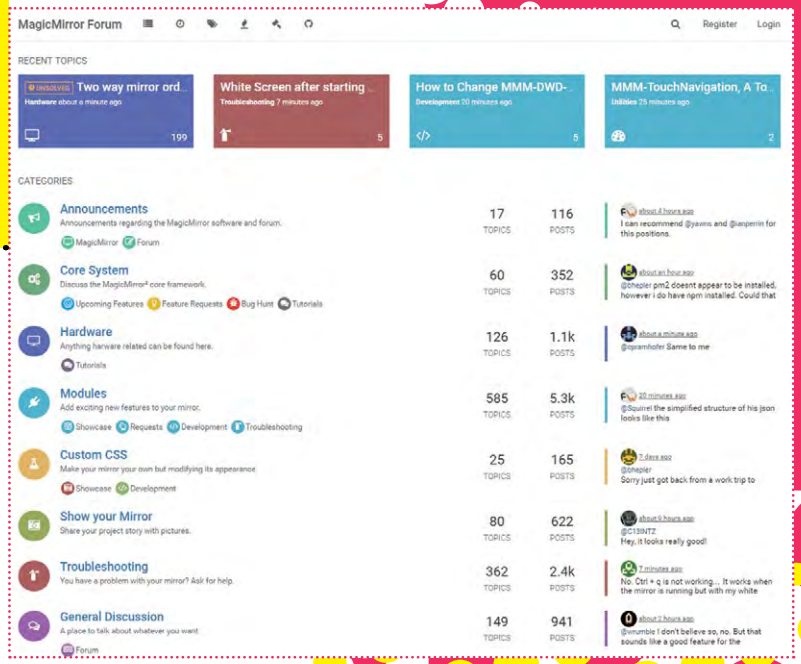
You've made your mirror, now here are some ways to keep up on its developments



MAGICMIRROR SITE

magicmirror.builders

Your first stop for MagicMirror info is the home site for the software. You'll find some handy links, such as those to the blog for updates on the project, GitHub for the source code and more in-depth configuration options for the mirror, as well as a link to the forum and available modules. It's also a good way to quickly introduce a friend to the concept.



MAGICMIRROR FORUMS

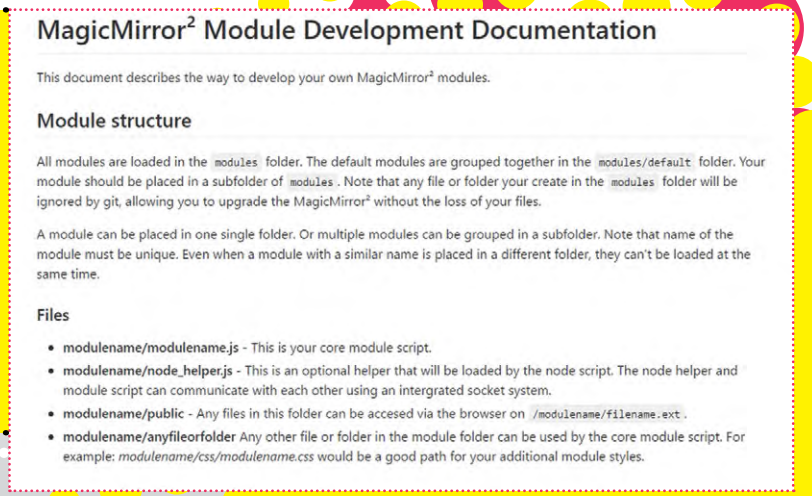
magpi.cc/2je2dXI

The community for the MagicMirror lives here, and the answer to many problems you might face in the long term have likely already been answered there. They're a friendly bunch, though, so if you can't find a solution, you can always have a chat with them to find out what might be going wrong. You'll also be able to compare notes about any modules you decide to make or any other little upgrades or building tips you might want to know about.

MAGICMIRROR MODULE DEVELOPMENT DOCUMENTATION

magpi.cc/2jebFux

Fancy making your own custom module for your mirror? You'll need to know how the API and code generally works, and you can do that with the documentation provided on the MagicMirror GitHub repo. We've seen some excellent add-ons for the mirror code, including seasonal variations and live train times. All you need is a way to get the data!



MagicMirror² Module Development Documentation

This document describes the way to develop your own MagicMirror² modules.

Module structure

All modules are loaded in the `modules` folder. The default modules are grouped together in the `modules/default` folder. Your module should be placed in a subfolder of `modules`. Note that any file or folder your create in the `modules` folder will be ignored by git, allowing you to upgrade the MagicMirror² without the loss of your files.

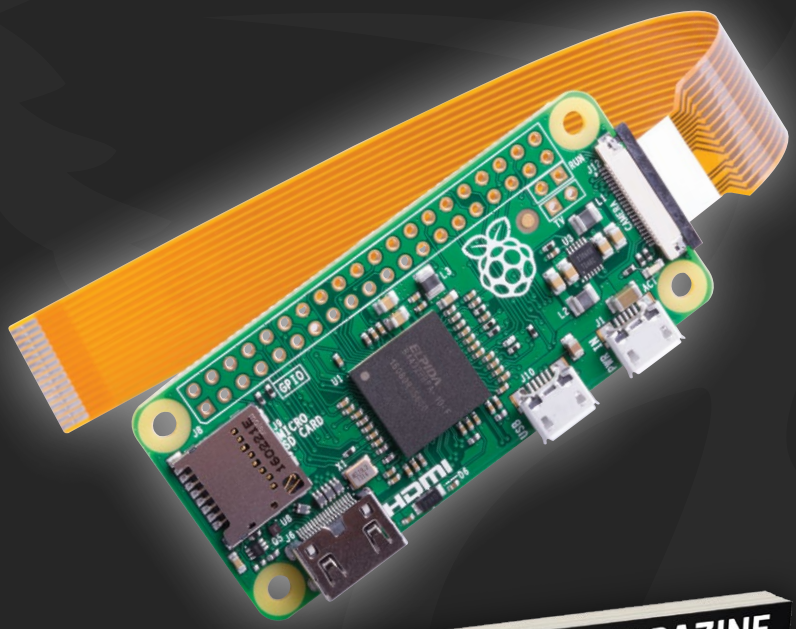
A module can be placed in one single folder. Or multiple modules can be grouped in a subfolder. Note that name of the module must be unique. Even when a module with a similar name is placed in a different folder, they can't be loaded at the same time.

Files

- **modulename/modulename.js** - This is your core module script.
- **modulename/node_helper.js** - This is an optional helper that will be loaded by the node script. The node helper and module script can communicate with each other using an intergrated socket system.
- **modulename/public** - Any files in this folder can be accesed via the browser on `/modulename/filename.ext`.
- **modulename/anyfileorfolder** Any other file or folder in the module folder can be used by the core module script. For example: *modulename/css/modulename.css* would be a good path for your additional module styles.

FREE PI ZERO!

Subscribe in print for six or 12 months to receive this stunning free gift



Subscribe today and receive:

- A free Pi Zero v1.3 (the latest model)
- A free Camera Module connector
- A free USB and HDMI cable bundle

Delivered with your first issue!

Other benefits:

- Save up to 25% on the price
- Free delivery to your door
- Exclusive Pi offers and discounts
- Get every issue first (before stores)

Pricing

Get six issues:

£30 (UK)
£45 (EU)
$69 (USA)
£50 (Rest of World)

FREE PI ZERO!

Subscribe for a year:

£55 (UK)
£80 (EU)
$129 (USA)
£90 (Rest of World)

FREE PI ZERO!

Get three issues:

£12.99 (UK) (Direct Debit)
$37.50 (US) (quarterly)

How to subscribe:

- **magpi.cc/Subs1** (UK / ROW)
- **imsnews.com/magpi** (USA)
- Call +44(0)1202 586848 (UK/ROW)
- Call 800 428 3003 (USA)

Available on the App Store

GET IT ON Google Play

SUBSCRIPTION FORM

YES! I'd like to subscribe to The MagPi magazine and save money

This subscription is: ☐ For me ☐ A gift for someone*

Mag#54

YOUR DETAILS Mr ☐ Mrs ☐ Miss ☐ Ms ☐

First name ..... Surname .....

Address .....

.....

Postcode ..... Email .....

Daytime phone ..... Mobile .....

**If giving* The MagPi *as a gift, please complete both your own details (above) and the recipient's (below).*

GIFT RECIPIENT'S DETAILS ONLY Mr ☐ Mrs ☐ Miss ☐ Ms ☐

First name ..... Surname .....

Address .....

Postcode ..... Email .....

PAYMENT OPTIONS

1 DIRECT DEBIT PAYMENT £12.99 every 3 issues (UK only)
Instruction to your bank or building society to pay by Direct Debit

DIRECT Debit

Please fill in the form and send to:
The MagPi, Select Publisher Services Ltd,
PO Box 6337, Bournemouth BH1 9EH

Service user number 8 3 8 7 7 3

Name and full postal address of your bank or building society:

To: The Manager Bank/building society .....

Address .....

..... Postcode .....

Name(s) of account holder(s) .....

Branch sort code ☐☐☐☐☐☐ Account number ☐☐☐☐☐☐☐☐

Reference ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐ (Official use only)

Instruction to your bank or building society
Please pay Select Publisher Services Ltd Direct Debits from the account detailed in this instruction subject to the safeguards assured by the Direct Debit Guarantee. I understand that this instruction may remain with Select Publisher Services Ltd and, if so, details will be passed electronically to my bank/building society.

Signature ..... Date ☐☐/☐☐/☐☐

Banks and building societies may not accept Direct Debit instructions for some types of account.

SUBSCRIPTION PRICING WHEN PAYING BY CHEQUE OR CREDIT/DEBIT CARD

6 ISSUES ☐ UK £30 ☐ Europe £45 ☐ Rest of World £50

12 ISSUES ☐ UK £55 ☐ Europe £80 ☐ Rest of World £90

2 CHEQUE

I enclose a cheque for ..... (made payable to Select Publisher Services Ltd)

3 CREDIT/DEBIT CARD ☐ Visa ☐ MasterCard ☐ Maestro ☐ Switch

Card number ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Expiry date ☐☐ ☐☐ Valid from ☐☐ ☐☐ (if shown)

Issue number ☐☐ (if shown) Security number ☐☐☐
(last 3 digits on the back of the card)

Signature ..... Date ☐☐/☐☐/☐☐

I would like my subscription to begin from issue ..... (month + year)

RETURN THIS FORM TO:
MagPi Magazine Subscriptions, Select Publisher Services Ltd, PO Box 6337, Bournemouth BH1 9EH

☐ Please tick this box if you DO NOT want to receive any other information from Select Publisher Services Ltd.

☐ Please tick this box if you DO NOT want to receive any other information from other companies.

☐ Please tick this box if you DO NOT want to subscribe to The MagPi newsletter.

MUSEUM IN A BOX

The core team consists of George Oates, Tom Flynn, Adrian McEwen, and Charlie Cattel-Killick.
museuminabox.org

Quick Facts

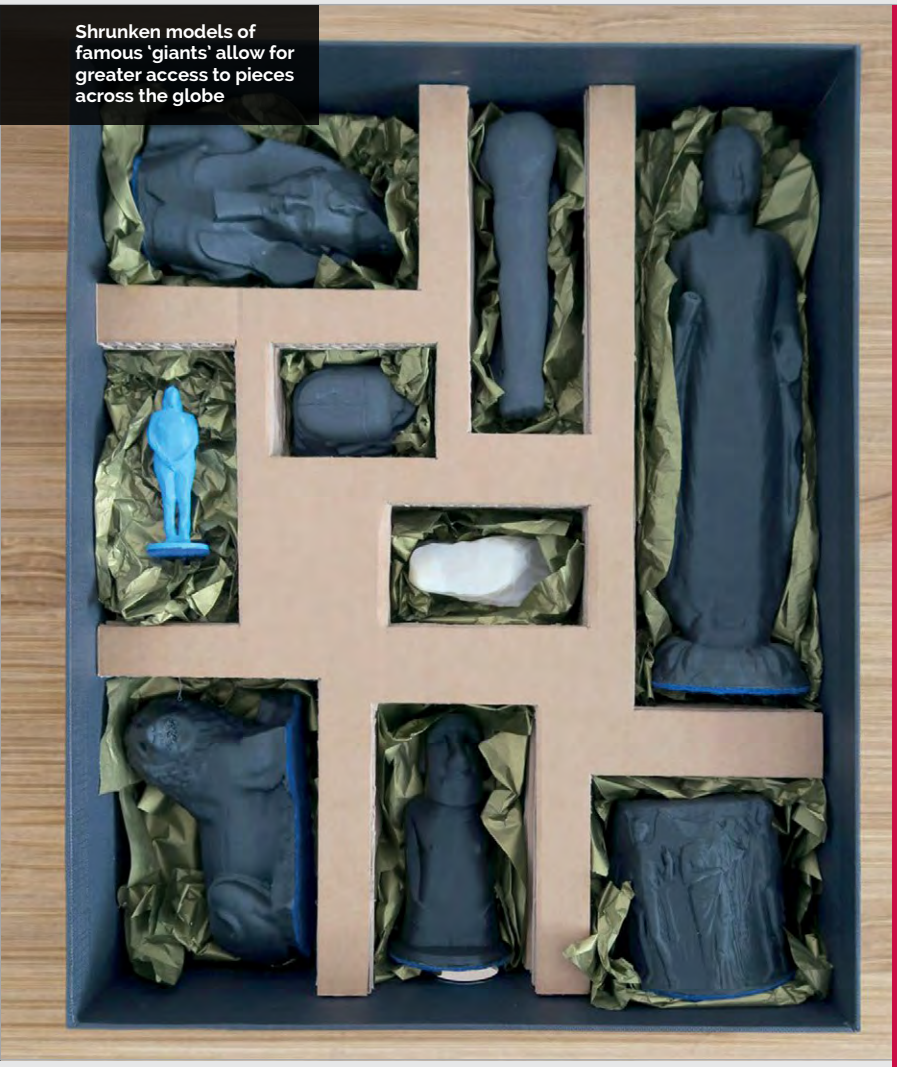
- Started as an R&D project by Good, Form & Spectacle
- The team are based in London and Liverpool
- The first box was built at Somerset House
- All the original pieces were scans from the British Museum
- The company incorporated in October 2015

MUSEUM IN A BOX

Museum in a Box gives us the chance to experience incredible pieces of art, artefacts, music, and more at our fingertips, anywhere in the world

You can visit Hoa Hakananai'a at the British Museum, London. A Moai, you'd likely recognise him as one of the Easter Island statues. He's 2.4 metres in height, nearly a metre wide, and is estimated to weigh around four tonnes. He sits on a high plinth surrounded by text regarding both his own history and that of his fellow Moai and it's fair to say that, unless you can go to the British Museum in person, you're unlikely to see him visiting your local museum, school, or library on loan.

Now imagine holding a smaller version of Hoa Hakananai'a in your hand. He fits perfectly on your palm and allows you to feel the texture of his surface and the shape of his features. You can pass him around, reposition him, and even drop him if you lose your grip. And as you 'boop' him on the top of the Museum in a Box's Raspberry Pi-powered 'brain', he starts to tell you the story of his



Shrunken models of famous 'giants' allow for greater access to pieces across the globe

sea voyage from Easter Island, of the history of his creators, and his first encounter with the explorer Captain Cook in 1774.

Bonding with history

This idea of forming stronger connections with objects through touch and sound is the heart of the Museum in a Box objective. Through 3D-printed models and wooden sculptures, 2D images such as postcards and photographs, and 'do it yourself' feedback cards, the team aim to break the disappointing, yet often necessary, Do Not Touch stigma of museums, allowing everyone the chance to get to grips with a history they may otherwise miss out on.

On the technical side of the project, the Museum in a Box consists of a wooden box 'brain' that houses a Raspberry Pi and RFID reader. There's also a volume control for playback and lights to indicate when the unit is powered and functional. Each object, whether it be 3D or flat, is kitted out with its own unique RFID tag. When placed upon the box, the tag is read by the reader beneath the surface and informs the Raspberry Pi to play back the appropriate sound file via a built-in speaker system.

> "The team aim to break the disappointing, yet often necessary, Do Not Touch stigma"

LETTING OBJECTS SPEAK FOR THEMSELVES

>BOX-01

magpi.cc/2iLlwqQ

The Planets

Box Prototype No. 13 – The Planets. Seven identical wooden balls sit within a plain black box. When 'booped', each plays a track from the USAF Heritage of America Band's rendition of *The Planets* suite by composer Gustav Holst.

>BOX-02

magpi.cc/2iLrd88

Frogs in a Box

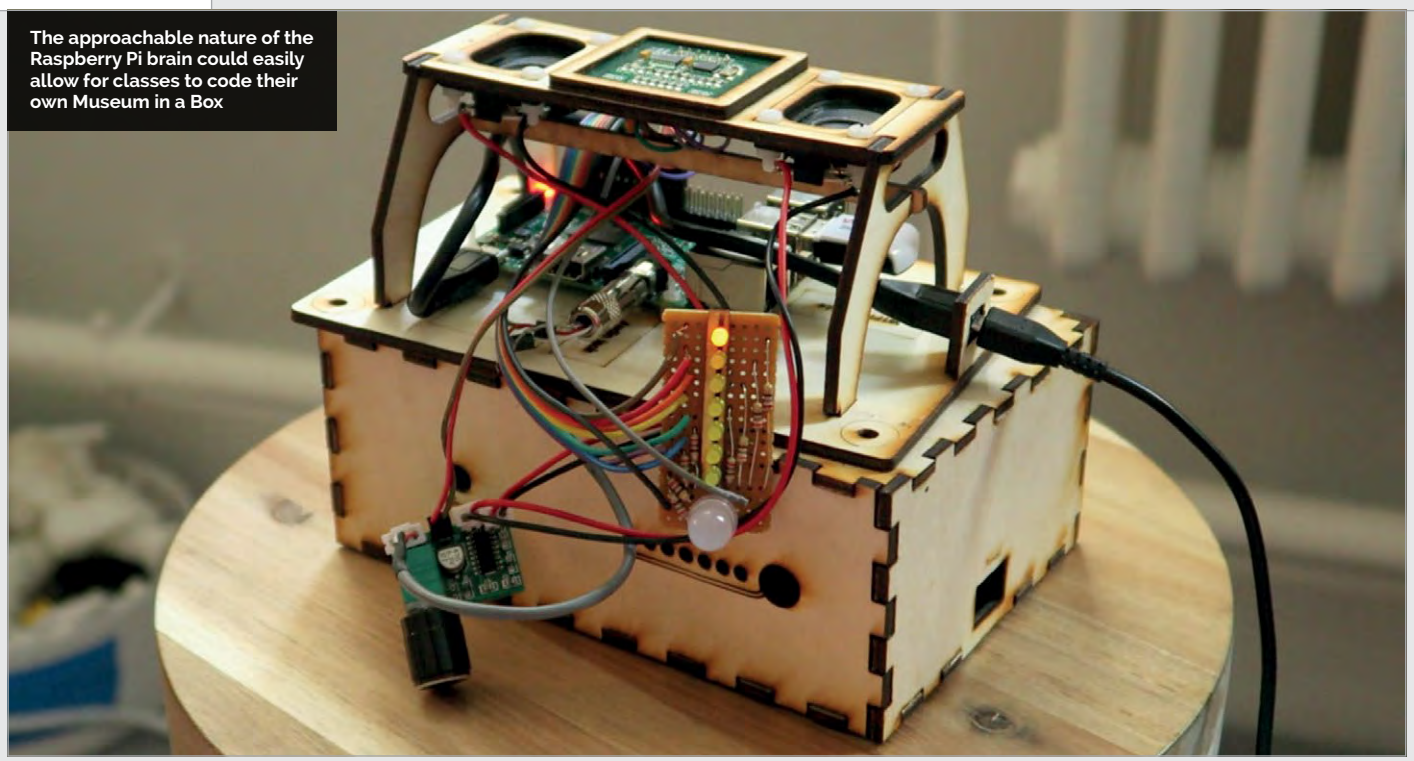
With a somewhat 'flatter' approach, this box uses postcards to play the various calls of the illustrated amphibians, all recorded by "a mid-20th-century herpetologist called Charles". Frogs in a Box is part of a larger pilot programme in conjunction with Smithsonian Libraries.

>BOX-03

magpi.cc/2iLmaES

Ancient Egypt: Daily Lives

The 3D-printed items within the Ancient Egypt box depict objects used in the daily lives of those living over 2,000 years ago. After it was shown to families at the British Museum, the team upgraded the box to include cards that offer visitor feedback when booped.



The approachable nature of the Raspberry Pi brain could easily allow for classes to code their own Museum in a Box

The sound files vary from object to object. Some play music, such as the Planets prototype box; home to seven identical wooden balls, it is set to play one of the seven tracks of Gustav Holst's *The Planets* suite. While the balls lack any differentiation on the outside, their insides are unique. Boop a ball and the experience is overpowering, thanks in part to the beautiful effect of the USAF Heritage of America Band as the sound of their instruments swells into life. Now imagine if these balls were 3D prints of the instruments that played famous solos. Or imagine if the ball was the 3D-printed bust of a famous composer who narrates the inspiration behind their work as their music plays in the background, captivating young and old musicians alike.

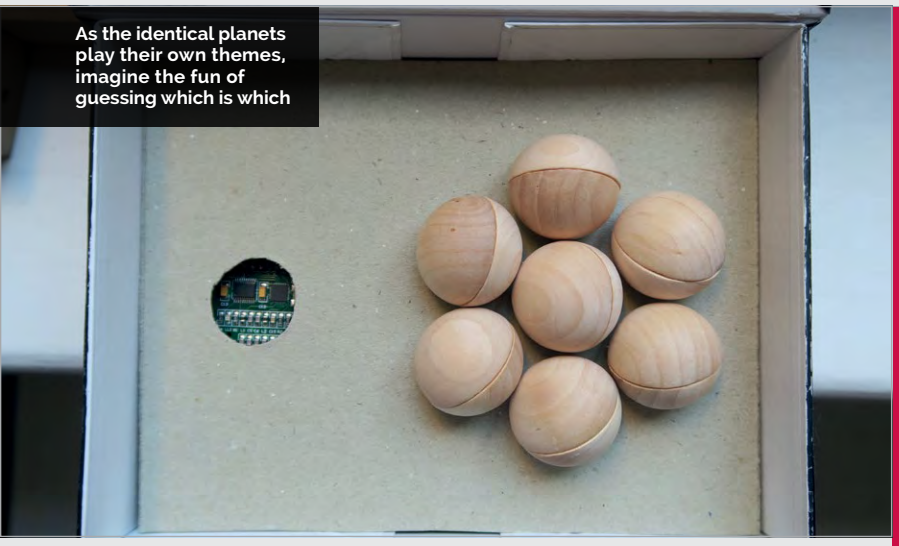
Inspired education

It's difficult to experience this project without finding yourself coming up with hundreds of your own ideas for its use.

Given this inevitable excitement and enthusiasm, the second intention of the project comes to life. Imagine if a classroom had its own Museum in a Box, and, alongside their brain box, the students had a second box allowing

RFID TAGS

A radio-frequency identification (RFID) tag is an inexpensive way of giving an item its own unique, readable code. Often confused with the NFC technology that allows you to touch your mobile phone to pay at your local supermarket or coffee shop, an RFID tag is a fairly simple piece of technology that can either be active, maintaining its own power source to allow you to use it to find lost keys, or passive, taking power from an RFID reader to let you into a locked building with the touch of a plastic card.



As the identical planets play their own themes, imagine the fun of guessing which is which

them to record content onto their own RFID tags. Maybe the class collects objects from their local town and records the items' history and their own thoughts directly to them. Once complete, they're able to send the objects to a different class in a different part of the world and share their experiences with others. Perhaps a museum records narration to a postcard and sends it out to teachers for them to share with their students. The nature of the Raspberry Pi allows for multiple data files, so a first boop could ask a question and further boops could provide more information to continue class discussion.

The team – whose core members include CEO and co-founder George Oates, an interactive designer and project manager; co-founder and designer Tom Flynn, an expert 3D creative; technical lead Alan McEwan, and junior designer Charlie Cattel-Killick – use their combined expertise to build constantly upon the core concept of the project. A great multi-platform use of the box is a recent integration with augmented reality. Cards depicting the fire-damaged ruins of pieces from South London's Cuming Museum can be scanned with the 'Augment' mobile app and brought back to life before your eyes. Pair this with an RFID tag and these lost pieces of history suddenly tell their story. Download the app and try it for yourself with the image here: **magpi.cc/2iLuCUA**.

At last count, the team had 13 prototype boxes, with others already commissioned for use in programmes such as the Smithsonian Libraries' 'I See Wonder' pilot; and with such incredible scope for use in education, it's not hard to see Museum in a Box thriving.

The British Museum displays a mere 1% of its collection, leaving so much kept from sight

>BOX-04

magpi.cc/2iL8V6W

Statues of Women in London

Only one audio recording survives of British writer Virginia Woolf and as her 3D replica is booped against the brain of the box, she begins to speak to you. "Words, English words, are full of echoes, of memories, of associations…"

>BOX-05

magpi.cc/2iLkM4Z

The Museum Fire

Alongside the box, the team also work on rather impressive content for an augmented reality app called Augment. When South London's Cuming Museum burnt down, this technology allowed for destroyed pieces to be revived with fascinating results.

>BOX-06

magpi.cc/2i8WMwf

The Brain Extension

The extension permits the recording of new content to tags, giving students, professors, museum visitors, and others the chance to personalise objects with their own unique viewpoints and information.

LAUREN EGTS

Lauren started interning for NASA at 15. She has been in NASA labs for three years.
magpi.cc/2hnYTuJ

NASA BOOT CUFF SURFACE IMAGER

Quick Facts

- The ALSCC was used by Neil Armstrong in Little Rock Crater
- The BCSI can be used all day on a single charge
- The ASLCC took stereograph photographs
- The BCSI captures 1920×1080 digital photographs
- The ALSCC was 11 inches (approx 28cm) from the floor)

When Lauren Egts spent time interning at NASA, she used a Raspberry Pi to put a moon camera on her foot

Lauren Egts has spent the last three years interning for NASA, where she has worked on a range of quirky and exciting projects.

The BCSI (Boot Cuff Surface Imager) is her latest project. It uses a Raspberry Pi Camera Module and a gyroscope to take close-up photographs of the floor.

This project is a modern take on a NASA device called the ALSCC (Apollo Lunar Surface Closeup Camera). This camera went all the way to the moon and back.

"The ALSCC was designed by Thomas Gold, a British scientist," explains Lauren.

"[He] created it to view surface material from the moon. The images that the ALSCC brought back to Earth show what the surface of the moon looks like from eleven inches up. This is about the equivalent of an Apollo astronaut putting his faceplate to the lunar surface and looking at it."

NASA don't want close-ups of the moon's soil for the fun of it, says Lauren: "It was used to discover more information about the moon's surface and structure. These are things that lunar soil brought back to Earth can't show because it has been disturbed.

"The ALSCC had a right and left lens that took a picture at the same time and from those 2D pairs of images, scientists

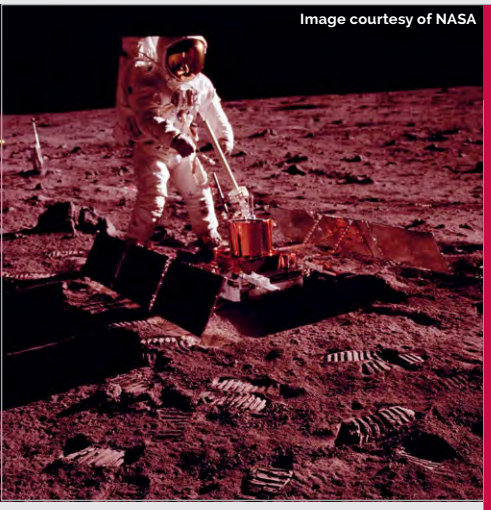


Image courtesy of NASA

Above Astronauts used the ALSCC manually while walking around the lunar landscape

back on Earth were able to create 3D pictures of the moon's surface."

The 3D effect enabled scientists to analyse the surface of the moon back on Earth.

The BCSI

"The concept of BCSI is very similar to ALSCC," says Lauren. "They both take up-close pictures of soil."

Thanks to modern components, the BCSI is much smaller than the ALSCC. And smart components like the Raspberry Pi automate the process of taking photographs.

"The BCSI is meant to sit on the ankle of an astronaut's boot," explains Lauren. It automatically detects deceleration with a gyroscope each time an astronaut takes a step. "Once deceleration is detected, the program that runs on the Raspberry Pi triggers the Camera Module.

"Of course, on the moon or Mars, there wouldn't be as much gravity, so the trigger value to take a picture would be much different," she reveals. "But through testing I've determined a consistent value that returns clear pictures."

Working for NASA

Lauren worked hard to get her NASA internship. "I met my mentor, Herb Schilling, at a Mini Maker Faire where I was presenting on Scratch," says Lauren. "He invited me to come shadow him at his lab at NASA for a day."

FLOOR PHOTOGRAPHY

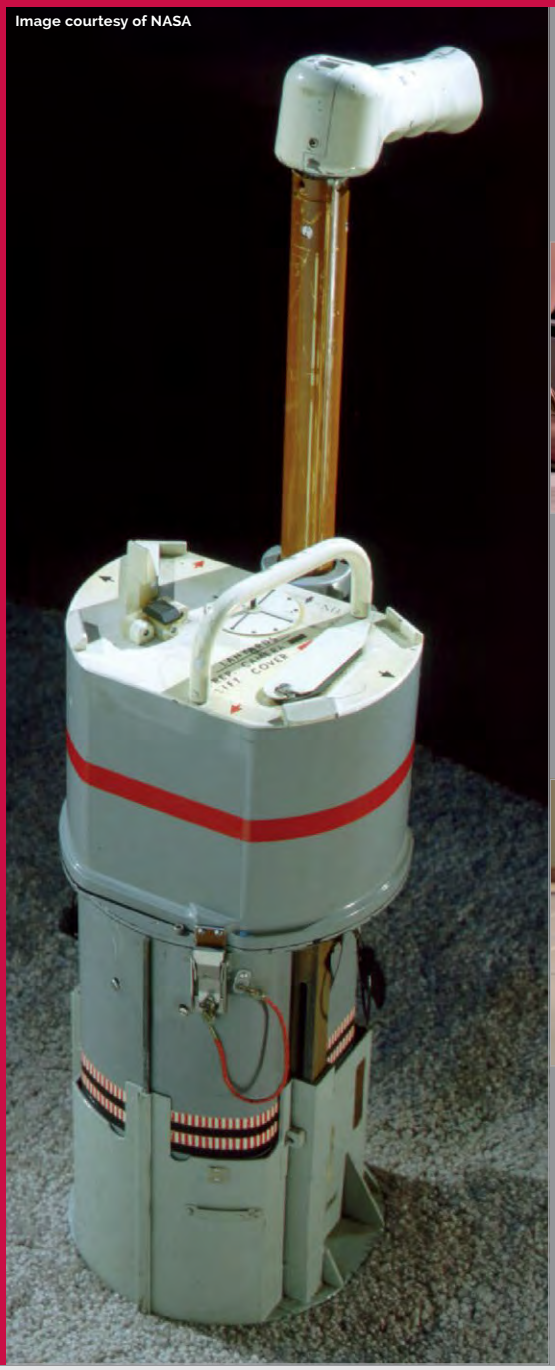


Image courtesy of NASA

>STEP-01
The ALSCC

The original NASA camera was used on the moon to take stereoscopic photographs of the floor, so scientists at home could view the undisturbed lunar surface.

>STEP-02
The test sock

The BCSI updates this idea. A Raspberry Pi, Camera Module, and gyroscope are attached to the astronaut's boot. To test the concept, Lauren first attached the equipment to a thick sock.

>STEP-03
Walk around

As the foot is lifted and put down, the gyroscope looks out for the deceleration. With every foot movement, it takes a single photograph of the floor. The project was tested in the gravity of Earth; Lauren then determined how it would work on the moon and Mars.

After the shadow, he invited her to intern at the lab for a few weeks that summer; she came back the next summer and the next.

"A friend of my mentor had heard about the ALSCC and thought that it would be interesting to create a modern version of it," recalls Lauren.

"I had a good idea of what I wanted the design to look like. I decided to sacrifice a pair of socks to science and use them to create the next version of BCSI. The sock prototype was much more robust but couldn't be used in the field, so I ended up getting an off-the-shelf pair of rainboots and attaching the BCSI to one of them."

Lauren tells us that working at NASA is a lot of fun. "I think my favourite part of the lab is the people. We range in age from high schoolers to graduate students, but we all have one thing in common: we love technology."

JOHANNES JAEGERS

A robot maker with many years of electronics and tech project management under his belt.
german-robot.com

GERO

A real humanoid robot powered by a Raspberry Pi, the German Robot (GeRo) is like a more advanced Rapiro

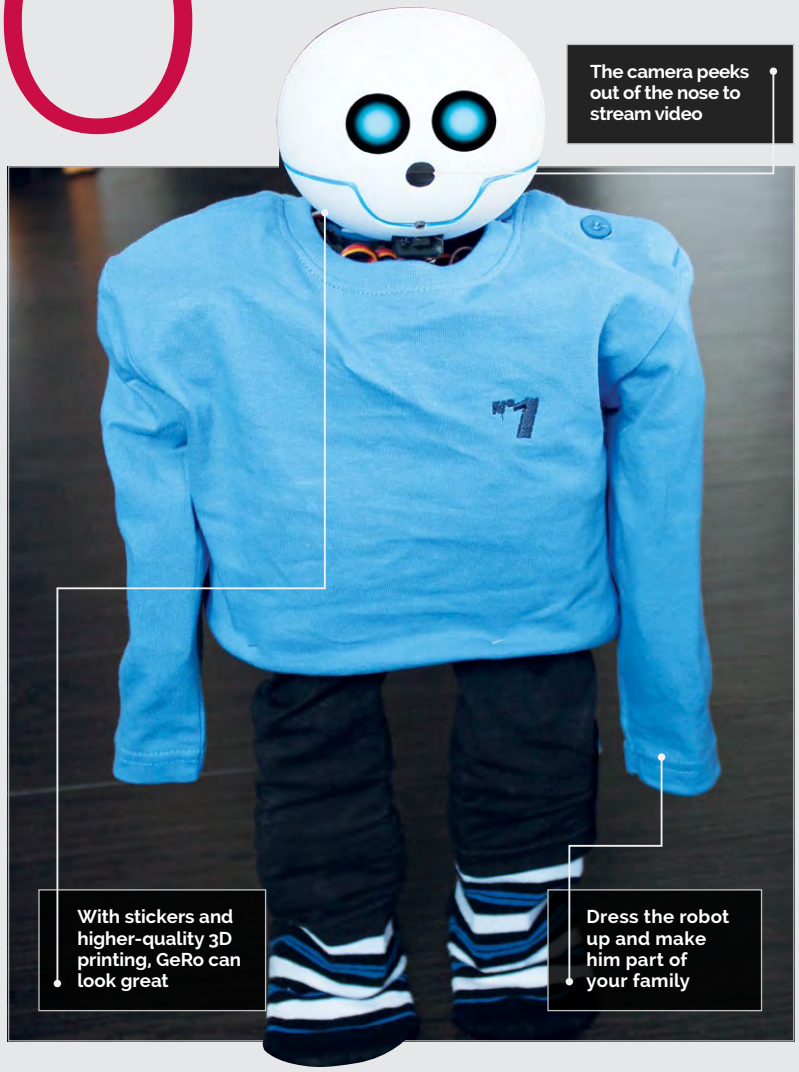
Quick Facts

- There are plans for a bigger, more manoeuvrable robot
- This is the first Raspberry Pi project Johannes has made
- Full build instructions, including an SD image, are available online
- With 3D printed parts, you can easily modify how GeRo looks
- GeRo has five more servos than Rapiro

"I've been fascinated by humanoid robots for a long time," Johannes Jaegers, the creator of GeRo, tells us. "I finally decided to make one myself when I saw the Rapiro project on Kickstarter."

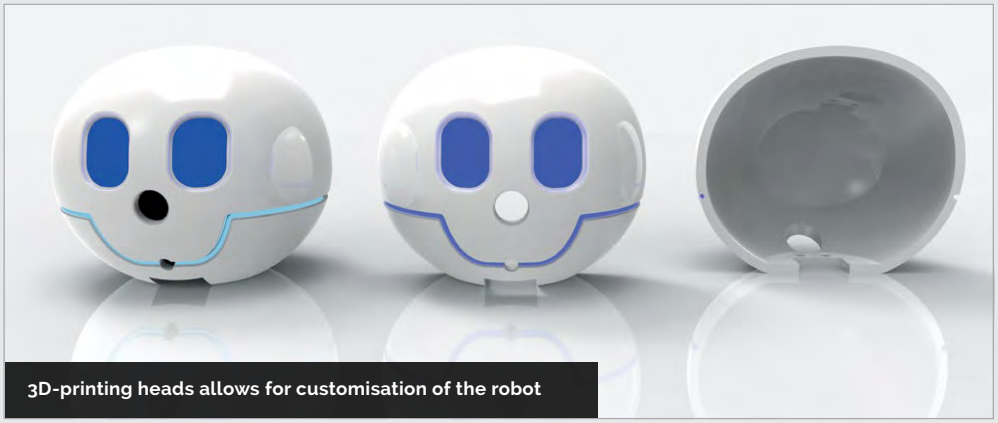
From this fascination comes GeRo, the German Robot. It's a walking, talking robo-project running on a Raspberry Pi. GeRo is fully programmable by the user and can even be controlled remotely over the network from any web browser. The camera and microphone also stream to the webpage connected to it.

For this type of complex robot, a Raspberry Pi is not always used, with some makers using Arduino controllers to do the job. "I preferred a computer over an Arduino," Johannes explains. "Using a computer like the Pi allows for many more features like the camera and microphone connection, ROS integration, sensor data evaluation, reactions, and so on. I chose the Raspberry Pi because it was cheap and it had a great online community which supports a lot of functions like software PWM which is needed to connect the 17 servos."



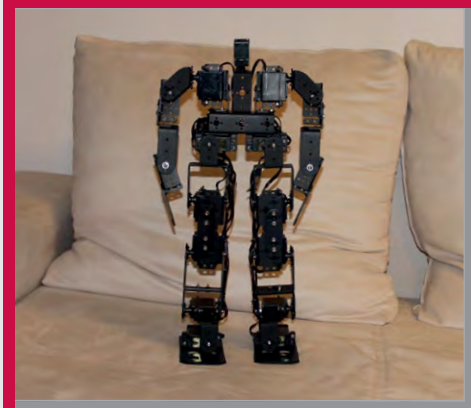
ROS is the Robot Operating System, an open-source OS specifically for creating and programming robots. It's used in the GeRo, making it fairly unusual among Raspberry Pi robots, which usually just work with Raspbian.

It took Johannes about two years on and off to build the robot. It consists of a metal skeleton and 3D-printed parts, along with a custom PCB to power the Raspberry Pi and connect to the 17 individual servos and the battery. There is

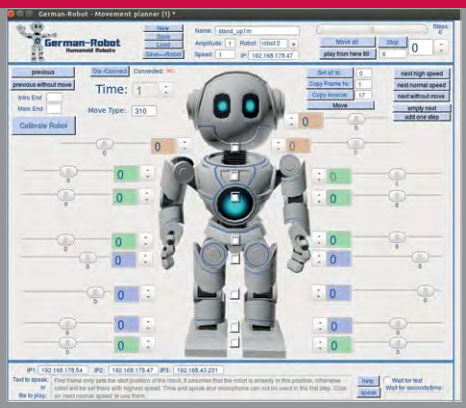


3D-printing heads allows for customisation of the robot

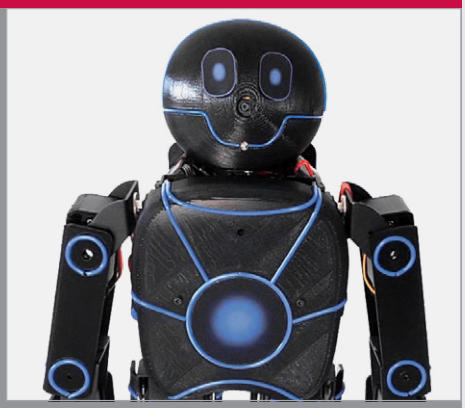
PROGRAM YOUR ROBOT



>STEP-01
Get connected
Once built, the robot needs to be connected to the wireless network so you can access it via a software interface on a separate computer.



>STEP-02
Slide around
The robot can be controlled with different sliders instead of pure code. You can also access the text-to-speech functions from here.



>STEP-03
Mobile control
Log into the browser page on a phone and you can stream the camera view and control the robot's movement without having to boot up a PC.

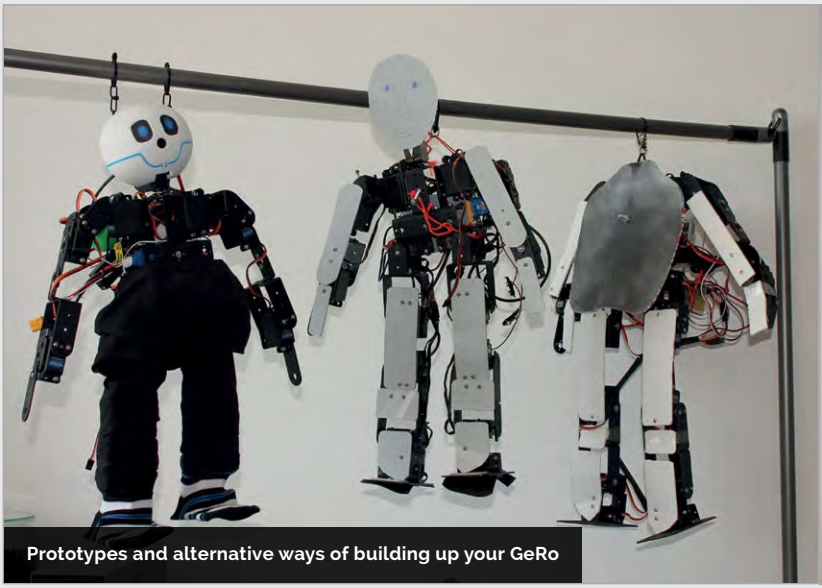
a full list of parts online you can read, which gets a little extensive when you get down to the nitty-gritty: **magpi.cc/2fQ7k0r**.

"In general it works quite well," states Johannes. "However to do anything with the robot, the movements and the text for speech have to be defined. I wrote software to make the positioning of each servo as easy as moving a slider; however, it still takes some work and time to perform, especially for the walking sequence. The robot has no servos to turn the legs around, so it's limited to walking in a straight line for now, and cannot easily turn. However, it can stand up from any position and speak any sentence in different languages."

Johannes has recently made the process of building the robot, along with some of the materials, accessible online from **german-robot.com**, and he reckons you can build the robot for less than 500 euros (about £430/$530). It's not overly complex, but Johannes reckons it's best to have some level of understanding of robot hardware and software and how they can interact, along with some soldering experience to boot. "I really enjoyed working with the Raspberry Pi, especially because it supports ROS and made it so cheap and easy to get direct hardware access from Python and C and thus build complex systems."

> "It took Johannes about two years on and off to build the robot"

Below You'll need quite a lot of parts to build this robot

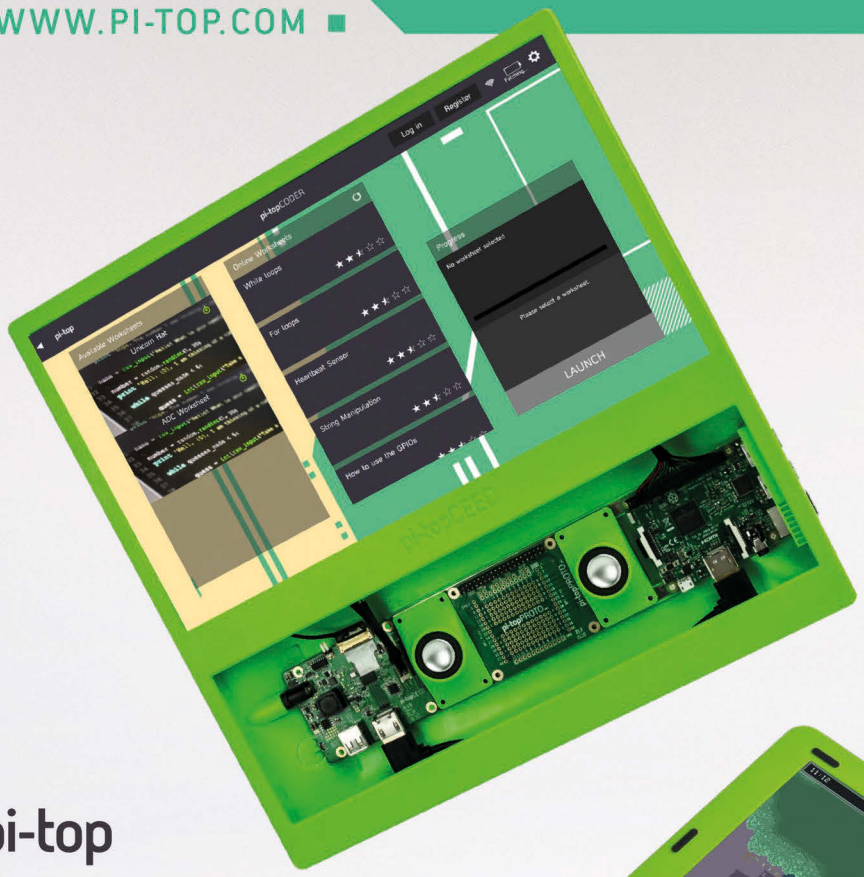


Prototypes and alternative ways of building up your GeRo

pi-top

LEARN
PLAY
CREATE

■ WWW.PI-TOP.COM ■



pi-topCEED

- Adjustable Viewing Angles
- 14" HD Screen
- Modular Components

$114.99
without Raspberry Pi
ex VAT

pi-top

- 10 Hour Battery Life
- 13.3" HD Screen
- Modular Components

$264.99
without Raspberry Pi
ex VAT

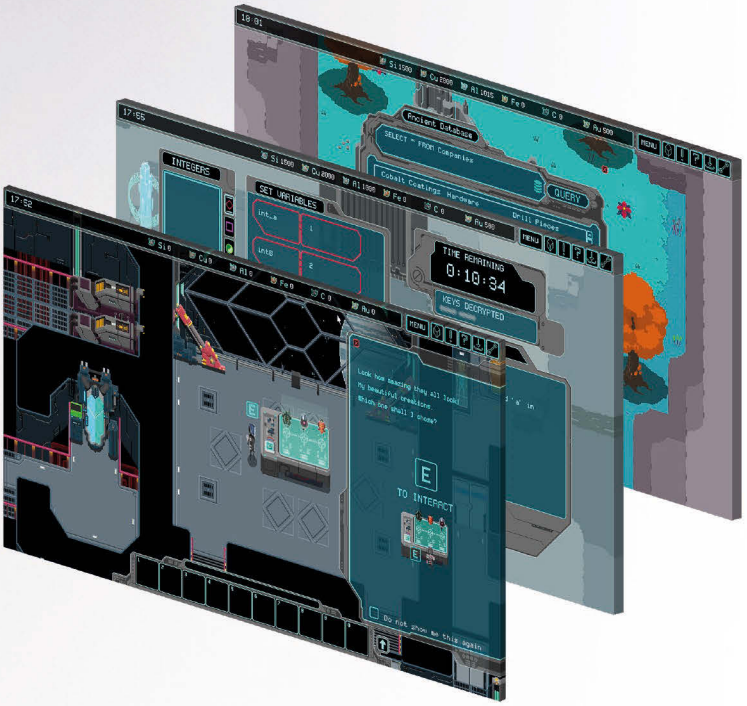
Available in green or grey colours

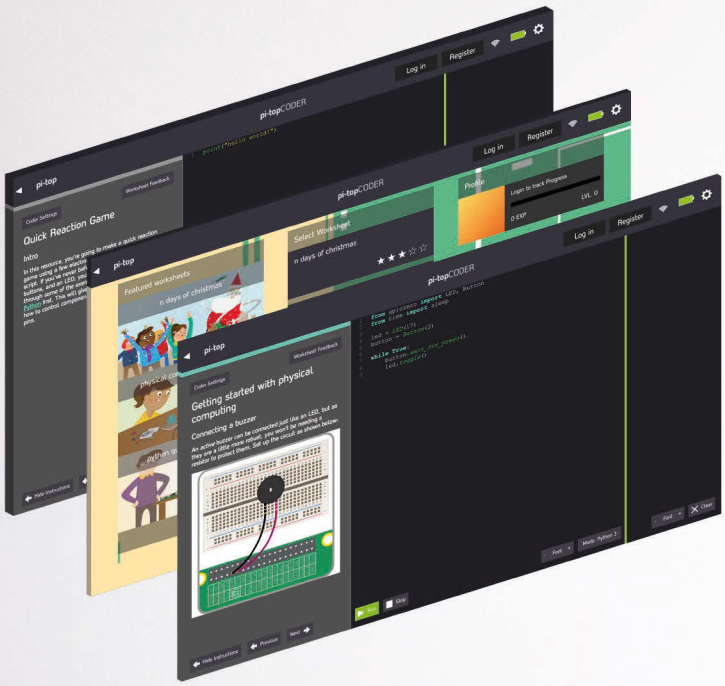
pi-top | Worldwide shipping available in green or grey at www.pi-top.com
Stay up to date with our latest news by following our social media

CEED**universe**

CEED**universe** is a world of fantasy grounded in computing reality! After crash landing on a strange new planet you will first encounter 'drag and drop' coding puzzles that improve your computational thinking skills.

Discover more about the planet you've landed on and the civilisation that used to inhabit it while learning harder and more complex concepts in fun and engaging ways! Before long you'll be writing your own conditional statements, loops and functions. Find out more about CEED**universe** at www.pi-top.com

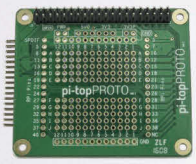




pi-topCODER

Influenced by the workflow of makers and hackers, pi-top presents **pi-top**CODER - an integrated code editor which allows you to learn, write and test code all in one view. With intuitive syntax highlighting, dynamic views and customizable themes it makes for a versatile learning tool for your projects.

pi-topCODER also has every Raspberry Pi Foundation lesson plan created and will track and save your progress as you go through dozens of fun hardware and software projects.



pi-topPROTO

pi-topPROTO is a HAT compatible Add-on Board for your **pi-top** or **pi-top**CEED that allows you to prototype electronics. Create a Weather Station, HAM Radio, Heart Rate Monitor, or integrate any Arduino based maker kits into your own Raspberry Pi compatible prototyping board!

pi-topSPEAKER

Give a voice to your **pi-top** device with **pi-top**SPEAKER!

- Modular design, attach up to three in a row to give true stereo sound.
- 2W per module
- Left, Right and Mono mix selection
- High quality SPDIF digital audio from HDMI
- I²C controlled

Our Official Distributors
adafruit
HANDS-ON SCIENCE
KIWI electronics
KUBII
MODMYPI
MICRO CENTER computers & electronics
Pi-Shop.ch
RS
sparkfun ELECTRONICS

SWITCH TO THE COMMAND LINE

By using the command line, you are able to work faster and smarter. Discover how to get started today...

You'll Need

- Raspberry Pi
- Raspbian with PIXEL

Unless you grew up in the 1980s or earlier, the chances are that you are accustomed to using only GUIs (graphical user interfaces) and desktop environments.

There's really nothing wrong with GUIs, and Raspbian comes with a rather fine one called PIXEL.

But beneath the icons sits a whole other world: the command line. This is where your real computer is. With the command line, you're not locked into doing just what desktop applications enable you to do. You can do just about anything to your computer, and you can do it much faster.

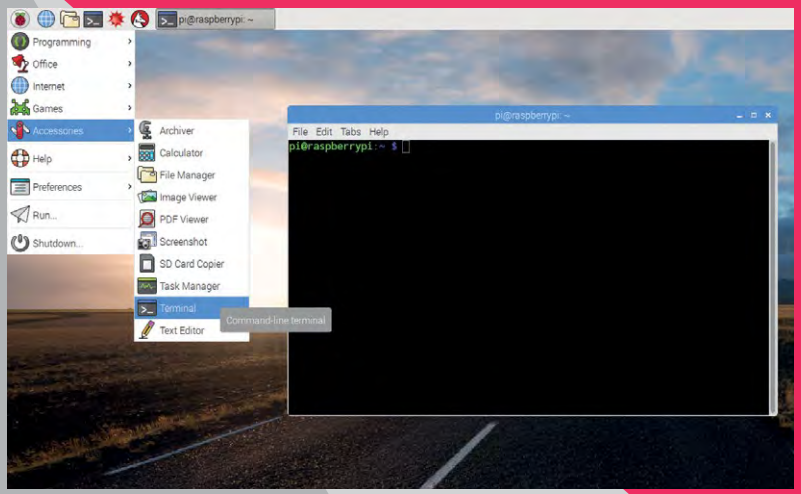
Think of it like driving a car. If you've only ever used a GUI then you're driving an automatic. The command line is like switching to manual. It's a lot trickier, but you get far more control and feel like a proper driver.

The command line can be daunting for newcomers, but it really needn't be. With just a few commands, you can master the command line.

Typing commands

When you boot a Raspberry Pi, you start by default inside the PIXEL desktop interface.

The fastest way to get access to the command line is through the Terminal app.



Most people access the command line through the Terminal app in the PIXEL desktop

Click on the Terminal icon in the top menu bar (or choose Menu > Accessories > Terminal). A window opens with a black background and some green and blue text. You will see the command prompt:

```
pi@raspberrypi:~ $
```

You are now at the command line. You enter commands using the text interface. Enter **`echo Hello World`** and press **RETURN**, you'll see 'Hello World' printed on the line. Below this is another **$** prompt, ready to accept another command.

Most users get to the command line via the Terminal app, but there is another way known as 'virtual console'. Press **CTRL+ALT+F1** and the desktop will vanish. A black screen appears, displaying 'Raspbian (or Debian) GNU/Linux 8 raspberry pi tty' and below it, 'raspberrypi login'. If you are not automatically logged in , enter **pi** and press **RETURN**, then enter your password (**raspberry** by default).

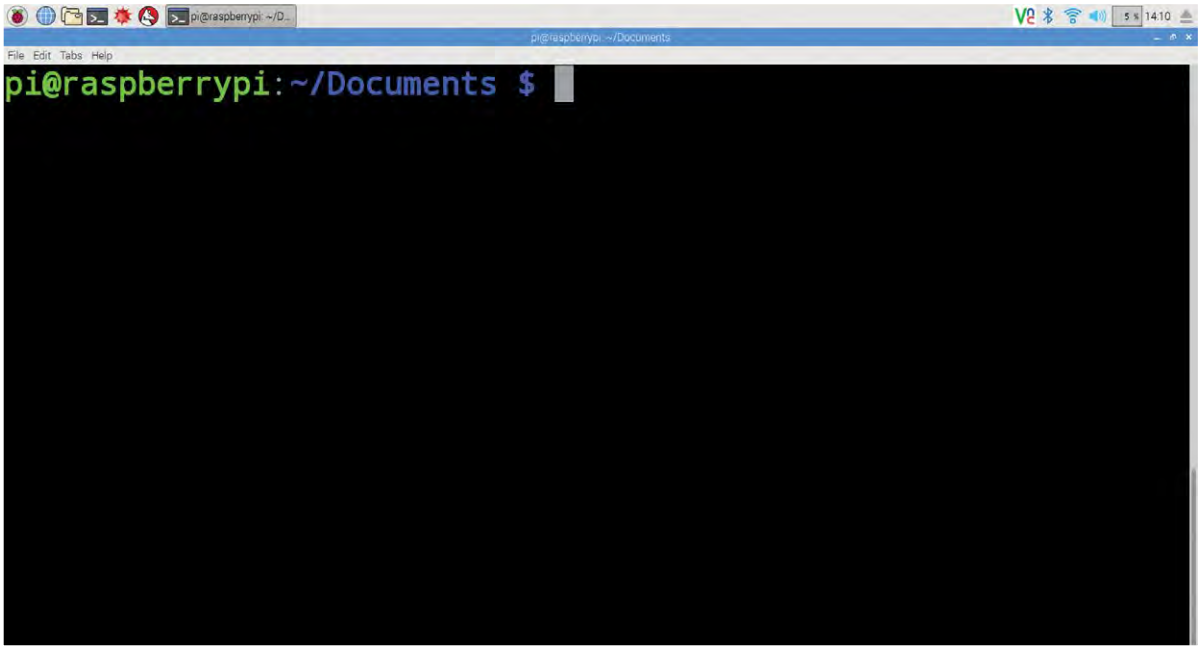
You can now use the command line in full-screen mode. You can get back to the PIXEL desktop using **CTRL+ALT+F7** and switch back to the virtual console using **CTRL+ALT+F1**. Additional virtual consoles can be accessed using **CTRL+ALT+F2** to **F6**. Each has its own login and operates independently.

If you prefer the command line, you can boot Raspbian directly to the command line instead of the PIXEL desktop. Open Raspberry Pi Configuration (Menu > Preferences > Raspberry Pi Configuration). Change the Boot setting to 'To CLI' and click OK. Now when you reboot, you'll start in the command line (enter **`startx`** to boot into the PIXEL desktop).

Locate yourself

The first thing you need to learn is how to find out where you are. You are in your home folder by default. Enter the following command and press **RETURN**:

```
pwd
```



pi@
The first part of the command line is your user name followed by an @ symbol. You can see this on the command line by entering `whoami`.

raspberrypi
After the @ comes your host name. It is the name of your computer: 'raspberrypi' by default.

~/Documents
After the host name is your current working directory. This displays just '~' when you are in your home folder.

$
The dollar sign shows you're operating as a normal user .

This command is short for 'print working directory' and it tells you where you are. The command line will return `/home/pi`.

The home folder is the same one that appears by default when you open the File Manager app. You view the files and directories inside the working directory using the list (`ls`) command:

```
ls
```

Here you'll see the same directories (or folders) that are in the File Manager app: **Desktop**, **Downloads**, **Documents**, and so on.

The file path

Before going any further with directories, you need to understand the file path and the difference between a 'relative' and 'absolute' path.

Files are placed inside folders (which are called 'directories' in the command line). In the visual GUI, you can see these as folders that you open, revealing files and more folders. Your home folder contains a **Documents** directory, and inside that are three more folders: **Blue J Projects**, **Greenfoot Projects**, and **Scratch Projects**.

```
/home/pi/Documents/Scratch\ Projects
```

In the file path above, the first slash is the root of your hard drive. Here you have a directory called **home** that contains all users. In here is another directory called **pi** (that's you), and inside that is another directory called **Documents**, and inside that is one called **Scratch Projects**.

The eagle-eyed reader may have noticed the weird backslash character: '\'. You can't have spaces in file names, so you use a backslash followed by a space at

UNDERSTAND THE LANGUAGE

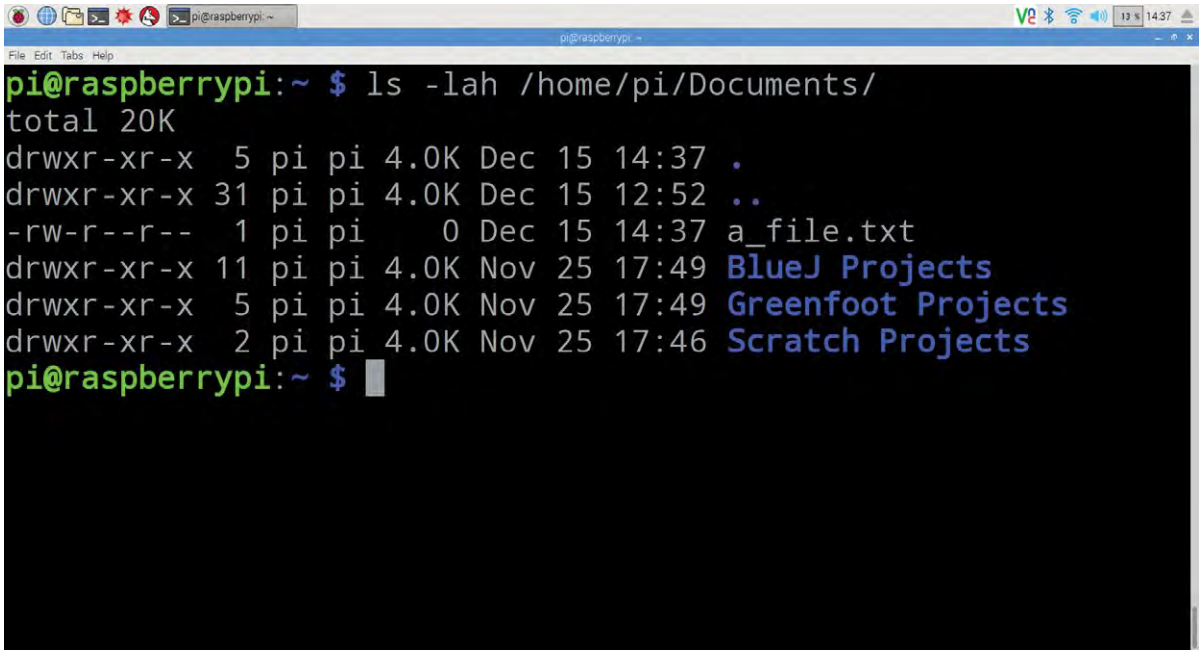
There's a lot of confusing jargon thrown around related to the command line. Terms like command line, shell, and terminal are often used interchangeably. Each has a precise meaning...

- **TERMINAL:** This is the program you use to access the command line from the PIXEL desktop in Raspbian (its full name is LXTerminal).
- **CONSOLE:** This is a physical terminal display with a keyboard. Consoles used to be empty computers that connected to a large mainframe computer.
- **VIRTUAL CONSOLE:** These are virtual versions of a physical console. In Linux, you have multiple virtual consoles accessed using **CTRL+ALT** and the function keys.
- **TTY:** Teletypewriter. In Linux, tty is used to display which virtual console you are using: tty1, tty2, and so on.
- **COMMAND LINE:** This is the text-based environment in general or the specific line you are working on. The command line starts with a dollar sign ($), known as the 'prompt'.
- **SHELL:** This is a command-line interpreter. It surrounds the computer's kernel (hence the name). To get to the kernel, you go through the shell. The shell interprets your text commands and turns them into code the kernel understands.
- **BASH:** This stands for 'Bourne Again Shell' and is the type of shell used in Debian (the version of Linux upon which Raspbian is based).

ls
The first part of a command is the command itself. Here we have ls, which lists the contents of a directory.

-lah
After the command come options. These start with a hyphen and are typically single letters. Each modifies the command. Here we have 'l', 'a', and 'h'. These stand for long listing mode, all files, and human-readable.

/home/pi/Documents
The final part of the command is the arguments. These are often file names or file paths. Here we are listing an absolute (direct) path to the Documents directory. If you omit the argument, it'll display the contents of the current directory.



the command line. Most of the time you'll also use the **TAB** button to quickly enter long file names (see 'Tab completion').

As mentioned, file paths come in two types: relative and absolute. Relative paths are 'relative' to your working directory, which is **/home/pi/** when you start. Entering **ls** alone shows the contents of the current directory. You can view the contents of a directory inside your working directory using **ls** and its name:

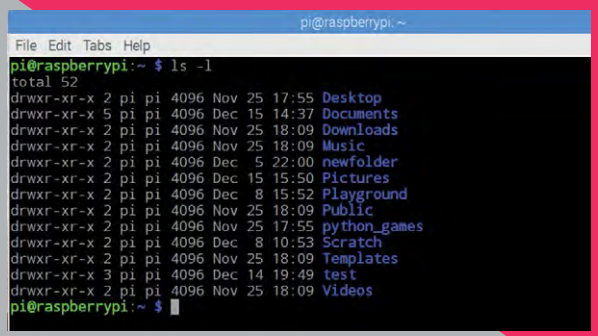
```
ls Documents
```

You can also view the contents of the directory above you using two dots (..):

```
ls ..
```

This displays files relative to where you currently are in the file system. If you moved into the **Downloads** folder and entered **ls Documents**, it'd cause an error, because there is no **Documents** directory inside the **Downloads** folder.

An absolute path, on the other hand, always starts with a slash '/', which is the root directory (the base of your hard drive).



The command line can be used to manage files and directories on your system

Enter:

```
ls /
```

...to view the root directory. Here you'll see all the directories and files that make up Linux. You'll see directories like **bin** (for binaries), **boot** (files used to start up the system), and **home**, which contains your user folder.

Enter:

```
ls /home/pi
```

...and you'll view the contents of your home folder, just as if you had entered **ls** from within it.

You can use absolute paths no matter what your working directory might be, because they always start from the root.

Moving around

Up until now we've stayed in the home folder and looked around using **ls**. You move from one directory and another using the **cd** (change directory) command:

```
cd Documents
```

Now enter:

```
pwd
```

...and you'll see a different working path: /home/pi/Documents. To move back up a directory (known as the 'parent' directory), you use two dots.

```
cd ..
```

Enter **pwd** again and you're back in the home folder. Now try it using an absolute path. Enter:

```
cd /
```

...and you'll be in the root directory. Enter **ls** to view the folders at the base of your hard drive. Enter:

```
cd /home/pi
```

...to move back to your home folder. There's a shortcut for this:

```
cd ~
```

The tilde (~) character is a shortcut for your home folder. You can use it at the start of an absolute directory too. For instance, entering:

```
cd ~/Downloads
```

...moves to your **Downloads** folder no matter where you are in the system.

Files

Throughout the file system, you'll find various types of files. A good selection is in the **python_games** folder, so enter:

```
cd ~/python_games
ls -l
```

The **-l** part (an option) makes it use 'long listing' mode, which displays items with lots of information:

```
-rw-rw-r--  1  pi  pi  973  Jan 27 2015
4row_arrow.png
```

From left to right, each item is:

- **Permissions:** The users and groups that can access a file.
- **Hard links:** The number of files that are linked to this file.
- **Owner:** The person who owns the file. Usually either **pi** or **root**.
- **Group:** The group the file belongs to.
- **File size:** The name of the file.
- **Modification:** When the file was last changed.
- **File name:** The name of the file.

The most obscure item is the list of letters and hyphens that make up the permissions. The first letter will be either a '-' or a 'd' depending on whether it's a file or a directory. Our **4row_arrow.png** is a file, so it's a '-'.

After that are nine letters arranged into three groups of three (see **Fig 1** overleaf):

- **Owner:** Typically this will be the person who created the account.
- **Group:** This is a group of users. You have only one group, **pi**, by default, containing just one user (also **pi**).
- **Other:** These are users from other systems.

Each of the three groups contains letters: 'rwx'. These letters are always in that order and each is either the letter or a hyphen. A letter indicates that the person, group, or other has access to read, write, or execute the file. A hyphen means they don't have that level of access.

Some examples include:

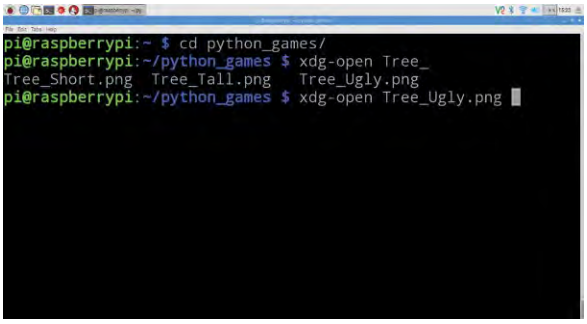
- **rwx** read, write, and execute
- **rw-** read, write, but don't execute
- **r-x** read and execute
- **r--** read only

Now that you've discovered how to move around the file system from the command line, it's time to learn what else you can do.

Take command

One of the first commands you need to learn is **mkdir**. This stands for 'make directory'. Move to the home folder and create a new directory called **test**:

```
cd ~
mkdir test
cd test
```

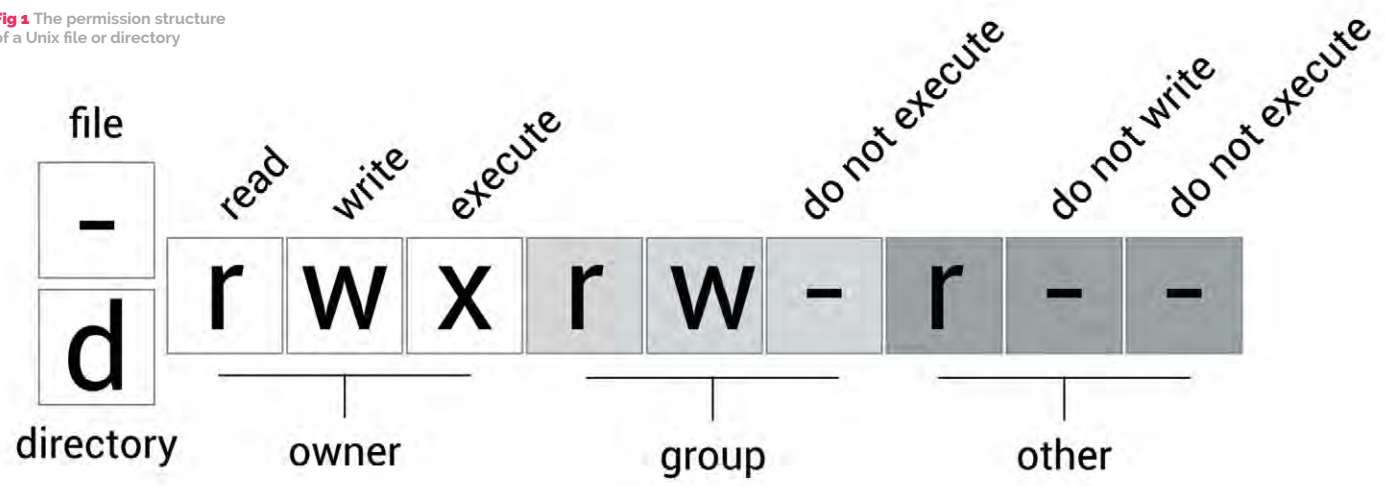


TAB COMPLETION

The single most useful tip you'll ever learn for the command line is tab completion. Pressing **TAB** at any time when entering a working path attempts to complete the file or directory name for you. Use **cd python_games** and enter **xdg-open Tr**, then press the **TAB** key. Notice how it fills it out to **'xdg-open Tree_'**. There are three files starting with Tree. Quickly press **TAB** twice and you'll see them: **Tree_Short.png**, **Tree_Tall.png** and **Tree_Ugly.png**. Enter **S**, **T**, or **U** and press **TAB** again to fill out the whole file name. Press **RETURN** to open it.

Tab completion can be invaluable for entering long file names packed with letters, numbers, and punctuation.

Fig 1 The permission structure of a Unix file or directory



To create files, you use a rather odd command called **touch**. Officially, touch is used to update the modification time of files (reach out and 'touch' them). If you touch a file, it updates the time next to it to the current time.

Few people use **touch** for that. A happy by-product of the command is that if you touch a file that doesn't exist, it creates an empty file. Enter:

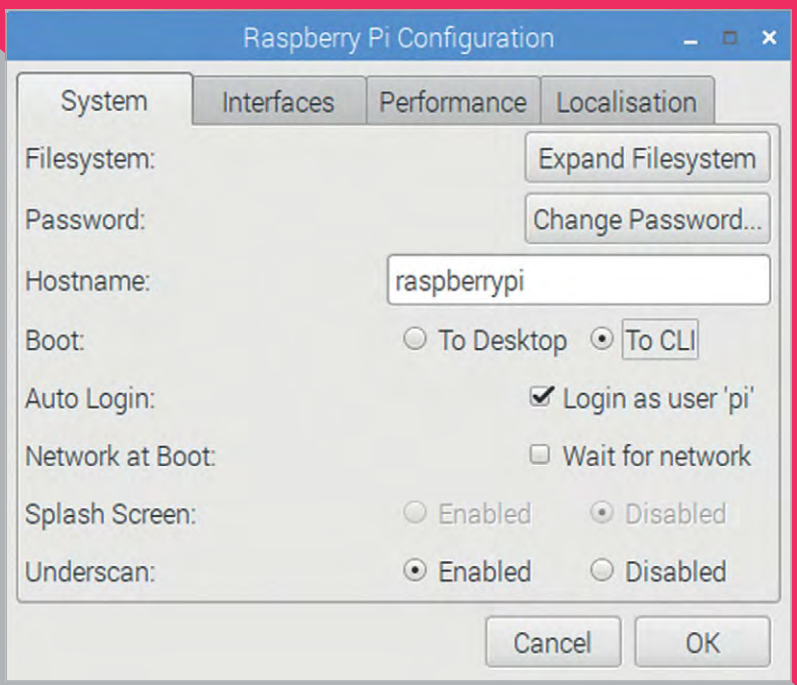
```
touch test.txt
```

You'll create a blank file called **test.txt**. Enter **ls -l** and you'll see the new file along with all its details. Notice that the file size is 0. This is because the file is completely empty.

We can edit the contents of the file using a text editor called nano:

```
nano test.txt
```

You can set Raspbian to boot into the command line (instead of the graphical interface) from the Raspberry Pi Configuration settings



You can enter and edit text in nano, but the Save and Exit commands predate the traditional **CTRL+S**, **CTRL+W** standards. Enter a single line, 'Hello World', and press **CTRL+O** followed by **ENTER** to save the file. Now press **CTRL+X** to exit.

Enter **ls -l** again; you'll notice that the file size has changed from 0 to 12. This is one for each letter (including space) and a newline marker at the end (you can see this character using **od -c test.txt** if you're curious).

Let's try deleting files. This command removes the file:

```
rm test.txt
```

Now move up to its parent directory and use another command, **rmdir**, to remove the empty **test** directory.

```
cd ..
rmdir test
```

Unfortunately, you'll rarely use **rmdir** to remove directories, because they typically have files inside them. You can see how this fails using these commands:

```
mkdir test
touch test/test_file.txt
rmdir test
```

It will say **rmdir: failed to remove 'test': Directory not empty**. The solution is to use **rm** with the **-R** option. This option stands for 'recursive' and means it goes inside each directory (and subdirectory) and removes every file and directory. Be careful when using **rm -R**, as it's permanent and deletes everything inside. Enter:

```
rm -R test
```

The **test** directory and everything inside will disappear. Unlike the desktop environment, there is no Wastebasket in the command line. When you remove files, they're instantly gone for good.

Options

Most commands have options that affect how they work. It's common to use these three options with the **ls** command:

```
ls -lah
```

Options start with a single hyphen '-' followed the letter for each option. The three options used here are:

- l = long listing format
- a = all including hidden files
- h = human-readable (makes large file sizes more readable)

Options are case-sensitive. So **ls -l** and **ls -L** are two different things (small 'l' is long listing format; large 'L' is dereference mode). Sometimes options are listed out in full. These start with two hyphens and have a single hyphen for spaces. This command is the same as **ls -lah**:

```
ls -l --all --human-readable
```

But it's more common to see (and use) the single letter approach.

Sudo

Sudo stands for 'substitute user do', although it's often also called 'superuser do'. If you have multiple users on your system, it can be used to perform commands as another user.

It's mostly used to get root access to your Linux installation. There is an account that controls your Pi user, called 'root'. This is an all-powerful account, which can change just about anything on your system.

Your default account can view files in the root of the hard drive, but it can't create or delete files at the root. Enter:

```
cd /
touch test.txt
```

You'll see **touch: cannot touch 'test.txt': Permission denied**. However, enter:

```
sudo touch test.txt
```

...and the **test.txt** file will be created on the root of your hard drive. You can see it using **ls -l**.

Now try to delete it:

```
rm test.txt
```

It will say **rm: remove write-protected regular empty file 'test.txt'?"** Enter **Y** and it'll say **rm: cannot remove 'test.txt': Permission denied**. You need to use sudo to remove the file:

```
sudo rm test.txt
```

You can see why sudo is such a powerful tool. Without it, you couldn't install software using apt or apt-get. But with it, you can remove or delete vital system files. Enter **ls /bin** and you'll see many programs (known as 'binaries') used by Linux. These include the **ls** command you just used. Accidentally deleting these files could make your system unstable.

So use sudo with care. In Raspbian you don't need to enter the password to use sudo. On many other Linux systems, however, you will be asked for the password before you can use sudo.

What's up, man?

There are lots of ways of getting help inside the command line. The first command you should turn to is **man**. This stands for 'manual' and gives you instructions on Linux commands and tools. Enter:

```
man ls
```

...and you'll see the manual for the list command. Notice under the SYNOPSIS it says:

```
ls [OPTION]... [FILE]...
```

This shows you the structure of the command. Almost all commands are in the 'command, option, argument' structure, although some arguments have more than one [FILE] argument (such as copy, which requires a source and destination file).

Press the space bar to move down the instructions. Here you will see a list of all the available options. With **man**, you can get detailed information on just about every tool on the command line. You can even get a manual for the **man** command with:

```
man man
```

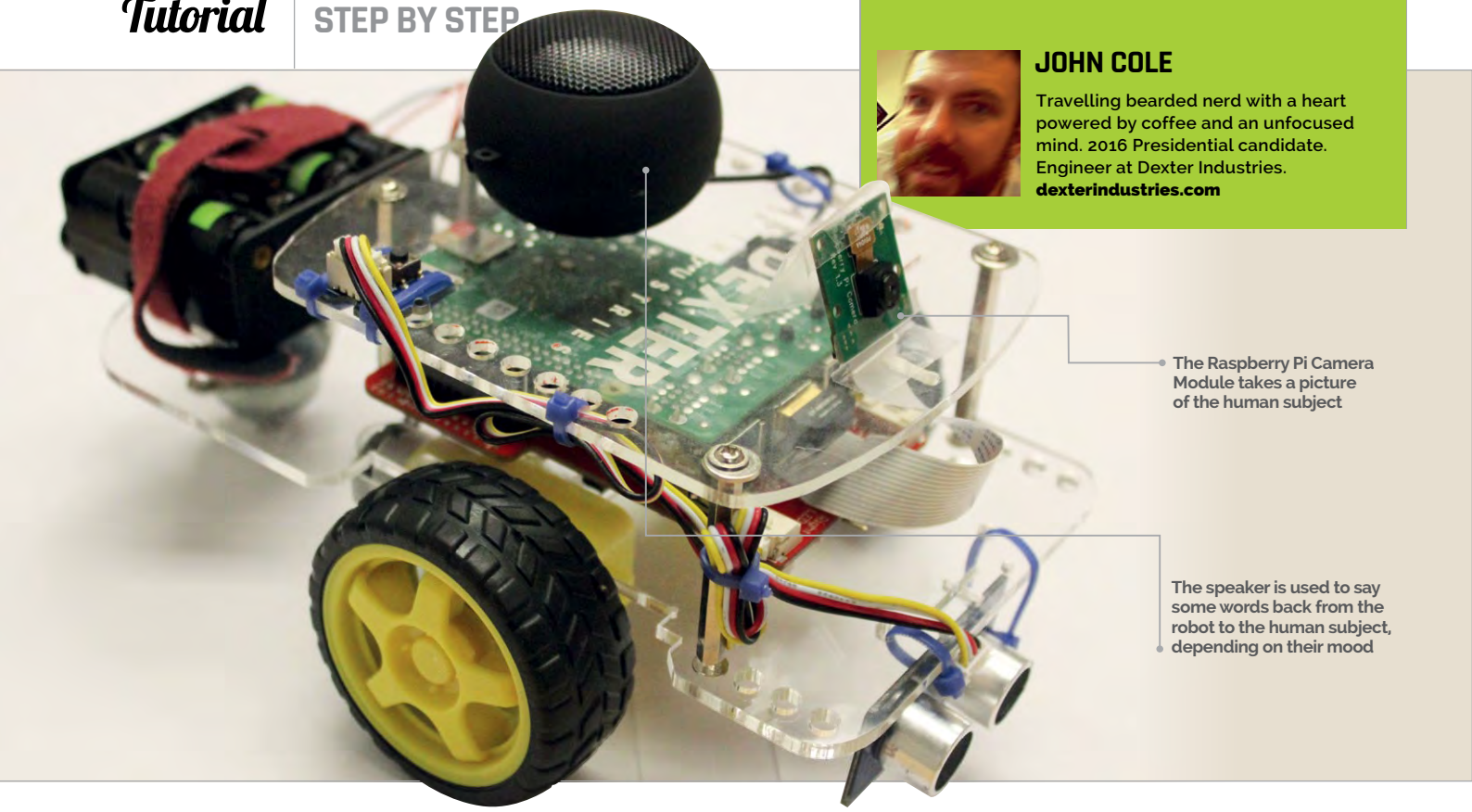
If you need a quick reminder on how to use a command, try using it with **-h** or **--help** as an option:

```
touch --help
```

...tells you what options are available with the **touch** command. You can use this with many command-line tools to get a quick refresher on how they work.

Moving from a GUI to a command line is a vital skill for hackers and coders. Everything on your computer, from programs to preferences, is stored in the file system somewhere. Learning to use the command line makes you a more capable Raspberry Pi user.

So, the next time you make a file, move a file, or delete something, don't head to the File Manager. Open Terminal and perform tasks from the command line. Soon it'll be second nature.



JOHN COLE

Travelling bearded nerd with a heart powered by coffee and an unfocused mind. 2016 Presidential candidate. Engineer at Dexter Industries.
dexterindustries.com

EMPATHYBOT: THE RASPBERRY PI ROBOT THAT READS EMOTIONS

In this project we'll build a Raspberry Pi robot with emotional intelligence using the Raspberry Pi Camera Module to tell if a person is happy, sad, angry, or surprised

You'll Need

- GoPiGo kit magpi.cc/2hA8f6i
- WiFi dongle magpi.cc/2hAgOOC
- Speaker magpi.cc/2hAgCi7
- Ultrasonic sensor magpi.cc/2hAazds
- Button sensor magpi.cc/2hAhqE1
- Raspberry Pi Camera Module magpi.cc/28ljlsz

What if you could build a robot with some empathy? This tutorial will show you how to build a robot that can read a person's face for emotions. In this tutorial we'll use the Raspberry Pi, the Raspberry Pi Camera Module, a GoPiGo, and a speaker to read some human faces and say something appropriate back. Our robot will roll up to its human master, take a picture, analyse the face with a free Google Cloud Vision account, and then say something appropriate to the human's current mood.

>STEP-01

Build the GoPiGo

You will need a small Phillips head screwdriver; the rest of the parts are included with the GoPiGo. There are written directions and video instructions on the Dexter Industries site showing how to assemble the GoPiGo robot kit. Attach the Raspberry Pi, and add eight AA batteries to power the GoPiGo. While programming the stationary GoPiGo, you may want to use a USB power supply to power the Pi.

>STEP-02

Add the camera

Add the Raspberry Pi Camera Module to the GoPiGo. In this tutorial, we'll use one of the slots on the top of the GoPiGo canopy to support the camera. You can also use the servo accessory to move the camera side to side.

>STEP-03

Add the speaker

The speaker can be mounted to the top of the GoPiGo using a few zip ties. Place the speaker on top of the GoPiGo, and connect the speaker aux cable to the Raspberry Pi headphone port. You can charge the speaker using the Raspberry Pi USB cable.

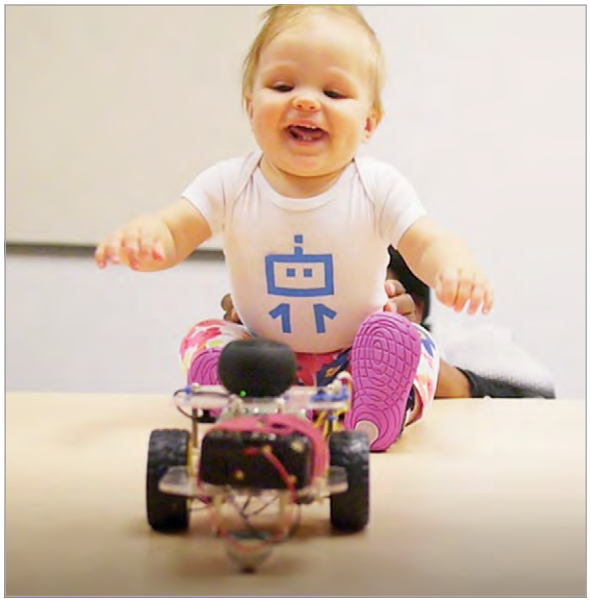
>STEP-04
Attach the sensors
We use the button sensor to give us an easy way to start the robot up once the code is running, and the distance sensor to judge how far we are from the human. Hang the ultrasonic distance sensor from the front of the GoPiGo, using small zip ties, and attach the button to the top of the GoPiGo, also using zip ties.

>STEP-05
Setup a free Google Cloud Vision account
You can use your Gmail or Google account to set this up. At the time of publication, Google offers a 60-day free account – **magpi.cc/2hAkotI**.

>STEP-06
Create a new project
This is an abbreviated version of the setup process. You can see a pictorial walk-through of how to set up a new project in Google Cloud Vision online here: **magpi.cc/2hAhHqc**.

Create a new project called 'vision1'. Enable the Cloud Vision API for vision1.

>STEP-07
Download and install your JSON credentials
Head back to the Console in Google Cloud. Find the box titled 'Use Google APIs' and click 'Enable and manage APIs'. Click on Credentials and Create Credentials. Credentials is on the left-hand side, with a picture of a key next to it. Select 'Create a Service Account Key'. Under Service Account, select New Service Account. We'll call this 'vision'. Finally, create a role. We'll give the new role full access, so select Project and Owner to give the Pi full access to all resources. A pop-up window should appear telling you that you have created a new key,



Above This project was tested on babies and it works just fine on them too

Above This project really only works well on non-bearded humans. Beards interfere with the software.

Language
>PYTHON

DOWNLOAD:
magpi.cc/2hAgqiT

and an automatic download of the JSON key should begin. Keep track of this file! You should now use an FTP program (such as FileZilla) or Samba (see our tutorial at **youtu.be/CEYwYqkAPfA**) to move the JSON file over to your Raspberry Pi. Place the JSON file in the home directory **/home/pi**.

>STEP-08
Prepare the Raspberry Pi
We need to run a few commands to prepare the Raspberry Pi. We only need to do this one time. In the command line, run the following commands:

```
sudo pip install --upgrade pip
sudo apt-get install libjpeg8-dev
sudo pip install --upgrade google-api-python-client
sudo pip install --upgrade Pillow
sudo apt-get install python-picamera
```

Finally, make the credentials we downloaded in the previous step available to Python:

```
export GOOGLE_APPLICATION_
CREDENTIALS=filename.json
```

Here, replace 'filename' with the name of your JSON file.

>STEP-09
Run the code
If you've taken the code from our GitHub repo, or typed it out yourself, you can now run the code:

```
sudo python empathybot.py
```

Point the robot towards your human subject, press the Grove button on the GoPiGo, and let your robot start interacting with some humans!

HENRY BUDDEN

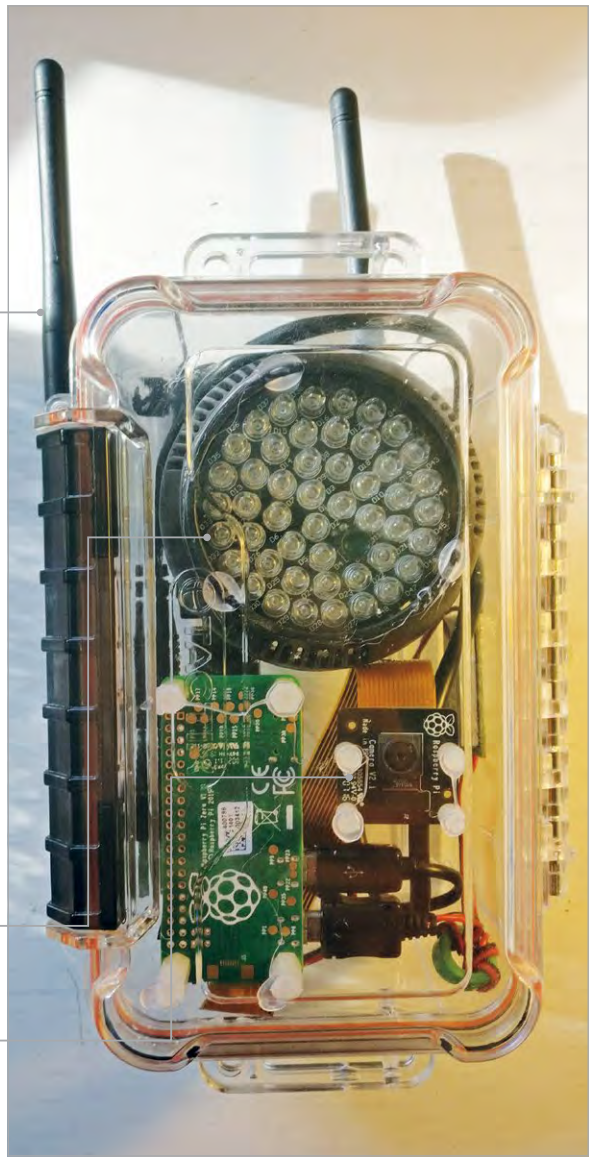
Following the release of the Raspberry Pi when he was 12, Henry taught himself to code and to use/break electronics, and has shared this process with the world on his website.
magpi.cc/2eCbaMf

MAKE A NIGHT-VISION CAMERA TRAP

Build a motion-detecting camera that streams to the internet and can see in the dark!

You'll Need

- Raspberry Pi Zero v1.3 and Raspberry Pi 1 B or B+
- Pi NoIR Camera Module **magpi.cc/2hND1tW**
- Pi Zero camera cable **magpi.cc/1V5N9dh**
- USB to micro-USB adapter
- Infrared illuminator **magpi.cc/2jxw3sT**
- 12V power supply
- Power supply adapters **magpi.cc/2hNFuEy**
- 5V UBEC **magpi.cc/2hNMv8z**
- WiFi dongle with antenna
- Waterproof case
- Two short lengths of jumper wire



With night-vision and the MotionEyeOS installed, the camera trap can be installed outside as a home security camera, a nature-trap, or a neighbour-watcher (careful!). The operating system MotionEyeOS allows the camera to be viewed online as well as alerting you when motion is detected. Built inside a waterproof case, and powered by mains electricity, the setup is also equipped with a high-power infrared illuminator to light up the darkness.

>STEP-01

Prepare case

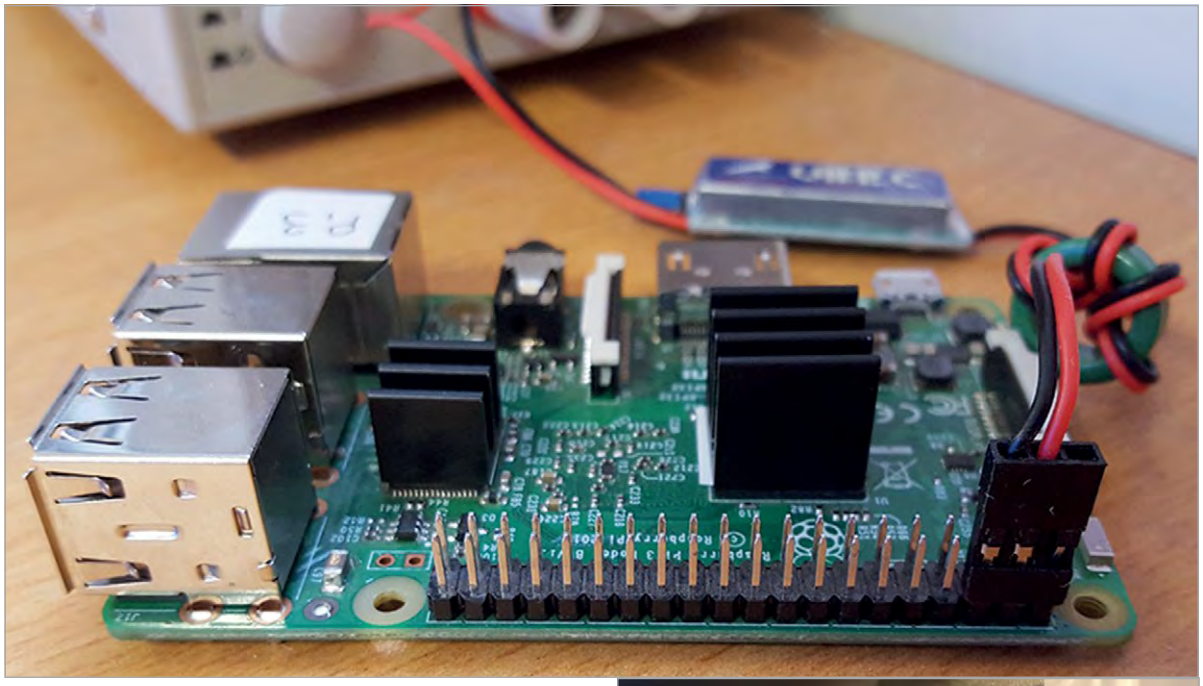
The first step is to drill all of the holes in the waterproof container of your choice (it must be transparent) in order to mount all of the necessary components. For our choice of case, we needed to drill two holes: one for the power socket and two for the WiFi antennae. Whilst the diameter and the distance needed between the holes for different WiFi antennae and power sockets may differ, we used a 7mm bit for the power socket and drilled two 10mm holes for the WiFi antennae. If the holes appear slightly too large once drilled, don't worry: the gaps will be filled with hot glue when mounting the components.

>STEP-02

Wire the power circuit

After preparing the case for the mounting of the components, the next step is to wire up the power circuit. The female barrel jack socket will be mounted inside the hole drilled earlier, with two connections from each screw terminal. One pair of wires connected to the negative and positive terminals will be connected to the male barrel jack connector screw terminals, with the other pair of wires connecting to the corresponding negative and positive inputs

Image courtesy of ModMyPi



Left Connecting the UBEC to the Pi via a pre-soldered header

on the UBEC, where the wires are already attached. Connecting the UBEC to the Raspberry Pi in order to supply power will be covered in the next step.

>STEP-03

Powering the Raspberry Pi

In order to supply power to the Raspberry Pi Zero, the UBEC can be connected to your Pi in two different ways: either by soldering the connections directly to the GPIO, or by attaching the three-pin socket on the UBEC to the Pi via a pre-soldered header. The photos show the output wires from the UBEC soldered directly onto the Pi's 5V and GND pins (pins number 2 and 6). Alternatively, you could solder a GPIO header onto the Pi and simply plug the UBEC's three-pin socket into the GPIO header – pins 2, 4, and 6 – making sure the black wire is connected to pin 6.

>STEP-04

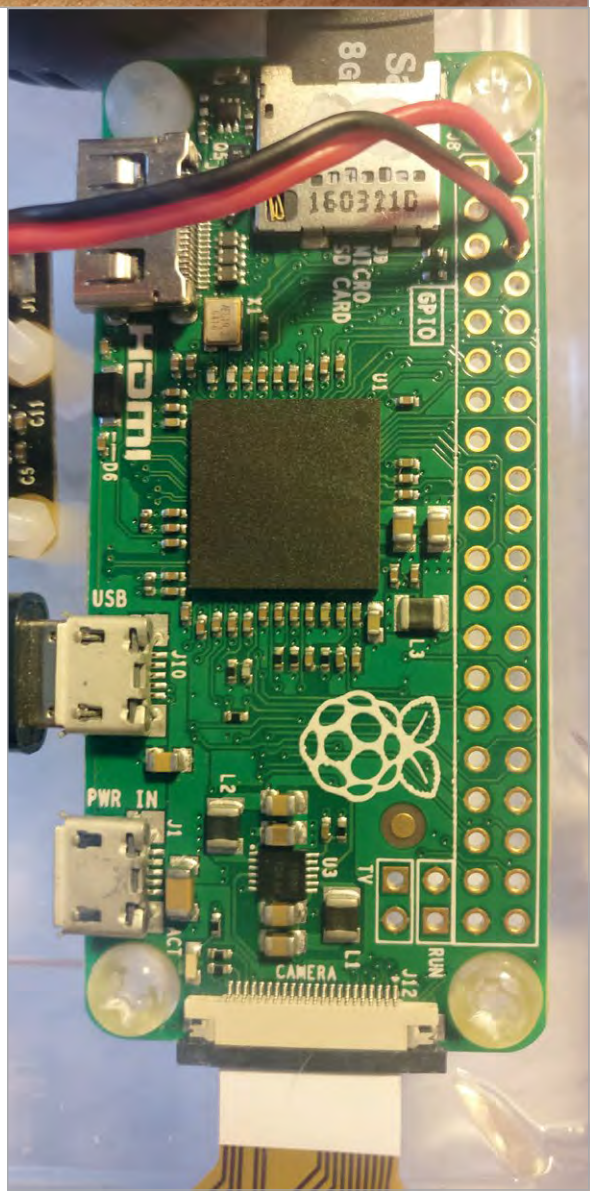
Connect the power circuit

The two input leads of the UBEC should be screwed into the corresponding terminals on the female barrel jack adapter, as well as two jumper wires connected to the correct terminals on the male barrel jack adapter. Using a hot glue gun, the female barrel jack socket can be attached to the hole that was drilled in the rear of the casing. Secondly, the UBEC can be attached to the inside of the casing in the same way. Finally, the male barrel jack adapter can be attached to the opposite side of the casing as the UBEC (see photos) to later be attached to the infrared illuminator.

>STEP-05

Mount the IR illuminator

Using a few small dots of hot glue, the infrared illuminator can be fixed to top of the case's lid, as



Left Connection from the output of the UBEC to the Pi's 5V and GND GPIO pins

MOTION IN BACKGROUND

Ensuring that there are no moving objects in the background of the shot (e.g. trees swaying) will prevent hours of blank footage from being recorded.

MOTION ALERTS

For a security setup, there is an option under the Motion Notifications menu for an email to be sent when motion is detected.

Above The inner layout and connections of the camera setup

shown in the photographs. The power lead from this module can then be connected to the male barrel jack adapter now fixed to the side of the case. More hot glue may be needed to ensure a good fix, especially accounting for the stress applied to the cable when the box is opened or closed.

CLOUD BACKUP

In order for the recorded files to be backed up, you can select Google Drive or Dropbox from the File Storage menu, clicking Obtain Key to gain permissions.

>STEP-06

Mount the WiFi and antennae

With the two holes for the antennae already drilled the correct distance from each other, the WiFi module can easily be attached to the casing by pushing the antennae through the holes from the inside of the case. Once the WiFi antennae are in position, they can be fixed (and waterproofed) by using the hot glue gun to create a circular seal around the outside of the holes. The WiFi module is connected to the Pi by connecting the USB to micro USB adapter to the socket marked 'USB' on the Pi, and the USB plug on the WiFi module.

>STEP-07

Mount the Raspberry Pi and Camera Module

In this tutorial, we mounted both the Raspberry Pi Zero and Pi NoIR Camera module to the casing using eight hexagonal stand-offs; however, this is not necessary. With some improvisation, makeshift stand-offs can be created from common materials like wooden dowel. Both the Raspberry Pi and Camera Module are attached to the casing using hot glue. As the pictures show, we mounted the Camera Module upside-down in order to remove any kinks in the camera cable. The software can correct this inversion at a later time.

Above The camera works well when positioned in a spot where interesting activity is to be expected, like a hole in a fence

>STEP-08

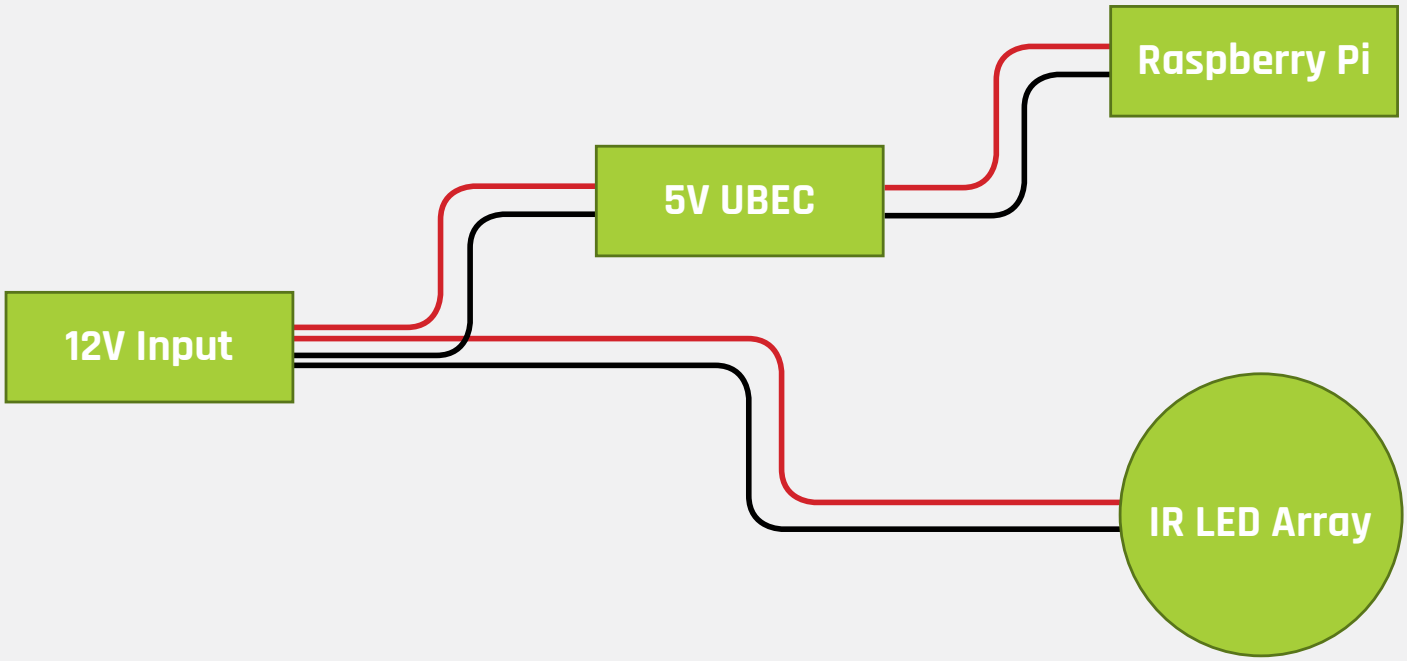
Install MotionEyeOS

On a computer, visit **magpi.cc/1UCw1Jk** and download the image shown as compatible with the Raspberry Pi Zero. Once downloaded, extract the IMG file from the `.tar.gz` archive. We did this using a program called 7zip. Finally, using a disk imaging tool like Win32 Disk Imager, write the IMG file to a micro SD card. As the OS does not create an interface on the Pi, initial setup will need to be performed on a Raspberry Pi 1 Model B or B+, as it works with the same version of the OS as the Zero and has Ethernet connectivity. This is because WiFi cannot be configured yet, unless you use one of the two methods for preconfiguring it before first boot (**magpi.cc/2iuKzhx**), in which case the older model Pi isn't needed and you can insert the card straight into the Pi Zero.

>STEP-09

Connect to MotionEyeOS

Insert the micro SD card into the older Raspberry Pi model and connect the Camera Module. Power on, ensuring it is connected to your home network via Ethernet. After approximately two minutes, use an



Above A simple diagram illustrating the power distribution circuit

IP address scanner like the mobile app Fing and look for the IP address of a device name beginning with 'meye-'. Enter this IP address into your browser's URL bar, which will direct you to the page generated by the Raspberry Pi, showing the live feed from the camera, if connected to the older model Pi. The next step will show you how to set up a WiFi connection so that the Raspberry Pi Zero (and WiFi dongle) can be used.

>STEP-10

Set up WiFi connection

When your browser is presented with the MotionEyeOS interface provided by the older Raspberry Pi model, navigate to the settings menu in the top-left corner of the screen. You may need to set up an admin user name and password beforehand. Under the Networks tab, switch Wireless Connection on, and fill in the Wireless Network Name and Wireless Network Key fields with the name and password for your wireless network, ensuring that the details are added with 100% accuracy. Once completed, click Apply and allow the system to reboot. Once rebooted, power off and insert the micro SD card into the Pi Zero in the casing, then boot it by connecting the 12V supply to the socket in the rear of the case.

>STEP-11

Configure MotionEyeOS

Connect to the Pi Zero in the same way as before, although you may need to rescan to find the IP address if it's changed. Once connected, there are a couple of settings that need tweaking before the camera can be let loose! Firstly, as the camera is mounted with a 180° flip, the Video Rotation setting under the Video Device menu should be updated to 180. Finally, if you want the camera to record video whenever it detects motion, make sure that the Motion Detection and Movies settings are switched on.

Above Putting the camera next to the front door allows you to see who's been visiting during the day!

>STEP-12

Using the camera

Now that everything is ready, the camera is free to be placed in the wild! When choosing a location for the camera, make sure that an extension power cable is able to reach it, and that the camera is pointing in a direction where you would expect some interesting animal activity, such as a gap in a fence. Finally, check that where you are placing the camera has a decent WiFi signal, otherwise the camera will continuously reboot until it finds a connection, and no footage will be collected. Good luck!

SET UP WITHOUT EXTRA PI

In order to set up MotionEyeOS without the need for an additional earlier model Pi, follow the instructions found here: **magpi.cc/1UCvYwV**

SIMON LONG
Works for Raspberry Pi as a software engineer, specialising in user interface design. In his spare time he writes apps for the iPhone and solves crosswords.
raspberrypi.org

AN INTRODUCTION TO C PART 08

THE STRING LIBRARY

Using the C string library to simplify common operations on strings

KEEP INSIDE YOUR STRING

The string library functions will not generally prevent you from writing off the end of a string; just as when using pointers, when using library functions you still need to make sure your string variables are large enough for the values you are writing into them.

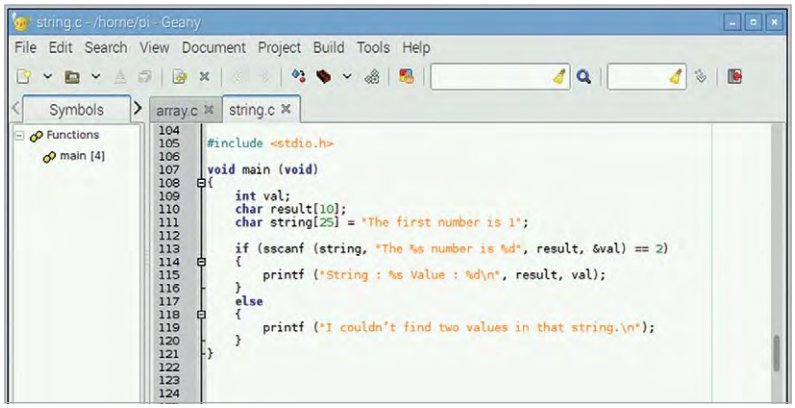
In the last instalment, we saw how to join two strings together using pointers. We're now going to do the same thing using the string handling library, which saves a lot of space:

```
#include <stdio.h>
#include <string.h>
void main (void)
{
  char str1[10] = "first";
  char str2[10] = "second";
  char str3[20];
  strcpy (str3, str1);
  strcat (str3, str2);
  printf ("%s + %s = %s\n", str1, str2,
str3);
}
```

Note the `#include <string.h>` at the start, which tells the compiler that we want to use functions from the string library.

This shows two string functions. **strcpy** ('string copy') copies the string at the second argument to the start of string at the first argument. **strcat** ('string concatenate') does the same thing, but instead it finds the terminating zero of the first argument and starts copying to its location, thereby joining the two strings together.

Below The sscanf function reads numeric values and words out of a formatted string, allowing you to parse text from elsewhere. All the arguments to sscanf must be pointers



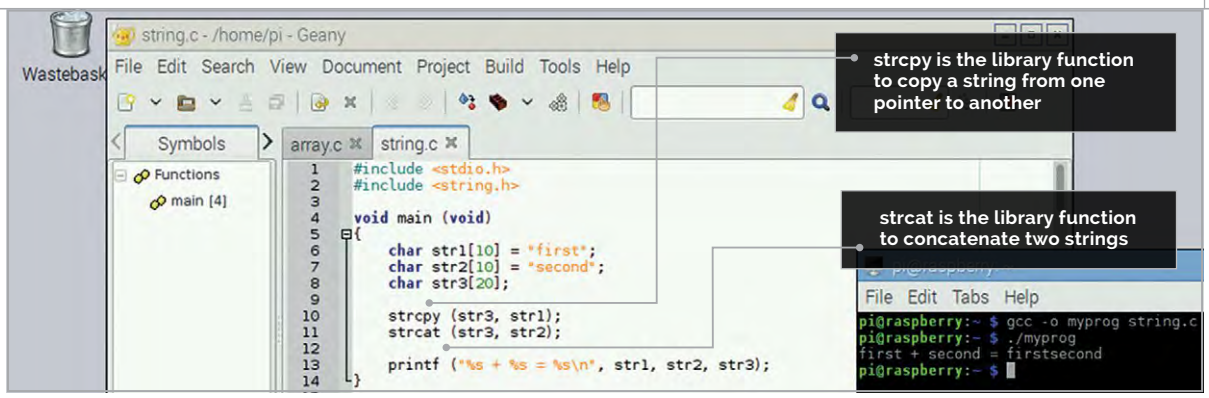
Comparing strings

We use the **==** operator to compare numeric values, but this doesn't work with strings. The name of a string is actually a pointer to a location in memory containing the string, so using **==** to compare two strings will only tell you if they are at the same place in memory.

You can use **==** to compare two **char** variables, and a string is an array of **char**s, so it is possible to write a simple piece of code that compares each character in a string in turn:

```
#include <stdio.h>
void main (void)
{
  char str1[10] = "first";
  char str2[10] = "fire";
  char *ptr1 = str1, *ptr2 = str2;
  while (*ptr1 != 0 && *ptr2 != 0)
  {
    if (*ptr1 != *ptr2)
    {
      break;
    }
    ptr1++;
    ptr2++;
    }
  if (*ptr1 == 0 && *ptr2 == 0)
  {
    printf ("The two strings are
          identical.\n");
  }
  else
  {
    printf ("The two strings are
           different.\n");
  }
}
```

The string library makes this much easier with **strcmp** (for 'string compare'):




```
#include <stdio.h>
#include <string.h>
void main (void)
{
  char str1[10] = "first";
  char str2[10] = "fire";
  if (strcmp (str1, str2) == 0)
  {
    printf ("The two strings are identical.\n");
  }
  else
  {
    printf ("The two strings are different.\n");
  }
}
```

strcmp takes two strings as arguments, and returns a 0 if they are the same; it returns a non-zero value if not.

To compare the first few characters of a string, use the function **strncmp** (for 'string numbered compare'). This works in the same way as **strcmp**, but it takes a third argument, an integer giving the number of characters to compare. So **strncmp ("first", "fire", 4)** would return a non-zero value, while **strncmp ("first", "fire", 3)** would return a 0.

Reading values from a string

The function **sscanf** ('string scan formatted') can be used to read variables from a string:

```
#include <stdio.h>
void main (void)
{
  int val;
  char string[10] = "250";
  sscanf (string, "%d", &val);
  printf ("The value in the string is %d\n", val);
}
```

sscanf uses the same format specifiers as **printf**, but all its arguments must be pointers to variables rather than variables themselves: as always, a function can never change the values of variables provided as arguments, but it can write to their destinations if they are pointers.

sscanf returns the number of values successfully read. So if a format specifier of **%d** is provided but the string supplied doesn't start with a decimal number, **sscanf** will write nothing to the supplied pointer and will return 0; if the string supplied does start with a decimal number, **sscanf** will return 1.

The format string supplied to **sscanf** can have multiple format specifiers and even other text:

```
#include <stdio.h>
void main (void)
{
  int val;
  char result[10];
  char string[25] = "The first number is 1";
  if (sscanf (string, "The %s number is %d", result, &val) == 2)
  {
    printf ("String : %s Value : %d\n", result, val);
  }
  else
  {
    printf ("I couldn't find two values in that string.\n");
  }
}
```

Note that, slightly inconsistently, the **%s** format specifier denotes a pointer to a string in both **printf** and **sscanf**, while the **%d** specifier denotes a *variable* in **printf** but a *pointer* in **sscanf**.

How long is a (piece of) string?

One other string handling function is **strlen** (for 'string length') – this tells you how many characters are in a string, including the terminating zero character.

```
#include <stdio.h>
#include <string.h>
void main (void)
{
  char str1[10] = "first";
  printf ("The length of the string '%s'
          is %d\n", str1, strlen (str1));
}
```

All the string operations above are possible by manipulating pointers, but the string library makes life much easier!

DON'T OVERWRITE

It looks like it ought to be possible using strcpy and strcat to copy part of a string over itself – **strcpy (a + 1, a)**, for example. Don't try it! The source and destination buffers for strcpy and strcat must be completely separate areas of memory; if not, their behaviour is unpredictable.

IGNORING CASE

There are versions of strcmp and strncmp which ignore the case of the letters in the strings being compared – they are called strcasecmp and strncasemp, respectively; they take the same arguments and return the same values.

MIKE'S PI BAKERY

MIKE COOK

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

magpi.cc/259aT3X

DRUM LIKE ME

THE RASPBERRY PI AND PRINGLE DRUM KIT

Make your own auto repeating pattern drum kit

You'll Need

- Empty Pringles cans
- 9 by 9 hole stripboard
- 2N7000 FETs
- BC237BG or similar transistors
- OVL-5521 white LEDs
- 3V3 Zener diodes
- 1N4868 or similar small signal diodes
- Piezo sensors
- Assorted resistors and wires
- Hot-melt glue

This is a fun project where you have a set of drums to play, but the twist is that what every sequence you play, it gets repeated, until you play something else. The idea is to have five finger drums, each one playing its own sample and when you stop playing, the sequence is automatically replayed. The drums are made from small empty Pringles tubes with piezo sensors on the plastic lids. When the sequence is playing back, a powerful LED is flashed in the appropriate drum. One of the easiest parts of making this was the eating of the Pringles before we could start.

The circuit

Each drum has its own circuit built into it, the schematic for this is shown in **Fig 1**. Note that there needs to be five of these circuits, one for each drum, and while the circuit is the same in each drum, the

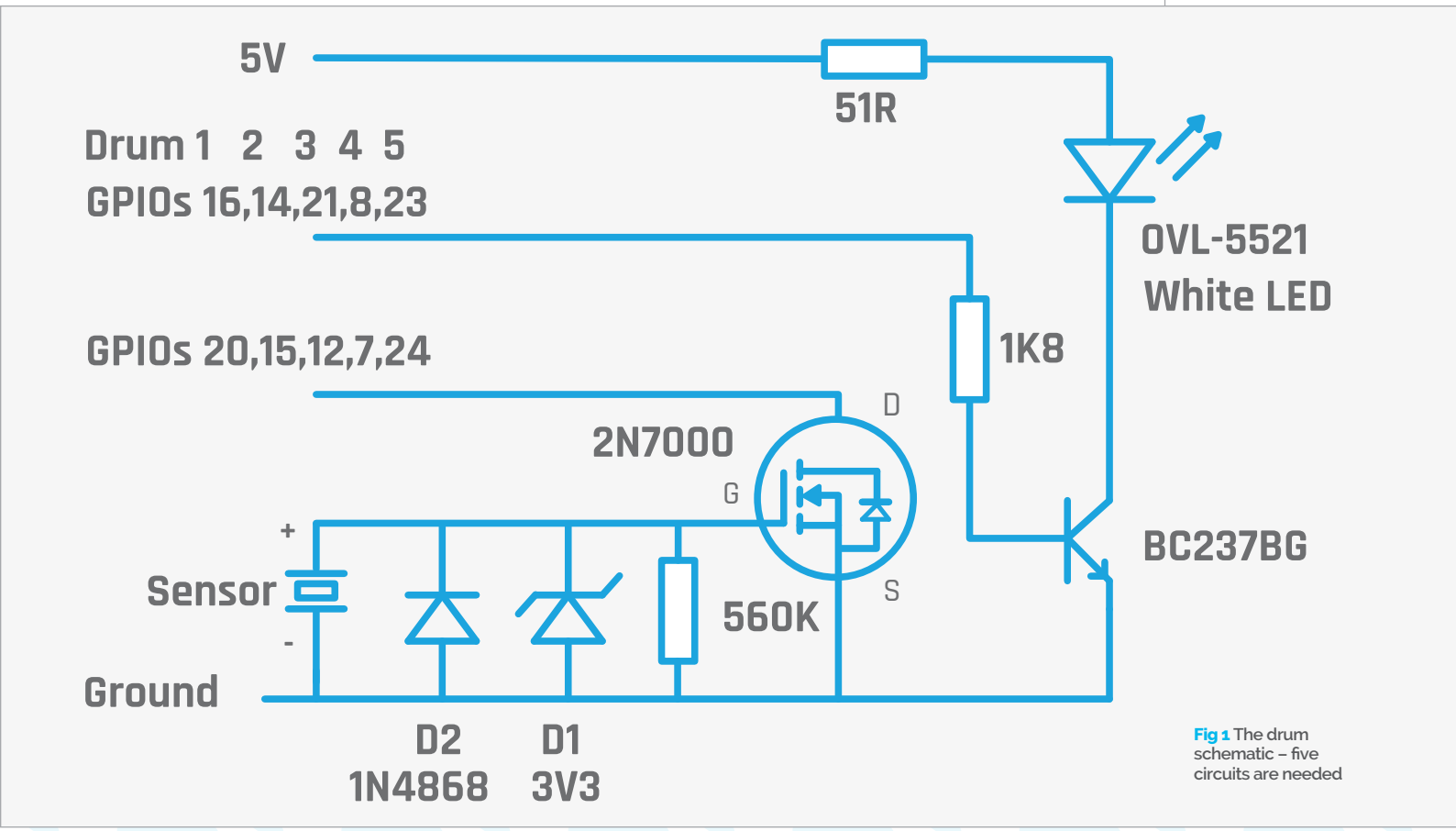


Fig 1 The drum schematic – five circuits are needed

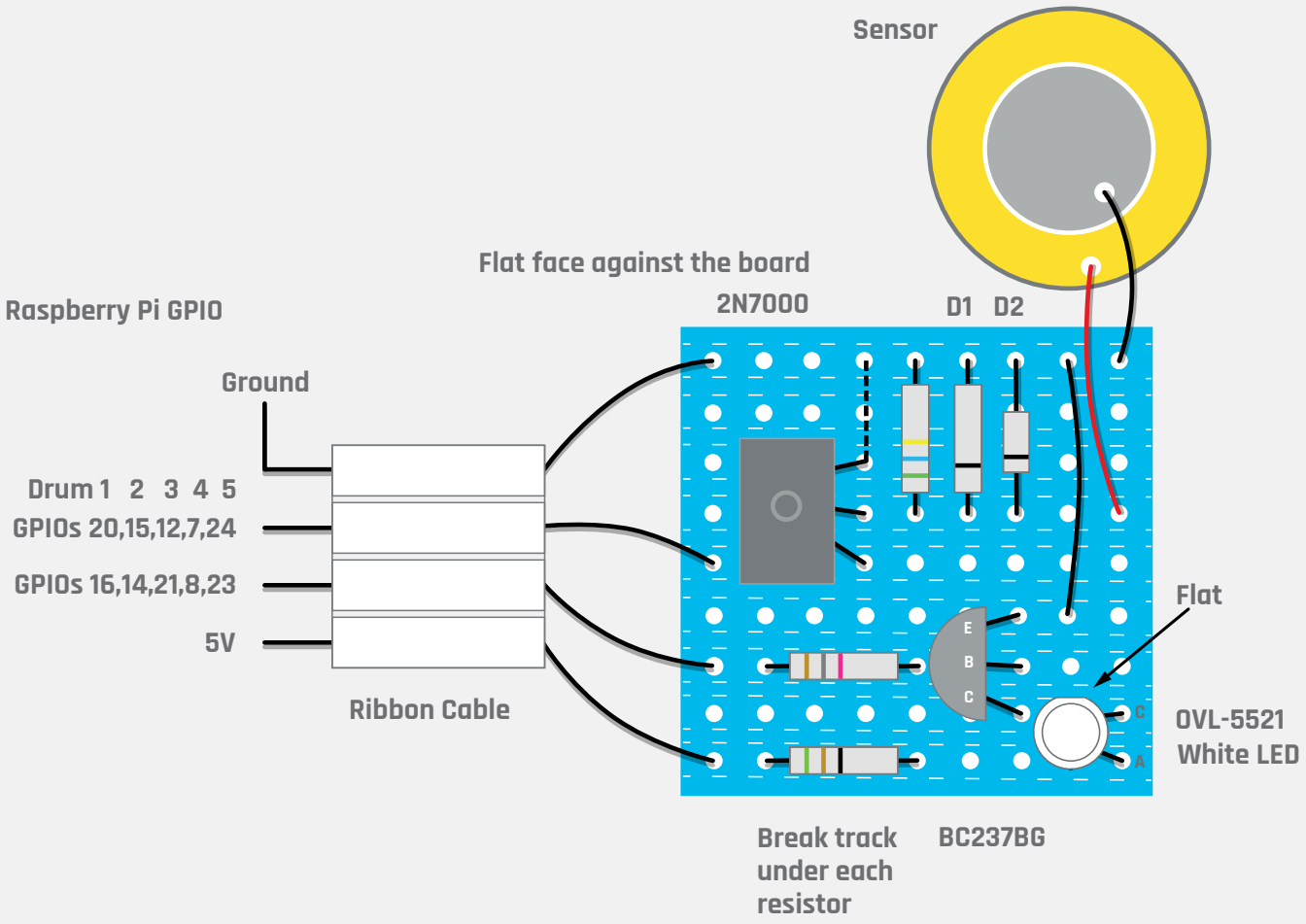
two GPIO pins the Pi uses is different for each one. We used an OVL-5521 white LED as this was the most powerful LED we had; it has a maximum output of 18 candela, whereas most LEDs produce light in the millicandela range. In part this is due to the narrow angle, but it reflects off the foil interior of the tube very well, and lights up the drum. The forward voltage for this LED is 4V when passing 20mA through it, which is why we need to use a transistor to switch it, and not connect it directly to the Pi's GPIO pins. You will also notice a 3V3 Zener diode protecting the gate of the FET from excessive voltages and a small signal diode protecting against negative voltage excursions. In theory, the Zener should have protected against this by itself, but in practice it was not good enough. Full construction details are shown in the step-by-step guide.

The Interface

Each drum has two GPIO pins: one to control the LED and the other to report back the sensor's response. When the sensor is hit, the FET conducts and puts a logic zero on the GPIO pin, which has been set as an input with the internal pull-up resistors enabled. The software has two modes, playing and listening, and will switch back automatically into the listening mode when a drum is played or when the space bar is pressed. When in the listening mode, it will switch into the playing mode when there has been a pause of 1.5 seconds without a drum being played; this delay is adjustable.

The sensor suffers from bounce, like virtually all mechanical switches; in this case it is from continued vibrations after the initial strike. The code has a debounce timer where once a sensor is triggered,

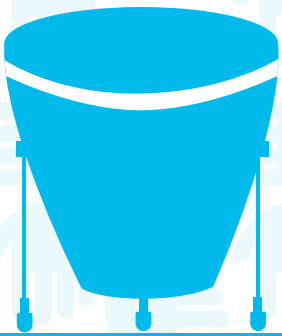
BUILDING THE DRUMS



>STEP-01

Build the circuit boards

Build five circuit boards for the drum electronics; do not attach the ribbon cable at this stage. Make sure there is a track break underneath the 1k8 base resistor and the 51R LED current-limiting resistor. Attach 6cm extension wires to the piezo sensors, and cover the joint with shrink sleeving, insulation tape or, as we used, self-amalgamating tape. Leave about 1cm of lead on the LED above the stripboard and push it to one side so that it shines directly on the tube's wall.



all further input from it is ignored until the end of this time. Any debounce time is always a compromise between blanking out any unwanted repeated triggering and allowing intentional rapid retriggering of the drum. Again, like the start delay, this debounce time is configurable.

The only things you'll need are the five sound samples for each drum. We used very short WAV files of drums and cymbals – you can find them on our GitHub page (**magpi.cc/1NqJjmV**) or download your own samples.

The software

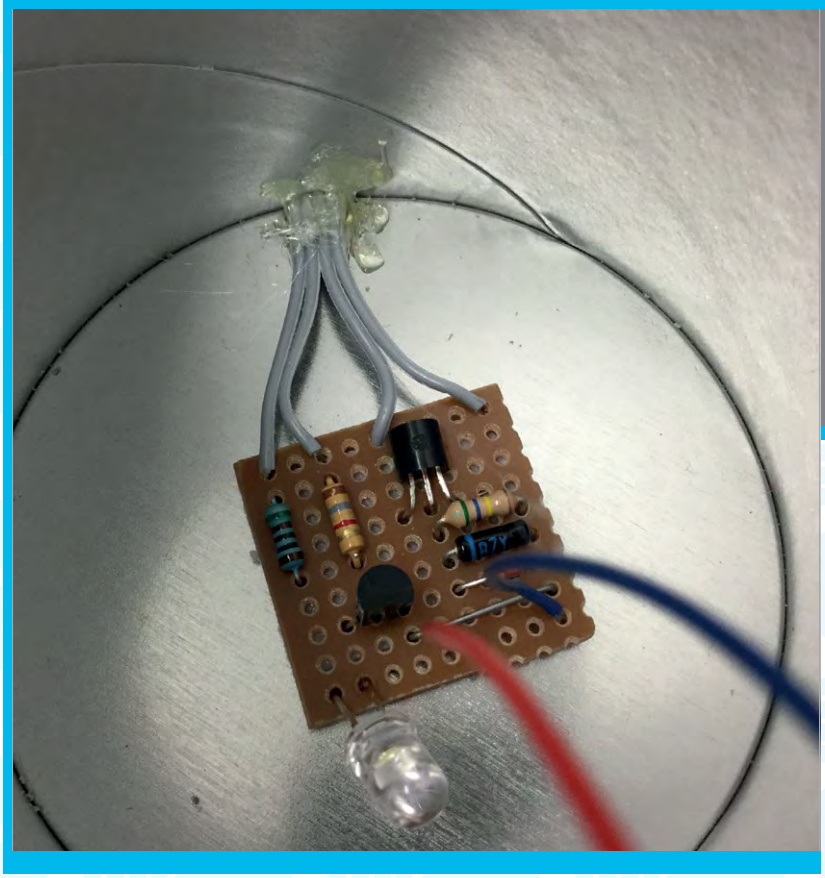
The full code is shown in **drum.py** overleaf; this is also available on our GitHub page. At the heart of the program is a couple of buffers where the time a drum is struck is recorded along with which drum was struck. At first, in the listening mode, the buffer is filled with the drum strike time, then, once the play mode is entered, the buffer times are altered to the time the note will be played back. In the playback mode, the buffer is constantly being searched to find if a sound is due to be played; when one is found, it is removed from the buffer and played and then that note is reinserted into the buffer but with a time that represents the next repeat of the sequence.

There are a number of variables that are important in the fine-tuning of the code. The **`startDelay`** variable determines how many seconds of no drum input triggers the transition into play mode. The **`debounceTime`** variable guards against the sensor signal's bouncing, but also determines the rate you can enter valid drum hits. This variable also controls

>STEP-02

Ribbon cable access

Cut a slot at the base of each drum to allow access to the ribbon cable. Drill two 1.5mm holes about 5mm apart (right-hand drum), then use a scalpel or sharp knife to cut between the two holes (centre drum), and clear away any debris. Finally, strip away five pieces of four-wire-wide ribbon cable, about 45cm long, and push it through the slot you have made (left drum). This is a lot easier than trying to push it through from the inside of the tube.



>STEP-03

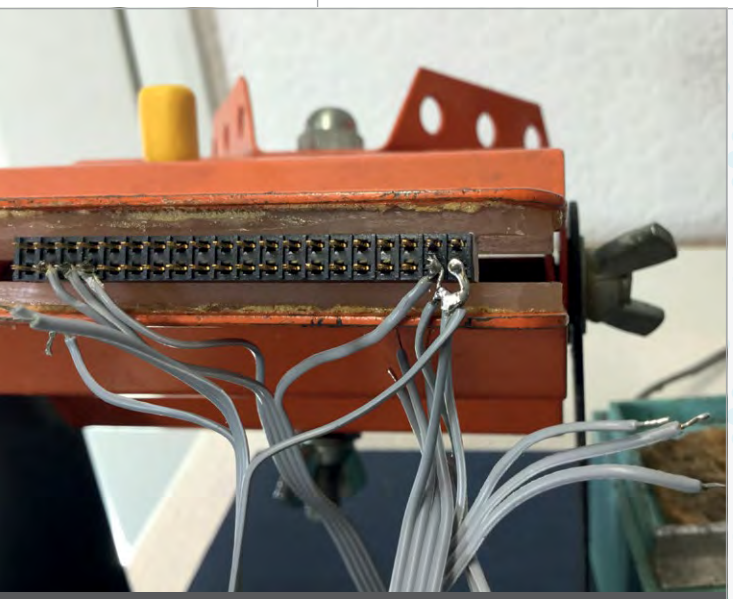
Fit the circuit board

Strip and solder the four wires of the ribbon cable to the board. Then cover the track side of the board with dual-sided sticky foam, pull the ribbon cable out, and manoeuvre the circuit board to the centre of the drum and press down. Then place some hot-melt glue over where the ribbon cable exits the drum. We could not get the glue gun into the tube, so we resorted to putting the glue on a wooden stick and transferring it to the ribbon cable – we used the wrong end of a small paintbrush for this.

how long the drum's LED is on during playback. The **`bufferLength`** variable controls the number of hit events you can record. While it might seem like a good idea to make this big, too big and you will waste time searching the buffer and that will affect the playback timing.

Taking it further

When using this program, we have found that sometimes the non-real-time nature of the Linux operating system causes the occasional drum hit not to be recorded. This might be improved by increasing the value of the 560k resistor across the



>STEP-04

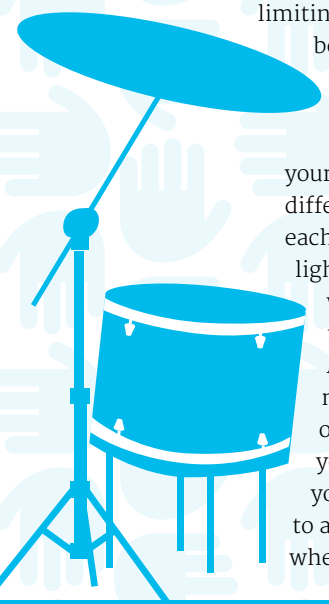
Wiring to a Pi connector

We used a 40-pin connector to wire the ribbon cable to the Pi's GPIO pins. While there are plenty of ground connections to go round, there are only two 5V ones. So we wired a small loop of tinned copper wire round the two and hung the wires off this. Finally, we applied a generous blob of hot-melt glue to the centre of the drum's lid and pushed the piezo sensor firmly onto this with a small circular motion to try to spread out the glue evenly, then held it down until the glue set.

FET gate to 1M, although this might mean you have to increase the debounce time.

You can change the number of drums quite easily: just have a longer list of LED and sensor pins. You might want to change the colour of the LEDs and have a different colour for each drum. Remember, you will also need to change the value of the LED's current-limiting resistor when you do this because the replacement LED will have a different forward voltage.

You might want to print out a drum graphic to wrap around your Pringles tube, or even use different flavours of Pringles for each drum. The drum's LEDs do not light up in play mode; you might want to see how you can change the code to make that happen. Also, you might want to limit the number of drum pattern repeats or gradually change the tempo as you have more repeats. Finally, you might want to fix the drums to a base board to stop them sliding when you hit them.



Drum.py

```
#!/usr/bin/python
# Drum Like Me Pygame framework
import pygame, time, os
import wiringpi as io

pygame.init()
os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'
pygame.display.set_caption("Drum like me")
screen = pygame.display.set_mode([300,40],0,32)
pygame.mixer.quit()
pygame.mixer.init(frequency=22050, size=-16, channels=2,
buffer=512)
pygame.event.set_allowed(None)
pygame.event.set_allowed([pygame.KEYDOWN,pygame.QUIT])
bufferLength = 40 # number of hits to store
delay = 0 ; startDelay = 1.6 # quiet time before playback
# minimum time between entry
lastEntry = 0.0 ; debounceTime = 0.05
lastInstrument = -1 ; ledOn = [0.0,0.0,0.0,0.0,0.0]
playing = False

def main():
  global lastPin, delay, playing, event, lastEntry,
lastInstrument
  initResource()
  print"Drum like Me - By Mike Cook"
  clearBuffer()
  timeOut = time.time()
  while True:
    startTime = time.time()
    while not playing:
      checkForEvent()
      pressed = getPins()
      if pressed:
        timeOut = time.time()
      for pin in range(0,len(sensorPins)):
        if currentPin[pin] == 0 and lastPin[pin] != 0:
          if time.time() - lastEntry > debounceTime or
lastInstrument != pin:
            drums[pin].play()
            placeInBuffer(pin,time.time())
            lastInstrument != pin
        lastPin[pin] = currentPin[pin]
      if time.time() > (timeOut + startDelay):
        playing = True # start playing if nothing received
        delay = time.time()-startTime # length of sequence
        adjustBuffer(delay) # add delay into buffer
    while playing :
      checkForEvent()
      lookAtBuffer(delay)
      pressed = getPins()
      if pressed:
        playing=False
        clearBuffer()
```

Language
>PYTHON 2

DOWNLOAD:
magpi.cc/1NqJjmV

PROJECT VIDEOS

Check out Mike's Bakery videos at:
magpi.cc/1NqJnTz

```
52.
53.  # see if something needs sounding
54.  def lookAtBuffer(delay):
55.    global event, instrument,ledOn
56.    for i in range(0,bufferLength):
57.      if instrument[i] != -1 and time.time() >= event[i]:
58.        toPlay = instrument[i]
59.        drums[toPlay].play()
60.        io.digitalWrite(ledPins[toPlay],1) # turn on LED
61.        ledOn[toPlay] = time.time()
62.        instrument[i] = -1
63.        placeInBuffer(toPlay,time.time()+delay)
64.    for i in range(0,len(ledPins)):
65.      if ledOn[i] != 0 and time.time() > (
     ledOn[i]+debounceTime):
66.        io.digitalWrite(ledPins[i],0)
67.        ledOn[i] = 0.0
68.
69.  def adjustBuffer(delay): # add delay into buffer
70.    global event,instrument
71.    for i in range(0,bufferLength):
72.      if instrument[i] != -1:
73.        event[i] += delay
74.
75.  def clearBuffer():
76.    global instrument,event, lastInstrument, lastEnrty
77.    lastInstrument = -1
78.    lastEnrty = 0.0
79.    for i in range(0,bufferLength):
80.      instrument[i] = -1
81.      event[i] = 0.0
82.
83.  def getPins():
84.    down = False
85.    for pin in range (0,len(sensorPins)):
86.      currentPin[pin] = io.digitalRead(sensorPins[pin])
87.      if currentPin[pin] == 0:
88.        down = True
89.    return down
90.
91.  def placeInBuffer(drum,strikeTime):
92.    global event, instrument,lastEntry,lastInstrument
93.    place = 0 # find free space
94.    while instrument[place] !=-1 and place <
     bufferLength-1 :
95.      place+=1
96.    event[place] = strikeTime
97.    instrument[place] = drum
98.    lastEntry = strikeTime
99.
100. def initResource():
101.   global sensorPins,samples,drums,currentPin,lastPin,
     event,instrument,ledPins
102.   sensorPins= [20,15,12,7,24] # GPIO pins for sensors
103.   ledPins = [16,14,21,8,23]
     # GPIO pins for the LEDs
104.   samples = [ "clap.wav",
     "ti.wav","drum.wav",
     "top.wav","ride.wav"]
105.   drums = [ pygame.mixer.
     Sound("sounds/"+samples[sound])
     for sound in range(0,
     len(sensorPins))]:
106.   currentPin = [1 for pin in range(
     0,len(sensorPins))]
107.   lastPin = [1 for pin in range(0,len(sensorPins))]
108.   event = [ time.time() for b in range(
     0,bufferLength)]
109.   instrument = [ 1 for b in range(0,bufferLength)]
110.   try :
111.     io.wiringPiSetupGpio()
112.   except :
113.     print"start IDLE with 'gksudo idle' on command
     line"
114.     os._exit(1)
115.   for pin in range (0,len(sensorPins)):
116.     # make pin into an input
117.     io.pinMode(sensorPins[pin],0)
118.     # enable pull up
119.     io.pullUpDnControl(sensorPins[pin],2)
120.   for pin in range (0,len(ledPins)):
121.     # make pin into an output
122.     io.pinMode(ledPins[pin],1)
123.     io.digitalWrite(ledPins[pin],0) # set output low
124.
125. def terminate(): # close down the program
126.   pygame.mixer.quit() ; pygame.quit()
127.   for pin in range (0,len(ledPins)):
128.     io.digitalWrite(ledPins[pin],0) # turn LEDs off
129.   os._exit(1)
130.
131. def checkForEvent(): # keyboard commands
132.   global playing
133.   event = pygame.event.poll()
134.   if event.type == pygame.QUIT :
135.     terminate()
136.   if event.type == pygame.KEYDOWN :
137.     if event.key == pygame.K_ESCAPE :
138.       terminate()
139.     if event.key == pygame.K_SPACE :
140.       playing = False
141.       clearBuffer()
142.
143. # Main program logic:
144. if __name__ == '__main__':
145.   main()
146.
```

WINFRIED PLAPPERT

Former IT consultant. Has enjoyed playing with Unix and Linux for many years. Received a Raspberry Pi as a Christmas present a few years back.

MULTI BOOT YOUR RASPBERRY PI 2

PART 02

You'll Need

- Das U-Boot: **magpi.cc/2iZ7P8g**
- Cross-compiler tools

Use 'Das U-Boot' to boot multiple Raspberry Pi operating systems from a USB hard drive

Last time we looked at multi-booting (in issue 52), we got ourselves set up to start putting several operating systems and Linux distributions on the Raspberry Pi in a multi-booting scenario. Let's finish that off...

The following distributions have been tested as working: Raspbian, Ubuntu MATE, SUSE, Manjaro, Fedora 25, Puppy Linux, Sabayon, and Arch Linux in console mode.

The concept of booting multiple operating systems from one USB hard disk using one kernel is not without its problems.

Firstly, you need a fast USB disk that connects directly to the Raspberry Pi, which means it's far preferable to use an external hard drive that has its own power supply.

The separation between partition 1 (**/boot**) and partition 2 (**root file system**) is not defined exactly. Debian-based distributions seem to adhere to similar standards, but other distributions do not, most notably Fedora 25 (download at **magpi.cc/2igrnGp**) and Manjaro (**magpi.cc/2j5gc4w**).

Whichever distribution you have downloaded, it is wise to install it on a micro SD card first and boot it for the first few times from the SD card. This is because some distributions run through some initial steps like creating a new user for you and doing other customisations.

Language
>C, BASH

DOWNLOAD:
magpi.cc/MultiBoot2

After these steps have been completed and you are happy with the resulting distribution, you can copy the root file system onto your USB disk. When you create new partitions on your USB disk, make sure they are created as ext3 partitions, since the U-Boot disk does not play well with ext4 partitions yet.

Modify in C

The U-Boot code can be extended with your own custom commands. The only way you can make full usage of all the built-in features of U-Boot is to generate a new command as part of it, which should then be linked into the newly created executable **u-boot.bin**. The most convenient place to put your new source code is in the **cmd** subdirectory of U-Boot. After having done this, modify the **Makefile** in the **cmd** directory to add your code to be compiled and linked into the U-Boot. Once you have transferred the resulting **u-boot.bin** file to the first partition of the SD card, as described in part one of this series in issue 52, you can test your modifications by hand. Interrupt the U-Boot sequence and type in the newly created command; in our example, this is called **wpl_cmd**. Create the executable file by issuing **make all** in the top-level directory of the source code tree of the downloaded U-Boot.

At the very end of **cmd/Makefile**, you need to add the following line:

```
obj-y += new_wpl_all.c
```

...where **new_wpl_all.c** is the C file name of your extension code. Add a custom **boot.scr** file to your **/boot** partition by issuing the command in the top-level directory of the source code tree:

```
./tools/mkimage -O linux -A arm -T script -n
"boot script for booting from usb hard disk"
-d wpl_prerate.txt boot.scr
```

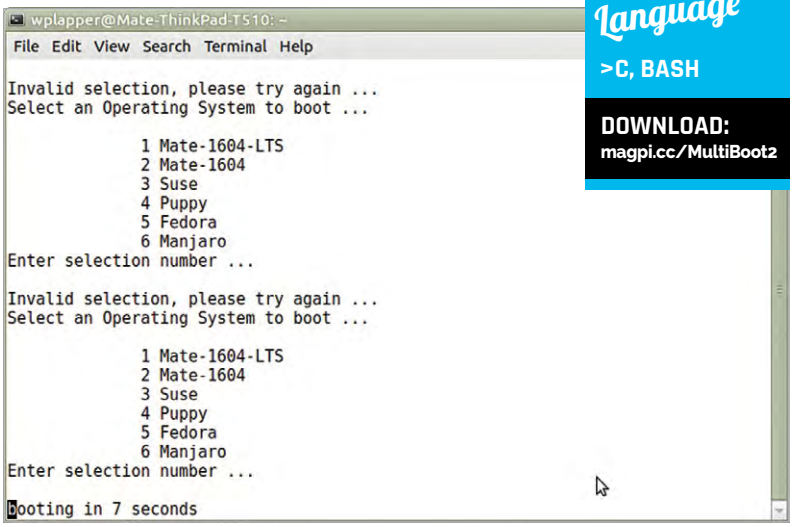
Source of the code

After including the normal C header files, some U-Boot-specific headers will be included to allow access to the file-system-specific routines and other bits of the U-Boot.

Specific to this command is **os_list**, which contains the names of the partitions and the OS path to the partitions.

Once the source code has been compiled successfully, it has to be transferred to the **/boot** partition of the micro SD card. This can easily be achieved on a Linux based system – the one which does the compilation and linking steps of the U-Boot. Mount the micro SD card on your Linux system, and issue a copy command:

```
cp -p u-boot.bin boot.scr /
media/<userid>/<boot-partition-name>/
```



Above If you follow the tutorial correctly, you should get a boot menu similar to this

...where 'userid' is your own user ID and 'boot-partition-name' is the name of the boot partition of the SD card.

Post-processing

These are a couple of tasks to be done on the Raspberry Pi before you boot from the USB hard disk. Firstly, modify the **/etc/fstab** file of the distribution to match the real root file system:

```
/dev/sda<x> / ext4 defaults,noatime,nodiratime,
errors=remount-ro 0 1
```

This should replace **/dev/mmcblk0p<y> / ext4 ...**

Next, copy all **/lib/modules/$(uname -r)/** from the distribution where the kernel came from originally to the distribution's **lib/modules/$(uname -r)/** via rsync. This step has to be repeated every time the kernel gets updated via **rpi-update** or implicitly by updating your packages via the package manager. Watch out and don't get caught without a complete backup of your **/boot** file system:

```
rsync -av /lib/modules/$(uname -r)/ /
media/<userid>/<Distribution-name>/lib/
modules/$(uname -r)/
```

If you forget this step, you might be able to boot your distribution from the USB disk, but the chances are very high that you won't be able to use a keyboard and mouse attached to the Raspberry Pi. The drivers just don't match the kernel or the hardware is simply not recognised properly.

In summary, U-Boot can be used to boot multiple partitions from one USB hard disk. This project is not without its problems, since there can only be one kernel, which lives in the **/boot** partition of the micro SD card. Different distributions have different requirements from the kernel. The other issue which has been observed is the dependency of the systemd service on the kernel.

SAM AARON

Sam is the creator of Sonic Pi. By day he's a research associate at the University of Cambridge Computer Laboratory; by night he writes code for people to dance to. **sonic-pi.net**

Sonic Pi PART 16

ADDITIVE SYNTHESIS

Did you know you can design sounds on Sonic Pi? **Sam Aaron** shows us how...

You'll Need

- Raspberry Pi running Raspbian
- Sonic Pi v2.9+
- Speakers or headphones with a 3.5mm jack
- Update Sonic Pi: `sudo apt-get update && sudo apt-get install sonic-pi`

This is the first of a short series of articles on how to use Sonic Pi for sound design. We'll be take a quick tour of a number of different techniques available for you to craft your own unique sound.

The first technique we'll look at is called additive synthesis. This may sound complicated – but if we expand each word slightly, the meaning pops right out. Firstly, 'additive' means a combination of things; secondly, 'synthesis' means to combine thing, in this case sounds. Additive synthesis therefore means nothing more complicated than combining existing sounds to create new ones.

This synthesis technique dates back a very long time – for example, pipe organs in the Middle Ages had lots of slightly different-sounding pipes which you could enable or disable with stops. Pulling out the stop for a given pipe 'added it to the mix', making the sound richer and more complex. Now, let's see how we can pull out all the stops with Sonic Pi.

Simple combinations

Let's start with the most basic sound there is – the humble pure-toned sine wave:

```
synth :sine, note: :d3
```

Now, let's see how this sounds combined with a square wave:

```
synth :sine, note: :d3
synth :square, note: :d3
```

Notice how the two sounds combine to form a new, richer sound. Of course, we don't have to stop there: we can add as many sounds as we need. However, we need to be careful with how many sounds we add together. Just like when we mix paints to create new colours, adding too many colours will result in a messy brown. Similarly, adding too many sounds together will result in a muddy sound.

Blending

Let's add something to make it sound a little brighter. We could use a triangle wave at an octave higher (for that high bright sound), yet only play it at **`amp 0.4`** so it adds something extra to the sound rather than taking it over:

```
synth :sine, note: :d3
synth :square, note: :d3
synth :tri, note: :d4, amp: 0.4
```

Now, you can try creating your own sounds by combining two or more synths at different octaves and amplitudes. Also, note that you can play around with a synth's opts to modify each source sound before it is mixed in for even more combinations of sounds.

Detuning

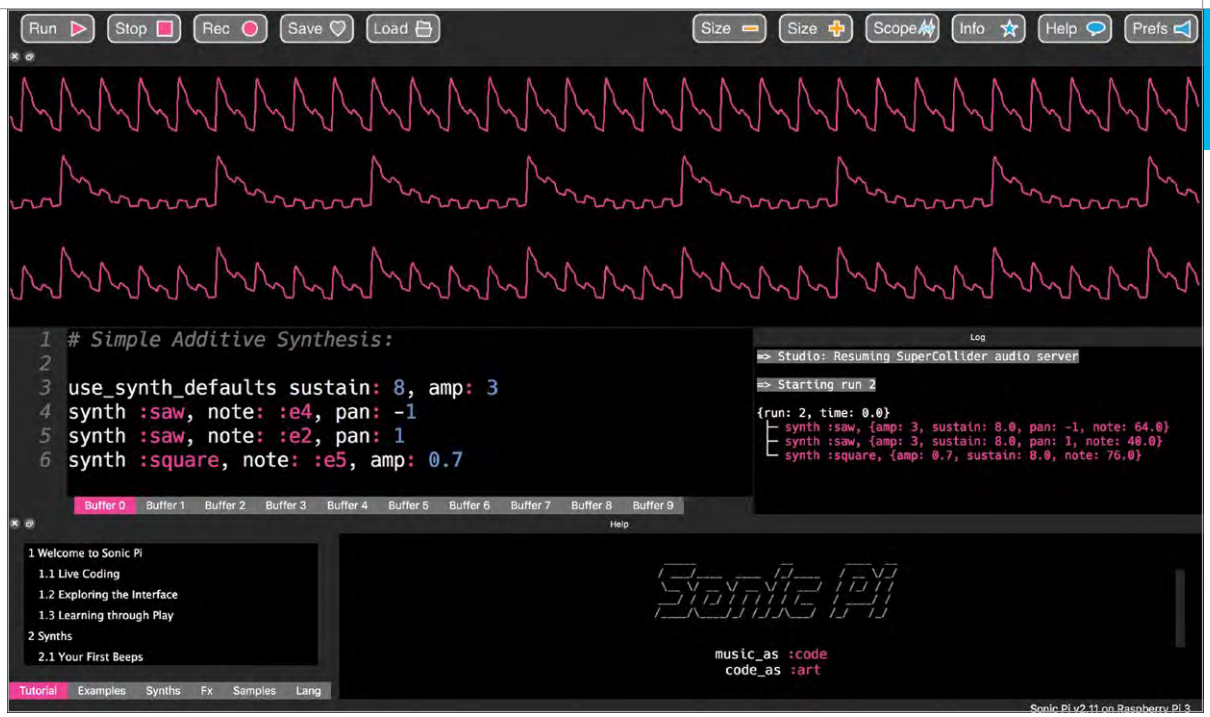
So far, when combining our different synths we've used either the same pitch or switched octave. How might it sound if we didn't stick to octaves but instead chose a slightly higher or lower note? Let's try it:

```
detune = 0.7
synth :square, note: :e3
synth :square, note: :e3 + detune
```

If we detune our square waves by 0.7 notes, we hear something that perhaps doesn't sound in tune or correct – a 'bad' note. However, as we move closer to 0, it will sound less and less out of tune as the pitch of two waves gets closer and more similar. Try it for yourself! Change the **`detune:`** opt value from **`0.7`** to **`0.5`** and listen to the new sound. Try **`0.2`**, **`0.1`**, **`0.05`**, **`0`**. Each time you change the value, take a listen and see if you can hear how the sound is changing. Notice that low detune values such as **`0.1`** produce a really nice 'thick' sound, with both slightly different pitches interacting with each other in interesting, often surprising, ways.

Some of the built-in synths already include a detune option that do exactly this in one synth. Try playing with the **`detune:`** opt of **`:dsaw`**, **`:dpulse`**, and **`:dtri`**.

Language
>RUBY



Left Combine sounds to make new ones with additive synthesis

Amplitude shaping

Another way in which we can finely craft our sound is to use a different envelope and options for each synth trigger. For example, this will allow you to make some aspects of the sound percussive and other aspects ring out for a period of time.

```
detune = 0.1
synth :square, note: :e1, release: 2
synth :square, note: :e1 + detune, amp: 2,
                                release: 2
synth :gnoise, release: 2, amp: 1, cutoff: 60
synth :gnoise, release: 0.5, amp: 1, cutoff: 100
synth :noise, release: 0.2, amp: 1, cutoff: 90
```

In the example above, we have mixed in a noisy percussive element to the sound along with some more persistent background rumbling. This was achieved firstly by using two noise synths with middling **cutoff** values (**90** and **100**), using short **release** times, along with a noise with a longer **release** time but with a low **cutoff** value (which makes the noise less crisp and more rumbly.)

Bringing it all together

Let's combine all these techniques to see if we can use additive synthesis to recreate a basic bell sound. We've broken this example into four sections. Firstly we have the 'hit' section, which is the initial onset part of the bell sound, and so uses a short envelope (e.g. a **release:** of around **0.1**). Next we have the long ringing section, for which we're using the pure sound of the sine wave. Notice that we're often increasing the note by roughly **12** and **24**, which is the number of notes in an octave. We've also thrown in a couple of low sine waves to give the sound some bass and depth. Finally, we use **define** to wrap our code in a function which we can then use to play a melody. Try playing your own melody and also messing around with the contents of the **:bell** function until you create your own crazy sound to play with!

```
define :bell do |n|
  # Triangle waves for the 'hit'
    synth :tri, note: n - 12, release: 0.1
    synth :tri, note: n + 0.1, release: 0.1
    synth :tri, note: n - 0.1, release: 0.1
    synth :tri, note: n, release: 0.2

  # Sine waves for the 'ringing'
    synth :sine, note: n + 24, release: 2
    synth :sine, note: n + 24.1, release: 2
    synth :sine, note: n + 24.2, release: 0.5
    synth :sine, note: n + 11.8, release: 2
    synth :sine, note: n, release: 2

  # Low sine waves for the bass
    synth :sine, note: n - 11.8, release: 2
    synth :sine, note: n - 12, release: 2
end

# Play a melody with our new bell!
bell :e3
sleep 1
bell :c2
sleep 1
bell :d3
sleep 1
bell :g2
```

FREQUENTLY ASKED QUESTIONS

Your technical hardware and software problems solved...

NEED A PROBLEM SOLVED?

Email **magpi@raspberrypi.org** or find us on **raspberrypi.org/forums** to feature in a future issue.

WIRELESS CONNECTIVITY

HOW IS THE RASPBERRY PI WIRELESS?

Wireless LAN

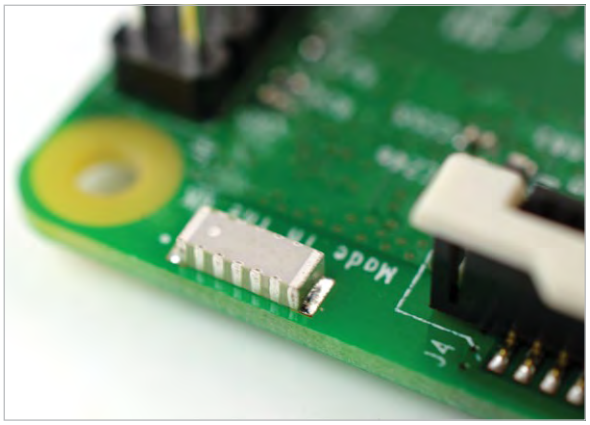
You can connect to a wireless network (WiFi) using the on-board wireless LAN on the Raspberry Pi 3. For other Raspberry Pi models, you can buy a WiFi dongle; most work just fine on it and can be managed from the graphical interface.

Bluetooth

As well as wireless LAN, the Raspberry Pi 3 has built-in Bluetooth connectivity. You can connect speakers and input devices to the Raspberry Pi via Bluetooth, although you may have a bit of trouble navigating a mobile phone's storage.

Radio

While no Raspberry Pi supports native RF radio, there are some HATs, add-ons, and USB devices you can plug into it that add support for this. You'll need to refer to any documentation on the device to clarify how to get it work on the Pi.



Right That's the wireless antenna. It's tiny!

HOW DO I CONNECT TO THE WIFI?

Check your SSID

Make sure that you know the SSID and password for the wireless network you want to connect to. Turn on the Raspberry Pi (with a WiFi dongle plugged in if you need to use one) and boot into the graphical desktop interface.

Connect to the network

In the top right of the screen you'll notice the wireless interface logo – a dot below quarter-circles. Click on it and search for your network name. Click on the network and enter your password. It should then connect and give you internet access.

Command line

If you've connected to the network on the graphical interface, the Pi will remember these details even if you boot to the command line. If you need to set it up via the command line, though, a guide can be found here: **magpi.cc/2hQhwW4**.

HOW DO I CONNECT TO A BLUETOOTH DEVICE?

Make it discoverable

All Bluetooth devices have a way to make themselves discoverable by other Bluetooth devices – refer to the device's manual on how to do this. It usually involves holding down the main button for a period of time.

Find it

On the Raspberry Pi graphical desktop, look for the Bluetooth logo in the top right. Click on that and begin scanning for devices to pair with. Click the one you want and enter a passcode if you set one up.

Command line

Pairing in the command line is rather tricky. It's better to do it via the desktop as most uses of Bluetooth with the Pi only work on the desktop anyway. There are some threads in the Raspberry Pi forums with some solutions on how to do it, though.

FROM THE RASPBERRY PI FAQ
RASPBERRYPI.ORG/HELP

DOES THE RASPBERRY PI COME WITH A CASE?

An official case for the Raspberry Pi is available from various retailers. There are also lots of homebrew case discussions on the forum, as well as several third-party cases available. We suggest stopping by the cases subforum and reading some of the threads about cases you can purchase or build yourself.

CAN I POWER THE RASPBERRY PI FROM A USB HUB?

It depends on the hub. Some hubs comply with the USB 2.0 standard and only provide 500mA per port, which may not be enough to power your Raspberry Pi. Other hubs view the USB standards more like guidelines, and will provide as much power as you want from each port. Please also be aware that some hubs have been known to 'backfeed' the Raspberry Pi. This means that the hubs will power the Raspberry Pi through its USB input cable, without the need for a separate micro-USB power cable, and bypass the voltage protection. If you are using a hub that backfeeds to the Raspberry Pi and the hub experiences a power surge, your Raspberry Pi could potentially be damaged.

CAN I POWER THE RASPBERRY PI FROM BATTERIES AS WELL AS FROM A WALL SOCKET?

Running the Raspberry Pi directly from batteries requires special care and can result in damaging or destroying your Raspberry Pi. If you consider yourself an advanced user, though, you could have a go. For example, 4×AA rechargeable batteries would provide 4.8V on a full charge. 4.8V would technically be just within the range of tolerance for the Raspberry Pi, but the system would quickly become unstable as the batteries lost their full charge. Conversely, using 4×AA Alkaline (non-rechargeable) batteries will result in 6V; this is outside the acceptable tolerance range and would potentially damage or, in the worst case scenario, destroy your Raspberry Pi. It is possible to provide a steady 5V from batteries by using a buck and/or boost circuit, or by using a charger pack which is specifically designed to output a steady 5V from a couple of batteries; these devices are typically marketed as mobile phone emergency battery chargers.

TAKE GIANT LEAPS FORWARD

Great gadgets for your next Raspberry Pi projects

WD SMART CABLE MODULE

Compact way to easily integrate a Raspberry Pi Compute Module* with a USB drive to create projects like a wireless NAS or USB camera recorder.

*Raspberrry Pi Compute Module not included.

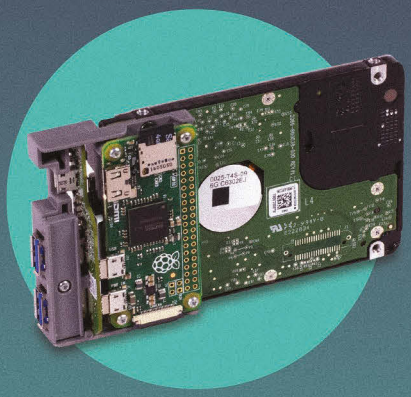
$18.99

MEDIA STICK FOR RASPBERRY PI

Combines with Raspberry Pi Compute Module* to quickly turn a TV or monitor into a powerful media hub or custom digital display.

*Raspberrry Pi Compute Module not included.

$24.99



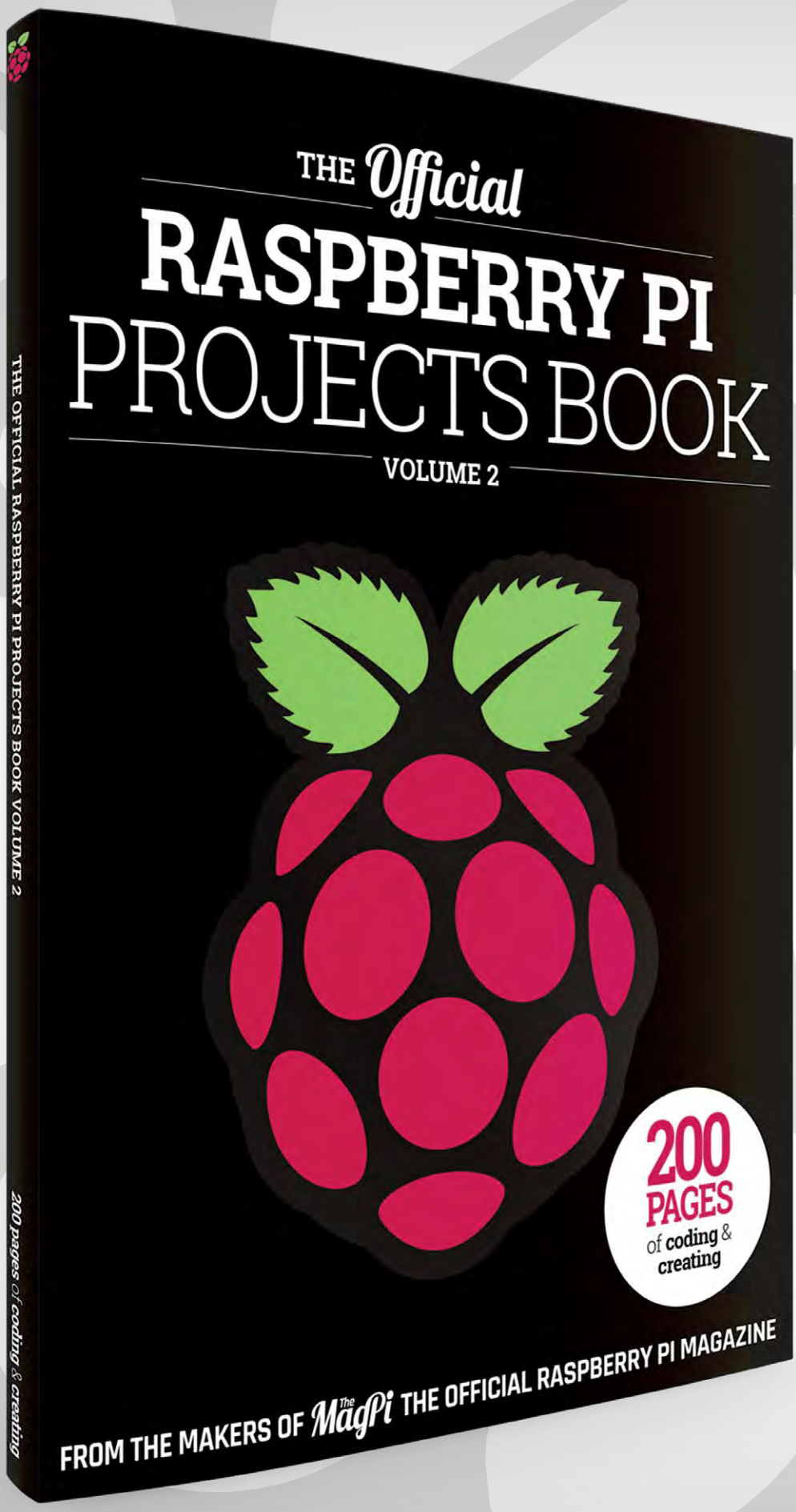
WD PIDRIVE™ NODE ZERO

Provides onboard computing capabilities with an included Pi Zero for autonomous projects like video recording, data logging, and more.

$44.99

Western Digital, WD, the WD logo, WDLabs and the WDLabs logo are registered trademarks or trademarks of Western Digital Corporation or its affiliates in the U.S. and/or other countries. Raspberry Pi is a trademark of the Raspberry Pi Foundation. microSD is a trademark of SD-3C, LLC. All other marks are the property of their respective owners. As used for storage capacity, one gigabyte = one billion bytes. Total accessible capacity varies depending on operating environment. Product specifications subject to change without notice. Pictures shown may vary from actual product. © 2017 Western Digital Corporation or its affiliates.

WDLABS wdlabs.wd.com/magpi54c


THE Official
RASPBERRY PI
PROJECTS BOOK
VOLUME 2
THE OFFICIAL RASPBERRY PI PROJECTS BOOK VOLUME 2
200 pages of coding & creating
200
PAGES
of coding &
creating
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE

£12.99
200 pages of
Raspberry Pi

THE *Official*

RASPBERRY PI PROJECTS BOOK

VOLUME 2

Amazing hacking and making projects

from the creators of *The MagPi* magazine

Inside:

How to get started with Raspberry Pi

The most inspirational community projects

Essential tutorials, guides, and ideas

Expert reviews and buying advice

Available now

magpi.cc/MagPiStore

plus all good newsagents and:

WHSmith BARNES&NOBLE

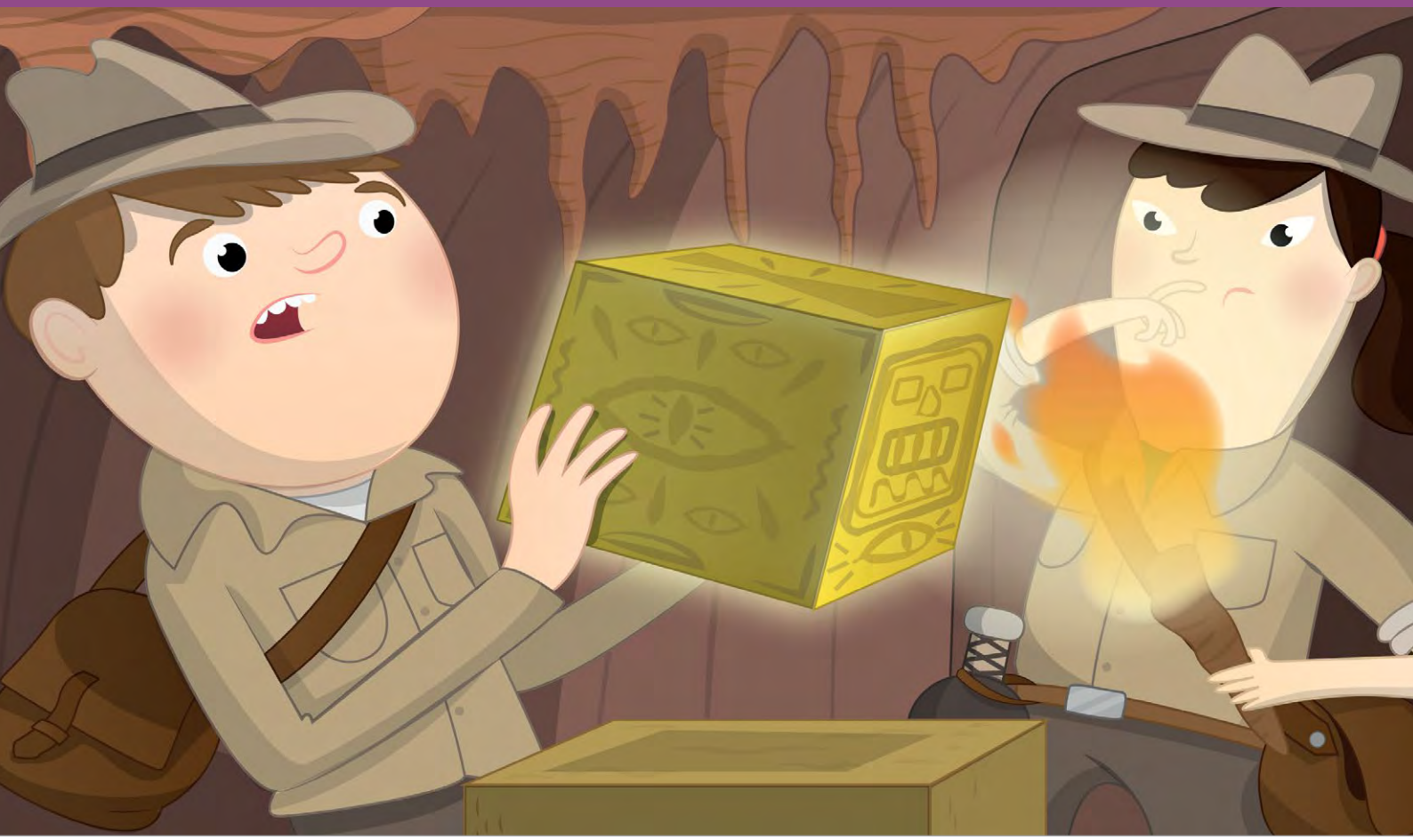
Available on the App Store

GET IT ON Google Play

Available on Amazon

UNDERSTAND OBJECT-ORIENTED PROGRAMMING

Get your head around OOP by using Scratch and Python to create the same programs



BEYOND PROCEDURE

When you first start programming, you'll begin by writing procedural code.

In good old-fashioned procedural programming, you typically create all your variables at the start of a program. Then you make some function definitions (these are blocks of reusable code).

We looked at procedural programming in *The MagPi* issue 53 (**magpi.cc/2iin6PQ**).

OOP takes all the building blocks of procedural programming – variables, functions, loops, conditions – but bundles them into self-contained blocks.

Most coders create procedural scripts that import objects (from modules and packages). So you're using objects without even realising it.

OOP concepts are found in almost all modern programming languages, including Python and Java.

In the modern world, almost all of the code you will encounter is created in a style called 'object-oriented programming', or OOP for short.

If you grew up with OOP, it is the obvious way to create computer programs.

In OOP, the code is used to create objects. These represent real-world things: a dog, a chair, or the wheels on a car.

Objects contain both the variables that make that thing: a person's height, age, or a name for example. They also contain the functions that object can perform: a dog can jump, or walk, or run; a wheel can rotate.

OOP bundles both the variables and functions together. This style makes it super-easy to cut and paste the code from one program to another. You don't even need to cut and paste, in fact: you use import statements to get functions and the variables they need.

With OOP, you don't need to create an object for a dog: you just find one somebody else has made and import it to your program.

Importing knowledge

At the start of most programs, you'll find a bunch of import statements. These are used to paste in code which has been created by other people.

OOP isn't perfect. It can be accused of overkill. "The problem with object-oriented languages is they've got all this implicit environment that they carry around with them," says Joe Armstrong, creator of Erlang. "You wanted a banana but what you got was a gorilla holding the banana, and the entire jungle."

There's also a whole bunch of decorative terminology surrounding OOP. You'll encounter lots of strange words like 'encapsulation' and 'instantiation'. These make the concept appear much more complicated than it is, and can also be rather off-putting to newcomers.

So OOP is a bit wordy and lends itself to navel-gazing. Many makers, hackers, and coders struggle to understand OOP, and indeed you can get a long way without understanding it.

Young coders, on the other hand, are increasingly introduced to programming via Scratch.

Software like Scratch is included with Raspbian with PIXEL (and Debian with PIXEL) and is designed specifically to teach students OOP stealthily.

In Scratch, objects are called 'Sprites'. They resemble video game characters. The idea is that children brought up on Scratch will inherently feel at home with objects when they migrate to a language like Python.

CREATE BUNCO IN SCRATCH

Create a game in Scratch where players play dice with one another

You'll Need

- Raspbian with PIXEL
- Scratch 2.0 account

First, we're going to create a game in Scratch. Then we'll recreate it in Python so you can see how it works in both languages.

Our dice game is based on Bunco (**magpi.cc/2hoZNcj**). It's a popular parlour game played in North America.

We've made the rules a little simpler. Each player rolls three dice and counts up the score. The player with the highest score wins.

We need two players, each with their own set of dice. Each player then rolls the dice, looks at their dice, and compares them with the dice of the other player.

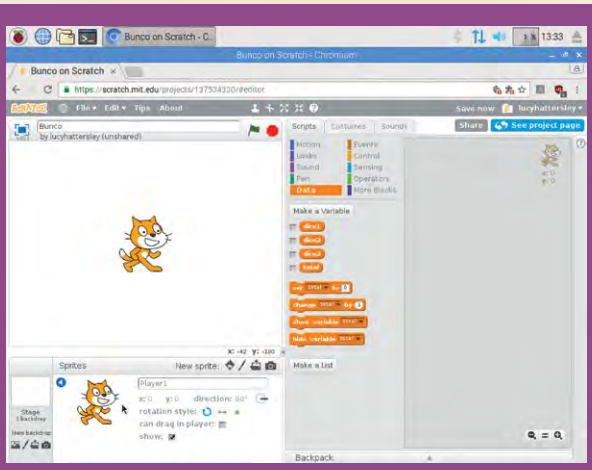
If they have the same score, both call it a draw. If a player spots that their total is higher than the other person's, they shout out "I win!".

This game introduces you to the concept of local variables. Each sprite has three local variables: their own set of dice. They can also look at the variables (or dice) that are local to other sprites.

The opposite of a local variable is a global variable. This is as if both players rolled a single set of dice and shared the result. They'd always draw.

Scratch works slightly differently to Python. In Scratch, you create one sprite and then clone (duplicate) it to create a second sprite. In Python, you create a blueprint for your sprites (known as a 'class') and then stamp out two player objects. We'll come to Python in a bit.

Let's create the dice game in Scratch first...



>STEP-01

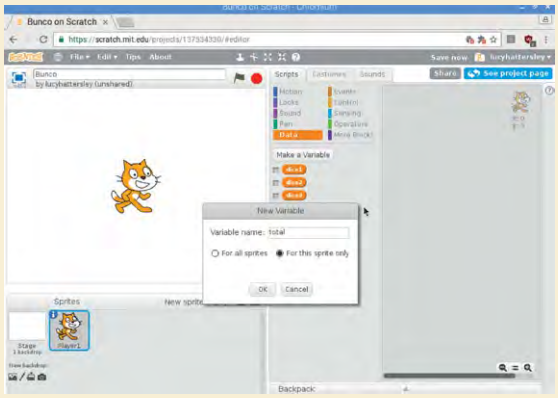
Scratch

Open the web browser and visit **scratch.mit.edu** to open Scratch 2.0. We need the clone features from 2.0, so don't use the Scratch 1.4 app. Log in (or create an account if you're new to Scratch). Create a new project and you'll see a single Scratch Cat sprite on the screen. Click the 'i' symbol next to the sprite in the bottom-left. Change its name to Player1.

>STEP-02

Three dice

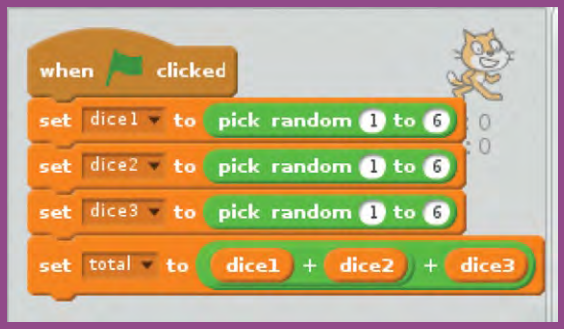
Click on Data and then Make a Variable. Enter **dice1** in the Variable name field and select 'For this sprite only'. Click OK and **dice1** appears in the blocks palette. Repeat the process to create **dice2** and **dice3**. Finally, create another variable called **total**. Remember to choose 'For this sprite only' for all three dice and total.



>STEP-03

Throw the dice

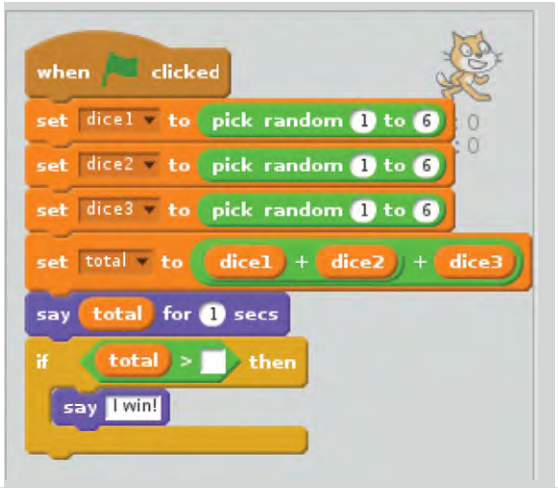
Click on Events and drag a **when green flag clicked** block to the scripts area. Below this you need to add three **set diceN to pick random 1 to 6** blocks. Below these blocks, add **set total to dice1 + dice2 + dice3** (you need to drag one **( ) + ( )** block inside another to add up three blocks.



>STEP-04

Speak

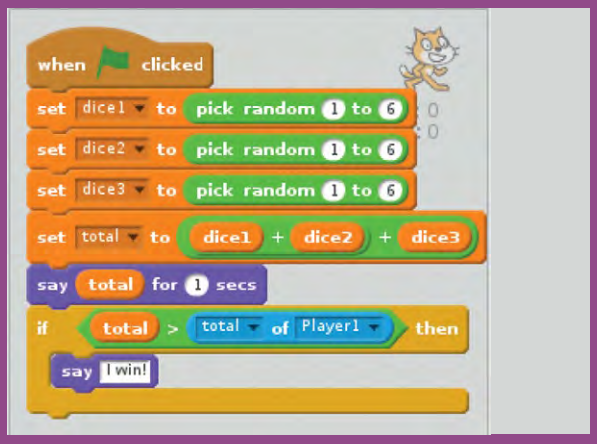
Now drag a **say for** block and attach it to the end of the code. Change it to **say total for 1 secs**. Below that, drag an **if** block. Inside, add a **[ ] > [ ]** (greater-than) block. Drag the **total** variable to the left side of the greater-than block. Drag a **say** block inside the **if** block and change 'Hello!' to 'I win!'.



>STEP-05

Compare scores

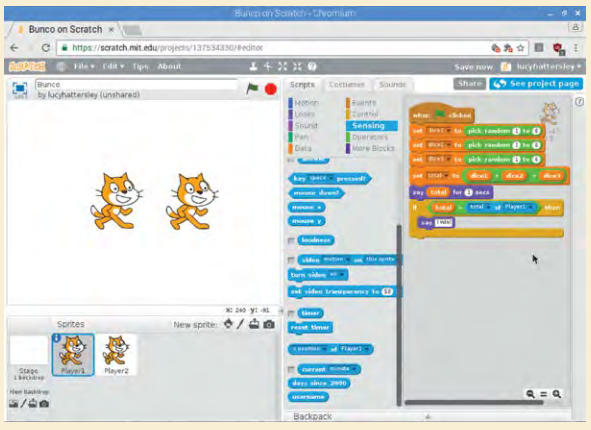
We've only got one sprite so far. But we're about to add another and compare one sprite's total with the other. Choose the Sensing selection of blocks and look for one marked **x-position of Player1**. Change '**x-position**' to total. Drag the block into the correct side of the greater-than block.



>STEP-06

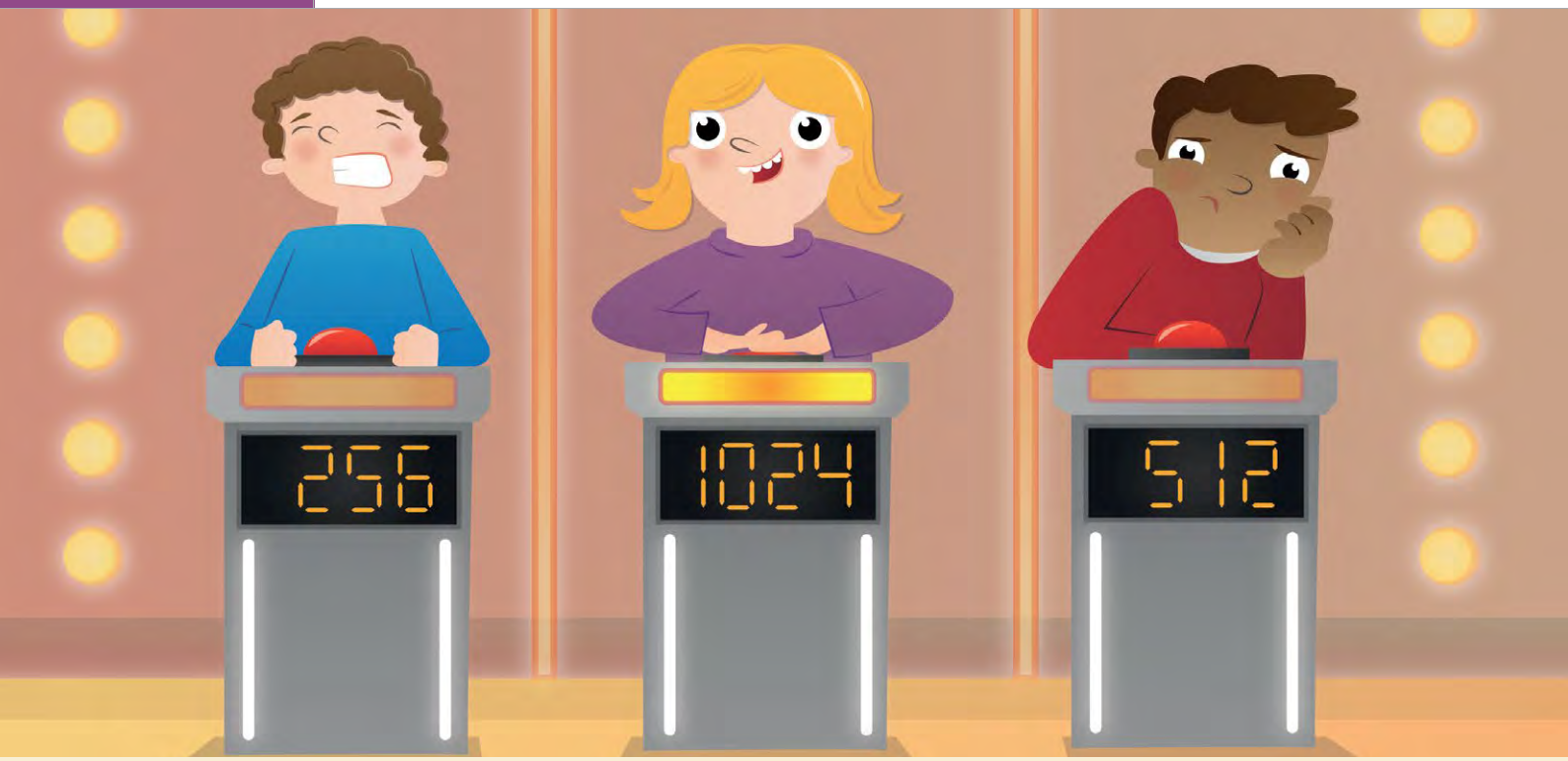
Player 2

Our Player 1 is ready. Now we're going to clone the sprite to create a Player 2. Right-click the sprite and choose Duplicate. The new cat sprite will be automatically called 'Player2'. Click on Player1 in the Sprites window. Change the **total of Player1** block to **total of Player2** (as shown below). Now Player1 compares their total to Player2 (and Player2 is comparing theirs to Player1). Click the green flag to run the program and see which player wins.



SCOPE

The concept of scope is important in object-oriented programming. In Scratch, it's so simple that you may not even notice it. But our two players each have their own **dice1**, **dice2**, and **dice3** variables, plus a **total** variable. These variables are local in scope. When Player1 announces **total**, it's their total. If both sprites accessed the same total, it would be global in scope. Variables in objects are local in scope.



CREATE BUNCO IN PYTHON

Recreate our game of Bunco in Python

CLASS AND INSTANCE

One of the biggest differences between creating sprites in Scratch and objects in Python is that you create an object using a class. This code acts as a blueprint for the object.

In Scratch, you create a sprite and then duplicate it. The second sprite has the same functions as the first. It also has its own set of variables.

In Python (and other programming languages), things work somewhat differently. You don't create an object directly. Instead, you create a blueprint for the objects. This blueprint is known as the 'class'. Don't think of a school, though. Class here means a category of similar items. It's rather like a 'Class M' planet in Star Trek: though different, these are all Earth-like planets.

Once you've created your class, you use it to create objects. These are known as 'instances' or 'object instances'. They all share similar properties. They all have the same variables and functions (called 'methods'). In Scratch, we create one sprite and then duplicate it (to get two sprites). In Python, we create one class definition. We then use this to create two object instances.

Our basic version of Bunco runs just fine in Scratch. Now we're going to recreate it in Python. The translation will help us get a good understanding of how objects work.

First, think about how we could make the game procedurally.

There is a module called **random** that we can import to create random numbers. So we'd need to import that. Then we could create a list for each player. And use the **`randint`** function to add three random numbers between one and six.

We could then use an **`if else`** block with the **`sum()`** function to add up each player's numbers. The player with the highest score wins.

Type out the code from **bunco_procedural.py** to test the program.

There are two problems with this procedural. Bunco is a much more complex game in real life. It is played in six rounds, and players score 21 points

Feature

Language
>PYTHON

DOWNLOAD:
magpi.cc/2jxE6WJ

if they roll all three dice that match the number of the round (three 1s in round 1, or three 2s in round 2, and so on). That's known as rolling a 'bunco'.

We're not going to create all that complexity here. But we are going to add extra types of player. Cheats! One scoundrel has loaded dice; the other rapscallion swaps out one die for a six.

We're then going to play thousands of games and see who wins.

This complexity would be extremely difficult in procedural programming. It requires us to rethink our approach to Bunco. And OOP is the answer.

OOPs upside your head

Instead of creating a list of variables for each player at the start, we're going to create a class called **Player**.

The code in **bunco_oop.py** represents a dice player. We then use it to create two players (see 'Class and Instance').

As with our procedural code, we start by importing the **randint** module.

Now we define our player objects. To do this, we create a class definition. It looks like this:

```
class Player:
```

Inside the class definition is indented code that describes the player object.

Notice that the class name is capitalised and, unlike function definitions, there are no parentheses.

The first thing we need to add is a list to contain the dice. Normally this would be just **dice = []**. But if we wrote it like this:

```
class Player:
    dice = []
```

...we'd have a problem. This code is equivalent to choosing 'For all sprites' in Scratch. Every player created using this code would share a single set of dice and get the same results. We want to use the equivalent of 'For this sprite only'.

To ensure that all our players have their own set of dice, we need to wrap the **dice = []** list inside a quirky function called **__init__()**.

It looks like this:

```
class Player:
    def __init__(self):
        self.dice = []
```

The **__init__()** function runs when you use a class to create an object.

This is known as a 'constructor' or 'initialiser'.

Later on, when we use this **Player** class to create player objects, the **__init__()** code runs each time

bunco_procedural.py

```
import random

player1_dice = []
player2_dice = []

for i in range(3):
  player1_dice.append(random.randint(1,6))
  player2_dice.append(random.randint(1,6))

print("Player 1 rolled" + str(player1_dice))
print("Player 2 rolled" + str(player2_dice))

if sum(player1_dice) == sum(player2_dice):
  print("Draw")
elif sum(player1_dice) > sum(player2_dice):
  print("Player 1 wins")
else:
  print("Player 2 wins")
```

bunco_oop.py

```
from random import randint

class Player:
  def __init__(self):
    self.dice = []

  def roll(self):
    self.dice = [] # clears current dice
    for i in range(3):
      self.dice.append(randint(1,6))

  def get_dice(self):
    return self.dice

player1 = Player()
player2 = Player()

player1.roll()
player2.roll()

print("Player 1 rolled" + str(player1.get_dice()))
print("Player 2 rolled" + str(player2.get_dice()))

if sum(player1.get_dice()) == sum(player2.get_dice()):
  print("Draw!")
elif sum(player1.get_dice()) > sum(player2.get_dice()):
  print("Player 1 wins!")
else:
  print("Player 2 wins!")
```

automatically. It creates a separate set of dice for each of our players.

The 'self' bit also needs explaining. Variables, like our **dice = []** list are normally disposed of when a function ends (or is returned).

So if we just put **dice = []**, the list would be created by **__init__()**, then immediately vanish.

Python gets around this problem using the keyword 'self'. You put 'self' inside the parentheses of the **__init__()**:

```
def __init__(self)
```

Then we use **self**, followed by a dot, to store the variable in this version of the object.

```
self.dice = []
```

You then use **self.** in functions when you want to access or change a variable, by writing **self** inside the parentheses of the function. Like this:

```
def roll(self):
```

The concept can be mind-boggling (it's passing a version of itself into itself). So focus on the practical steps rather than the esoteric theory of how it works:

- Put a special function at the start of a class called **__init__(self)**.
- Put the variables you want to use inside **init**.
- Create the variables with **self.**, like **self.name** or **self.age** or **self.dice = []**.
- Place **self** inside the parentheses of functions that need to access the variables.
- Use **self.** and the name of the variable inside the function to use it.

Got that? Don't worry too much if it seems weird. That's the hardest part and it will get easier with practice.

Now we've got our dice list sorted, what about the other functions?

Methods in the madness

Now that our class has a list for the dice, we need to roll the dice. For that, we'll create a more regular function definition.

```
def roll(self):
  dice = [] # clears any current dice
  for i in range(3):
    self.dice.append(randint(1,6))
```

An object's functions are called 'methods', but they are created in the same way.

There are lots of different types of methods, and you can create whatever you like, but common ones are called 'setters' and 'getters'.

Our **roll** method is a 'setter'. It sets the **dice** list to three random numbers.

What do you think a 'getter' does? That's right. It gets the variables inside the object and returns them. We have just the one getter:

```
def get_dice(self):
  return self.dice
```

Getters and setters seem a bit odd at first. After all, you could just reach into an object and access the variables.

Well, you could, at least in Python, but this is considered a bad thing to do. One of the points of OOP is that objects contain their variables and keep them safe from other objects. So you don't just reach inside an object and access variables.

Instead, you create methods (functions) that set the variables and get them. Then you use these methods to set and get variables.

Now we've created our class definition, we can use it to create objects.

Create away

You create objects just as you would a variable. You use the assigns operator (**=**). We're going to create two dice players:

```
player1 = Player()
player2 = Player()
```

Note that **player1** and **player2** are not called 'variables'. They are called 'object instances'.

We access the object instance's methods using dot notation. That is where you use the name of the object instance, followed by a dot, then the name of the method you want to use.

We created a method, **get_dice()**, that returns the dice stored. We would access this method using dot notation, such as **player1.get_dice()**.

First, we use the **roll** method to get each player object to roll its dice:

```
player1.roll()
player2.roll()
```

The rest of our **bunco_oop.py** program is really very similar to **bunco_procedural.py**. The difference is that here we use the **.get_dice()** method in place of **sum()**.

Inheritance cheats

We've already mentioned that one advantage of OOP is that we can create hundreds or thousands of players with their own set of variables.

There's little point in our program as it is, as the players are all using the same dice and have the same chance of winning. What would happen if we introduced some cheats?

We're going to create a cheat who swaps out one die for a six. The blighter.

Our cheat is not alone. We have another one who uses three loaded dice. They always roll one higher (unless they're already a six).

Both cheats would beat a regular player over a few hundred games. But which of the cheats would win against the other?

It's pretty close. To find out the answer, we'll need to run a simulation that plays hundreds of thousands of games.

We're going to create our cheats using a technique called inheritance. This technique is where you create

> "But which cheat would win against the other?"

a class that takes on all the properties (instance variables and methods) of another class. It also adds a few of its own.

Think of a child inheriting its parents' features. It might get its dad's big nose but grow knobbly knees all by itself.

Our cheats will inherit the same **dice** and **roll** functions as the parent, but they will have cheat functions all of their own.

Our **bunco_module.py** program defines the **Player()** class and two children:

```
class Player:
class Cheat_Swapper(Player):
class Cheat_Loaded_Dice(Player):
```

Objects that inherit from a parent are defined using the same class keyword.

However, the name of the parent is placed inside parentheses of the child.

Our two cheats inherit all the variables and methods (functions) from the parent. So they already have a **dice** list and **roll()** and **get_dice()** methods.

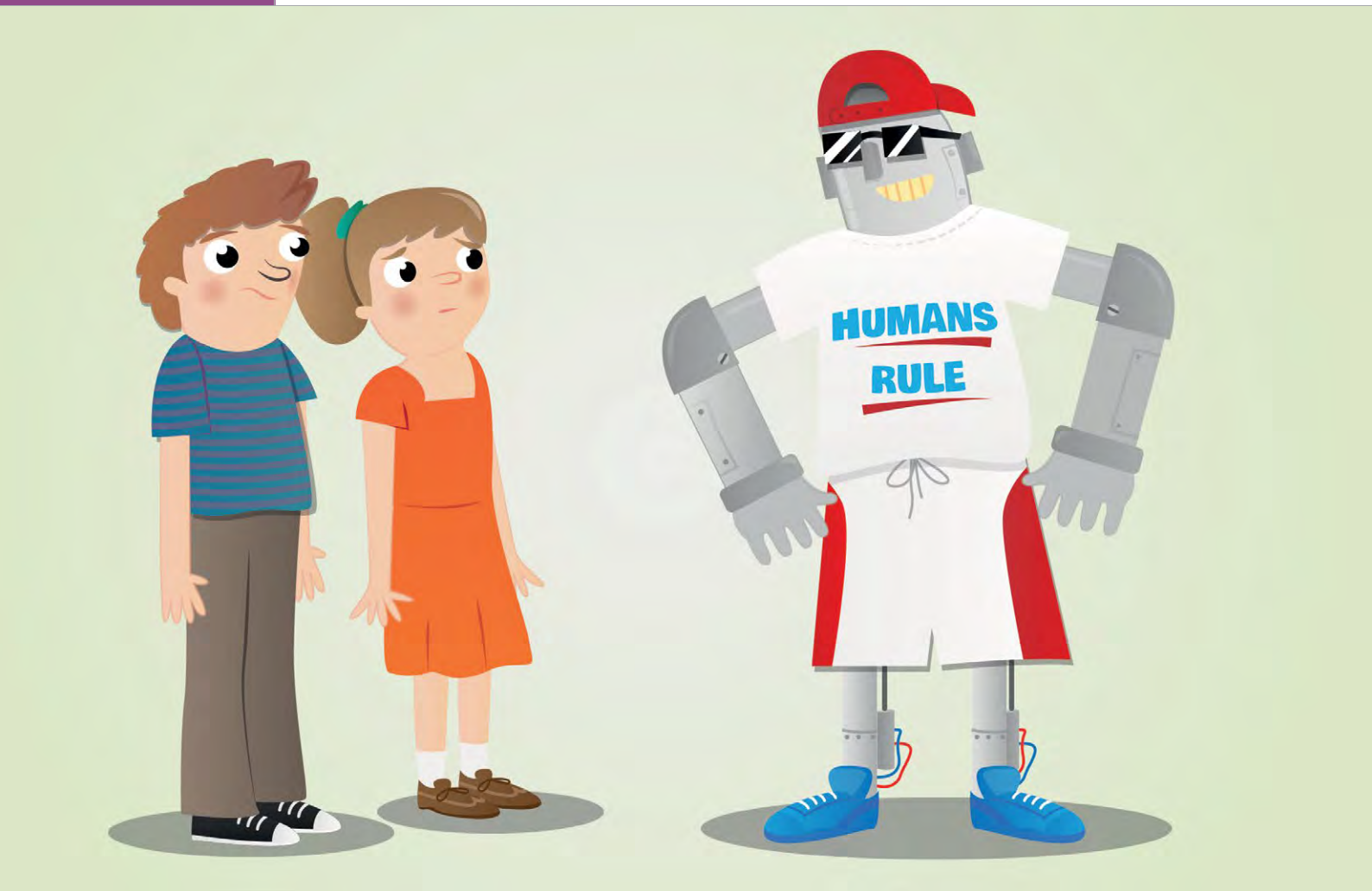
We now give each cheat an additional method, called **cheat**. This is implemented in a different way for each type of cheat.

bunco_module.py

```
from random import randint

class Player:
  def __init__(self):
    self.dice = []

  def roll(self):
    self.dice = [] # clears current dice
    for i in range(3):
      self.dice.append(randint(1,6))

  def get_dice(self):
    return self.dice

class Cheat_Swapper(Player):
  def cheat(self):
    self.dice[-1] = 6

class Cheat_Loaded_Dice(Player):
  def cheat(self):
    i = 0
    while i < len(self.dice):
      if self.dice[i] < 6:
        self.dice[i] += 1
      i += 1
```

bunco_single_test.py

```
from bunco_module import Player
from bunco_module import Cheat_Swapper
from bunco_module import Cheat_Loaded_Dice

cheater1 = Cheat_Swapper()
cheater2 = Cheat_Loaded_Dice()

cheater1.roll()
cheater2.roll()

cheater1.cheat()
cheater2.cheat()

print("Cheater 1 rolled" + str(cheater1.get_dice()))
print("Cheater 2 rolled" + str(cheater2.get_dice()))

if sum(cheater1.get_dice()) == sum(cheater2.get_dice()):
  print("Draw!")

elif sum(cheater1.get_dice()) > sum(cheater2.get_dice()):
  print("Cheater 1 wins!")

else:
  print("Cheater 2 wins!")
```



The **Cheat_Swapper** class definition has a relatively straightforward cheat method:

```
class Cheat_Swapper(Player):
  def cheat(self):
    self.dice[-1] = 6
```

Cheat_Swapper's cheat method finds the last item in the **dice** list and sets it to 6.

Our **Cheat_Loaded_Dice** class definition has a slightly more complicated cheat method:

```
class Cheat_Loaded_Dice(Player):
  def cheat(self):
    i = 0
    while i < len(self.dice):
      if self.dice[i] < 6:
        self.dice[i] += 1
      i += 1
```

This method iterates through the dice in the list, checking whether each die is lower than six. If so, it increases its value by one.

Make sure that you create the code in **bunco_module.py** and save it with the same name.

Notice that this code doesn't have any procedural programming below it. This is because we're going to import it (so you can see what happens when you use **import** in your Python programs).

Now we will create the code that uses these objects in a separate file. Enter the code from the **bunco_single_test.py** listing and make sure you save it in the same directory as **bunco_module.py**.

The first line imports the **Player** class definitions from our **bunco_module.py**.

```
from bunco_module import Player
```

Extra points to you if you spotted that **bunco_module** is listed without the '.py' file extension. This is how you import code from other files into your program.

The **import Player** line pastes in the **class Player** code from **bunco_module.py**. It's as if you had included that code in your program.

Compare this line to the **from random import randint** code at the start of **bunco_module.py**. The idea is the same.

We import the other two class definitions we created:

```
from bunco_module import Cheat_Swapper
from bunco_module import Cheat_Loaded_Dice
```

The rest of the **bunco_single_test.py** code creates the same game as our earlier **bunco_oop.py** program.

Now we create two object instances using the **Cheat_Swapper** and **Cheat_Loaded_Dice** definitions we imported from **bunco_module**:

```
cheater1 = Cheat_Swapper()
cheater2 = Cheat_Loaded_Dice()
```

We then use the **roll()** method. Notice that neither **Cheat_Swapper()** or **Cheat_Loaded_Dice()** has a **roll** method definition. This is a function they both inherit from their parent class, **Player()**:

```
cheater1.roll()
cheater2.roll()
```

Next we call the **cheat()** method from each object instance:

```
cheater1.cheat()
cheater2.cheat()
```

Although each object has a method called **cheat()**, the objects have different implementations. So **cheater1** changes the last die to a 6, and **cheater2** increases each individual die's value by one.

Run the program by pressing **F5** and see which of the players wins. Run it again and you'll get different results. Keep running the program and you'll find it's a pretty close call.

Look inside the folder containing the code and you'll see a new file has appeared called **bunco_module.pyc**. This is a 'compiled file' and is created the first time you run a program that imports code. You don't usually see compiled files because you import code tucked away inside Python on your computer. Don't worry about it. You can't open and make sense of it in a text editor. Delete it if you wish. It'll be recreated when you use **bunco_module.py** in our final program. You can just ignore it for now.

To discover which of the two cheats has the edge, we need to run a simulation. We need to play hundreds of thousands of games and keep track of who wins the games.

Our final program, **bunco_simulation.py**, does just that. This program brings together everything we've learned about OOP. The code in **bunco_simulation.py** creates two cheats and plays 100,000 games. It imports class definitions from our **bunco_module.py** program (so make sure you save it in the same folder).

bunco_simulation.py

```
from bunco_module import *

swapper = Cheat_Swapper()
loaded_dice = Cheat_Loaded_Dice()


swapper_score = 0
loaded_dice_score = 0

number_of_games = 100000
game_number = 0

print("Simulation running")
print("==================")
while game_number < number_of_games:
  swapper.roll()
  loaded_dice.roll()

  swapper.cheat()
  loaded_dice.cheat()

  #Remove # before print statements to see simulation running
  #Simulation takes approximately one hour to run with print
  #statements or ten seconds with print statements
  #commented out

  #print("Cheater 1 rolled" + str(swapper.get_dice()))
  #print("Cheater 2 rolled" + str(loaded_dice.get_dice()))

  if sum(swapper.get_dice()) == sum(loaded_dice.get_dice()):
    #print("Draw!")
    pass

  elif sum(swapper.get_dice()) > sum(loaded_dice.get_dice()):
    #print("Dice swapper wins!")
    swapper_score+= 1

  else:
    #print("Loaded dice wins!")
    loaded_dice_score += 1

  game_number += 1

print("Simulation complete")
print("-------------------")
print("Final scores")
print("------------")
print("Swapper won: " + str(swapper_score))
print("Loaded dice won: " + str(loaded_dice_score))

if swapper_score == loaded_dice_score:
  print("Game was drawn")
elif swapper_score > loaded_dice_score:
  print("Swapper won most games")
else:
  print("Loaded dice won most games")
```

Maker Says

"Bring your data home"

Nextcloud

NEXTCLOUD BOX

Could a supply-your-own-Pi bare-bones box be your ticket to data independence and security?

The low power draw of the Raspberry Pi family means they are very well suited to always-on server applications. A network-attached storage (NAS) device is one possible use case, but dealing with external USB-connected hard drives and multiple power bricks is an unwelcome side effect of rolling your own solution.

That's where the Nextcloud Box comes in. Created as a partnership between storage giant Western Digital's WD Labs division and the open-source Nextcloud project, the Nextcloud Box takes WD Labs' PiDrive and uses it as the home for a clever variant of the Nextcloud server software.

Unpacking the box, the first thing you'll notice is that no Pi is included. The Nextcloud Box comes with a 1TB PiDrive USB hard drive, clever magnetic housing, power splitter cable, data cable, mounting screws and screwdriver, a 4GB micro SD card, and a power supply capable of driving both a Raspberry Pi and the hard drive. It's then up to the user to supply the Pi itself; at the time of testing, only the Raspberry Pi 2 was supported, with Pi 3 compatibility promised in a free software update which should have been released by the time this issue hits shelves.

Cloud setup

Installation is quick and easy: simply fix your Raspberry Pi into the casing using the provided screws, run the cables, and route a network cable. The small 4G micro SD card is used only for the initial boot process, which copies the Ubuntu Snappy Core operating system onto the 1TB hard drive in an entirely automatic installation process which takes about ten minutes to complete.

The first surprise with the Nextcloud Box is that there's never a need to connect a display or keyboard, unless you want to. All management is carried out using a web interface, bar a few advanced tasks which require SSH access to the Pi. From here, using the Nextcloud Box is like using any other installation of Nextcloud: user accounts, encryption settings, and additional features from media streaming through to a centralised calendar system can all be set up and configured directly within the browser.

By default, the Nextcloud Box is accessible only from your local network, either using the browser interface or dedicated client software. Configuring it for external access is one of the few parts of the process which could be fairly described as inaccessible to less technical types: you'll need to log in via SSH, configure a dynamic DNS service, run a script which retrieves a cryptographic

Related

WD LABS PIDRIVE BERRYBOOT EDITION

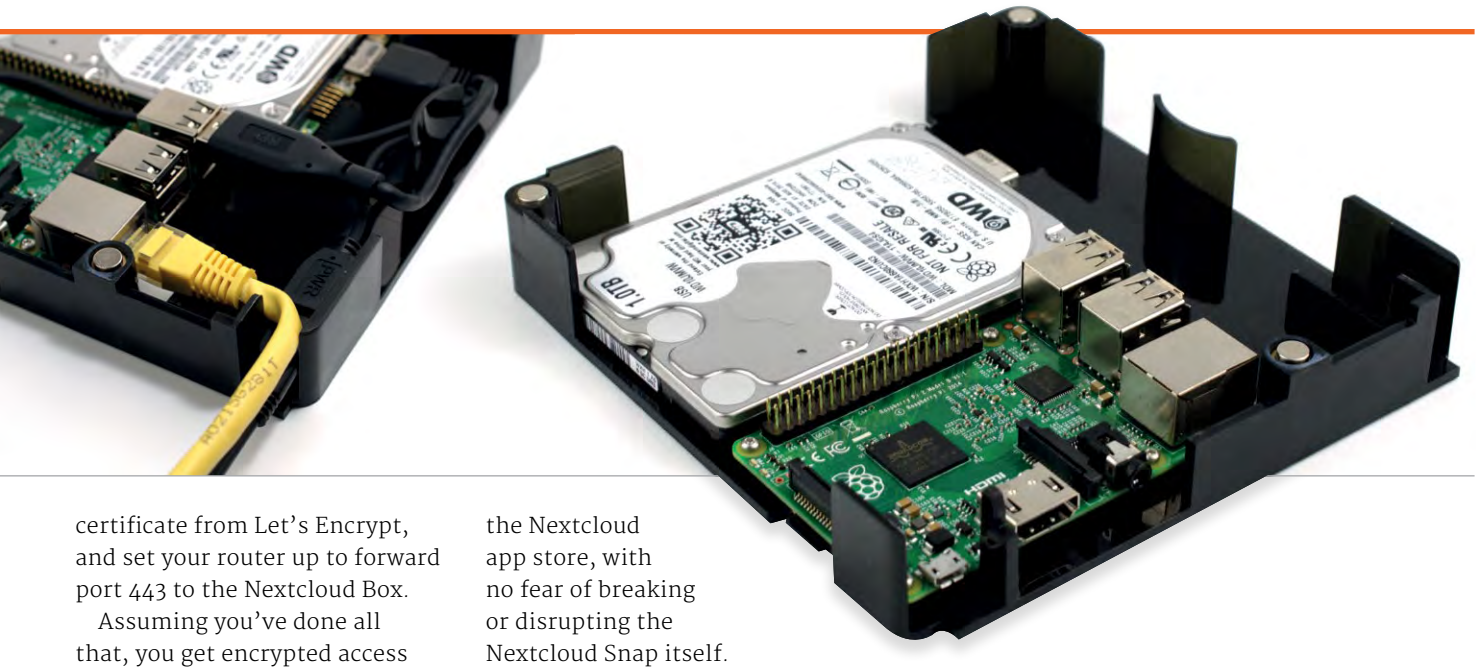
If you don't need the Nextcloud software, the PiDrive is available as a standalone option direct from WD Labs, but you'll need to buy the case and power supply separately.

£58/$60

wdlabs.wd.com

magpi.cc/2ivwjou

£60 / $80



certificate from Let's Encrypt, and set your router up to forward port 443 to the Nextcloud Box.

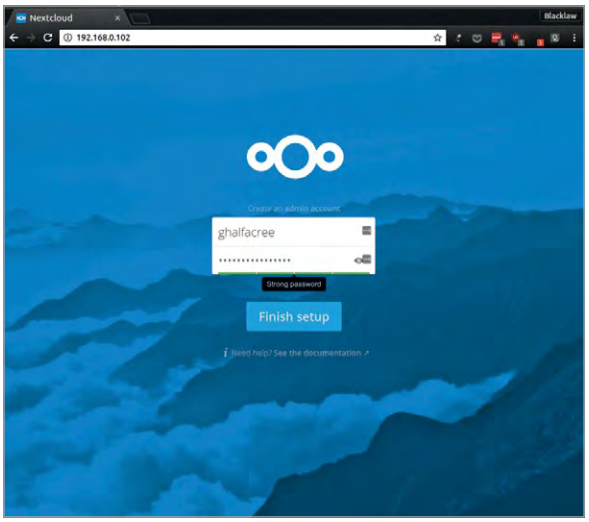
Assuming you've done all that, you get encrypted access to your data – including two-way synchronisation, a great way to keep photos and videos from your phone safe – anywhere in the world with no ongoing costs. Performance is, surprisingly, very smooth: despite the network port and hard drive both sharing the Pi's single USB channel, transferring files to the Nextcloud Box felt snappy with little in the way of annoying pauses or hitches.

If you don't feel like dedicating a Pi to just Nextcloud use, there's the entire underlying Ubuntu Snappy Core operating system to play with. Nextcloud itself is installed as a 'Snap,' a self-contained collection of all the dependencies required to run the software. You're free to install as many other Snaps as you like, adding extra features above and beyond what you'll find in the Nextcloud app store, with no fear of breaking or disrupting the Nextcloud Snap itself.

Sadly, not everything with the Nextcloud Box is as smooth as its software. The supplied Nextcloud-branded housing, one of several designs created by WD Labs, requires that the cables go through some tight bends before being routed out through gaps in the walls. The biggest flaw by far is the power cable for the Pi; taking the form of a clever pass-through cable providing power to the hard drive as well, it exits the case only to turn a full 180 degrees and go back in again in order to reach the Pi's micro-USB power socket.

Providing you're not constantly connecting and disconnecting the Nextcloud Box, though, it's largely a set-and-forget system: install the cables, configure the software, and it'll keep itself automatically updated through Ubuntu Core's Snappy system.

> "By default, the Nextcloud Box is accessible only from your local network"



Above The Nextcloud box runs Ubuntu

Last word

Powerful, flexible, and extremely configurable, Nextcloud is a fantastic bit of software which runs perfectly on the Raspberry Pi 2. Sadly, WD Labs' contribution to the project still needs a bit of work, with the cabling inside the box going through some worryingly sharp bends.

★★★★☆

nostarch.com/scratchcards

£20 / $25

Maker Says

"Want to introduce kids to coding in a fun and creative way?

No Starch Press

SCRATCH CODING CARDS

Flash cards with Scratch code on to help kids learn and experiment

We like Scratch here at *The MagPi* – it's an amazing piece of software which is very easy to use, and it can create some truly astonishing projects. Perfect for helping kids learning to code to get to grips with the logic and language of programming. While we've definitely had tutorials in the magazine (and in a dedicated Scratch book!), we're not the only place you can learn about Scratch. Meet Scratch Coding Cards.

These flash cards provide a unique way to go about learning Scratch, presenting a concept or program (animating your name, creating a horse race) and then working you through step-by-step on how to build the project. Each project is colour-coded as well, so even if they do go everywhere, you'll be able to figure out how to reassemble the pack.

The USP

One of the most interesting things about the projects is that in some of them you don't have to follow the prescribed order. Some of the programs have an end goal you're working to create, but others just have you experimenting with code and Scratch. This is honestly one of the best ways to do it, as playing about with your code to see different results is a great way to test the limits of what you can do.

Most importantly, parts of the code are explained as you make blocks, although we feel like they could go a bit more in-depth in some cases. There's a size issue they have to work with, and admittedly young ones don't want an essay on each function. At least they're getting to use them in practice, which will help.

Price-wise it's interesting; it's a premium product at least and the stock on the cards is very nice (when you make magazines you notice these things); however, the set might be a touch too expensive for what you're getting. Still, it is a very good pack.

Last word

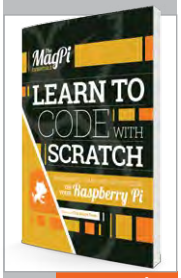
Possibly a little overpriced, and maybe it doesn't quite explain itself enough, but it should lead to hours of fun and a fairly thorough knowledge of Scratch.

★★★★☆

Related

LEARN TO CODE WITH SCRATCH

We've got a lot of great stuff in our book for learning Scratch, along with a whole host of cool tutorials.



£4 / $6

magpi.cc/Scratch-book

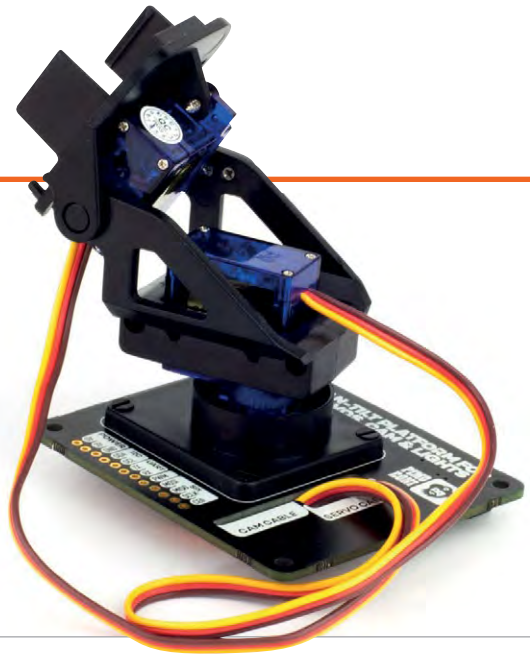
pimoroni.com

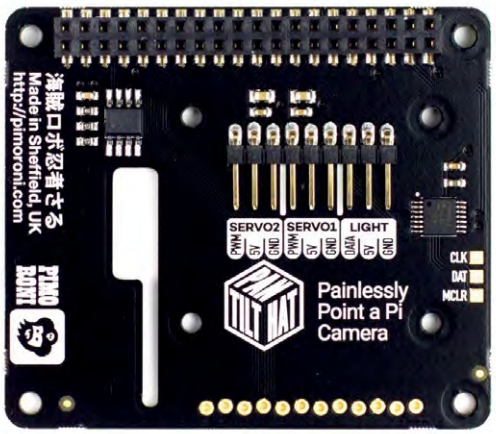
£30 / $37

Maker Says

"Let your Raspberry Pi and camera look at the world around you"

Pimoroni





PAN-TILT HAT

Raspberry Pi camera stand with horizontal and vertical motion servos

The Raspberry Pi Camera Module is one of the best accessories you can get, enabling cheap photography on the Pi. But it doesn't sit upright on its own – a stand is required.

The Pan-Tilt HAT fulfils this function and a whole lot more. The Camera Module is mounted on the end of a robotic arm that sits on top of the HAT. Thanks to the arm's horizontal and vertical joints, the camera can be angled precisely by the two servo motors.

The finished effect is adorably cute, instantly imbuing your Raspberry Pi with personality as it looks around the room.

It's really useful too. You could set the Pan-Tilt HAT up to monitor a room, and then use VNC or SSH to adjust its viewing position remotely.

Alternatively, you can set up a Raspberry Pi with face-tracking software and connect it to the Pan-Tilt HAT. Pimoroni, the HAT's makers, also suggest mounting it on top of a robot for a set of eyes.

Setting it up

First, you have to set the HAT up. Fortunately, there is an online setup guide (**magpi.cc/2hR4NFC**).

The board has a GPIO connection on one side, and servo connections on the other. The two sets of cables on the arm are connected to Servo 1 and Servo 2 on the board (1 for pan, 2 for tilt). A third servo channel can be used to control an optional NeoPixel strip for lighting.

Camera control

You can download all of the code from Pimoroni's GitHub page (**magpi.cc/2hRrjyo**). You need to install the **pantilthat** module to access the controls.

After importing the **pantilthat** library in Python, you use **pan()** and **tilt()** methods to change the camera position. These accept any value between -90 and 90. To set the camera straight forward, for example, you would use:

```
pantilthat.pan(0)
pantilthat.tilt(0)
```

To look up by 45 degrees, use:

```
pantilthat.tilt(-45)
```

To look all the way to the camera's left, you'd put:

```
pantilthat.pan(90)
```

We would have dearly loved more software examples. There are ones for motion and NeoPixels, but none for recording from the camera or face-tracking. A few more sample programs and it'd be perfect.

Even so, we had a lot of fun setting up the Pan-Tilt HAT and look forward to researching and coding a face-tracking program.

Last word

A highly enjoyable and extremely cute accessory. With a bit of research, you should be able to create some fun things with it.

★★★★☆

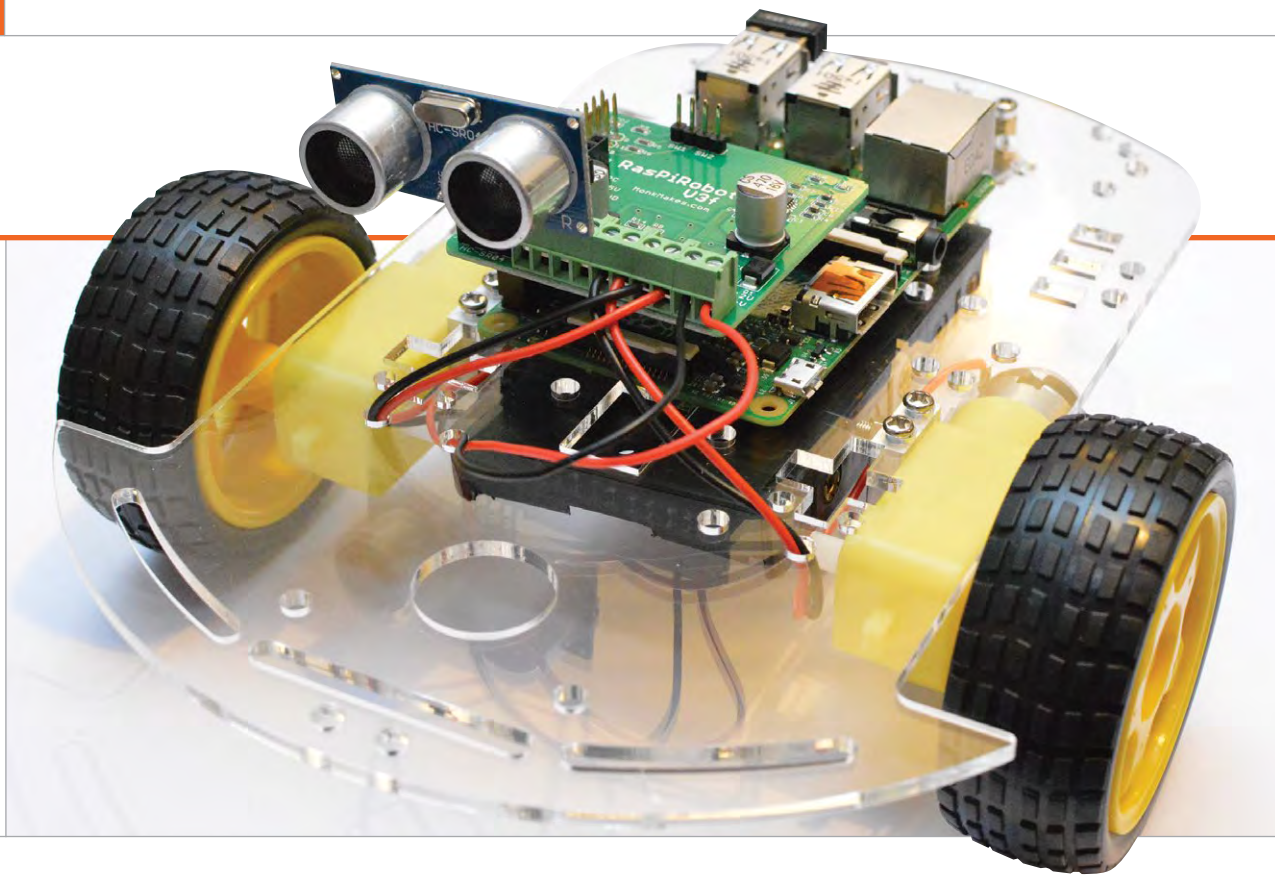
Related

RASPBERRY PI CAMERA MOUNT

It doesn't move, but if all you're after is a camera mount, you can pick one up from The Pi Hut for a tenth of the price.

£3 / $3.69

magpi.cc/2hRtkdT



Maker Says

"Build your own Robot Rover for Raspberry Pi – just add Raspberry Pi and batteries"

Monk Makes

RASPIROBOT ROVER KIT

Related

GOPIGO

The GoPiGo is another fairly simple-looking robot kit. However, it too can be programmed to do a lot of robotics tasks.

£80 / $99

magpi.cc/2hA8f6i

A kit that includes more than just a robot, how well does it work for teaching youngsters about coding?

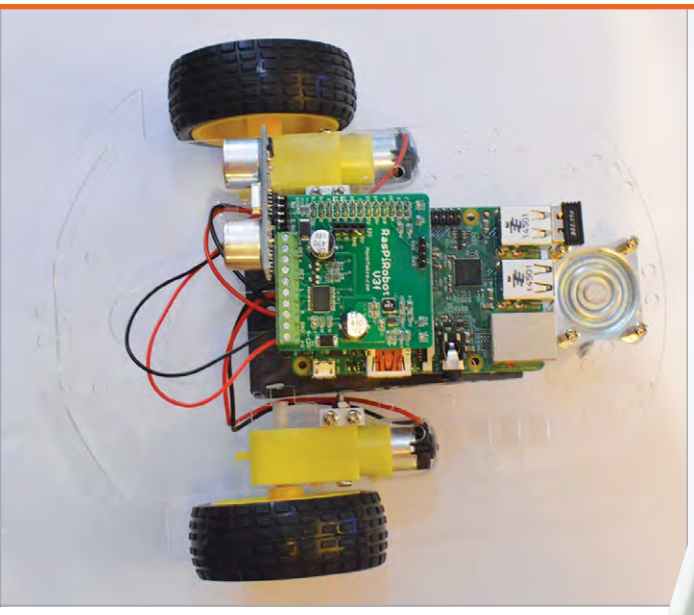
We always like the kit from Monk Makes. It's functional, comes with great documentation and most importantly is always a good price. The RasPiRobot board that you can currently buy scored well in our review a little while ago, so it's nice to see a kit that uses it for a Pi robot in the way Simon Monk intended.

This particular robot kit is two kits in one. As well as being able to create a basic robot that you can program, you have access to a push button and the Raspberry Squid RGB LED. These work on the RasPiRobot board without it needing to be built as a robot; however, it does also mean you can attach them once the robot is built. It's an interesting concept as in theory you can use the button and LED as a 'getting started with GPIO programming' step and then move onto building the robot and doing even more.

As for the robot build itself, the process is nice and easy. The chassis is made up of one piece of laser-cut acrylic and it uses two motors to drive it. A caster wheel is attached to the rear so it can pootle about, and the battery pack can be attached to the underside. The Pi is merely stuck to the surface using some Blu-Tack or double-sided tape and the rest of the components slot neatly into the motor controller which attaches on top of the Pi. It's very simple and there's a couple of video guides from Simon that show you how to do it, although it would be nice to have a step-by-step visual guide as well. It took us an episode of the original *Star Trek* to build, so under an

magpi.cc/2i7kJzh

£40 / $48



Above There's not much to build with the kit and you'll be done very quickly. There are lots of extra parts, though

hour, although that does include us struggling to peel off the protective layer of paper from the chassis and find the rechargeable batteries for it.

Tough enough

Once built it's quite sturdy. The chassis isn't large enough to be bendy and the Raspberry Pi can be affixed firmly enough in place so it doesn't fly off. The kit also comes with speed sensors in the form of timer discs you can attach to the motors if you want to get the extra sensors to add them. Speaking of adding sensors, there are plenty of spots along the chassis to add them. The front is especially good for line and proximity sensors, as well as adding a pan-and-tilt arm. You'll have to do the research yourself to figure out how to get it connected up to the board, although that's all part of the fun.

Programming it is very easy, using the same standard Monk Makes libraries. The instruction booklet that comes with the robot also provides some little tutorials on getting the LED and button working, as well as programming the robot for remote control and autonomy. The ultrasonic rangefinder also has a bit in there, so you can start programming with that as well. The great part about all these instructions is that you can combine them to do crazy things. For example, you could have the button start a little mission where it drives towards the nearest wall and uses the rangefinder data to change the colour of the LED as it gets closer.

> "You can use the button and LED as a 'getting started with GPIO programming' step"

While a minor spin on the normal robot kit, it does open up a lot more types of customisation and play options that are much simpler to understand.

We very much like the RasPiRobot Rover and its kit. We've built a few other similar and simple robots and while they're always quite good, they don't quite go the extra mile like this. Not only that, but we can easily get our Raspberry Pi off it without having to fiddle around with tiny screws to do so. For the price, as well, it's an excellent starter kit for kids interested in the Raspberry Pi and programming electronics.

Last word

A wonderfully put together piece of kit, the RasPiRobot Rover is a great intro to robotics that can also be used far beyond the beginner level.

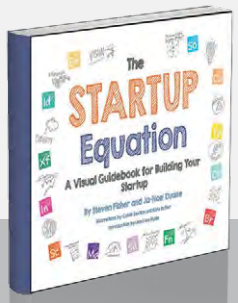
★★★★☆

RASPBERRY PI BESTSELLERS STARTUPS

Not just an IT book company, McGraw-Hill has the books to help you turn your project into a business.

THE STARTUP EQUATION

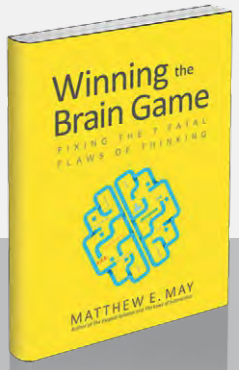
Authors: Steve Fisher, Ja-Nae Duane
Publisher: McGraw-Hill
Price: £27.99
ISBN: 978-0071832366
magpi.cc/2igcc1E



Will your project or idea make you one of the 'next billion entrepreneurs' referred to by the authors? They give you all of the tools necessary in a detailed, yet nicely visual guide.

WINNING THE BRAIN GAME

Author: Matthew E May
Publisher: McGraw-Hill
Price: £16.99
ISBN: 978-1259642395
magpi.cc/2ig7lO3



Learn to 'Think Different', and run rings around your competitors: your mind can overcome the seven fatal flaws of thinking using framestorming, jumpstarting, prototesting, and May's well-tested toolkit.

ONE PERFECT PITCH

Author: Marie Perruchet
Publisher: McGraw-Hill
Price: £19.99
ISBN: 978-0071837590
magpi.cc/2ig5SqX

You know your solution is brilliant, elegant, and way ahead of the competition, but backers and customers need to be engaged by more than dry facts. Learn to tell your story.

INVENT YOUR OWN COMPUTER GAMES WITH PYTHON

Author: Al Sweigart
Publisher: No Starch
Price: £23.99
ISBN: 978-1593277956
magpi.cc/2igb91v

This is a very well-paced introduction to coding and to Python 3, all done through games programming. Starting gently, but always thorough in its teaching approach, each game carefully introduces new concepts, from loops to working with lists. For example, in the Hangman game, the task is nicely broken down into separate tasks by flowchart, then composed into functions in the code, all nicely modular. Stepping through programs with the debugger, and entering breakpoints, is a welcome early chapter.

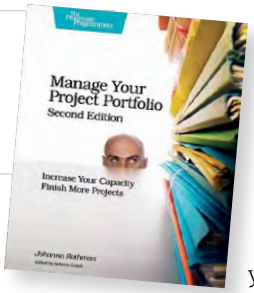
Games make a great foundation, with early emphasis on user interaction, data structures, and even AI, as well as the motivation of 'look what I've made; would you like a go?'. By the time the reader reaches Pygame in the last quarter of the book, she should have a firm grasp of some very important basics.

After four editions (the previous three through CreateSpace), and a number of other titles, Sweigart is close to perfecting the formula for painlessly teaching what, one occasionally forgets, is actually quite difficult. Should work equally well with adult beginners, and with those in the Code Club age group, leaving them ready to tackle more in-depth programming texts. Fun and educational – our favourite combination.

Score ★★★★☆

MANAGE YOUR PROJECT PORTFOLIO

Author: Johanna Rothman
Publisher: O'Reilly
Price: £28.99
ISBN: 978-1680501759
magpi.cc/2ig4LHA



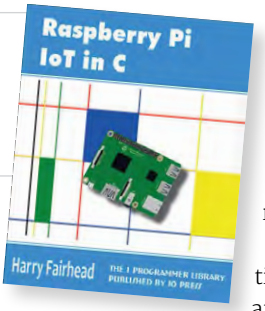
Never mind 'those who can, do', we find it's 'those who can, take on far too much'. This is such a familiar pattern among makers and free and open source software programmers, as much as amongst IT teams of medium and large enterprises for whom this book is written, that we think that anyone feeling overwhelmed beneath a mountain of unfinished – and quite possibly unfinishable – projects will find some useful nuggets here, if not a whole plan of escape.

For those of us who do work in larger organisations, trying to shield our teams from the clashing demands of (seemingly unthinking) management, this book offers not just snippets of wisdom, but a practical plan. Those snippets range from practical ways of bringing in Agile technologies to waterfall organisations, to a simple high-level view to show that you have no spare capacity, and plenty of relevant advice such as: "with enough multitasking, you can bring all work to a dead stop."

Drafting your portfolio and evaluating your projects, deciding which to commit to, making the decisions collaboratively – after all this, the last chapter is called 'Start Somewhere … But Start'. This survival manual for the hard-pressed will decrease stress levels in numerous organisations.

Score ★★★★☆

RASPBERRY PI IOT IN C



Author: Harry Fairhead
Publisher: I/O Press
Price: £14.85
ISBN: 978-1871962468
magpi.cc/2igfd22

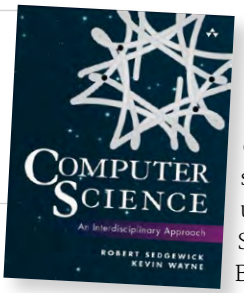
Get closer to the metal. Programming with C gives you the chance to work directly with the hardware, bypassing Raspbian's Linux drivers, for speed, custom features, and to understand the interfaces in the face of occasionally contradictory documentation. Harry Fairhead's book eschews HATs in favour of off-the-shelf sensors, so that you can work directly with all of the Pi's data buses.

The bcm2835 library is used, but sometimes interfacing through Linux's everything-is-a-file interface is suitable, and that is discussed; then implementing your own version of sysfs is shown, for a speedier version of the Linux API. Only minimal parts are needed, but use of a logic analyser or oscilloscope lets you follow along when measuring the actual speed of change on the GPIO lines – interesting if you're after close to real-time performance.

After GPIO, the real-time scheduling problem, and generating musical notes, the I²C bus is demystified, then an excellent section on the 1-wire bus; after that, the SPI bus. Old-school serial interfaces are problematic, with the kernel seeing them as somewhere to log on from, and the Pi 3 using it for Bluetooth: various workarounds are discussed. Web and WiFi round off an invaluable guide. Recommended.

Score ★★★★★

COMPUTER SCIENCE



Authors: Robert Sedgewick, Kevin Wayne
Publisher: Addison-Wesley
Price: £49.99
ISBN: 978-0134076423
magpi.cc/2igbocZ

Sedgewick and Wayne of *Algorithm* (book and MOOC, see *The MagPi* #45) fame, give us an 1,168-page guide to computer science, with Java programming. It's not just an intro to the science of computing, but is imbued throughout with a love of science, engineering, and mathematics – found particularly in many of the excellent example problems and exercises, which draw on equations and algorithms from all areas of STEM.

This helps lift it above run-of-the-mill text books, and combines with clear-headed writing to make it a joy to follow along with. Structure is logical: from basics (variables, conditionals, arrays, etc.); functions; OO programming; algorithms and data structures; theory of computing; and then some practical elaborations on that. Along the way, multimedia is introduced early on, always in the service of interdisciplinary understanding, with Sierpinski triangles and Barnsley ferns asking 'what does computation tell us about nature?', and vice versa.

Sounds like just an old-school alternative to those zany, cartoon-laden introductions to programming we all prefer nowadays? Well, there's more. Each section features dozens of exercises which, should you work your way through all of them, will make you a better coder than many programmers of longer experience. Outstanding.

Score ★★★★★

ESSENTIAL READING: CLASSIC SCI-FI

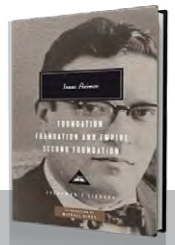
Don't be out-geeked! Catch up on these sci-fi classics, old and modern.

Brave New World

Author: Aldous Huxley
Publisher: Vintage Classics
Price: £8.99
ISBN: 978-0099518471
magpi.cc/2igbG3v

Happiness without meaning. "A more astute guide to the future than any other 20th century novelist," says JG Ballard.

Foundation trilogy



Author: Isaac Asimov
Publisher: Everyman
Price: £13.99
ISBN: 978-1841593326
magpi.cc/2ig87uw

Now 70 years old, it remains stunning in its breadth of vision, and immediate in its individual versus collective narratives.

Neuromancer

Author: William Gibson
Publisher: Gollancz
Price: £8.99
ISBN: 978-1473217386
magpi.cc/2igjGld

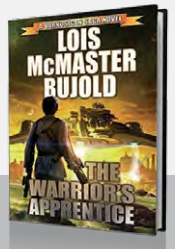
Noir reborn in a dystopian internet-saturated future. More than 30 years old, yet both familiar and futuristic.

Do Androids Dream Of Electric Sheep?

Author: Philip K Dick
Publisher: Gollancz
Price: £8.99
ISBN: 978-0575079939
magpi.cc/2igc3vb

Filmed for its dark thriller elements as *Blade Runner*, the original is more reflective, and more disturbing in its questions.

The Warrior's Apprentice



Author: Lois McMaster Bujold
Publisher: Baen
Price: £10.99
ISBN: 978-1476781303
magpi.cc/2igff9S

Warm-hearted, humorous; engineering, bio-tech, and classic war strategy; this ongoing series sets the standard for modern sci-fi.

APL: EDUCATING WITH OLD CODE

How a programming language last popular in the Eighties is making a comeback

The oddly named A Programming Language (APL) isn't exactly new. Invented in the 1960s, this language is somewhat different to Python and other classic languages like C. While it declined in popularity from the mid-1980s on, in the last few years it has been making a comeback, including a recent port to the Raspberry Pi.

"It's a slightly curious language because it's called A Programming Language, but it was created by a mathematician," Morten Kromberg tells us. Morten is the chief experience director at Dyalog, a company that specialises in APL-based solutions, and the force behind APL's resurgence.

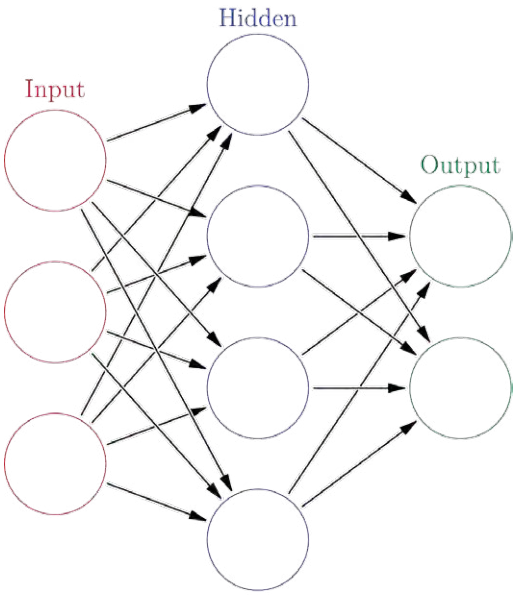
"APL in [its creator's] mind was trying to be mathematical... most of the people using computers in the early years had some other background – software engineering hadn't been invented at the time. All software in that period was hand-made, so APL was a more comfortable fit for many of these people than the languages being invented. It was very widely used on mainframe computers and then there was what we think of as the dark age where the focus of software engineering became very structured, with a lot of effort being spent on GUI frameworks and stuff like that, which was really quite daunting and off-putting for many of the people who were trying to use it to solve problems."

Now that platforms are becoming more friendly, he feels like APL is more relevant than it has ever been.

Educational APL

A few years ago, APL was ported to Raspbian, following on from ports to Windows, Mac, and other Linux systems. Dyalog thinks APL has a distinct advantage in many areas over other code.

"As far as I'm concerned I think the education applications where APL on the Pi would work

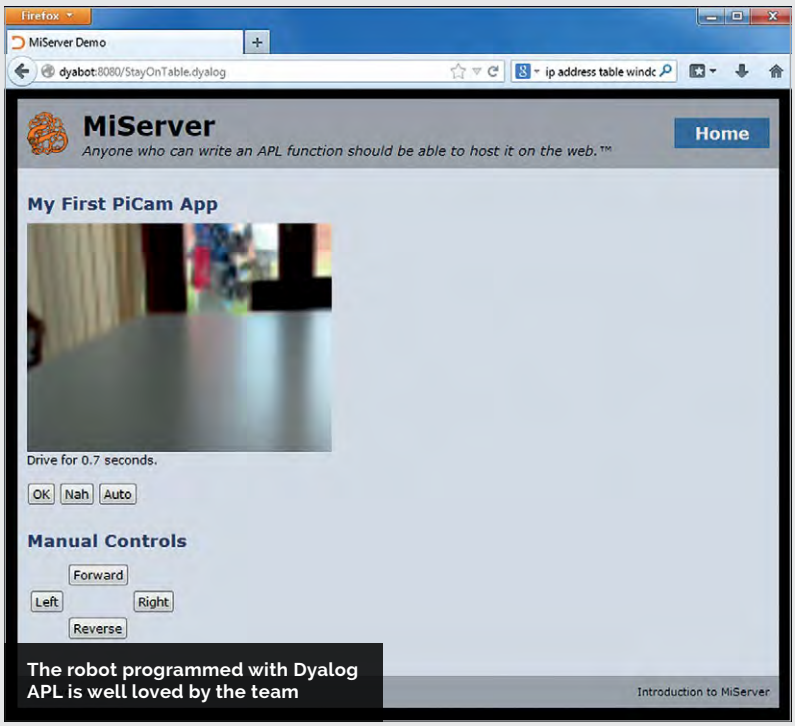


PI BRAIN

The concept of neural networking is using the brain as a model for processing data, which is something Romilly Cocking has been experimenting on for years with APL.

"Back in 1984 as part of my LSC, I did a simulation which burnt up pretty much up all the department's usage of computing power in a week trying to simulate a terribly cut-down model of part of the brain called the cerebellum," says Romilly. "These days, something that's about a million times larger, in terms of number of nerve cells, runs happily on a Pi. It's allowed me to carry on the research I started in 1984."

Above Plenty of people show up to APL conferences, proving its lasting popularity



The robot programmed with Dyalog APL is well loved by the team

really well are a sweet spot where three things combine," Romilly Cocking, an APL veteran, reveals. "Firstly, if you're tackling a problem [that] you don't initially know how to solve, you are exploring and testing. Secondly, that problem involves fairly complex algorithms. Thirdly, performance matters. Probably because you've got a lot of data, or possibly just because you're trying to build a robot and the robot needs to react very quickly. In any problem that satisfies those three things, APL would be both a good thing and a fun thing to use."

> "It's called A Programming Language, but it was created by a mathematician"

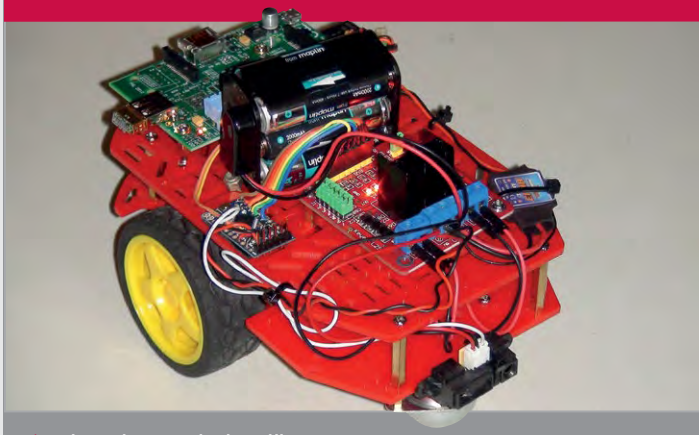
Though UK schools may not have taken APL to heart yet, it has been successful in education elsewhere, as Vibeke Ulmann, PR representative for Dyalog, explains. "For the past two years, although they have not been using the Raspberry Pi, an Italian class of students aged about 14 and 15 have been using APL for mathematics teaching. They've done some absolutely stupendous presentations about how they understood the language and how they came to understand [it]... Using the Pi for mathematical work in a school environment would, I think, be quite fun."

You can watch a video about it online here: **magpi.cc/2hzwmCe**.

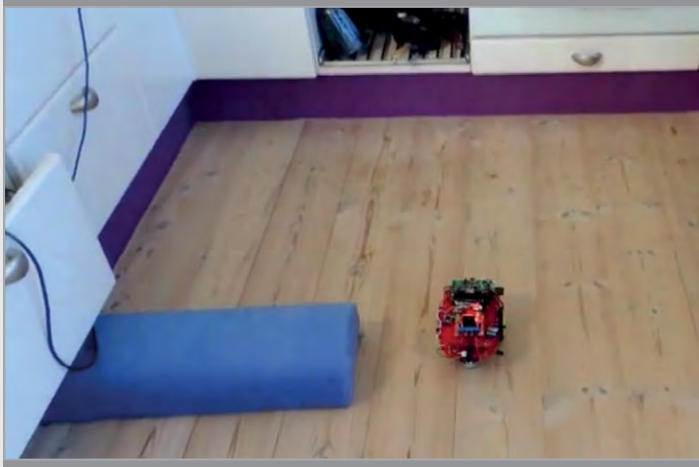
Work is still being done on APL on the Raspberry Pi, with future plans to incorporate OpenCV and the Camera Module. It's also very good for parallel computing, with other plans to implement clusters with APL for some excellent (and cheap) number crunching on the Raspberry Pi. Dyalog is hoping all of this will create a new generation of APL users ready to improve and work with the venerable language.

ROBOTS!

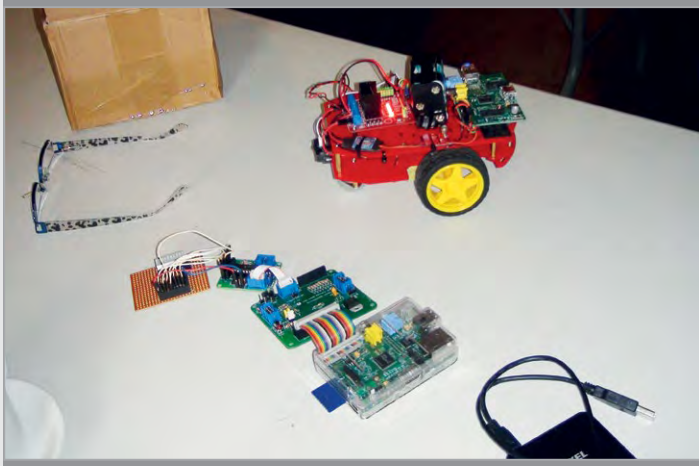
ONE OF THE PROJECTS MORTEN AND ROMILLY HAVE WORKED ON WITH APL IS AN APL-PROGRAMMED, PI-POWERED ROBOT



> The robot works just like any other Pi robot programmed in Python



> It can navigate a room using sensors to make sure it doesn't hit any obstacles



> The robot also has the ability to check if it's near a ledge. We've lost many a robot off the edge of a table, so we welcome this functionality

THE MONTH IN RASPBERRY PI

Everything else that happened this month in the world of Raspberry Pi

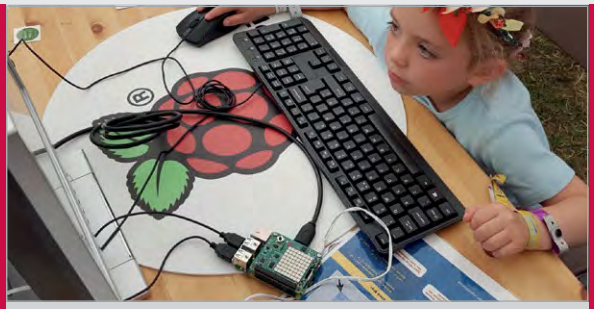
GEARING UP FOR 2017

This year the Raspberry Pi will be five years old. It's been a great few years for the little computer, but the Raspberry Pi Foundation team aren't resting on their laurels yet.

Pi Foundation CEO Philip Colligan detailed the plans for Raspberry Pi in 2017 in a blog on the website (you can read it here: **magpi.cc/2ijU9rn**). In it he talked about all the amazing things that happened just in 2016:

"By any measure, the Raspberry Pi Foundation had a fantastic 2016. We ended the year with over 11 million Raspberry Pi computers sold, millions of people using our learning resources, almost 1,000 Certified Educators in the UK and US, 75,000 children regularly attending over 5,000 Code Clubs in the UK, hundreds of Raspberry Jams taking place all over the world, code written by schoolkids running in space (yes, space), and much, much more."

Here are some of the exciting things Philip talked about that are in store for Raspberry Pi in 2017:



YOUNG DIGITAL MAKERS

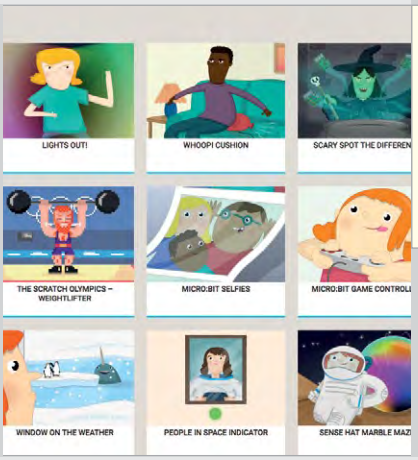
Getting young people into computing has always been a big goal of the Raspberry Pi Foundation, and more is coming in 2017 to aid this. Code Clubs are opening up around the world and a brand new competition called Pioneers has launched to get teens excited about making.

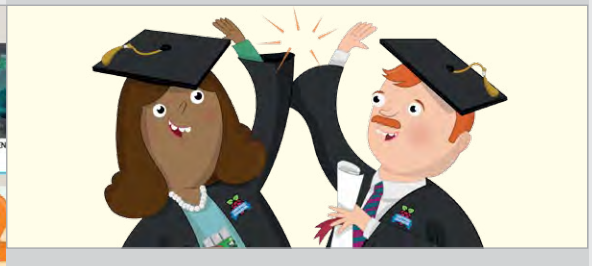
COMMUNITY

The community is extremely important to Raspberry Pi – it's one of the selling points of the hardware, to be quite honest. The Foundation will be increasing its outreach in the UK, and setting itself up as a charity in the US to help do more there.

FREE EDUCATIONAL RESOURCES

There are already loads of free online tutorials and lesson plans for learning to code with the Raspberry Pi, and many more will be coming in 2017. Raspberry Pi will also release a digital making curriculum (possibly by the time you read this) to help aid educators structure learning. There's also a lot of video in the works...



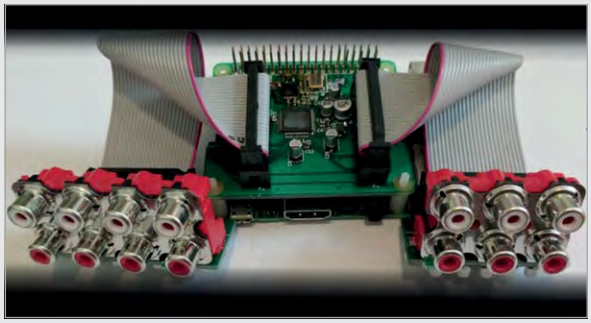


EDUCATORS

As well as continuing to certify educators through Picademy courses, a brand new series of free online courses is available, with more to come. Check out our news section for more details.

CROWDFUND THIS!

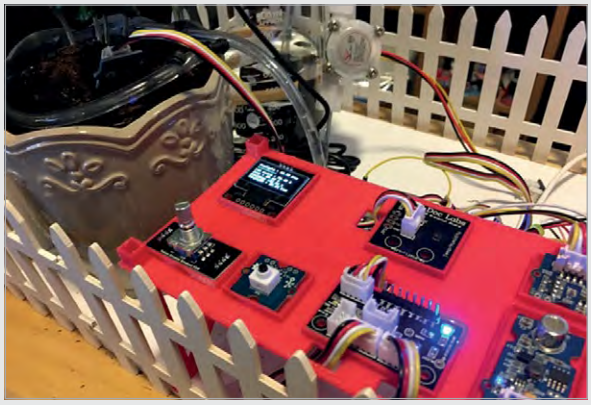
The best crowdfunding hits this month for you to check out...



AUDIO INJECTOR OCTO SURROUND SOUND

kck.st/2i5frs0

There are a few audio cards for the Raspberry Pi, but we've never seen anything quite like this. The Audio Injector team have put together a surround sound HAT for the Pi with both inputs and outputs using RCA so that you can add surround sound to your projects. We'd like to see it attached to a Pi running some form of Kodi, but we assume it will be more practical for interesting audio projects in the short term.



SMARTPLANTPI

kck.st/2idtpsG

You may have heard of Farm Pi, the farming robot for communities, but the SmartPlantPi is a much more scaled-down affair that keeps an eye on your house plant. We love that the pictures on the Kickstarter have a little picket fence around the plants. Anyway, it waters your plant and monitors several other environmental factors. No need to get your neighbours to water your plants when you're away, then!

BEST OF THE REST

Here are some other great things we saw this month

magpi.cc/2j4RUFx

PI-POWERED COFFEE

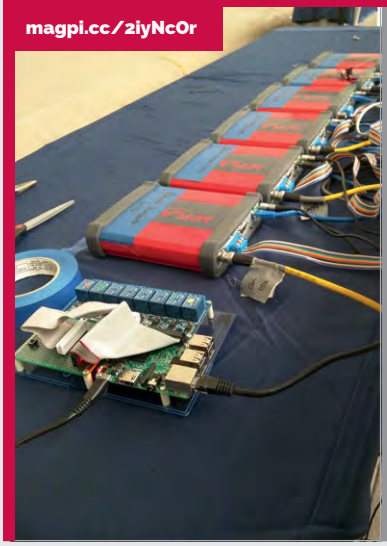
"Wrote a cool little application for my @wilburcurtisco coffee machine at work," matty.mcc says on Instagram. It gives info about when the last cup was brewed and emails folks when their coffee is ready. Raspberry Pi Towers recently got a new coffee machine, so it's only a matter of time before the same thing happens here...

magpi.cc/2j51Yyk

BMO

We all want a Korean gaming pal like BMO, so NinjaBunny9000 created a 1:1 model of *Adventure Time*'s premier games console and powered it up with a Raspberry Pi. It even has a headphone out jack. It's all 3D printed, but there's a lot of custom electronics work going on inside it as well.

magpi.cc/2iyNcOr



OLYMPIC TRAINING CENTER

Apparently, the Olympic Training Center Velodrome in Colorado Springs is using this Raspberry Pi setup to test athletes. We remember doing all those little Olympics projects a few months ago, so it's nice to know the Pi can be used for real-life sports.

FEMI & GRACE OWOLADE-COOMBES

Hacker Femi and his Certified Educator mum Grace rock the South London Jam scene

Profile info

Category: Hacker and educator

Day job: Femi is still in school while Grace works in further education

Website: hackerfemo.com
twitter.com/gowolade

Grace Owolade-Coombes heard about coding through her work in further education and found herself intrigued. Mother to a 'young, inquisitive son', she felt the urge to investigate further. However, it wasn't until a year later, while attending the National STEM Centre in York, that she finally discovered more about coding and what the term actually meant.

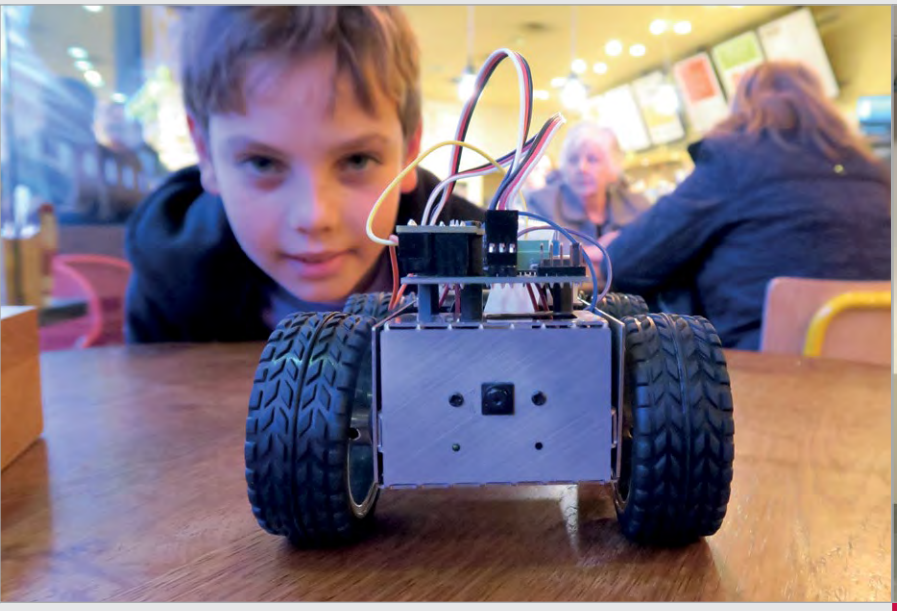
She was sent in the direction of the Raspberry Pi Foundation and attended a few events with her son, Femi, where they discovered the warmth of the community and the fun to be had with projects such as Minecraft hacks and robot builds. From there, Grace found herself on the Picademy programme and, after the two-day course, she felt excited and eager to get more involved in the community. "I particularly loved the physical coding workshop which I took back and delivered to my son, who also loved it."

With no prior background in computing, Grace, along with Femi, was able to take her Picademy experience and run a workshop at Covent Garden's Dragon Hall – the first of many workshops the pair would lead.

At age nine, on 11 September 2015, Femi wrote on his website:

Below right A visit to the BBC was on the cards, thanks to the Make it Digital campaign

Below Femi continues to run workshops for others, improving on his worksheets for events such as MozFest, PyCon, and Raspberry Jams



In June, along with fellow Raspberry Pi community member Jonathan Tyler-Moore, Femi won the NanoSat Hack at Microsoft AzureCraft

"This is my very first blog and I'm getting ready to do a talk at CamJam in the morning." He later went on to describe his experience at Festival of Code, an event he attended the summer after their first workshop at Dragon Hall. Femi is very open about his experiences with Tourette's syndrome, a condition he was diagnosed with around the time of his introduction to the Raspberry Pi community. While attending a Tourettes Action support group, he wanted to share his new passions with fellow group members. It was this desire to create a safe, comfortable place for his friends to learn that brought about the South London Raspberry Jam movement. "He asked if we could set up a jam that was inclusive – both autism and Tourette's syndrome friendly. There was such a wealth of support, advice, and volunteers who would help us set up… it really wasn't a hard decision to make." The pair ran an Indiegogo campaign to raise money, hitting 241% of their target, and the event ran in October 2015, welcoming ages 5 to 15. It was a great success, leading to a code club and a further Jam. By their third Jam, Femi and Grace were working in conjunction with various makers and producers, introducing attendees to Crumble inventor Joseph Birkes and Paul Hayes from Unity, all from within the Kano HQ.

Above Femi is forever building on his skill set; from robots to radios, Minecraft, and Python, he loves discovering new ideas

> "It was a great success, leading to a Code Club and a further Raspberry Jam"

Femi continues to document his journey through his site, **hackerfemo.com**, relating his experiences as both a student and teacher within the community. Heading into 2017, the South London Raspberry Jam aims to expand into South London Makerspace. Meanwhile, Femi has been invited join the University of the Arts London Digital Makers Collective and asked to lead robot workshops at the Tate Modern, while also looking forward to taking part in both the European Astro Pi Challenge and Pi Wars.

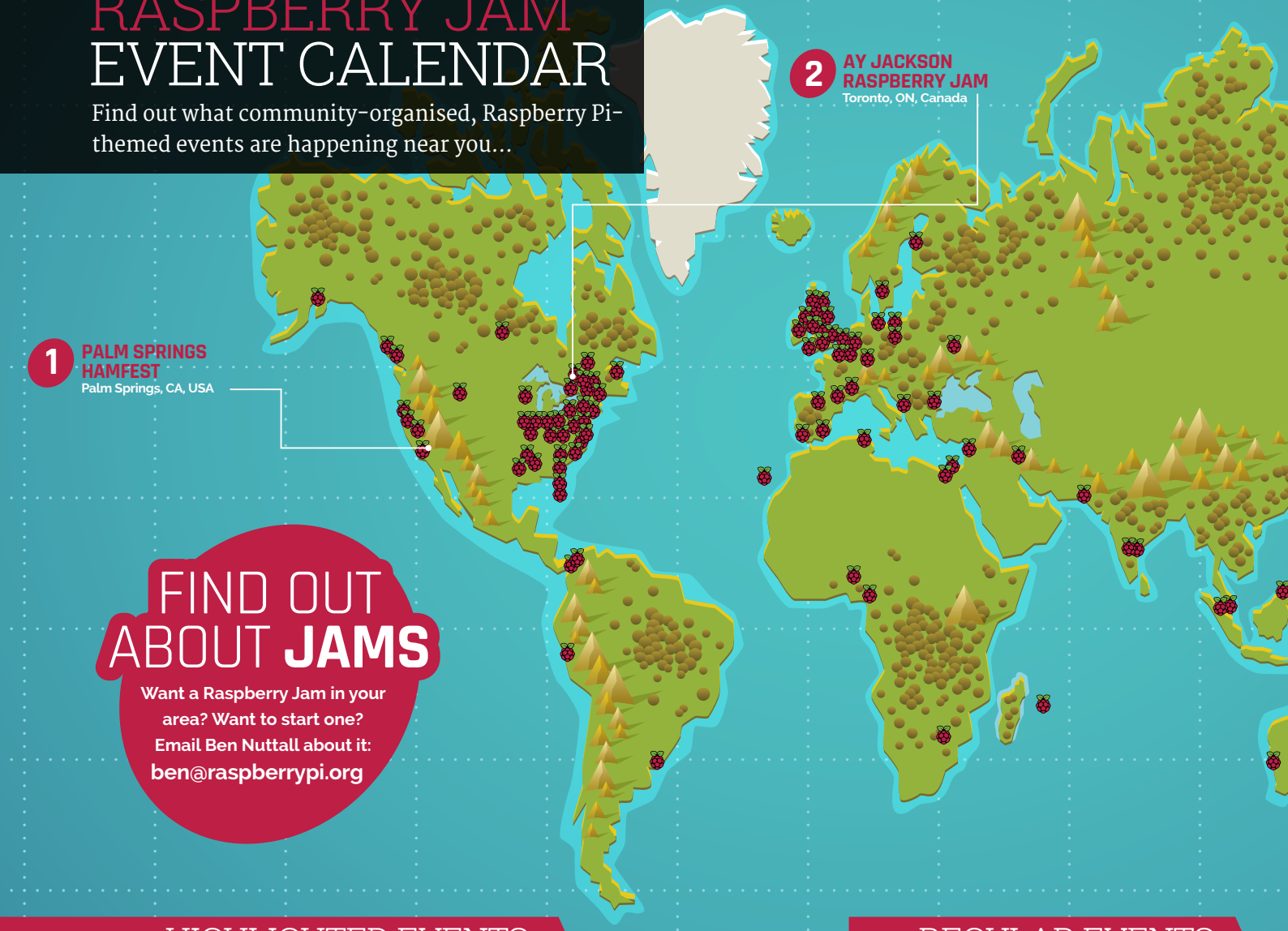
HIGHLIGHT

BT YOUNG PIONEERS AWARD

Femi found himself shortlisted for the Tech4Good BT Young Pioneer Award for his work in making coding accessible to young people such as himself and members of his support group. His Indiegogo campaign had been so successful that the industry couldn't help but notice. "Femi Owolade-Coombes is making coding accessible to young people with his infectious enthusiasm and determination for inclusive Jams," read the info on **tech4goodawards.com**. The South London Jam also gained recognition, earning a place as finalist for the Tech4Good Digital Skills award in the same year.

RASPBERRY JAM EVENT CALENDAR

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



FIND OUT ABOUT JAMS

Want a Raspberry Jam in your area? Want to start one? Email Ben Nuttall about it:
ben@raspberrypi.org

HIGHLIGHTED EVENTS

1 PALM SPRINGS HAMFEST

When: Saturday 4 February
Where: Palm Springs Air Museum, Palm Springs, CA, USA
palmspringshamfest.com
A demo of the Astro Pi will be held at Hamfest, one of the largest gathering of ham radio enthusiasts in California.

2 AY JACKSON SCHOOL RASPBERRY JAM

When: Sunday 11 February
Where: AY Jackson SS Library, Toronto, ON, Canada
magpi.cc/2hE6ghB
An event for you to share and learn about the Raspberry Pi and computer engineering concepts.

3 RASPBERRY JAM: BITESIZE

When: Saturday 18 February
Where: Leamington Library Royal, Royal Leamington Spa, UK
magpi.cc/2hPVqml
A stripped-back Jam where you can get more help than you would at a standard Jam.

4 RASPBERRY PI BIG BIRTHDAY WEEKEND

When: Saturday 4 March
Where: Cambridge Junction, Cambridge, UK
magpi.cc/2hEjDyh
Robots, gaming, animations, and much more at Raspberry Pi and Code Club's 5th birthday!

REGULAR EVENTS

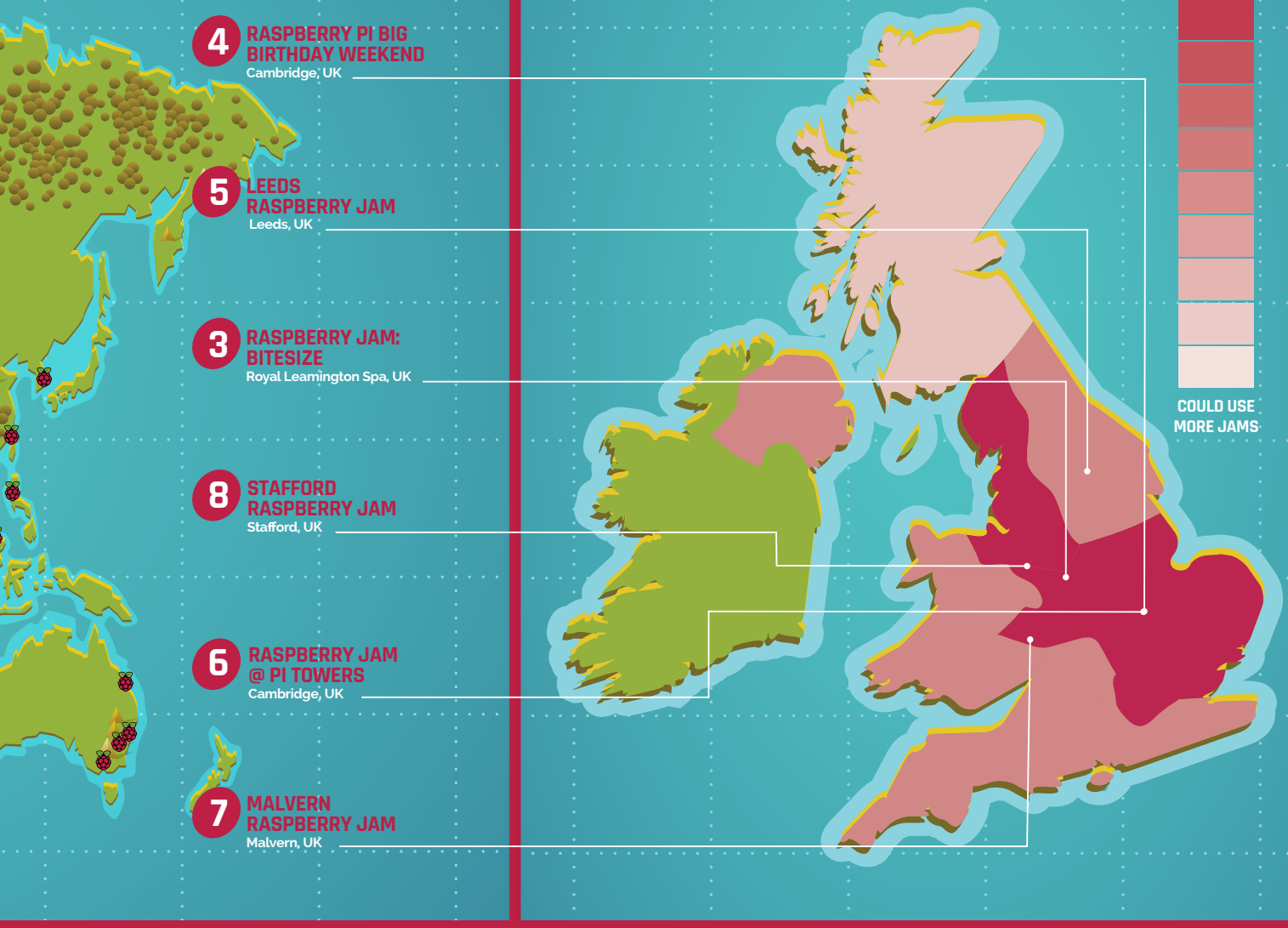
5 LEEDS RASPBERRY JAM

When: Wednesday 1 February
Where: Swallow Hill Community College, Leeds, UK
magpi.cc/2hDzJYS
There'll be chances to get hands-on with more digital making activities through workshops and a hackspace.

6 RASPBERRY JAM @ PI TOWERS

When: Saturday 11 February
Where: 30 Station Road, Cambridge, UK
magpi.cc/2hDILoz
Come and visit the Raspberry Pi Foundation's headquarters and take part in a Jam like no other.

JAM HEAT MAP



7 MALVERN RASPBERRY JAM

When: Wednesday 15 February
Where: Wyche Innovation Centre, Malvern, UK
magpi.cc/2hEb8D8
An opportunity to come along with your own Raspberry Pi to show people what you have been up to.

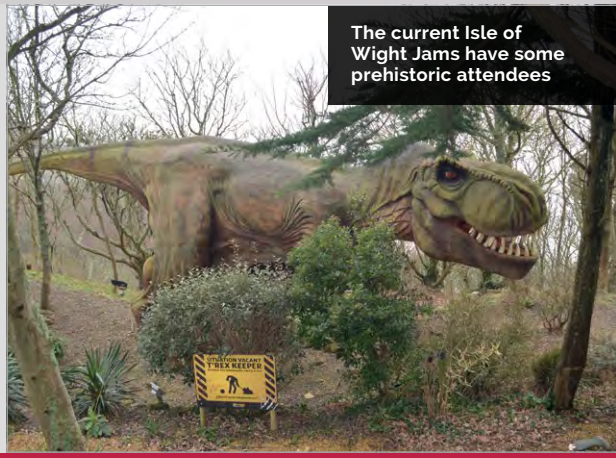
8 STAFFORD RASPBERRY JAM

When: Tuesday 14 February
Where: The Signpost Centre, Stafford, UK
magpi.cc/2hEjiva
A big meetup of Pi enthusiasts who love to share ideas, help each other, and have a lot of fun!

FILL IN THE GAPS!

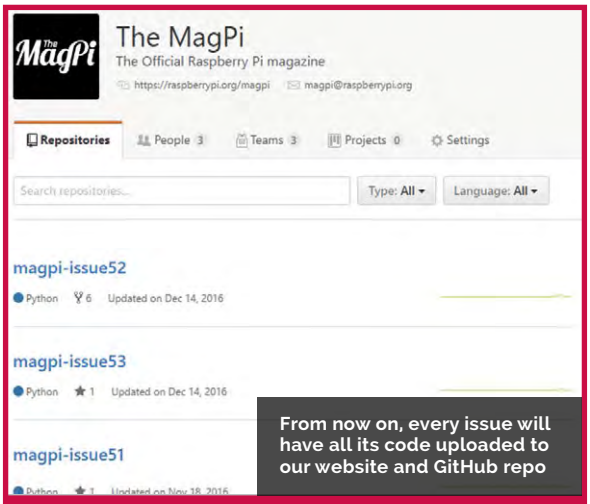
THE ISLE OF WIGHT NEEDS JAMS

The Isle of Wight is a lovely place and while it's small, plenty of people live there. It's also not always convenient to get to the mainland, and while Lucy Rogers does run the occassional Pi Wight event, more could go on for young folk on the island. If you fancy starting your own Jam on the Isle of Wight, you can get more info from the Raspberry Pi site: **raspberrypi.org/jam**.



The current Isle of Wight Jams have some prehistoric attendees

YOUR LETTERS



Code concerns

I've been working through your Beginner's Guide to Coding feature and really enjoying it. However, I've unfortunately hit a snag. When I go to run the code, it doesn't seem to be working and I get an error about the syntax of a line. Is there any way you can help?
Tim

When it comes to a syntax error, our best advice is to see which line it has an issue with, and work the code back until you find a problem. This can be as simple as forgetting a colon somewhere, or not having the right tab spacing on a line.

This is the kind of difficulty people who code face even when they've been doing it for years – after practice, you become better at spotting the errors, though!

If all else fails and you can't find the problem, all the code is now available for each issue on our GitHub repository, which you can find here: **magpi.cc/2ikX9DO**

Christmas cover

Thanks for the wonderful Christmas cover last issue [#52]! I really loved the idea of the official Raspberry Pi magazine being officially interactive with the Raspberry Pi in some way. I bought a second copy and wired it all up and it sits proudly among some of my other decorations. I did still put up a tree next to it; I hope you don't object. Maybe I'll upgrade it with the NeoPixel lights in the future.

Will you be doing more of these styles of covers in the future? Maybe an Easter one with an egg hunting game, or something with the next Pi release! Anyway, I'm going to preserve the magazine with the rest of my decorations and use it next year as well. Thanks a lot!
Phil Brown

We're glad you enjoyed it; it was a pleasure to put it together! We hope more people made the cover into their latest decoration as well (email or tweet us if you did!). You can also be sure that we'll definitely try to do something like it again in the future, although we don't want to spoil any plans we have.

You should try out the NeoPixel lights if you have a chance next year – we might even add a few more functions into the code in our GitHub repo (**magpi.cc/2juNw4J**), so take a look if you do decide to make it.

Right We hope to do more themed covers in 2017 – watch this space!

Resolve to change

I really enjoyed the New Year's resolutions article you did last year. In fact, I tried out a couple of them myself. I managed to lose a bit of weight! Hopefully I don't put it all back on over Christmas. I was wondering if you were planning another New Year's article for this year as well?
Laura B

We're definitely not averse to seasonal themes, as you can probably tell by our Christmas issue. Unfortunately, though, we don't have a New Year's article this year. Instead, maybe you can treat the programming guide from issue 53 as your New Year's resolution and start to learn to code?

We'll keep it in mind for next year, though, and maybe we'll do another New Year's article in the mag!

FROM THE FORUM: PRINT IS NOT DEAD

The Raspberry Pi Forum is a hotbed of conversations and problem-solving for the community – join in via **raspberrypi.org/forums**

I would love to have paper copies of *The MagPi*, but my local retailers want close to $20+ for copies. Has there been any talk of a printing company in the USA printing copies and stocking them locally for a cheaper price?
kryptalivian

We've been looking into it as an option to get the magazine out to the US faster and more cheaply, but that's probably not going to happen in the short term. Long story short, it's very complicated, which is not hard to imagine.

For the short term, though, what you can always do is actually print out the magazine. Remember, each issue is licensed under Creative Commons so you're allowed to go to a good printers and get it printed out for yourself.

Otherwise, the subscription does work out a lot cheaper than both these options, and you get a free Pi Zero as well. Check out **magpi.cc/Subs1** for more info.

WRITE TO US

Have you got something you'd like to say?
Get in touch via **magpi@raspberrypi.org** or on The MagPi section of the forum at: **raspberrypi.org/forums**

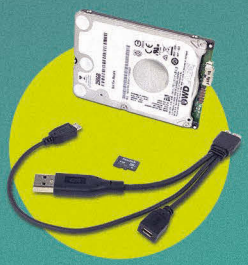
LAUNCH YOUR NEW PROJECT

with WD PiDrive Foundation Edition

Create Raspberry Pi projects using the WD PiDrive Foundation Edition device with our custom NOOBs for a simpler onboarding experience, and Project Spaces for quick and easy access to your projects from one Raspberry Pi setup.

Choose the version that's right for you:

WD PIDRIVE FOUNDATION EDITION
USB FLASH 64GB
Includes MicroSD™ Card
$18.99



WD PIDRIVE FOUNDATION EDITION
USB HARD DRIVE
250GB / 375GB
Includes MicroSD™ Card and WD PiDrive Cable
$28.99 / $37.49

WDLABS
wdlabs.wd.com/magpi54b

Western Digital, WD, the WD logo, WDLabs and the WDLabs logo are registered trademarks or trademarks of Western Digital Corporation or its affiliates in the U.S. and/or other countries. Raspberry Pi is a trademark of the Raspberry Pi Foundation, microSD is a trademark of SD-3C, LLC. All other marks are the property of their respective owners. As used for storage capacity, one gigabyte = one billion bytes. Total accessible capacity varies depending on operating environment. Product specifications subject to change without notice. Pictures shown may vary from actual product. © 2017 Western Digital Corporation or its affiliates.

WIN

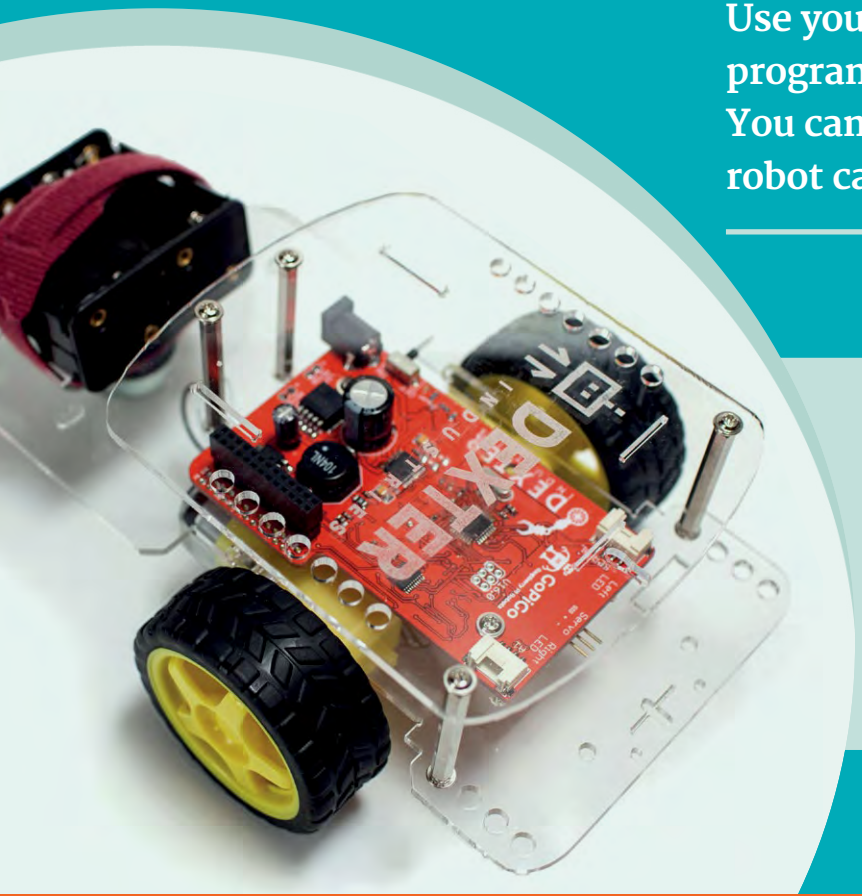
In association with

DEXTER INDUSTRIES

dexterindustries.com

1 OF 3 SETS OF A GOPIGO ROBOT BASE KIT AND A PIVOTPI

Use your Raspberry Pi and the PivotPi to program anything in your world to move! You can also build and program your own robot car with the GoPiGo and Raspberry Pi.



The BrickPi lets the Pi talk to **what LEGO product?**

Tell us by 27 February for your chance to win!

Simply email **competition@raspberrypi.org** with your name, address, and answer!

Terms & Conditions

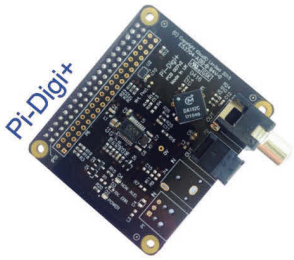
Competition closes on 27 February 2017. Prize is offered worldwide to participants aged 18 or over, except employees of the Raspberry Pi Foundation, the prize supplier, their families or friends. Winners will be notified by email after the draw date. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine (unless otherwise stated upon entry). 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.

Love your Pi? Love Music?

Designed by audio experts, enjoyed by everyone

Made in the UK

10% Discount voucher "MagPi10"



Jaw dropping audio quality for your Raspberry Pi

Connect Headphones, Speakers, RCA, Toslink, S/PDIF or XLR

We work with all the major music solutions to ensure compatibility

Sonic Pi

MAX2PLAY

moOde audio player | OSMC | piCorePlayer SQUEEZE AUDIO | VOLUMIO | roon | Archphile | PiMusicBox | KODI | rune audio

Twitter: @IQ_audio
Email: info@iqaudio.com
Web: www.iqaudio.com

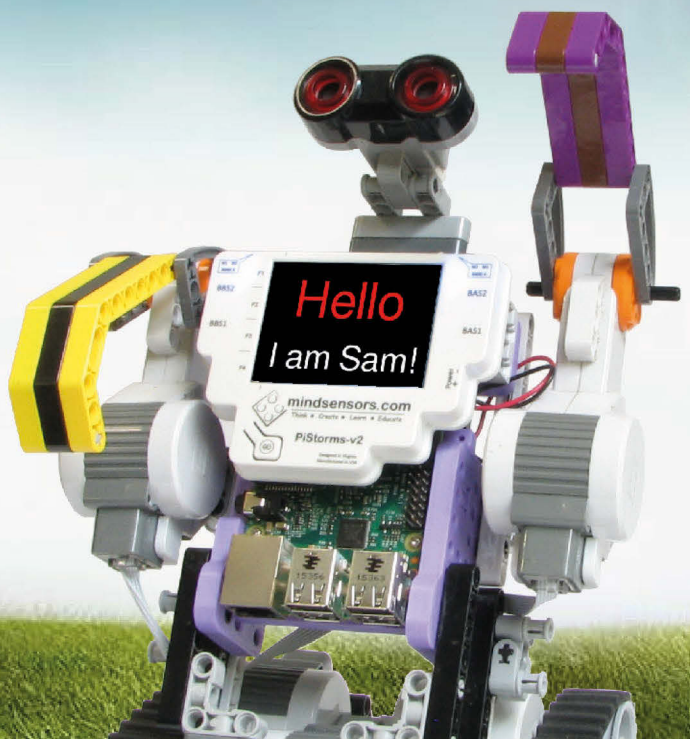
IQaudIO

IQaudio Limited,
Cricklade Wiltshire.
Company No.: 9461908

mindsensors.com

Think • Create • Learn • Educate

!! Special !!
15% discount:
Use Code:
MAGPISPECIAL



PiStorms-v2

Make Stunning Robots with LEGOs and Raspberry Pi !

MATT RICHARDSON

Matt is Raspberry Pi's US-based product evangelist. Before that, he was co-author of *Getting Started with Raspberry Pi* and a contributing editor at *Make:* magazine.

LIFELONG LEARNING

Matt Richardson on how education with computers should continue well into adulthood

When you contemplate the Raspberry Pi Foundation's educational mission, you might first think of young people learning how to code, how computers work, and how to make things with computers. You might also think of teachers leveraging our free resources and training in order to bring digital making to their students in the classroom. Getting young people excited about computing and digital making is an enormous part of what we're all about.

We all know that learning doesn't only happen in the classroom – it also happens in the home, and at libraries, code clubs, museums, scout troop meetings, and after-school enrichment centres. At the Raspberry Pi Foundation, we acknowledge that and try hard to get young people learning about computer science and digital making in all of these contexts. It's the reason why many of our Raspberry Pi Certified Educators aren't necessarily classroom teachers, but also educate in other environments.

Even though inspiring and educating young people in and out of the classroom is a huge part of what we set out to do, our mission doesn't limit us to only those that are young. Learning can happen at any age and of course we love to see kids and adults using Raspberry Pi computers and our learning resources. Although our priority is educating young people, we know that we have a strong community of adults who make, learn, and experiment with Raspberry Pi.

I consider myself among this community of lifelong learners. Ever since I first tried Raspberry Pi in 2012, I've learned so much with this affordable computer by making things with it. I may not have set out to learn more about programming and algorithms, but learned them as a by-product of trying to create an interesting project that required them. This goes beyond computing, too. For instance, I needed to give myself a quick maths refresher when working on my Dynamic Bike Headlight project. I had to get the speed of my bike in miles per hour knowing the radius of the wheel and the revolutions per minute from a sensor. I suspect that – like me – a lot of adults out there using Raspberry Pi for their home and work projects are learning a lot along the way.

Internet of Tutorials

Even if you're following a tutorial to build a retro arcade machine, set up a home server, or create a magic mirror, then you're learning. There are tons of great tutorials out there that don't just tell you what to type in but also explain what you're doing and why you're doing it at each step along the way. Hopefully it also leaves room for a maker to experiment and learn.

Many people also learn with Raspberry Pi when they use it as a platform for experimental computing. This experimentation can come from personal curiosity or from a professional need.

They may want to set up a sandbox to test out things such as networking, servers, cluster computing, or containers. Raspberry Pi makes a good platform for this because of its affordability and its universality. In other words, Raspberry Pis have become so common in the world that there's usually someone out there who has at least attempted to figure out how to do what you're looking for with it.

To take it back to the young people, it's critical to show them that we, as adults, aren't always teachers. Sometimes we're learning right beside them. Sometimes we're even learning from them. Instil in them the idea that learning doesn't stop after they graduate. We must show young people that none of us stops learning.

The MagPi ESSENTIALS

LEARN | CODE | MAKE

OUT NOW IN PRINT

ONLY **£4/$6**

from

raspberrypi.org/magpi

The MagPi ESSENTIALS From the makers of the official Raspberry Pi magazine

GET THEM DIGITALLY:

Available on the App Store

GET IT ON Google Play

DEXTER INDUSTRIES

SAVE 15%
coupon code
magpi

GrovePi
Plug-and-play Sensor Kit for Raspberry Pi
$89.99 USD-$179.99 USD

GoPiGo
A Robot Car with the Raspberry Pi
$99.99 USD-$199.99 USD

BrickPi
Raspberry Pi + LEGO MINDSTORMS
$99.99 USD-$199.99 USD

www.dexterindustries.com

Expand your Pi

Stackable expansion boards for the Raspberry Pi

Serial Pi Plus

RS232 serial communication board. Control your Raspberry Pi over RS232 or connect to external serial accessories.

Breakout Pi Plus

The Breakout Pi Plus is a useful and versatile prototyping expansion board for the Raspberry Pi

ADC Differential Pi

8 channel 18 bit analogue to digital converter. I²C address selection allows you to add up to 32 analogue inputs to your Raspberry Pi.

IO Pi Plus

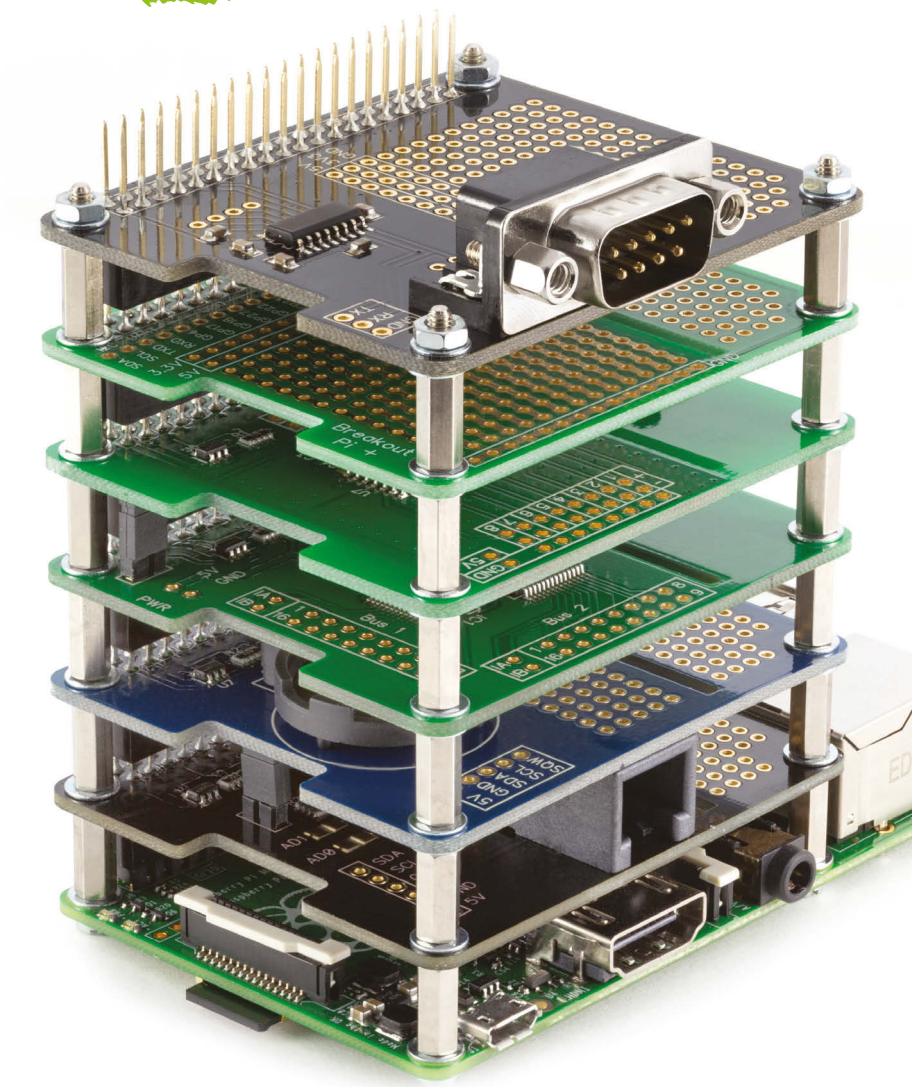
32 digital 5V inputs or outputs. I²C address selection allows you to stack up to 4 IO Pi Plus boards on your Raspberry Pi giving you 128 digital inputs or outputs.

RTC Pi Plus

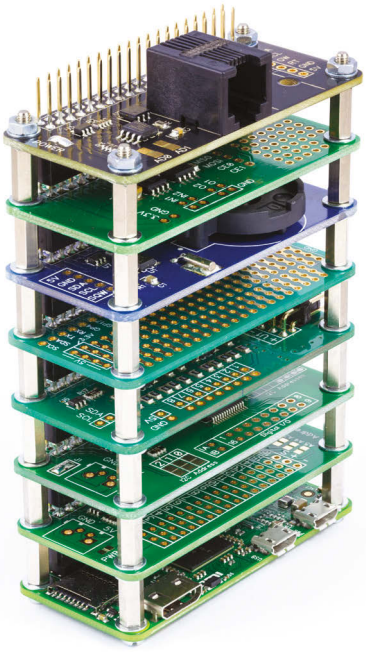
Real-time clock with battery backup and 5V I²C level converter for adding external 5V I²C devices to your Raspberry Pi.

1 Wire Pi Plus

1-Wire® to I²C host interface with ESD protection diode and I²C address selection.



Also available for the **Pi Zero**

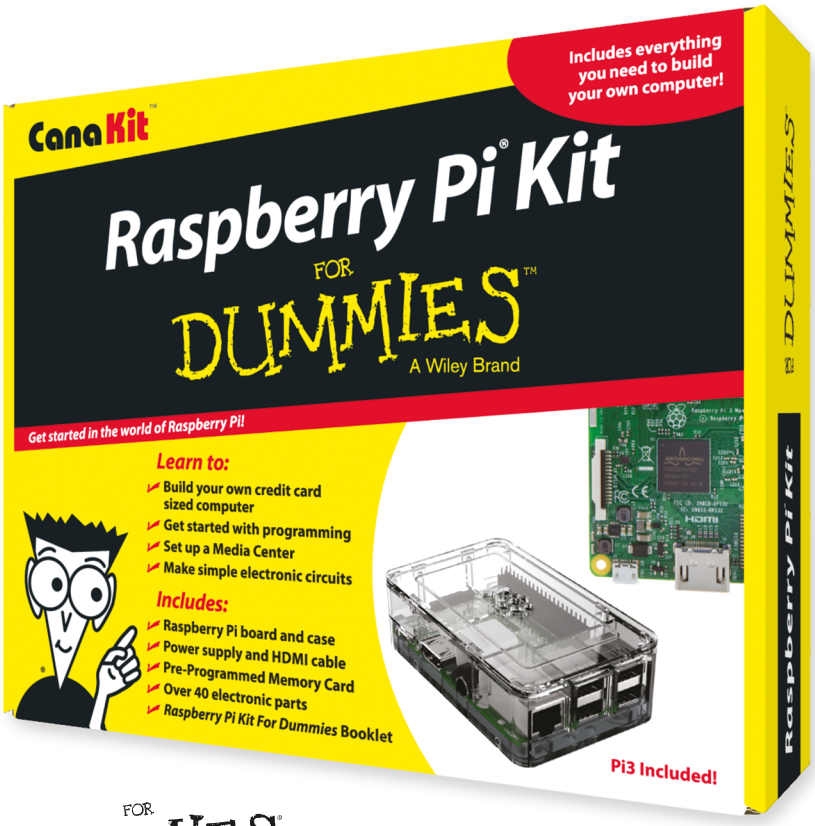


ABelectronics UK

www.abelectronics.co.uk

CanaKit™

The Perfect Holiday Gift!



FOR DUMMIES® A Wiley Brand

Available for worldwide shipping at:

WWW.CANAKIT.COM

Available in Europe through RS Components

Kit Includes:

- *Raspberry Pi For Dummies* Booklet
- Raspberry Pi 3 Board
- Memory Card
- Plastic Case
- 2.5A Power Supply
- HDMI Cable
- Resistors
- LEDs
- Push Button Switches
- Prototyping Breadboard
- Jumper Wires
- Heat Sinks

$89.99 US DOLLARS

£69.99 EXCLUDING VAT

Raspberry Pi is a registered trademark of the Raspberry Pi Foundation. For Dummies and the Dummies Man logo are trademarks or registered trademarks of John Wiley & Sons, Inc. Used under license. RS logo is a registered trademark of RS Components Ltd. CanaKit is a registered trademark of Cana Kit Corporation.