70 PAGES OF RASPBERRY PI TIPS, TRICKS AND TECH

The MagPi

The official Raspberry Pi magazine

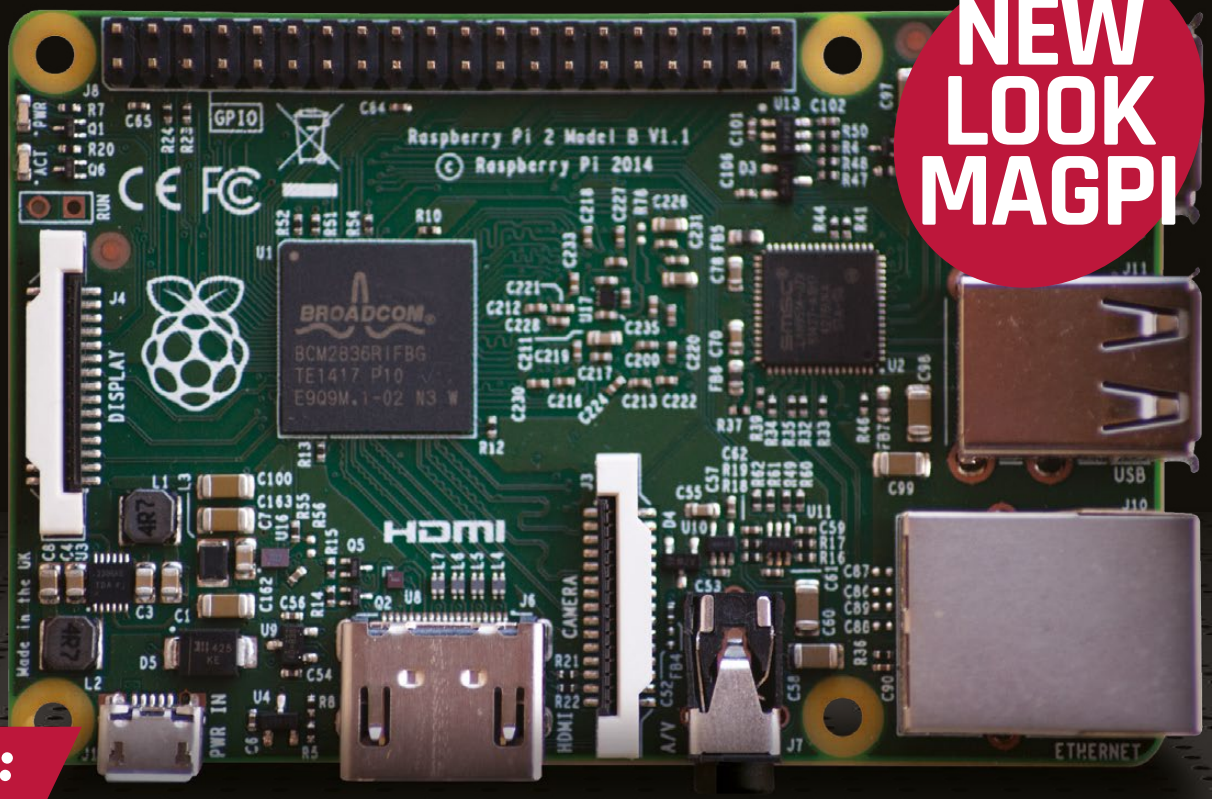
Issue 31 Mar 2015

raspberrypi.org/magpi

Win! 5 RASPI 2s + a case of your choice from Pimoroni

RASPBERRY PI 2

The $35 PC reinvented. You really can have your cake & eat it



NEW LOOK MAGPI

YOUNG HACKERS &MAKERS

Meet tomorrow's technologists today

MAKING GAMES IN EASY STEPS

Grasp the basics with part one of an epic new game-making series

Also inside:

- GET CREATIVE WITH MINECRAFT: PI
- EVERYDAY ENGINEERING WITH PI
- ANONYMEBOX MAKES TOR EASY
- SHOOT SLOW-MOTION VIDEOS
- RETRO GAMING GROUP-TEST

CODE IN SPACE!

The Astro Pi countdown begins, but will your project make it to the ISS?

Plus LEARN TO LOVE THE COMMAND LINE

WELCOME TO THE MAGPI

It's not just the Raspberry Pi that's had a facelift for 2015. Welcome to the new MagPi, now the official Raspberry Pi magazine. Before we go any further, massive thanks and congratulations need to go to the original MagPi team, and all the volunteers. You'll be seeing many of them again, once they've had a well-earned break. Running a monthly magazine in a full-time capacity is challenging enough, let alone as a group of enthusiasts taking on the task during evenings and weekends. The success of The MagPi over the last few years is testament to the amazing Raspberry Pi community and a clear indication of what it can achieve. What else can it do? That's what The MagPi is here to share!

The magazine might look a bit different, but it's still made for and by the Raspberry Pi community and we'll be sharing your amazing projects, tips and tricks every single issue. The MagPi is still very much committed to open source, too. You can download it free online and it still operates under the same Creative Commons licence. To download the magazine (in PDF format) and find out more about the licence, visit **raspberrypi.org/magpi**.

If you've made something with the Raspberry Pi, want to share your tips and tricks, need help with a technical issue, or simply want to let us know what you think of the new magazine, please get in touch via **magpi@raspberrypi.org**. I'm really looking forward to hearing from you and sharing what you're doing with the world's favourite credit card-sized PC.

Russell Barnes

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
Technical Editor: **David Whale**
Sub Editors: **Phil King, Laura Clay**

DESIGN
Critical Media: **criticalmedia.co.uk**
+44 (0)1202 399824
Head of Design: **Dougal Matthews**
Designers: **Mike Kay, Lee Allen**

PUBLISHING
For advertising & licensing:
magpi@raspberrypi.org
Publisher: **Liz Upton**
CEO: **Eben Upton**

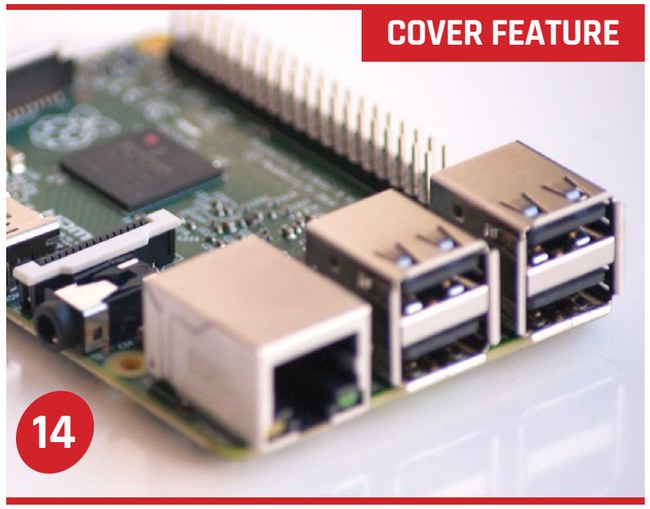
With thanks to this month's contributors: **David Crookes, Liam Fraser, David Hunt, Phil King, Simon Long, Simon Monk, Martin O'Hanlon, Les Pounder, Matt Richardson, Richard Saville, Richard Smedley, Sean M Tracey** and **Robin Withers.**

Contents

Issue 31 March 2015 raspberrypi.org/magpi

TUTORIALS

COVER FEATURE



RASPBERRY Pi 2

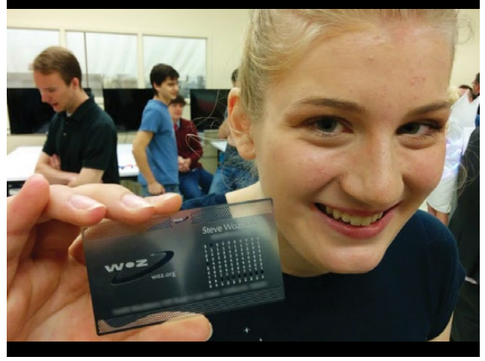
It's priceless and infinitely powerful, at least according to our account of the Raspberry Pi Foundation's new credit-card-sized PC. Here's why...

NEWS FEATURE

MEET THE MAKERS

30

Meet the technologists of tomorrow as we interview four young hackers and makers, and learn what they do with Pi



REGULARS

REVIEWS

5 Pi 2s must be WON!

66

What would you do with a Pi 2? Let us know for your chance to win!

YOUR PROJECTS

LEGO-LUTION OF Pi

To celebrate the third birthday of the Raspberry Pi, the Hayler family did what they do best and built a massive LEGO diorama

RASPBERRY PI SALES PASS FIVE MILLION

But does that make it the bestselling UK computer of all time? **The MagPi** delves into the past to find out...

When the Raspberry Pi was launched in February 2012, Foundation trustee Eben Upton's firm belief was that the tiny and inexpensive bare-bones computer would sell 10,000 units over the course of its lifetime. As it turned out, Mr Upton could hardly have been more wrong – albeit much to his delight.

For in the first year alone, the Pi hit sales of 1 million, as demand outstripped supply to such an extent that it proved very difficult for the manufacturer to keep up. But on 17 February this year, the sales figures reached a staggering 5 million – making the Raspberry Pi the fastest-selling UK computer of all time, as well as one of the most successful.

It's been a long time coming for a British-made home computer to come close to anything like the dominance of the machines of the 1980s and 1990s. Back then, multimillion-selling computers were far more common, as entrepreneurs and inventors ranging from Clive Sinclair to Alan Sugar lined up for a slice of the emerging market, making their fortunes and earning lordships and knighthoods in the process.

Yet there is still some way to go before the Pi can take the undisputed crown of the UK's biggest-selling computer. While the Pi has more than surpassed the 1.5 million BBC Micros sold and bettered the entire Amstrad CPC range (which, despite being especially popular in France, shifted just 3 million units), it still lags behind the Amstrad PCW's 8 million sales. Significantly, though, it has gone neck and neck with the most iconic British-made computer of all time: the ZX Spectrum.

The Speccy

The Speccy, as it was affectionately called, was launched by Sinclair Research in 1982 and it attracted the attention of 5 million punters – the exact same number as the Raspberry Pi. There is some debate over whether or not the figures should include the sales of the ZX80 and ZX81 – that would add an extra 1.6 million – but when you look at the pattern of Pi sales, you get a sense that it's only a matter of 'when' and not 'if' it becomes top dog in terms of sales.

TAKING UP ARMS

When sales of the Raspberry Pi eventually exceed that of the ZX80, ZX81 and ZX Spectrum combined, the consensus is that the small-form computer will become Britain's most successful of all time. But will it?

According to Stephen Furber, one of the designers of the BBC Micro, a lot depends on how the terms are defined. "More than 60 billion ARM processors have been shipped in total, and the rate is now over 12 billion a year (a small subset of which are, of course, going into the Raspberry Pi), so ARM is clearly the most successful British computer of all time, and indeed the most successful computer in the world," he says. "But maybe ARM doesn't count as a 'computer' because it is a microchip – or a part of a microchip – that needs other components to operate?"

The Raspberry Pi took 20 months to achieve sales of 1.75 million, yet just 16 months more to add 3.25 million. Should this trend continue, we could (in theory) see an extra 6 million sales in a year's time, though Upton touted 3 million as a target for the year. Whatever the truth in that, far from waning, the Pi is becoming more popular. Even Sir Clive Sinclair, the brains behind the Spectrum, has been impressed by its impact.

"It's very exciting," he said. "I think it's dramatic and terribly clever." He has praised both its low cost and its accessibility which, like his Spectrum, allows users to quickly start coding. "Suddenly people can again get their hands on computing power and play with it, manipulate it and really understand it."

All of this is undoubtedly brilliant news for the Foundation, which created the computer as a tool to get children coding. Over the past three years, the Pi has become as ubiquitous in British schools as the BBC Micro was in the 1980s, and the irony of this situation is rather delicious.

After all, the BBC Micro was produced to complement the BBC Computer Literacy Project, which aimed to familiarise pupils with the ins and outs of these newfangled machines back in 1981. When affordable PCs became common in the home and consoles began to seize control of the gaming industry, it became more difficult for children to 'get under the hood' of computers, and so the number of youngsters growing up with programming skills nosedived. The Raspberry Pi was created as a response to this.

> "Should this trend continue, we could, in theory, see an extra 6 million sales in a year's time"

A touch of Micro magic

It was hoped that by invoking the spirit of the BBC Micro, coding skills would rise again. On the back of this, former education secretary Michael Gove proposed that Computing, with a firm emphasis on coding, would replace ICT as a subject in schools. As the new curriculum was introduced last September, so many more Raspberry Pis were purchased.

Teachers view the computer as the perfect way to introduce children to programming and so the Raspberry Pi has become the new BBC Micro, albeit with greater sales and better penetration. Best of all, instead of having one computer per class, the Pi is so inexpensive that some schools have enough for every pupil. It's a triumph in every sense.

"The success of the Raspberry Pi is to be highly welcomed," says Stephen Furber, one of the designers of the BBC Micro. "It has created a real buzz of excitement around learning to use computers, reminiscent of the early days of the BBC Micro."

Image: Bill Bertram CC BY-SA 2.5

Above Playground debate raged over whether the UK's limited-colour Spectrum was better than the American-made Commodore 64

Image: Stuart Brady (public domain)

Above Very few pupils went through a British school in the 1980s without learning to program (or play *Chuckie Egg*) on a BBC Micro

Image: Bill Bertram CC BY-SA 2.5

Above Lord Alan Sugar's Amstrad made the CPC range, including the 464 (pictured). The disc-based 6128 sold well in France

FROM BEDROOM TO BOARDROOM

The Raspberry Pi's remarkable success has spawned a cottage industry of entrepreneurs, fuelled by crowdfunding platforms like Kickstarter. Here are some recent Raspberry Pi success stories...

› FLOTILLA

Build great ideas in minutes, not hours, with Flotilla for Raspberry Pi by the pirates of Pimoroni. Flotilla recently rocked Kickstarter, surpassing its goal by 447% to raise nearly £147,000.
flotil.la

› OPENPi

For inventors, makers and coders, OpenPi is designed to make it easier and cheaper to design, make and sell products based on the Compute module.
wirelessthings.net/openpi

› RASPiO DUINO

Learn to code Arduinos from the comfort of your Raspberry Pi with the latest Kickstarter project from the co-creator of HDMIPi and RasPi.TV blogger, Alex Eames.
kck.st/1zrHIFI

GET A SLICE OF THE ACTION

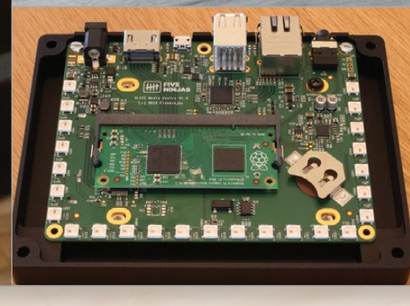
The crowdfunded media player powered by the Raspberry Pi Compute Module is shipping soon, with online pre-orders coming immediately after…

The case design is now slimmer and sleeker than was originally planned during the funding process

The unique LED light ring puts on quite a show and offers stunning (but optional) visual feedback

The Slice remote control was made available as a £15/$25 pledge on Kickstarter and has already been delivered to backers

Right Slice can output full 1080p via HDMI and play network media via a customised version of OpenELEC (which is powered by KODI)



Right The anodised aluminium chassis houses a Raspberry Pi Compute Module, a real-time clock and can hold a slimline 3TB 2.5" hard drive

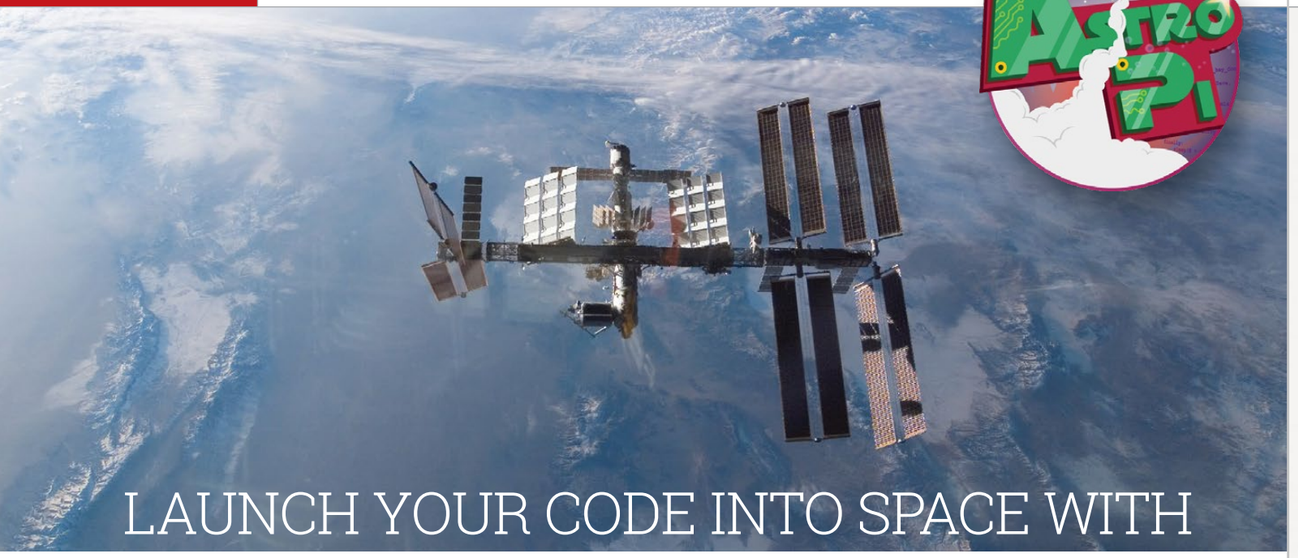
Slice, the Raspberry Pi-powered media player, was successfully funded on Kickstarter in September last year, shattering its £90,000 goal on its way to raising over £227,000 ($345,000). While it was originally due for launch in December 2014, the team behind the HD media box, known as FiveNinjas, have now started the delivery process.

"The backers who just wanted a Slice remote control have already got them," explains Paul Beech, member of FiveNinjas, a co-founder of **Pimoroni.com** and the designer of the Raspberry Pi logo. While delivery of actual Slice hardware still hasn't taken place, it is coming soon. "Everything is ordered and in production, and we've already got things like cables, Wi-Fi, remotes, and power supplies in hand.

"We've had delays when getting the test plan perfect on the PCB [printed circuit board]. It's a complex beast, so getting that right has taken more time and communication, which has been through intermediaries, slowing things down. A bad product is bad forever, so we're being super-careful about the process. Even though we're late on our deadline, this is still the quickest I've seen a complex hardware Kickstarter delivered," he said.

It seems Slice buyers will be getting much more than they originally bargained for, with Beech revealing that a large amount of future-proofing has taken place: "It's a box you'll still be using years from now. The Compute Module is a good part of that. When the Raspberry Pi Foundation update the Compute Module, we expect it to be a slot-in replacement to upgrade your Slice."

We also asked Beech what the FiveNinjas' plans were, post-Kickstarter delivery. Will Slice be available to those that missed out on the Kickstarter? "Backers come first," said Beech. "Once we're happy that delivery to them is going smoothly, we'll spin up a shop on **FiveNinjas.com** so everyone can get a Slice."



LAUNCH YOUR CODE INTO SPACE WITH ASTRO Pi

UK astronaut Major Tim Peake offers UK schoolchildren the chance to launch their code into space with a competition that sees two Pis travel to the International Space Station…

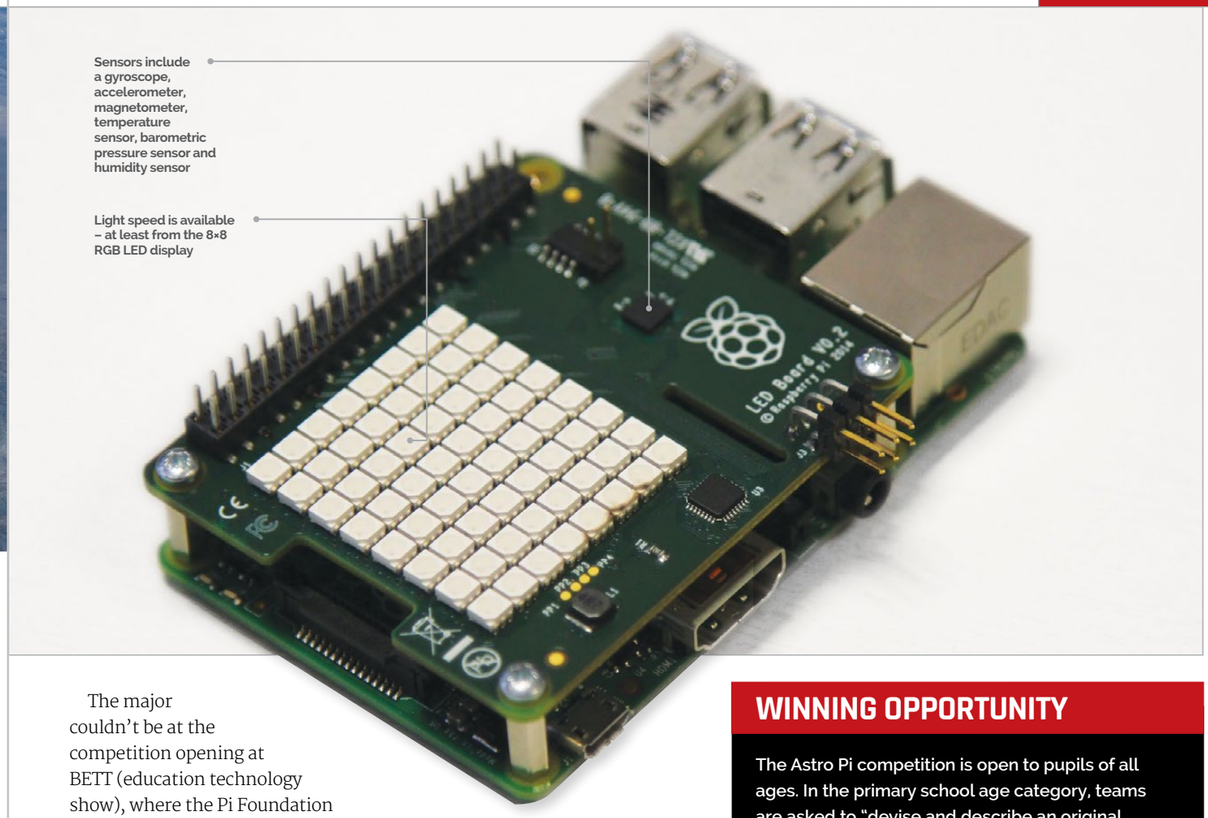
Ever dreamed of getting into space? It takes many years of training before those with the right stuff are selected for missions beyond planet Earth, but thanks to the Raspberry Pi, UK schoolchildren have the chance this year to send their code up to the International Space Station (ISS), when Major Tim Peake takes off in November for a six-month mission.

Each ISS-bound Raspberry Pi will carry a new Astro Pi board, loaded with sensors and gadgets well-suited to getting real science done in space. The gadget list of the Astro Pi is impressive and too long to detail in full here, but includes a gyroscope, accelerometer, barometric pressure sensor, and magnetometer, among other things. Add in the regular camera module or infrared camera (both versions will be travelling into space) and, like us, you may already be thinking of many possible experiments for more Earth-bound Pi boards as well.

Speaking at the UK Space Agency during the launch of the Astro Pi in December, Business Secretary Vince Cable spoke of the government's industrial strategy to create a new generation of engineers with world-class skills: "So much technology relies on big data, but not enough people are being trained in this field. This challenge helps the next generation to have fun whilst learning the skills that industry need."

Major Tim, also attending the launch event, spoke about his excitement at the project's cooperation between UK industries and institutions, and the potential of the Astro Pi on board the ISS. "There is huge scope for fun science and useful data gathering, using the Astro Pi sensors on board the International Space Station. This competition offers a unique chance for young people to learn core computing skills that will be extremely useful in their future," he commented. "It's going to be a lot of fun!"

Above Major Tim Peake will be launched in a Soyuz rocket – the only means of reaching ISS since the Shuttle programme ended



The major couldn't be at the competition opening at BETT (education technology show), where the Pi Foundation was out in force – being an astronaut involves a lot of training and not much getting out to events – but he was there virtually, in cartoon form. Take a look at the competition launch video at **vimeo.com/117274487**, where Major Tim explains that "brilliant as the tech is, it's nothing without you. We need your ideas for space experiments using the kit, and your code, to make it work."

The possibilities seem endless, but to help focus creative thinking about the uses of the Astro Pi, there are five themes to the competition:

- Spacecraft Sensors,
- Satellite Imaging,
- Space Measurements,
- Data Fusion and
- Space Radiation.

The entries will be judged on creativity, originality, practicality and usefulness.

Ground control

We got to speak about the Astro Pi with Libby Jackson – the UK Space Agency's astronaut flight education programme manager, who is supporting Tim Peake's flight to the International Space Station in 2015/16. Before joining the UK Space Agency, she was a Columbus flight director, working at the Columbus Control Centre in Munich, Germany – the European control centre for the ISS – and she is passionately enthusiastic about every aspect of the space programme: "I've been working on manned space flight for most of my career, and to be back in the UK working on the first government-supported manned British mission is fantastic."

We ask her about the possibilities of real and useful measurements with the Astro Pi on board the ISS. What is the UK Space Agency expecting? "I know we'll be surprised," Jackson tells us. "Every time we run competitions, children in the UK – and children generally – always have amazing ideas that we

WINNING OPPORTUNITY

The Astro Pi competition is open to pupils of all ages. In the primary school age category, teams are asked to "devise and describe an original idea for an experiment or application which can be conducted on the Astro Pi by Tim during his mission." The two best submissions will get the opportunity to work with the Astro Pi team to interpret their ideas; the programmers at the Raspberry Pi Foundation will then code them, ready for flight on the ISS.

For secondary schools, there are three age categories – 11-13, 14-16, and 16+, corresponding to Key Stages 3, 4 and 5 in England and Wales. In the first phase, competitors submit their ideas for experiments and applications, with at least the best 50 submissions in each age category winning a Raspberry Pi computer and an Astro Pi board on which to code their idea. In phase two, teams will "develop code based on their original concept," and two winning teams will be selected in each age category. The deadline is 3 April 2015 for concepts, and 29 June 2015 for final code.

The winning teams' code will be readied for flight by the Raspberry Pi Foundation and CGI, have their code uploaded to the ISS, each receive a class set of Raspberry Pi and Astro Pi boards, meet the Astro Pi team, and participate in a winners event during Tim's flight. Extra prizes will be supplied by each of the UK space firms supporting the project.

READING SPACE

Don't miss our essential reading list for Astro Pi and other extraterrestrial projects...

The Astro Pi site
> **astro-pi.org**

Astro-Pi video
> **vimeo.com/117274487**

More details about the Astro Pi HAT
> **astro-pi.org/hardware**

Astro Pi FAQ
> **astro-pi.org/faqs**

Teachers' resources
> **nationalstemcentre.org.uk/elibrary/collection/2008/astro-pi**

ESERO, the UK Space Education Office
> **esero.org.uk**

High-altitude ballooning with the Raspberry Pi
> **daveakerman.com**

NASA's Space Gambit programme
> **spacegambit.org**

ISS switches to GNU/Linux
> **fsf.org/blogs/community/gnu-linux-chosen-as-operating-system-of-the-international-space-station**

ISEE-3 Reboot Project
> **spacecraftforall.com**

magnetometer to come up with where you are," and speaks of the other sensors' possibilities. She is enthusiastic about the Raspberry Pi ("fantastic British innovation, [part of a] coding renaissance") and the part it can play in STEM (science, technology, engineering and mathematics) education.

Open source space

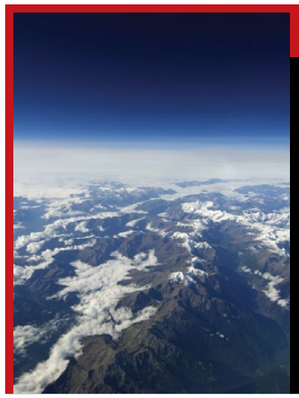
This isn't the Raspberry Pi board's first brush with space. Many readers will have followed with interest the helium-fuelled Pi missions of Dave Akerman (see box below), or may have experimented with astrography, attaching the Pi camera board to an amateur telescope. Beyond the Pi, free and open source software has been democratising space research elsewhere: NASA's SpaceGAMBIT is a US government-funded open source space programme, reaching out to makerspaces and hackspaces across the world to collaborate, last year focusing on projects for NASA's Asteroid Grand Challenge.

More hands-on was the ISEE-3 Reboot project, which last year – with NASA's blessing – took temporary control of the ISEE-3. Launched in 1978, the International Cometary Explorer (ICE) spacecraft – designed and launched as the International Sun/Earth Explorer 3 (ISEE-3) – came close to Earth in 2014 on its 17-year orbit. NASA had decommissioned the hardware necessary to communicate with the craft in 1999, but provided the archive material necessary for a group of amateurs to crowdfund provision of the software and hardware to communicate with ISEE-3 when it passed close to the Earth again last summer. Sadly, power issues on the craft meant the new data-collecting mission didn't get far, but the project both showed the scope of collaborative amateur efforts, and underlined the importance of making the most of space hardware. The chance to control the Astro Pi's sensors while aboard the ISS is a phenomenal opportunity.

The ISS itself made headlines two years ago when it switched to Debian GNU/Linux, the basis of the Pi's Raspbian operating system, following a number of what Eugene Kaspersky called "virus epidemics" among the Windows laptops on board. Other space efforts have gone further towards open source – at the silicon level – with nano satellites based on the OpenCores system-on-chip, a MIPs design maintained and modified by academics, amateurs and several specialist chip-design companies.

Below The Raspberry Pi Foundation's David Honess excitedly tweets a link to the Astro Pi competition launch video

HIGH-ALTITUDE BALLOONING



"I've been interested in space exploration since watching the Apollo flights as a child," Dave Akerman tells us. "Then in 2011 I saw a YouTube video of two guys launching a weather balloon containing just cameras and a tracking device, and thought: well, if it's that easy to get photos from 'space', then I should do it. This started a hobby which has largely taken over my life during the last four years!"

Akerman has brought his near-space exploration expertise and enthusiasm to many schools, particularly since he incorporated the Raspberry Pi into his flights: "The Pi brought about large change, firstly because it allowed me to send down live images during the flight, and secondly because of the resultant 'internet fame' and occasional media appearance. The most rewarding thing has been to help schools and scouts to do their own launches – it's really nice to see the look on their faces when they see their own images of the planet from near space."

Asked if he'd be putting Astro Pi sensors on his flights, Akerman suggests: "If the Raspberry Pi Foundation fancy testing one of their Astro Pi boards in a near-vacuum at -50°C, rather than the warm pressurised environment of the ISS, I'll be pleased to oblige!"

Above Major Tim Peake training inside the full-scale mock-up of the Soyuz capsule, at the Gagarin Cosmonaut Training Centre in Russia

> "Brilliant as the tech is, it's nothing without you. We need your ideas for space experiments using the kit, and your code, to make it work"

Careering ahead

Meanwhile, back on Earth, the shortage of engineers and scientists – and not just in the space industry – has led to initiatives from the grassroots (like Code Club), as well as government and industry, to inspire more people to consider STEM careers. The Astro Pi competition is a result of UK Space companies, the UK Space Agency, and the Raspberry Pi Foundation working together: "UK Space – the industry consortium group – responding to Innovation Growth Strategy, the government's call for ideas in space." Space and science organisations involved include SSTL, Airbus DS, CGI, the Space KTN, National Nuclear Laboratory, and National Physical Laboratory.

Libby Jackson tells us, "Studying science and maths at GCSE and A-level doesn't close your options down, but does the opposite and opens you up to a whole range of careers," citing her own discovery of chances to work in the space industry while studying physics. The UK Space Agency's educational aims encompass both Space for Education and Education for Space: "We need to encourage people in," Jackson explains, "but we also know space is a fantastic inspiration for learning in general. We'd like everyone to move one step up the ladder. So we'd like a lot of people who've never considered science before to start looking at it, the people who are interested in it maybe having a career in it, the people who are thinking about a career in it – maybe bring them into the space industry."

Supported by £2 million of government outreach money, ESERO-UK (the UK Space Education Office) and Raspberry Pi are developing teaching resources to link to the curriculum and assist teachers of STEM subjects in engaging their students in the competition, as well as explaining how to use and write code for the Astro Pi and its sensors.

Having kept the project secret for most of 2014, David Honess – education resource engineer at the Raspberry Pi Foundation – is delighted to talk about it at last: "We think this competition has the power to motivate a whole generation of coders for the UK science, technology and space industries to employ in the future... We want every school in the UK to enter!"

We know from some of the great Pi projects that we've seen that whatever makes it to the ISS is bound to be something special. From astrography, through space-bound balloons, to the Astro Pi, the Raspberry Pi is making giant leaps for a little computer. So, in the words of Major Tim at the competition launch, "What are you waiting for, Earthlings? Get those intergalactic hats on and let's get coding. See you in space!"

RASPBERRY PI 2

Better, faster, stronger

The Raspberry Pi is a really tiny, really cheap computer. So cheap, in fact, you could sacrifice your Starbucks coffee for a week to afford one. Because it's so small and affordable, it's an excellent tool to teach computer education in schools. As the Raspberry Pi Foundation learned, however, the 'bigger' kids among us quite enjoy playing with the Pi too.

Regardless of how old you are, you can hack, make, watch movies, and even play games with it. With the Raspberry Pi, though, it's just as easy to start making your own game or movie as it is to passively consume one. This is one of the key things that sets the Raspberry Pi apart from its growing competition.

With the Raspberry Pi 2, those goal posts have shifted. With its sixfold increase in power, whether you're making or playing games, movies or music, the experience is all the better for it.

On paper, the upgrade itself is pretty mundane stuff. The Pi 2 is essentially identical to a Model B+ in almost every respect. Other than its four 900MHz ARM Cortex-A7 cores and 1GB of RAM (as opposed to one 700MHz ARM Cortex-A6 core and 512MB of RAM), you could easily confuse the two.

The fact that there's no killer application exclusive to the Raspberry Pi 2 is, conversely, its biggest asset and a super-weapon primed to stave off even the stiffest competition in the burgeoning 'maker' marketplace. Best-of-British design and engineering is one thing, but a real commitment to powerful and flexible open source software, that offers near-total cross-compatibility between models, ensures this tiny, cheap computer is one of the most powerful in the world today, regardless of the model you're using...

The Pi 2 is compatible with almost every Model B+ case on the market

Add-ons designed for the Raspberry Pi Model B+, and any HAT add-on boards, will work on the Raspberry Pi 2 Model B

The 900MHz quad-core ARM Cortex-A7 CPU on the Broadcom BCM2836 SoC. The 1GB of RAM sits on the rear of the board

The combination analogue audio and video port is useful if you don't have an HDMI screen to connect to

Plug your Pi into any modern monitor or TV to turn it into a powerful quad-core Linux PC

Pi 2 uses the same amount of power as the Model B+ when idle, but can use as much as the original Model B under load

"We released the original Raspberry Pi on the 29th February 2012," says its creator Eben Upton. "It's been successful beyond our wildest dreams. Three years in, we've sold five million, and we think somewhere between one and two million Raspberry Pis are in the hands of children."

While the launch of the Raspberry Pi 2 on the 35th floor of the Shard, one of London's most impressive new landmarks, was quite grand, the humble aim of the Raspberry Pi Foundation was never far from view. The goal has always been to get more kids into computing; to give children of today the same kind of experience people growing up during the home computing boom of the '80s and early '90s had – people of Eben Upton's generation.

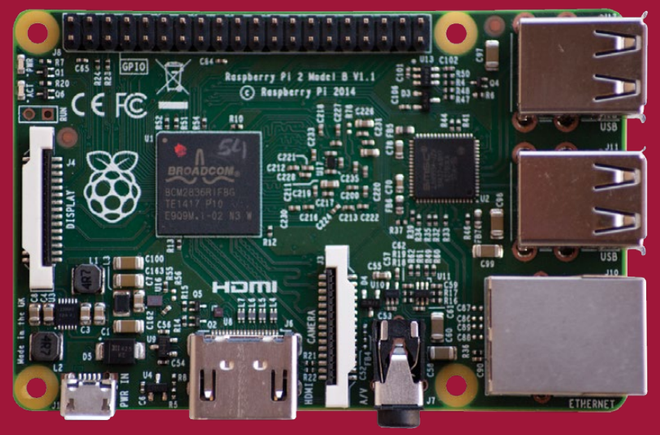
"It's the idea of having a computer in the bedroom that's hackable and fun," says Upton. "In the first few months we were concerned they were only going in the pocket of people like me, but over time it's become clear there is interest from children in learning computing with the Raspberry Pi. As much as anything, there's interest from children in learning something their parents don't understand."

Of course, even the Raspberry Pi Model B+ wasn't perfect. You can't build a $35 computer without making compromises. "The Raspberry Pi has a level of computing power of a PC from the turn of the century. Even when we doubled the RAM six months in, it still only had half a gigabyte." The Raspberry Pi 2 has been released to address these deficiencies, and more besides.

"The Raspberry Pi 2 takes us to a level of performance that makes it a genuine PC. We have power users in the office today that are using the Pi 2 as their PC at home," continues Upton. "While the Raspberry Pi was a great little PC, insofar as you had to be a little forgiving given the price, Pi 2 costs $35 and is now just a great PC – there is no caveat anymore."

> "The Raspberry Pi 2 takes us to a level of performance that makes it a genuine PC"

That's the money shot: the realisation that actually you can have your cake and eat it. It's also the moment you realise that the single-board computer revolution just got interesting. The Raspberry Pi was the hacking and making board with brains, but the Raspberry Pi 2 takes the formula much further: real-time physics calculations, complex computer vision projects with the Camera Module, and anything else – up to and including complex weather simulation – with minimal investment in hardware. The Raspberry Pi 2 represents the backbone of the perfect university computer cluster.



Below Imagine what could be achieved with a cluster of 40 Pi 2s running side by side

Above Eben Upton eyes Element14's past success, but there's much more to come

Software, software, software

The Raspberry Pi has popularised the format of the affordable development platform beyond all recognition, but it wasn't the first single-board computer and it won't be the last. The market has become saturated with competitors, many already as powerful as the Pi 2, others more powerful still. So what makes the Raspberry Pi special? The same thing that led to the iPad's runaway success in the tablet market – useful, targeted, easily accessible software.

In the three years that the Raspberry Pi has been with us, the Foundation has been working tirelessly on improving its software offering. Much like Ian Bell and David Braben dedicated themselves to crafting their BBC Micro assembly code to cram their most famous game, *Elite*, into a 22KB footprint, the Raspberry Pi Foundation has crafted Raspbian, its official Linux-based operating system, into an efficient, capable interface for its credit card-sized PC. Even before Pi 2 hardware, the Pi had become much faster and more usable because of Raspbian.

Adding a further three CPU cores and 512MB of RAM, while retaining complete compatibility with the same lean, polished software, is like stripping all the excess weight out of your car and fitting a roll cage. The Raspberry Pi 2 was born track-ready.

It was a must that all users of all Raspberry Pi models have the same experience. As Eben Upton so eloquently puts it, "We don't want to orphan 4.5 million Raspberry Pi users." Raspbian is Raspbian, whether you're running a Pi or a Pi 2, a Model B or an A+, the latest add-on HATs (Hardware Attached

RASPBERRY PI 2 IN NUMBERS

10 The version of Windows supported by the Pi 2

6 Average speed increase from the Pi Model B+

10,000 The number of Pis the Foundation thought it might sell

20 The number of full-time Raspberry Pi employees

3m The number of Pis the team hope to sell during 2015

900 The speed, in MHz, of the new ARM Cortex-A7 processor

02/02/15 The date the Raspberry Pi 2 was officially released

2012

May 2011
BBC news goes viral online
Elite game developer David Braben reveals an early prototype of the Raspberry Pi to BBC correspondent Rory Cellan-Jones. The video goes viral.

Feb 2012
Raspberry Pi Model B released
The Raspberry Pi Model B breaks the internet as overwhelming demand crashes the websites of its makers, Premier Farnell and RS Components.

March 2012
Ethernet bork slows progress
Due to a manufacturing error in China, the wrong networking ports are soldered to the first Pis, holding up production while they are replaced.

April 2012
Short supply fuels fire
The Raspberry Pi becomes available, but only in very short supply. Demand for the Pi and pre-order waiting lists continue to grow unabated.

May 2012
A community magazine is born
The MagPi magazine comes into the world! The magazine was designed by enthusiasts for enthusiasts and it's still alive and kicking!

July 2012
Raspberry Pis on the edge of space
Dave Akerman puts a Pi where no Pi has gone before: 40km straight up, attached to a weather balloon. He's repeated the feat many times since.

September 2012
New jobs for Pi bakers in Wales
The Foundation announces a deal with Sony that sees Pi production start up in a plant in Wales, meaning UK jobs for a UK computer.

2013

October 2012
Thanks for the memory
The Model B gets an upgrade from 256MB to 512MB RAM, in the hope it will become more useful as a home computer. It seems to do the trick.

November 2012
Three. Two. One. Say cheese!
The Camera Module is released. Its 5MP sensor and ability to record in HD ensure it's a resounding success and must-have Raspberry Pi accessory.

January 2013
Google puts Pis in schools
Google's Eric Schmidt donates 15,000 Raspberry Pis to be placed in schools all around the UK, alongside other educational material.

February 2013
The Model A is finally released
The Foundation announces the cheaper, lower-power Raspberry Pi Model A. It has 256MB RAM, no Ethernet, and only one USB port.

March 2013
One million (and counting)
After celebrating its first year on sale, the Raspberry Pi also breaks the one million sales barrier. Less than two years on and five million have been sold.

April 2013
Half a million Welsh Pis
In the six months since production of Raspberry Pis started in a Sony plant in Wales, 500,000 have already rolled off the production line.

May 2013
Can we have all the swag?
The Foundation launches the Swag store. Every purchase from the store helps fund the Foundation's educational activities in the UK and abroad.

SIX THINGS THAT ARE BETTER WITH RASPBERRY PI 2

Home office
You can now use the Raspberry Pi as a productivity machine. Now packages like LibreOffice (**Libreoffice.org**, the open source alternative to Microsoft Office) are responsive and usable.

Web browsing
Enjoy loading times four times faster than the old Model B – the Pi 2 hardware is much more adept at running modern websites. It's still not perfect, but it's nowhere near as frustrating as it used to be.

Minecraft: Pi
Minecraft is easily three times faster than before, and coders will be able to create much more elaborate and 'explodey' scripts. It's also now more realistic to create a *Minecraft* server with the Pi 2, as **stuffaboutcode.com**'s Martin O'Hanlon recently demonstrated (**bit.ly/1DzFJqx**).

Computer vision
With the addition of the affordable Raspberry Pi Camera Module, the Raspberry Pi 2's four faster cores make the evaluation and processing of images and video streams much easier.

Sonic Pi 2
Despite a cruel bug that initially meant the live coding music application couldn't use the full amount of RAM afforded to the Pi 2, the ability to create better beats and more complex compositions has drastically increased. Check out **sonic-pi.net** to get started today.

Retro gaming
The Pi is already a very popular solution for playing retro games, but the extra power from the Pi 2 opens up a whole world of new possibilities, including the emulation of fifth-generation consoles like the Nintendo 64.

on Top) are designed to 'just work'. The pin layout is the same for GPIO projects, and even 99% of the cases designed for the B+ fit the Pi 2. The Foundation has made a lot of its decisions based on the community and the cottage industry that surrounds the Pi, and it shows.

It wasn't a completely smooth getaway, though, with a small amount of wheelspin as the Pi 2 crossed the start line. There were some avoidable GPIO issues, an unforeseeable Sonic Pi 2 bug, and a rather amusing glitch that sees the Pi 2 proving rather camera-shy (crashing when a particular xenon flash is used in close proximity to a certain photosensitive component on the board). All these things were overshadowed, though, by the performance benchmarks the Raspberry Pi community were quick to supply.

Chief among them were **RasPi.TV**'s comparison videos. Alex Eames's short videos almost perfectly sum up the added user-friendliness that the extra power provides. According to the side-by-side comparisons (which you can see at **raspi.tv/?p=7589**) the Pi 2 boots in just 15 seconds, half the time it takes the Model B+.

> This isn't the same Windows 10 we'll be seeing on most other devices

Perhaps an even more accurate reflection of the Pi 2's new capability as a productivity machine, or second home PC, is the massive boost in performance seen from Epiphany, Raspbian's relatively new HTML5- and JavaScript-capable web browser. Browsing has always been the Raspberry Pi's most frustrating Achilles heel. It struggles to reliably serve all but the simplest of webpages and keels over at the prospect of handling JavaScript-heavy content. But in his testing of the Pi 2, Alex Eames saw webpage loading times reduced by 75%. **RasPi.TV** demonstrated **raspberrypi.org**'s homepage loading in six-and-a-half seconds. It's still no spring chicken, but side by side with the B+ it's the Usain Bolt of Raspberry Pis, breaking the previous record by a whopping 16 seconds.

2015 SALES PREDICTIONS
"We'd like to sell a total of 3 mill in 2015; that would be a good year," says Eben Upton. "The interesting thing about Pi 2 is that it does broaden out the addressable market, so I'm hoping that with this product we stand a chance of having people buy them as their second PC in their house. We've got to that level now where we can address these different market segments. We sold between 2 and 2.5 million in 2014; it would be great to hit 3 million this year."

Microsoft joins the party
Probably the biggest story surrounding the release of the Raspberry Pi 2 was the news that the Foundation has been working with Microsoft for six months to bring Windows 10 support to the Pi 2. "This is a really exciting thing for us. Raspberry Pi 1 uses an ARM 11 processor, which implements the ARM v6 instruction set architecture. In moving to this new core we've moved to the ARM v7 instruction set architecture, which broadens out the range of operating systems we can run on the Raspberry Pi," explains Upton. "Windows 10 runs; I've seen it and it's pretty cool."

Below *Minecraft: Pi* becomes even more usable with the advent of Pi 2

While much of the world saw the headlines about Windows 10 support and got excited, they went away misinformed. This isn't the same Windows 10 we'll be seeing on most other devices, despite Microsoft's forthcoming refresh being sold as a universal operating system that encompasses desktop PCs, smartphones, and tablets. "This is a version of Windows 10 primarily targeting Internet of Things (IoT) applications, so the intention here is to have a device you can use to build IoT devices that have screens attached, and it participates in the broad range of Windows 10 API support," explains Upton. You'll be able to take a Windows 10 application that runs on a PC or smartphone and run it on a Raspberry Pi 2. Ultimately, this implementation of Windows 10 support sees the Raspberry Pi 2 flourish in another area in which it excels – as a development platform.

So what's next for Raspberry Pi, now the Raspberry Pi 2 Model B has landed? When you apply its full title, it's clear the potential for a Model A is implied. While Eben Upton has confirmed the intention of making a smaller, lower-power version of the Pi 2, the Foundation is very much committed to the $20 price for the Model A and – as it stands – the maths simply doesn't stack up to produce a quad-core Raspberry Pi in that price range. That said, it's only a matter of time until it happens. Eben has also gone on record to confirm that a Compute Module version of the Raspberry Pi 2 is also in the works. Since there's no inherent price barrier at play, we'll certainly see this before the Pi 2 Model A, but likely no earlier than September.

2014

June 2013
Silver is better than gold
Eben Upton, creator of the Pi, receives the Royal Society of Engineering's Silver Medal for outstanding contributions to UK engineering.

August 2013
A weekly slice of Pi
Ben Nuttall and Ryan Walmsley start a weekly Pi newsletter, **piweekly.net**, which – as of Feb 2015 – has over 10,000 happy readers and counting.

October 2013
Infrared-y on set? Action!
The Pi NoIR version of the Camera Module is announced. The NoIR is designed to operate in the dark, so it's ideal for capturing nature shots and doing science.

November 2013
Wolfram: It all adds up
The Wolfram Language (as used for Mathematica) is launched. The Pi is the second computer ever to feature it installed free as standard.

December 2013
Adventures in Raspberry Pi
Carrie Anne Philbin's award-winning *Adventures in Raspberry Pi* is released. Find it on our top reads list in this issue's book reviews (page 62).

February 2014
A birthday gift from Broadcom
On its second birthday, the Foundation reveals that Broadcom has released the graphics stack of the Pi's VideoCore GPU under a BSD 3-Clause licence.

March 2014
The MagPi comes of age
The 21st issue of The MagPi magazine is released to the world, and includes articles on PIR motion detection and weather stations.

April 2014
A new Pi? It does compute!
The Raspberry Pi Compute Module is announced, **raspberrypi.org** gets a redesign, and a £1 million education fund is also set up.

July 2014
The Model B+ launches
The new and improved Model B+ offers a new layout, lower power requirements, and lots more GPIO pins. It's the Pi the world has been waiting for.

August 2014
Ben goes to America
The Raspberry Pi undertakes a mammoth tour of America, courtesy of the Foundation's Ben Nuttall. He drives over 4,000 miles.

September 2014
An internet Epiphany
A new, improved, hardware video-decoding, ARMv6-optimised, HTML5-supporting web browser is released for the Pi: Epiphany.

2015

November 2014
The Model A+ is released
The Raspberry Pi Model A+ arrives. It's smaller, cheaper and uses less power, making it ideal for battery-powered and small-form-factor applications.

January 2015
Astro Pi is announced
The Astro Pi competition is announced at BETT 2015. Read all about this amazing spacefaring mission elsewhere in this issue.

February 2015
Raspberry Pi 2 has arrived!
The Raspberry Pi 2 is launched. If your memory is so short you need reminding of this already, you require medical assistance, not a timeline.

Mad scientists working on the original Model B Rev 1. A surprisingly accurate account

The Raspberry Pi 2 takes centre stage on this elaborate rocket. A brilliant nod to Astro Pi

A Raspberry Pi Model A rolls off the production line, ready to be loaded onto a truck

Jasper (9) and Ozzy (8), the creators of the LEGO timeline of Raspberry Pi

Right Who are these likely-looking lads?!

LEGO-LUTION OF Pi

Ozzy, **Jasper** and **Richard Hayler** celebrate their collection of Raspberry Pis the only way they know how...

When Richard Hayler isn't working for the Foreign Office he's a Raspberry Pi enthusiast, CoderDojo mentor and Code Club volunteer. "Pretty much everything else revolves around my sons, who love getting involved with all things Pi," he says.

Besides test-driving his educational material, Richard's sons Ozzy (aged 8) and Jasper (9) are often to be found hacking and making with the Pi. Their latest creation is this rather marvellous LEGO scene designed to celebrate the evolution of everyone's favourite credit-card-sized PC.

"I recently liberated my Rev 1 Model B from the BrickPi robot and thought that it would be nice to take some photos of all the different versions I own," explains Richard. "I didn't get round to it straight away, and it languished on my list of 'things to do'. Then I was lucky enough to get a free Pi 2 on the day of launch by tracking down the Element14 PiCycle (**bit.ly/1DpL9Es**), which reinvigorated my interest in the idea.

"I asked the boys if they had any ideas of how to make the pictures more exciting than just a bunch of Pis on a desk, and they immediately suggested this."

After discussing a few ideas, Richard's youngest, Ozzy, suggested creating a LEGO timeline showing the Pis being used in different ways. "This morphed into a scene which follows the Pi from the design phase, through manufacture in a Pi factory, to being loaded onto a lorry. Then we have some children using it in a school, and finally, a Pi being strapped to a rocket, ready to launch up to the ISS to celebrate Astro Pi."

Each stage of the design boasts a more modern model of the Raspberry Pi, Richard explains, not to mention the addition of a couple of 'Easter eggs', including a rather suspicious-looking group consisting of a pirate, monkey, robot, and ninja.

What's next for the Haylers? "[We're] putting together some hardware based around the Model A+ for kite-mapping photography, that will record the altitude and orientation of the kite, and use it to have some intelligence about when (and when not) to capture an image." You can see more of the Haylers' Pi timeline, and learn more about the family's other projects, at **richardhayler.blogspot.co.uk**.

RASPBERRY Pi CLUSTER

David Guill shows us what happens when he's left in a room with 40 Raspberry Pis, two 24-port switches, 5TB of storage, and an ATX power supply

A computer cluster is 'a set of connected computers that work together so that, in many respects, they can be viewed as a single system'. Clusters can be anything from a few cheap computers networked together to supercomputers made up of thousands of individual 'node' systems, designed to undertake complex tasks like modelling weather or trying to beat humans at chess.

Back in early 2014 David Guill, a recent MSc Computer Engineering graduate, showed the world his rather impressive project to create a computer cluster consisting of 40 Raspberry Pis.

He created his cluster entirely single-handedly, right down to the custom laser-cut acrylic case.

A new direction

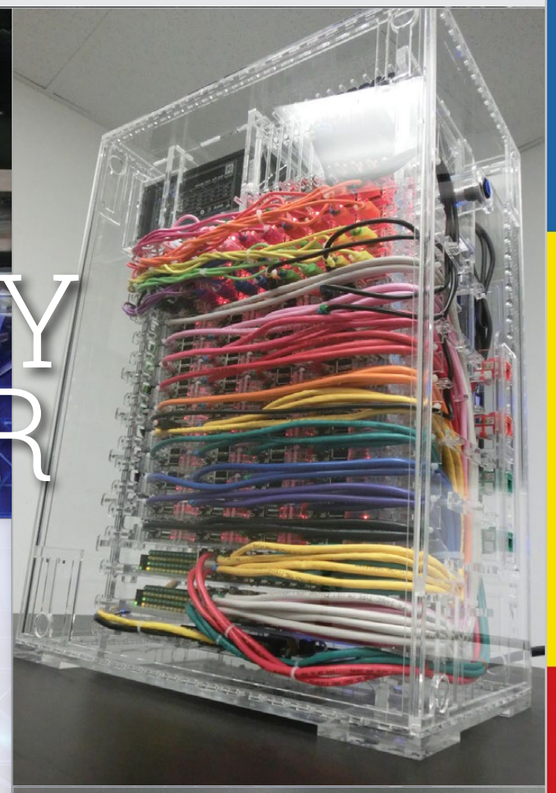
A year on, we caught up with David to find he's still working hard on his pet project, and it seems it's taking him in new and exciting directions. "While it wasn't one of my original goals, the most important work I've done on it so far has been in porting software to ARM," says David. "I spent some time trying to get Apache Mesos working properly on it." It's a worthy distraction since ARM is fast becoming a real player in the server market, meaning David's work could have real value in the coming years.

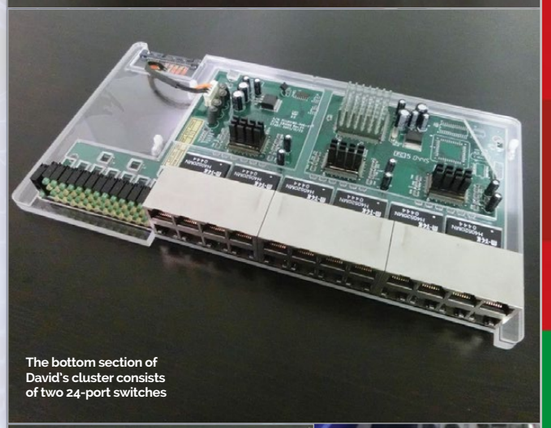
"While I've mostly been fixing supporting tools as I discover they aren't ready for ARM, I'll also be writing some of my own tools . My objective is to have a suite of tools with insignificant diminishment of returns for expansion, where the millionth node in a system would contribute nearly as much as the tenth did when it was new."

Virtual worlds

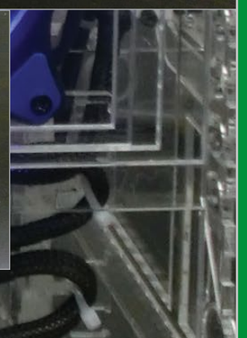
David's ultimate goal, though, is quite different – he wants to move into virtual reality. "My end goal is to develop detailed virtual reality simulations, like you might see in a hybrid of *Minecraft*, *Little Big Planet*, and role-playing games in general, with deformable planetary worlds. Of course, this is still hobby work – I have no guarantee that it'll ever get close to completion."

You can learn more about David and his work on the Raspberry Pi Cluster at **likemagicappears.com**.

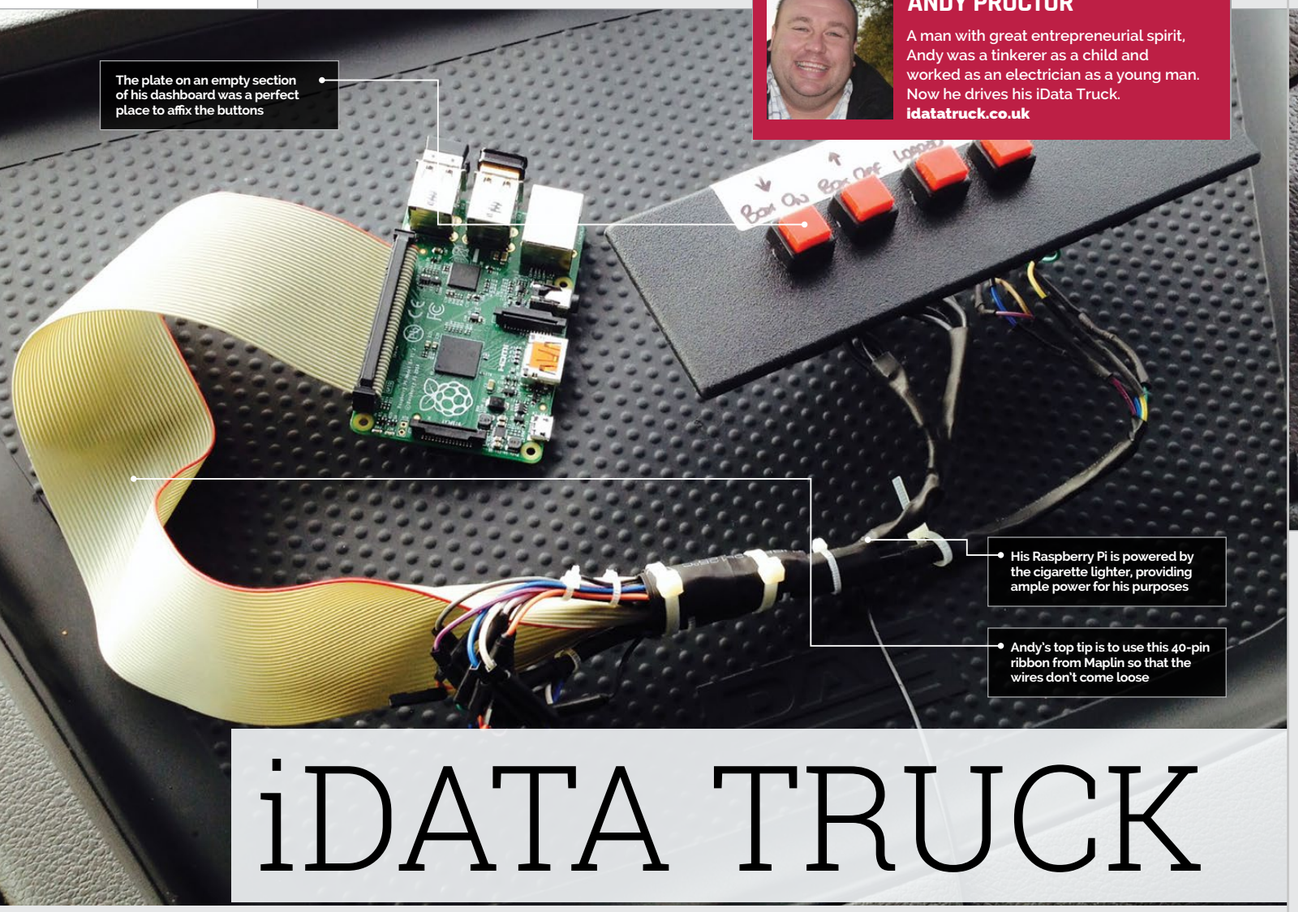




The bottom section of David's cluster consists of two 24-port switches



Above Most of the mid-section is made up of ten rows of these bad boys



iDATA TRUCK

ANDY PROCTOR

A man with great entrepreneurial spirit, Andy was a tinkerer as a child and worked as an electrician as a young man. Now he drives his iData Truck.
idatatruck.co.uk

Quick Facts

- Andy learned to program for this project
- The community has already begun making his code better
- Most of the electronics come from a SunFounder starter kit
- Andy uses an iPhone to connect his Pi to the internet
- There will soon be a live camera stream from his lorry

A Raspberry Pi-powered lorry? It's not as strange as you think, as **Andy Proctor** shows us how he automates deliveries with Pi

Braving the often-congested motorways of Great Britain, we find our hero Andy Proctor – lorry driver and truck hacker extraordinaire – live-tweeting his schedule as he picks up, and delivers, the nation's shipping containers. There's much more to his tweets than meets the eye, though, and it all started with him finishing up his previous business and becoming a lorry driver.

"I was tweeting '#m25' and '#m12', and I noticed it was being retweeted automatically. I contacted the guy who was doing it and he told me it was powered by a Raspberry Pi and a bot. I looked up what a Raspberry Pi was and decided that I wanted to push a button and send a tweet."

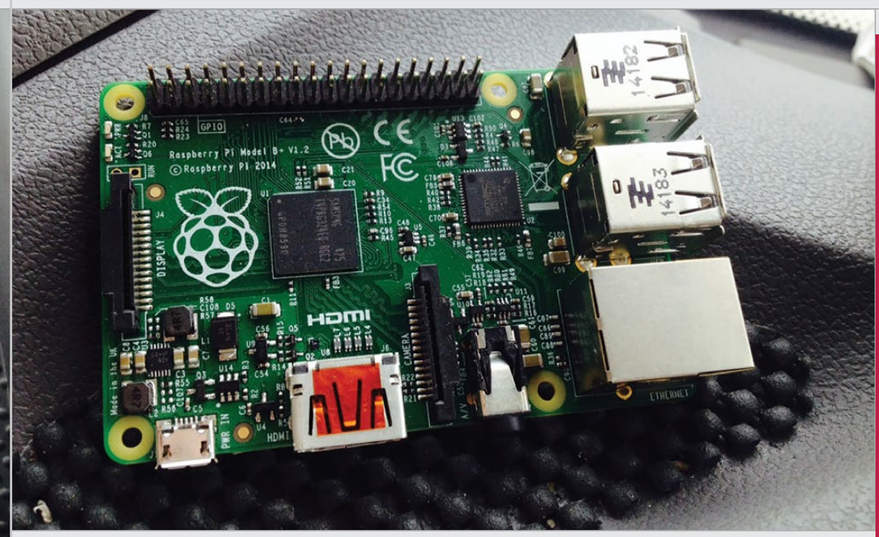
The transition from being a successful business owner to a lorry driver hadn't had the best impact on Andy, and his wife encouraged him to play about with his new Raspberry Pi on their honeymoon. With a background as an electrician, website builder and tinkering with computers and electronics as a kid, some of the Pi came naturally to him.

Humble beginnings

"I started off with Tweepy and Scratch to make some lights flash, and built a little box with a board for the lights, which made me happy! I then did the same in Python, learning along the way, and within six weeks I had created the box with the four buttons that you see now."

The iData Truck was born and not only did the buttons tweet out his current status, it emailed his office – a task he would have been doing manually anyway. He published a video on YouTube describing his setup, which got picked up by the Raspberry Pi Foundation and even the BBC. Andy isn't finished yet, though.

"I only have four things I can transmit," he laments. "So now what I'm doing is a barcode-scanning version. I've printed off loads of barcodes which I can stick to the back of my time sheet and then I'll be able to scan them. If it's just scanning to say 'start of day', 'end of day' or 'on a break', it will tweet that but not email it. If it's 'running 30 minutes later' or 'box on'/'box off', it will still email it to the office. So there will be a split of what data gets sent where."



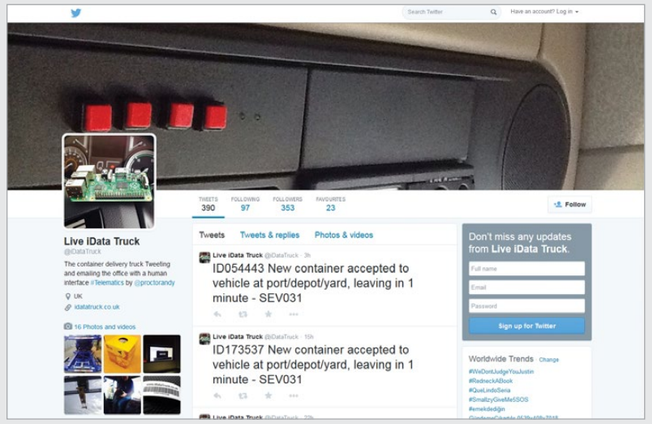
> "Not only did the buttons tweet his current status, it emailed his office"

What's next?

Next on the list for Andy is a camera – Pi-powered, of course – in a blind spot of his lorry that will display on his dashboard and hopefully make it easier to manoeuvre while reducing the risk of accidents. He also has further plans for the iData Truck beyond his personal use of it.

"I've approached the people that make the software that everybody uses in the industry and they said if one of their customers wanted to use that, then that's fine, they'd support it… one person's been in touch that can make the hardware, a box to put it in, the switches in the panel and the software, should I want to develop it further."

So next time you pass a container lorry on the M3, give it a wave and you might end up on iData Truck TV.



Left The information on the iData Truck stream is always expanding. Andy initially had trouble with duplicate tweets until he started using timestamps

HOW TO REPORT IN WITH iDATA TRUCK

> STEP 01
Press the button

When Andy loads or unloads, he presses one of the four preselected buttons on the iData Truck so he can let his company know what he's up to.

> STEP 02
Wait for the beep

You need to hold the button down for half a second: "It kept getting really hot, so I had to put a delay in to stop the processor working so hard from all the loops!"

> STEP 03
Email and tweet sent

An email is sent off to Andy's company to let them know of his status, and a tweet is sent to the @idatatruck Twitter feed for everyone to see what he's up to.

MARK PARRISH

A .NET developer with a software consulting firm, who grew up playing Nintendo's ground-breaking consoles.
imgur.com/a/1RUwa

SNES Pi CASE

Quick Facts

- It took two weekends to complete
- This is a US SNES, hence the purple and angles
- Some of the printed circuits are scratched off to make it work
- The original power light is fixed to a GPIO pin
- *F-Zero* is one of Mark's favourite SNES games

What happens when you turn a Super Nintendo into a Raspberry Pi? F-Zero becomes Raspbian, among other things…

You walk into a room and see a SNES. A classic, a legend, one of the greatest videogame consoles to ever be crafted by the hands of man. Beside it is a cartridge of the original *F-Zero*, perhaps not the best in the series but an excellent game nonetheless. You slam it in (gently though, they're both 25 years old), flick the power switch and look for a controller. Suddenly a Raspberry Pi logo shows up.

This isn't a Super Nintendo. It's a Raspberry Pi case that used to be a Super Nintendo.

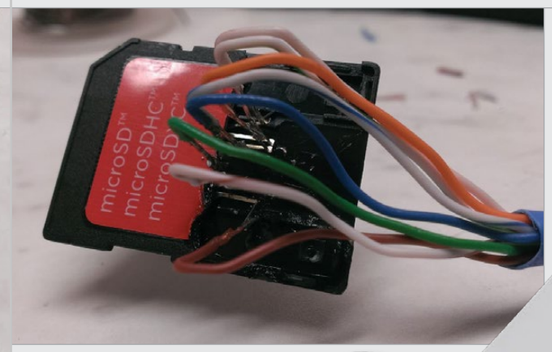
"Like most great ideas, [I got it] from watching others and seeing what they were building," Mark tells us. "[Also] how they were solving particular problems… then taking those ideas and improving on them in my own creative way."

His own creative way is frankly incredible. Instead of just fitting the Raspberry Pi into an empty case, he soldered parts of it directly to the original motherboard of the broken SNES he was working on. Most of the work on this project was the physical customisation part.

"The software side is easy since there are a numbers of solutions out there that have already been proven to be successful. The majority of the work I've done is with the physical part, and is easily 90-95% of the time invested."

Just about everything he could connect through the SNES has been done in that way, and just about everything uses the original port locations. USB and Ethernet are routed through the two front controller ports, the HDMI is in the old AV out, the power has been converted, and an on/off switch has been fitted into the aerial connector. That's not the best part though:

"In my design, I've moved the SD card from the Raspberry Pi and connected it inside an actual game cartridge. I've noticed a few more failed boots than normally would be expected. Other than that, it works beautifully!"



Left Mark interfaced directly with a microSD card converter to allow booting from the cartridge

Image: Evan Amos CC BY-SA 3.0

A common sight in '90s households – but this SNES holds a secret

A Nintendo console needs Nintendo controllers, surely? "I have two ideas," Mark tells us. "One is to take an original USB controller that works natively with the Pi and cram that into a controller housing. The result would look like the original controller. The other would be to map the current controller to the USB spec that the Pi expects. I haven't done a lot of research on this approach; however, figuring out these kinds of problems is the fun part of tinkering with gadgets."

> Figuring out these problems is the fun part of tinkering with gadgets

While the internet likes to go a little bit mad whenever someone posts a new classic console mod like this, it sounds like this is nothing compared to the joy of actually doing it:

"I'm always amazed at the beginning of a project like this, that you have a workbench full of parts that by themselves do little or nothing. Then at some point while putting the parts together, something new and useful is created and essentially 'comes alive'."

POWERING UP THE SNES Pi CASE

>STEP-01
Jack in

Plug your HDMI, power, Ethernet and USB devices into the various ports – the controller connectors hold the latter two, in case you were wondering.

>STEP-02
Grab a game

Search through the game library for *F-Zero*. Make sure it's the right version of *F-Zero*, the one with an SD card of Raspbian on it. Slot it in the top of the SNES.

>STEP-03
Flick the switch

Unfortunately, the original power button won't help you here. Behind the SNES and next to the power cable is the on/off button – flick it to bring the SNES Pi to life.

SIMON LONG

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 *really difficult crosswords*.
raspberrypi.org

UPDATING THE DESKTOP

To improve the Raspbian user experience and give the Pi its own identity, the Foundation has begun to work on customising and improving the desktop. **Simon Long** tells us more…

The term 'user interface' covers two aspects of software. First, its appearance – does it look good, does it draw you in and make you want to interact with it? Second, its behaviour – does it work well, is it intuitive and logical? I've been working on improving both of these for Raspbian's default desktop environment, with the first changes appearing in the December 2014 release.

The Raspbian desktop is an X Window system called LXDE (Lightweight X Desktop Environment). There are many good things about it, the most important from the point of view of the Pi being its low usage of memory, disk space, and processor. It is also very customisable, even without changing a line of code.

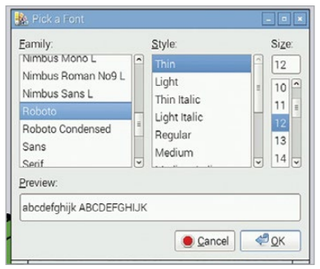
However, the default appearance of LXDE on Raspbian was a bit dated and unfriendly-looking, and I wanted to improve on that: to provide a desktop that looks more like those people used to Mac OS X or Windows might expect.

Starting small

I started out with some minor tweaks to the appearance; I added a new font (Google's Roboto, which is also used on the Pi website), cleaned up the colours and decluttered by removing seldom-used taskbar and desktop icons. I also chose a different icon theme, in which the icons are less complex graphically and therefore easier to understand. All these are aimed at making the interface nicer to look at and less intimidating when you first see it.

For me, the more interesting part of UI design is the second one I mentioned above: making it intuitive. This is where some basic psychology comes in. User interface design is mostly about applying consistency – you get used to the way something works and if something else works slightly differently, it jars. (As a designer, you can use that to your advantage sometimes, for drawing attention to something, but you don't want it happening all the time!)

Some of the changes are quite subtle. For example, when you move the mouse pointer over the menu bar at the top, everything now highlights in the same colour – previously, hovering the mouse over something on the menu bar had different effects on different items. This helps to add consistency, to make the desktop behave the way you expect.



Above The new Roboto font (as used on **raspberrypi.org**) gives a more modern and unified appearance to labels and menus

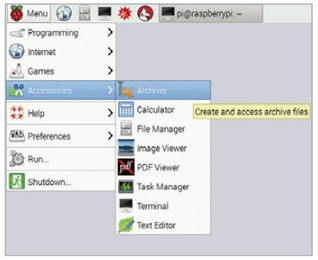
SIMPLIFYING RASPBIAN

In coming releases, some of the more complex settings – for things such as connecting to Wi-Fi networks – are going to be simplified, all with the intention of making the Pi feel as good to use as a PC or a Mac. Watch this space…

RASPBIAN DESKTOP THEN & NOW

Out with the old: The original appearance of LXDE on Raspbian prior to December 2014

In with the new: A cleaner appearance, new colours and font, and a revised menu bar



Above The menu design is now much cleaner, with new icons, clearer labelling and tooltips for applications

> "The intention is to make it easier for new users to get to grips with Raspbian"

Taskbar placement

A big change was to move the taskbar up to the top of the screen. I know this is contentious, as people are used to the taskbar being at the bottom. The reason is simple: the main application menu button is on the taskbar. At the point when you click on the menu button, your eyes are on the mouse pointer. If the menu then drops down, your eyes will naturally read downwards to find the item you want. If the menu rises up, not only do you then have to read in the 'wrong' direction – from bottom to top – but generally the items nearest the bottom of the menu are the ones (like Shutdown) that you click least often, so you need to move the mouse pointer further more of the time.

Another fundamental of UI design is MECE – 'mutually exclusive, collectively exhaustive'. In other words, the menu should have everything you need in it, but each thing should only be in one place. This is why I removed the 'Other' category from the menu – it was a catch-all that didn't really hold 'other' stuff; it actually held everything, which is why it was so huge. Many things in it had confusing names (and if you tried selecting them, many didn't even work).

It's bad UI design to use long lists, because it's hard to find what you are looking for in them, so I removed the 'Other' category and tried to bring a bit more MECE-ness to the remaining menu categories. I also changed some of the names and tooltips, to make it a bit easier to understand what is actually in the menu.

The intention with all the changes is to make it easier for new users to get to grips with Raspbian, and for them to feel comfortable with how it works as quickly as possible. Some of these changes may seem awkward to experienced users; if you don't like them, it's pretty straightforward to undo most of them.

PERSONALISE YOUR PI

This is just the start of the UI work planned for Raspbian. There's much more in the pipeline, including a new configuration application to enable you to customise the colours, desktop picture and more, so you can make your Pi look the way you want it to. There are going to be other graphic tweaks too, including a new set of icons we are having custom-designed.

MAKERS: THE NEXT GENERATION

The Raspberry Pi is inspiring a new generation to learn how to hack and make amazing projects. We chat to four young makers about their impressive creations and achievements, and why they do them with the Raspberry Pi…

Zach

Name: Zachary Igielman
Age: 14
Location: London
Studying: 12 GCSEs
Twitter: @ZacharyIgielman

After teaching himself to code in Visual Basic at just 11 years old, Zach moved on to Objective-C and released several apps on the iOS App Store. Since discovering the Pi, he's learnt to code in Python and has built his own autonomous robots, enhancing them with a variety of sensors. He also helped 4tronix to develop the Pi2Go. A regular Raspberry Jam attendee, he has run many different workshops. Remarkably, Zach recently conducted a successful crowd-funding campaign on Indiegogo for his PiPiano musical add-on board (**bit.ly/1wsBmci**), raising nearly twice the original goal.

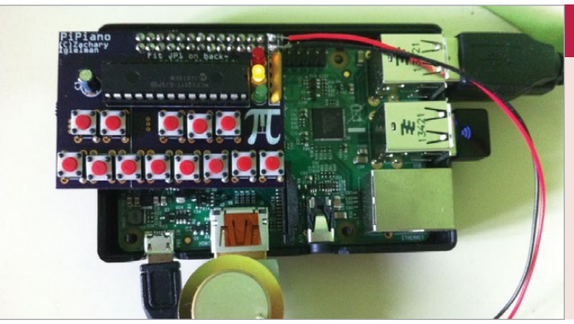
Tell us about some of the Pi projects you've worked on.
Along with my robots, I've done some cool work with the camera module: I programmed my Pi to do time-lapses. I made a tutorial for using an accelerometer with the Pi. I also ran a sensors workshop teaching line and distance sensing with my own materials.

Elsewhere I've been working on a self-balancing robot, which led to giving a talk on PID control theory for robotics at a Raspberry Jam… I have tested all sorts of Pi add-on boards, from LED to analogue, leading me to build my own.

How did the idea for the PiPiano come about?
I wanted to create a piano with my Pi, but adding switches to a real piano would be a wiring nightmare… My solution was to create a simple, piano-style add-on for the Raspberry Pi, which includes buttons in a piano octave formation and a piezo transducer for sound output. After trying it on a breadboard, and being swamped with bundles of spaghetti wiring, I opted to lay out the idea on a PCB (printed circuit board).

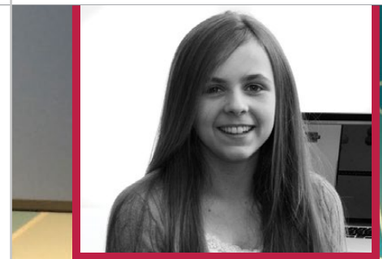
PiPiano teaches programming, soldering and electronics, through reading the buttons, making traffic lights with the LEDs, playing a scale on the buzzer, and finally making a PiPiano. It uses a special chip so all 17 components connect to three [GPIO] pins.

What is it you love about the Raspberry Pi?
It's really cheap; I can afford a few of my own and I'm not always worrying about breaking an expensive computer. It's portable, easy to ferry from event to event. It's got an amazing community, which enables me to meet cool people, learn lots and have amazing opportunities. It has lots of documentation and a great forum for learning anything with it. It is very programmable [and] it can plug into electronics, unlike most computers.



BIG PROJECT: PIPIANO

Plugging directly into the top of a Pi, this musical add-on board features 13 buttons in a piano key formation, a piezo transducer for sound output, and three LEDs. Designed to be educational, it comes with documentation which takes you from the basics of soldering the board and setting up the software, to programming a fully-working piano at the end. PiPiano comes soldered (ready-made) or as a kit, and with either a standard or stacking header. When not used as a piano, it's also a handy controller with an ample supply of buttons which could be used in automation projects and so on. Learn more at **pipiano.com**.



Amy

Name: Amy Mather
Age: 15
Location: Manchester
Studying: 9 GCSEs (already has an A* in Computing)
Twitter: @minigirlgeek

Following an invite from Jimmy Wales, Amy presented a keynote speech at the Campus Party EU 2013

Already into electronics at the age of 12, Amy integrated an Arduino kit into a model volcano for a school homework project, which she was then asked to demonstrate at Manchester's first Mini Maker Faire. After getting hold of a Pi, she learnt Python and created her own version of Conway's Game of Life, even outputting the display to an LED matrix. Most notably, Amy teaches both adults and children to code and works closely with the STEM network to inspire other young people to get involved in computer science. This has led to her giving keynote speeches at many prestigious technology events. At ICT 2013 in Lithuania she received an award as the European Digital Girl of the Year.

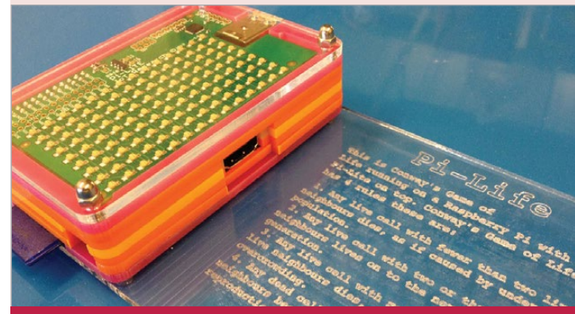
How did you get started with programming?
I first got interested in coding when I was about 12, following a Manchester Girl Geeks workshop that I attended that was an introduction to JavaScript using Codecademy. I thought Codecademy was an amazing platform for learning how to code and I continued with their courses (you can find out more at **codecademy.com**).

What's so great about the Raspberry Pi?
It doesn't matter if you accidentally blow bits up on it – you can get another one! Or if the SD card corrupts, it's not the end of the world: you can reformat it. The Pi allows you to make all kinds of remarkable projects and there are so many awesome add-on boards. Also, the community's really welcoming and friendly, so they're open to any of your questions. I think the whole environment is just amazing.

Do you have any new Pi-based projects planned?
My school has just asked me if I have any ideas for ways that we can link the coding club and the STEM club. So I'm thinking about how to help them through the use of Raspberry Pi-based projects. I've helped out with teaching coding workshops and I'm currently leading a series of soft electronics workshops as the volunteering section of my Silver Duke of Edinburgh Award.

And you also make speeches at numerous events?
Yes, I speak at quite a lot of conferences, about how we can get more kids involved in STEM and why it's really important that we do so.

What advice would you give to other young coders?
Just get involved, find out where all the local events are, and get involved with the community; you'll definitely learn a lot more from talking with other people who are interested in similar things, rather than just sitting alone at home and doing it by yourself.



BIG PROJECT: PI-LIFE

Amy first came to the attention of the international Raspberry Pi community after giving an impressive presentation of her Python version of Conway's Game of Life – a zero-player game simulating cellular replication – at the 2013 Manchester Raspberry Jamboree (**raspberrypi.org/amys-game-of-life**). In it, she enthused about her love of coding and detailed how she developed various implementations of Life, including one with the Pi outputting the resulting patterns to an 8x8 LED matrix via a connected Arduino Mega. Since then, she's created a more compact version using a Pi-Lite LED add-on board.

Right Lauren created a portable Pi-powered system to sync the lights in Charles Peachock's juggling clubs to a music track

Lauren

Name: Lauren Egts
Age: 15
Location: Stow, Ohio
Studying: High School (Hathaway Brown)
Twitter: @laurenegts

When she was just nine, Lauren's father taught her how to write some Bash scripts. Using Scratch on the Pi, she later created The Great Guinea Pig Escape game and demonstrated it at a local Maker Faire. A long-time member of Akron Linux User Group, she has presented talks on GlusterFS and teaching kids to code on the Pi. In 2013 she was a National Center for Women in Technology Ohio Affiliate Award Winner, and is also an intern at the NASA Glenn Research Center G-VIS Lab. Recent projects include creating a Pi video wall (at NASA) and designing a portable LED lights system for professional juggler Charles Peachock.

What do you especially like about the Pi?

One thing that I really love is its versatility. I've used it in a variety of projects, and seen it used in even more! The Pi is so small that it can be used practically anywhere, which means it can be used in [so many different] projects.

What was it like being an intern at NASA? How did it happen?

It happened at the Cleveland Mini Maker Faire. I had my booth where I was presenting on Scratch and the Raspberry Pi. I was helping a friend take her booth out to her car, and my dad shows up with some other guy who I later found out was Herb Schilling, now my mentor at NASA. Turns out Herb had gone to my booth while I was away, and my dad had told him all about me. When Herb and I met, he was so impressed with what I had done with that Pi that he invited me to shadow him at NASA. After my shadow day, Herb invited me to come back for a few weeks over the summer! We figured out some dates, and that's how my internship happened!

Are you planning to do any more Pi projects when you return to NASA?

It depends on what projects I am assigned to but I would absolutely like to finish my work on the Pi video wall this summer. Herb is very interested in showcasing the power of low-cost computing devices like the Pi. Finishing my work will require taking care of a few bugs in code, as well as fixing the aforementioned hardware issues. After the video wall project is done, hopefully it will be displayed outside the G-VIS Lab, playing a video that explains what the lab does when people walk by.

BIG PROJECT: PI VIDEO WALL

During her internship at NASA, Lauren – working with fellow intern Nick Patterson – did a proof-of-concept project to create a Pi-powered video wall (**1.usa.gov/1DYEa2v**). Since they only had access to what was in the NASA G-CVIS Lab at the time, different-sized monitors were used, but the end result was still impressive. The setup involved connecting four Pis to a master computer via a router. The PiWall software package (**piwall.co.uk**) was used to split up the video display into four tiles, one for each monitor. Lauren hopes to improve the setup when she returns to NASA this summer.

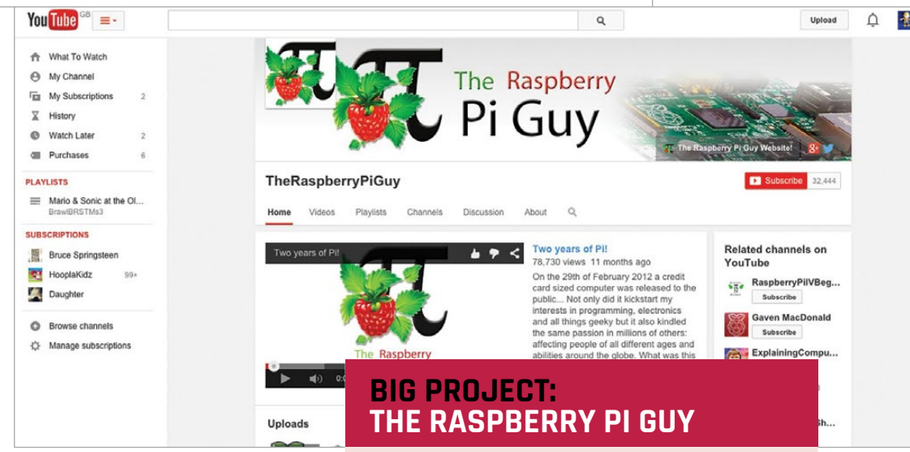
Matt

Name: Matt Timmons-Brown
Age: 15
Location: Bartlow, near Cambridge
Studying: 11 GCSEs
Twitter: @RaspberryPiGuy1

Better known as The Raspberry Pi Guy, Matt runs a YouTube channel dedicated to Raspberry Pi video tutorials. Amazingly, he only started coding and making three years ago, upon discovering the Pi Foundation's credit card-sized PC. So far he's created projects of varying complexity, including a Pi-controlled model railway, but his real passion is robotics – in particular, making two-wheeled robots and pushing their abilities to the max. His latest project involves building an accessible robotics platform, with a complete set of learning materials, using 4tronix's Pi2Go-Lite. To this end, he's filming a YouTube series called 'Raspberry Pi Robots', for which he's just released the first tutorial.

How did you first get involved with making?

I am the quintessential Raspberry Pi product: a schoolboy who now has a love of computer science because of the Pi... In the summer of 2012 I managed to get my hands on my first Pi and ever since then I have been in love with programming, making (robots!) and computer science. The Pi opened my eyes to computing and for that I am incredibly grateful!



Why did you decide to set up your own YouTube channel?

I am a very recent convert to the ways of Pi. As a result, I originally found computing a fairly hard subject to get into; there is all of this foreign jargon and sometimes it can be incredibly confusing... In September 2012 I realised that I had built up a considerable amount of knowledge on the subject and I thought it would be a great side project to teach people some of the stuff I'd had so much fun learning. I turned to YouTube as a way of doing this because I found the most easy way to learn something is by watching someone go through something step by step... On 1 September 2012, The Raspberry Pi Guy was born and I have been publishing videos ever since; I am just about to hit the 2 million view barrier on YouTube, something I never imagined!

What's so great about the Pi?

There is no other product out that there has the same ethos: to teach people about computing by introducing them to a [new] experience. After all, how many single-board computers has the average person seen? How many terminals have they programmed in? Scratch that, how many people have actually programmed?! The Pi is a gateway to the world of computers and [has] introduced me to a lifetime interest... It has inspired millions and continues to do so.

BIG PROJECT: THE RASPBERRY PI GUY

Matt's YouTube channel (**youtube.com/user/TheRaspberryPiGuy**) has been running for over two years now and has proven immensely popular, amassing over 32,000 subscribers. "Dedicated to teaching the masses how to make the most of their Raspberry Pi computer", it provides a plethora of step-by-step video tutorials. These range from basic setup to attaching various add-ons and creating numerous projects – including, of course, robotics. The latest addition is the Raspberry Pi Robots series, which Matt hopes "will engage people in computer science through the most exciting medium: world-conquering robots."

Get making!

Our young experts offer a lot of good advice for how to get started with coding and making...

- There are lots of free online resources, such as Codecademy, to help you learn to code. Just Google 'coding courses'.
- Look out for local events and get involved – it's much more fun than trying to do it alone at home!
- Events such as Maker Faires and Raspberry Jams can give you ideas to try and a place to start exhibiting your projects and make contacts.
- Join a robotics team if you're interested in making robots and entering them into competitions.
- Even if something seems hard to start with, keep trying: if you set your mind to it you'll achieve it. And you're never too young to start!
- However, remember to always have fun with what you're doing. If you don't like something, don't waste your time with it.

EVERYDAY ENGINEERING PART 1

SIMON MONK

Simon Monk is the author of the *Raspberry Pi Cookbook* and *Programming Raspberry Pi: Getting Started with Python*, among others.
simonmonk.org
monkmakes.com

BUILD YOUR OWN

PARKING SENSORS

You'll Need

- Half-size breadboard
- 7x male-to-female jumper wires
- 2x male-to-male jumper wires
- 4x 470 ohm resistors
- 2x HC-SR04 ultrasonic sensors
- Code: **bit.ly/1KutV7K**

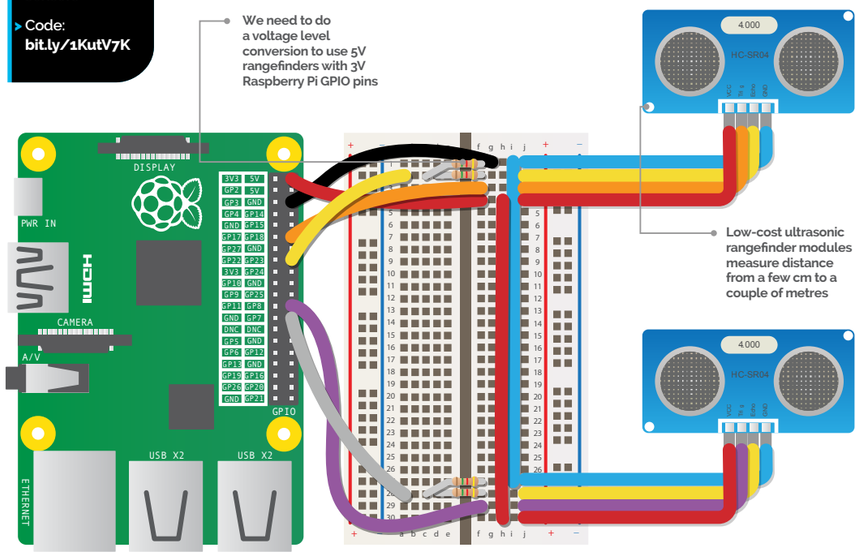
Solve real-world electronic and engineering problems with your Pi and the help of renowned technology hacker and author, **Simon Monk**

Electronics permeates every aspect of modern life and it's easy to take such technology for granted without ever stopping to think just how these things work. Small, low-cost computers like the Raspberry Pi make it possible for hobbyists to put their own take on commercially available products, and also invent new gadgets simply for the fun of it.

In this series, we will be exploring the use of the Raspberry Pi to make all kinds of everyday electronic devices, starting with an ultrasonic parking sensor designed to show you how far the rear corners of your car are from any obstacle.

Each of these projects will be constructed using a solderless breadboard and readily-available components, so even if you don't want to develop and install these projects for real, you can prototype them to learn more about engineering and electronic invention.

As you'll see from the list of required components nearby, our first project uses two low-cost ultrasonic rangefinders to measure the distance from the sensor to any obstacle in its path.



While you could attach the rangefinders to your car bumpers, with a sensor at each of the two rear corners of the car and a display positioned so that it is visible from the driver's seat, you could also place the sensors on the wall of your garage (assuming yours is not full of rubbish), so that the display can guide you in and tell you when to stop.

The distance to any obstacle for the right and left sensors is indicated by a rectangle that extends further down the screen as the distance to an obstacle increases. In addition, the actual distance to the obstacle is displayed in cm

Left The HC-SR04 Ultrasonic Rangefinder. Cheap-as-chips sonar

and the rectangle is colour-coded: red if closer than 30cm, green if greater than 100cm, and orange if it's in between.

The ultrasonic rangefinders are of the type that you can buy on eBay, sometimes for less than a pound. These sensors are often used in robot projects to detect obstacles. They use pulses of sound waves to measure the distance to an obstacle over a range of a few cm to several metres. Just search for HC-SR04 and remember to order two.

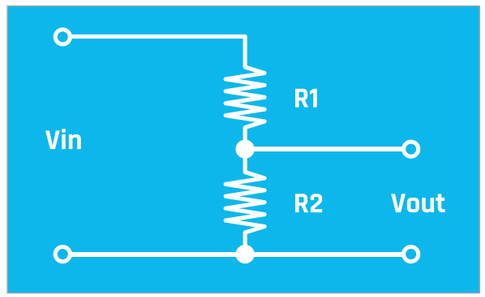
The HDMI display is only needed if you plan to install the project for real in your car or garage; otherwise, you can just use your usual Raspberry Pi monitor. Again, you will find mini HDMI displays at a very reasonable price on eBay. The model we used had a 7-inch display and separate controller board. Look for a display that will operate from 12V if you are going to connect it to your car.

The other parts are probably best bought as an electronics starter kit. The Monk Makes Electronic Starter Kit for Raspberry Pi includes the breadboard and all the parts and wires except the rangefinders. Most starter kits for the Raspberry Pi will include the breadboard, jumper wires and some resistors.

Why use four resistors?

A Raspberry Pi's GPIO pins operate at 3.3V, whereas the rangefinder module's pins operate at 5V. This does not cause a problem when connecting the output of the Raspberry Pi to the input of the rangefinder (for example, a GPIO output on the Raspberry Pi to the Trigger input on the rangefinder) because even though the voltage at the input is a bit low, at 3.3V it will still be high enough to activate the trigger input.

The problem arises when you are going in the opposite direction and the 5V Echo output of the rangefinder needs to connect to a GPIO input on the Raspberry Pi. Putting 5V on a GPIO pin only designed for a maximum of 3.3V could easily damage the pin. Therefore, we use an arrangement of two resistors to reduce this voltage from 5V to 2.5V, where it will still be high enough to register as a high input, but still be well below the maximum of 3.3V.



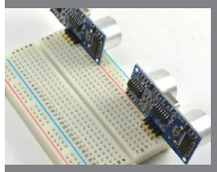
Above 5V in, 3V out – protecting your Raspberry Pi's GPIO pins

More advanced readers may prefer to use different combinations of resistor to set the voltage a bit closer to 3.3V, but the advantage of just halving the voltage is that all four resistors can be of the same value.

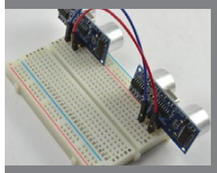
Building your parking sensor

Even if you plan to install the project for real, it's a good idea to start with the rangefinders plugged directly into the breadboard, with the ultrasonic transducers pointing outwards. This will allow you to experiment with the project and make sure that everything is working as it should be, before you commit to some more permanent setup.

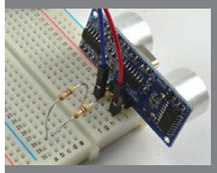
PROTOTYPING THE PROJECT



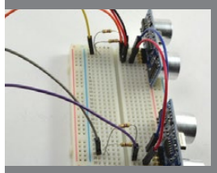
>STEP-01
Fit the rangefinders onto the breadboard
We consructed the prototype build of this project in five steps. First, plug the rangefinders into the breadboard holes at the far ends of the breadboard, as shown in the picture.



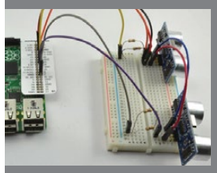
>STEP-02
Join the power connections
Use a male-to-male jumper wire to connect the 5V (labelled Vcc on the rangefinder) pins together, by plugging the jumper wires into the same row as the pins. Do the same thing for the GND connection.



>STEP-03
Add the resistors
Push the leads of the four resistors into the breadboard, as shown in the diagram on the left page. It doesn't matter which way around they go, but be careful that the leads don't touch each other.



>STEP-04
Add the male-to-female jumper wires
Attach the male-to-female jumpers that will connect the breadboard to your Pi. Colour-coding the leads will help you to identify which connection is which when you attach it to your Raspberry Pi.



>STEP-05
Link the breadboard to the Raspberry Pi
Finally, connect all the female ends of the headers to the GPIO pins on your Pi. Working out which can be tricky, but you could use a template like the Raspberry Leaf or the Pi GPIO Reference Board.

Now that the hardware side of the project is done, we need to get the software running. The program is written in Python, using the Pygame library to provide graphics. You can download the program from the internet by typing the following into the command prompt:

```
git clone https://github.com/simonmonk/pi_magazine.git
```

> The Python code for this project is very well commented on GitHub

The program has a graphical user interface, so to run the program, the windowing system must be running. If your Pi is not set to automatically boot into the windowing system, then type the following command to start it up after you have logged in:

```
startx
```

Open an LXTerminal window and type the following commands into it to run the program:

```
cd pi_magazine
sudo python 01_parking_sensor.py
```

After a short delay, the Pygame window will appear.

Try putting your hands in front of each sensor in turn to make sure they are both working okay. If one isn't, check over your wiring carefully.

How the code works

The Python code for this program is very well commented on GitHub (**bit.ly/1KutV7K**), so you'll probably find it handy to have the code up in an editor while we go through it.

The program starts by importing the Python libraries that will be used, and some constants for the GPIO pins. So, if you wanted to swap things around and use different pins, you could just change the numbers to the right of the equals signs. Variables are also used to define the colours that will be used in the user interface by Pygame.

After this, we have some code that initialises the four GPIO pins we want to use. Two of them are set to be outputs, so that they can send out a pulse that causes the rangefinders to send out an ultrasound 'ping'. The other two pins (the 'echo' pins) are set to be inputs, as we need to be able to read them in the program so that we know when the echo has returned, and therefore how long the delay was, so we can calculate distance.

The next three functions contain all the code relating to measuring distance using an ultrasonic rangefinder. The first of these (**send_trigger_pulse**) outputs a pulse of just 0.0001 seconds on the pin supplied as its parameter. This will cause the rangefinder to send out a pulse of ultrasound. The next function (**wait_for_echo**) is responsible for waiting until the echo from that pulse of ultrasound is received, so that the distance can be calculated by the length of time it took for the echo to arrive.

The function **get_distance** puts all this together, first sending a trigger pulse and then timing how long it takes for the echo to arrive. Actually, it's slightly more complicated than that, because the echo cannot be detected until after the pulse has finished sending. If we check too early, we will get a false reading. That is why there are two calls to **wait_for_echo**. The first waits until sending of the pulse is complete and the second actually times the delay until the echo arrives.

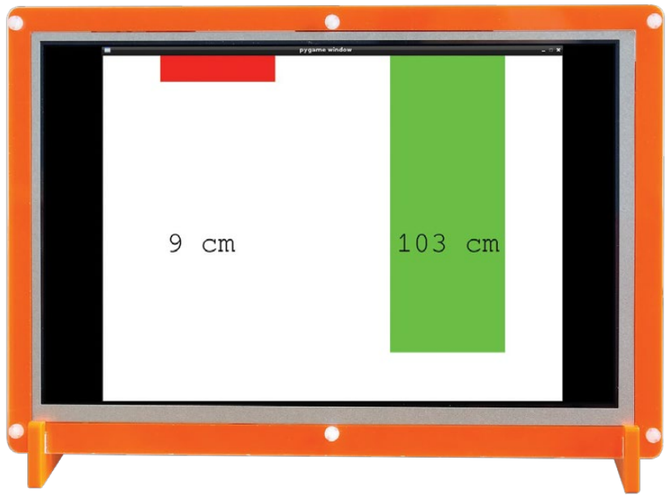
Graphical user interface

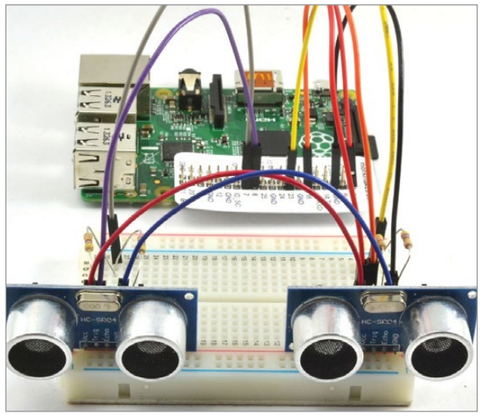
The rest of the code is concerned with the user interface for the project. The function **colour_for_distance** returns a colour to use when drawing the rectangle for that sensor, depending on how large the distance detected is.

The next few lines initialise the Pygame window and define a font to use for the distance readout. While

Below right The complete prototype

Below You could use the affordable HDMIPi screen in your garage





Pygame is designed primarily to make games, it's excellent for any project that uses graphics, like this one. You will likely want to alter the **width** and **height** variables to match the resolution of your display.

The **while** statement starts the main loop. The program will keep looping around the instructions inside, from **while** until the end of the file, until the program window is closed – this kind of loop is known as an infinite loop.

Each time around the loop the Pygame events are checked, and if the Pygame window has been closed (by clicking the little cross in the corner of the window), then the GPIO pins are set to a safe input mode using the **GPIO.cleanup** function and the program exits.

Most of the time the window will not have been closed, so the remainder of the loop will measure the two distances from the rangefinders and then draw rectangles on the screen, using the distance readings to set the height of the rectangle. The height of each rectangle is the distance in cm multiplied by 5 pixels.

Finally, there is a delay of half a second to stop the distance figures updating too fast to read clearly.

Using your parking sensor

Although you can extend the leads to the ultrasonic rangefinder by perhaps a metre or so, any longer than that and you are likely to have problems with the signal. So, if you are installing this project for real in a vehicle, it may be better to site the Raspberry Pi fairly near the sensors and use a longer HDMI lead to connect the Raspberry Pi to the display.

If you are installing this project for real, then you will probably want to make the program start automatically. That way, you don't even need to have a keyboard and mouse attached to the Raspberry Pi. You can find links on how to do this on the Raspberry Pi Forum: **bit.ly/1DaEURv**

The ultrasonic rangefinders are great little devices. You can take the range-finding part of the program for this project and use it in lots of other projects. You could, for instance, use it to just make a distance meter, perhaps displaying the distance in inches or cm. You could also use it to create a theremin-like musical instrument that changes the pitch of the note, depending on the distance of your hand from the rangefinder.

NEXT MONTH

In the next project of this series, we will turn our attention to making a web-controlled door lock that lets you unlock your door remotely.

01_parking_sensor.py

Language >PYTHON

DOWNLOAD: bit.ly/1KutV7K

```
import RPi.GPIO as GPIO
import time, sys, pygame

trigger_pin_left = 8
echo_pin_left = 7
trigger_pin_right = 18
echo_pin_right = 23

green = (0,255,0)
orange = (255,255,0)
red = (255,0,0)
white = (255,255,255)
black = (0, 0, 0)

GPIO.setmode(GPIO.BCM)
GPIO.setup(trigger_pin_left, GPIO.OUT)
GPIO.setup(echo_pin_left, GPIO.IN)
GPIO.setup(trigger_pin_right, GPIO.OUT)
GPIO.setup(echo_pin_right, GPIO.IN)

def send_trigger_pulse(pin):
    GPIO.output(pin, True)
    time.sleep(0.0001)
    GPIO.output(pin, False)

def wait_for_echo(pin, value, timeout):
    count = timeout
    while GPIO.input(pin) != value and count > 0:
        count -= 1

def get_distance(trigger_pin, echo_pin):
    send_trigger_pulse(trigger_pin)
    wait_for_echo(echo_pin, True, 10000)
    start = time.time()
    wait_for_echo(echo_pin, False, 10000)
    finish = time.time()
    pulse_len = finish - start
    distance_cm = pulse_len / 0.000058
    return int(distance_cm)

def colour_for_distance(distance):
    if distance < 30:
        return red
    if distance < 100:
        return orange
    else:
        return green

pygame.init()
size = width, height = 800, 600 # the variables alter window size
offset = width / 8

screen = pygame.display.set_mode(size)
myfont = pygame.font.SysFont("monospace", 50)

while True: # the main loop starts here
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            GPIO.cleanup() # set GPIO pins to be inputs
            sys.exit() # quit the program entirely

    left_distance = get_distance(trigger_pin_left, echo_pin_left)
    right_distance = get_distance(trigger_pin_right, echo_pin_right)

    screen.fill(white)
    pygame.draw.rect(screen, colour_for_distance(left_distance),
                     (offset, 0, width / 4, left_distance*5))
    pygame.draw.rect(screen, colour_for_distance(right_distance),
                     (width / 2 + offset, 0, width / 4, right_distance*5))

    left_label = myfont.render(str(left_distance)+" cm", 1, black)
    screen.blit(left_label, (offset + 10, height/2))
    right_label = myfont.render(str(right_distance)+" cm", 1, black)
    screen.blit(right_label, (width/2 + offset + 10, height/2))

    pygame.display.update()
    time.sleep(0.1)
```

MARTIN O'HANLON

Martin 'Minecraft' O'Hanlon is an active member of the Raspberry Pi community, co-author of *Adventures in Minecraft* and keeps an excellent account of his projects on his blog.
stuffaboutcode.com

You'll Need

- Raspbian
- Minecraft: Pi Edition
- Python 2 editor (IDLE)
- Getting started with Minecraft: Pi Edition: **bit.ly/1Ep9LKC**

MINECRAFT: Pi CODING TIPS

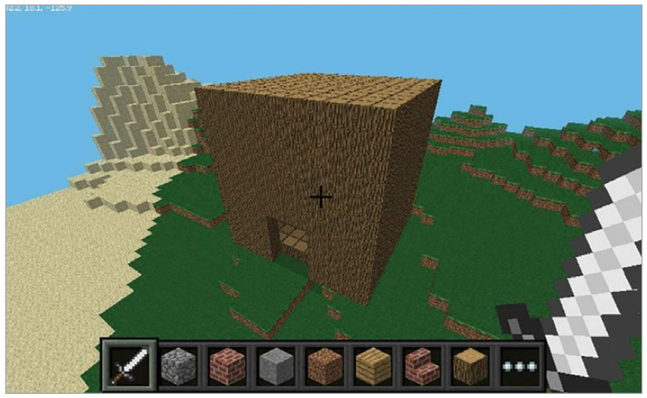
If you've completed the Minecraft Pi learning resources at **raspberrypi.org**, check out these pro tips and mini programs to learn more about the coding in Minecraft...

BUILD A HOUSE

The quickest way to make a house in *Minecraft: Pi Edition* is to use code and the API. By programming a house rather than building it by hand, it can be any size you want – 10 blocks across or 100!

Create a simple program which will use the **`setBlocks()`** function, once to create a cube 10×10×10 of wood (17) and then again to create a cube of air (0) 9×9×9 inside the wooden cube.

Below Create massive houses in the blink of an eye using just a few lines of code



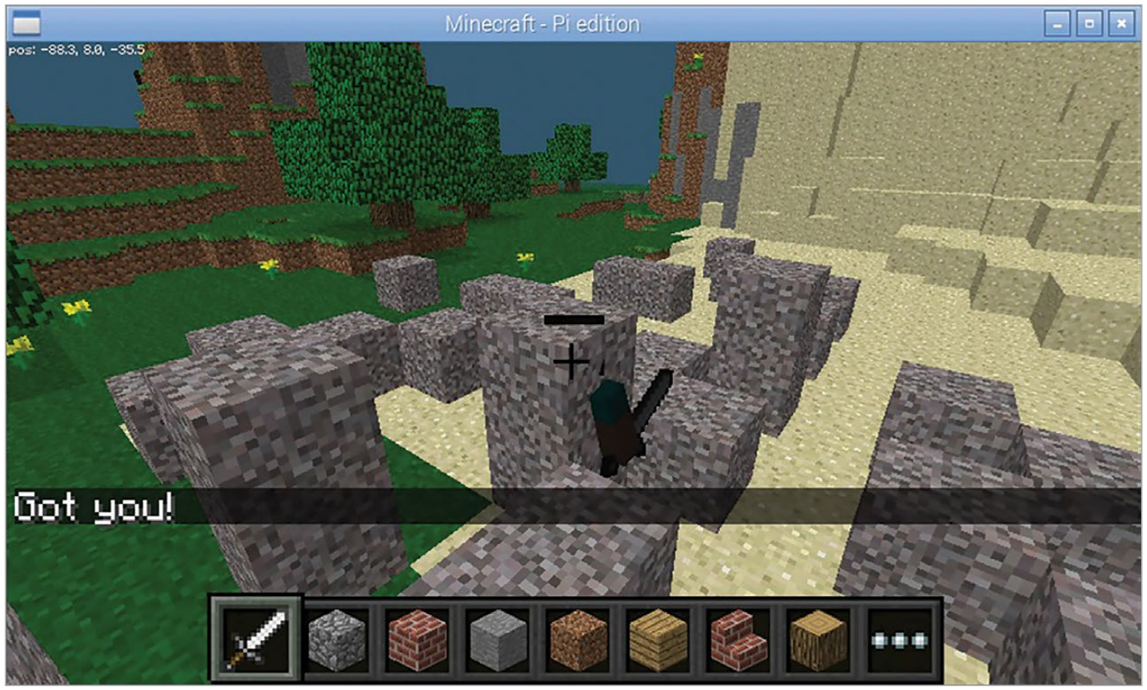
```
from mcpi import minecraft
mc = minecraft.Minecraft.create()

pos = mc.player.getTilePos()

mc.setBlocks(pos.x + 0, pos.y + 0, pos.z + 0,
       pos.x + 10, pos.y + 10, pos.z + 10, 17)

mc.setBlocks(pos.x + 1, pos.y + 1, pos.z + 1,
         pos.x + 9, pos.y + 9, pos.z + 9, 0)
```

You can then use **`setBlocks()`** again to create an entrance by building another block of air (0).

```
mc.setBlocks(pos.x + 4, pos.y, pos.z,
      pos.x + 6, pos.y + 3, pos.z, 0)
```

The limits of coding a house are endless – why not add a stone roof, a wool floor, some torches to the outside?

Below Use blocks affected by gravity such as gravel to create your own *Minecraft* mini-game



> Using the gravity effect of blocks is a great way to add something new to your Minecraft programs

USE GRAVITY-EFFECTED BLOCKS

Sand and gravel block types in *Minecraft* are affected by gravity and will fall down if the block below is air.

The same gravity effect occurs if a block is placed in the world using the API. So if you were to create a block of gravel (13) 25 blocks above the player, it would fall on the player's head. In a new program, type:

```
from mcpi import minecraft
mc = minecraft.Minecraft.create()
pos = mc.player.getTilePos()
mc.setBlock(pos.x, pos.y + 25, pos.z, 13)
```

Using the gravity effect of blocks is a great way to add something new to your *Minecraft* programs. Here is a simple program which loops until it manages to drop a gravel block onto Steve's head. If Steve wants to stay in the game, he has to keep moving so the gravel misses him. Start a new program:

```
from mcpi import minecraft
from time import sleep
mc = minecraft.Minecraft.create()

pos = mc.player.getTilePos()

while mc.getBlock(pos.x, pos.y, pos.z) != 13:
    mc.setBlock(pos.x, pos.y + 25, pos.z, 13)
    sleep(1)
    pos = mc.player.getTilePos()

mc.postToChat("Got you!")
```

Right Change the position of the 'camera' in *Minecraft* and get a different view of the world

CHANGE THE CAMERA

Bored of always following Steve around? You can alter the position of the 'camera' in *Minecraft* to change how you see the world.

You can change the camera to follow Steve while looking directly down at him, or to look down at the world from any coordinate in *Minecraft*.

The **camera.setFollow()** function will change your view so you are looking down at Steve. In a new script:

```
from mcpi import Minecraft
mc = minecraft.Minecraft.create()

mc.camera.setFollow()
```

To change the camera to look down on any position, you use the **camera.setFixed()** function before using **camera.setPos()** to change the position of the camera. If you wanted to set the camera 25 blocks above the spawn position, you would use:

```
mc.camera.setFixed()
mc.camera.setPos(0,25,0)
```

To set the camera back to normal, you would use the **camera.setNormal()** function.

```
mc.camera.setNormal()
```

Using the camera functions, you could hide a diamond block (57) in the world, then tease the player by changing the camera to show them where it is before challenging them to find it. Try this in a new program:

```
from mcpi import minecraft
from time import sleep
mc = minecraft.Minecraft.create()

mc.postToChat("Here is the diamond block I have hidden.")
mc.setBlock(100,25,100,57)
mc.camera.setFixed()
mc.camera.setPos(100,30,100)
sleep(10)

mc.postToChat("Go find it!")
mc.camera.setNormal()
```

You could change the program above to drop the diamond block in a random position and use **getHeight()** so the diamond block is always on the top of the world.

MAKE THE WORLD 'READ-ONLY'

Are you fed up with Steve having free rein to destroy your beautifully crafted world? Or would you prefer it if lava didn't burn down your creation?

Using the **setting()** function in the API, you can make your world 'immutable' – something which is unable to be changed. Start a new script with:

```
from mcpi import minecraft
mc = minecraft.Minecraft.create()

#make the world read-only
mc.setting("world_immutable", True)
```

> "Would you prefer it if lava didn't burn down your creation?"

Now, the only way you can change the world is through code – any attempt to place or destroy blocks in the game won't work.

You can make your world writable (or mutable) again by making the setting **False**.

```
#make the world writable
mc.setting("world_immutable", False)
```

You can use this setting to create a new script which will pit your building skills against a friend, giving you 1 minute to make the best building you can before turning the world read-only again.

```
from mcpi import minecraft
from time import sleep
mc = minecraft.Minecraft.create()

mc.setting("world_immutable", True)

mc.postToChat("In a moment you will have 1 minute to create the best building.")
sleep(10)
mc.postToChat("Go")

mc.setting("world_immutable", False)
sleep(60)
mc.postToChat("Stop - Who's is the best?")
mc.setting("world_immutable", True)
```

LEARN THE HEIGHT OF THE WORLD

If you want to code structures to always be 'on top' of the land, you need to know how high the world is – or, put another way, how far the air comes down!

In *Minecraft* the height is the Y coordinate, while X and Z are the horizontal dimensions – if you pass X and Z coordinates to the API function **getHeight()**, it will return the Y coordinate. In a new program type:

```
from mcpi import minecraft
mc = minecraft.Minecraft.create()

y = mc.getHeight(0,0)

mc.postToChat("Height of the world at spawn is")

mc.postToChat(y)
```

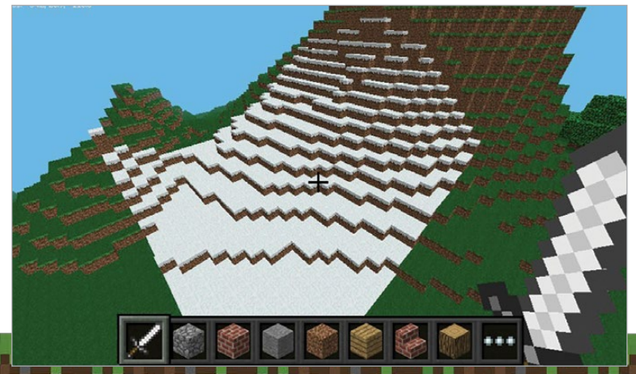
If you know the height of the world, you can cover the top layer of world in a different type of block by looping through the X and Z coordinates. What about covering the world in snow?

You can do this in a new script by looping through the coordinates around your player, finding the height for that position and setting the block to snow (78).

```
from mcpi import minecraft
mc = minecraft.Minecraft.create()

pos = mc.player.getTilePos()

for x in range(pos.x, pos.x + 10):
    for z in range(pos.z, pos.z + 10):
        y = mc.getHeight(x,z)
        mc.setBlock(x,y,z,78)
```

What other types of block could you cover the world in? Lava perhaps?!

Below Cover *Minecraft* in snow by using the API to find the height of the world



RICHARD SAVILLE
Richard runs a popular tutorial and projects blog about an average guy learning the Pi and sharing his less-than-average experiences with the community.
AverageManVsRaspberryPi.com

SHOOT IN SLOW-MOTION WITH THE CAMERA MODULE

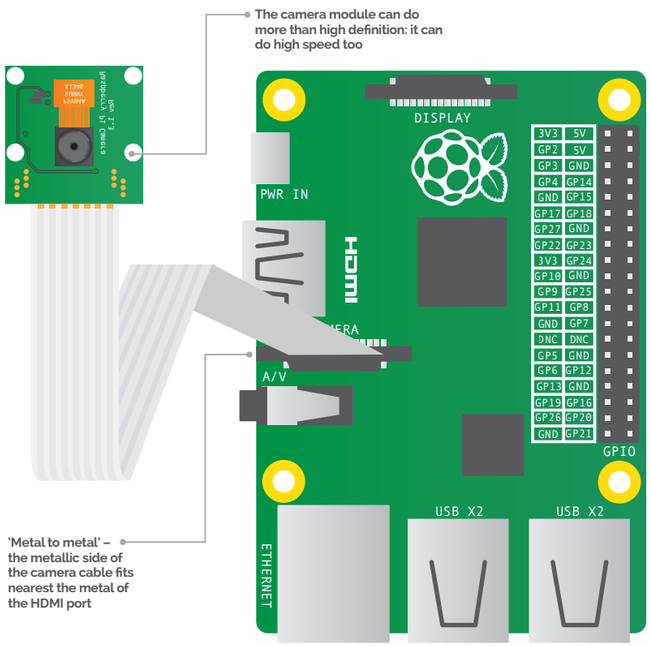
You'll Need

- Raspberry Pi Camera Module
- Internet connection
- Something fun to film

In this tutorial, the Average Man shows us how to shoot slow-motion videos with the Camera Module and convert them to play on almost any device

Slow-motion video has been used in the film industry for years – think of all those great action movie scenes with people jumping from explosions, or 'Bullet Time' made famous by the Wachowski Brothers in *The Matrix* trilogy.

It's actually really easy to make your own slow-motion videos with your Pi using the camera module. We'll get you set up and guide you through a short code listing that will let you record short 30-second videos that will automatically convert to MP4, so you can play it back on just about any device...



>STEP-01

Connect the camera module

The first thing you need to do is connect the camera module to your Pi. Make sure your Pi is turned off first. Be careful – the camera module is very sensitive to static, so ground yourself by touching something like a radiator before you start.

The camera module ribbon cable connects to the socket on your Pi nearest the HDMI port. Use the phrase 'metal to metal' to remember which way round to push it in – the metallic side of the camera cable should face the metal HDMI port. Gently pull up to release the clip and slip the ribbon cable in, then just push the clip back down firmly and check it's secure.

>STEP-02

Configure the camera module

You need to make sure the camera module is enabled, so connect your Pi to a screen and keyboard, turn it on and log in, then type **`startx`**. Open a terminal window and type **`sudo raspi-config`** to enter the configuration menu. Using the arrow keys, scroll down the list that appears and select 'Enable Camera' using the right arrow key. In the next menu, select 'Enable' with the right arrow key to turn on the camera module, then hit **Return**. You should be prompted to reboot after this; otherwise type **`sudo reboot`** to restart your Pi.

>STEP-03

Install a video converter

The Pi records video into raw H.264 files which don't work on most of our devices. We can get the Pi to convert them to a playable format straight after we've recorded them in our script on the right. To do this, we can install a package called gpac. At the command prompt, type the following, then follow the on-screen instructions:

```
sudo apt-get update
sudo apt-get install gpac
```


RECORDING TIPS

Be prepared

Make you have good lighting and a steady mount for your camera module. Also, charge your portable power options if you're shooting outside. Finally, don't forget to use an SD card with enough storage space.

What to shoot

How about shooting a remote-controlled car skidding round a corner? Perhaps a ball being thrown or other sports? What about an animation drawn on a notepad slowed right down?

>STEP-04

Test the camera

Let's make sure everything's working as it should by testing the camera with a couple of terminal commands. With a screen connected, open a terminal window and type **`raspistill -o test.jpg`**. The picture should appear on the screen for a short time and an image should be saved to your Home directory.

If it doesn't work, check you typed the command correctly, or turn off your Pi and reconnect the camera ribbon cable before trying again.

>STEP-05

Create a Python script

We'll be using Python to create our slow-motion video script. Open your favourite text editor (the Leaf text editor in Raspbian is perfect) and copy the code opposite, being careful not to misspell anything along the way. You don't need to copy the comments (lines starting with **#**) – Python just ignores them. The script uses the OS Python library to carry out terminal commands like you've typed them in directly.

Save your file as **`slowmotion.py`** in your Home directory (**`/home/pi`**).

>STEP-06

Run the Script

To run the script, simply open a terminal window, type **`cd`** and hit **Return** to ensure you're in the Home folder, then type **`sudo python slowmotion.py`**.

You will see the status of the script printed in your terminal window as it carries out its commands, and the camera module's LED will light up while it's recording.

The script will end when the video has been converted. You can watch the video on your Pi straight away by using omxplayer, which is included in Raspbian. Simply type **`omxplayer vid.mp4`**.

You could also copy your video onto any other device, like your tablet or smartphone.

The camera module has a fixed-focus lens and a wide recording angle

Slowmotion.py

Code Language >PYTHON

```
import os
import time

print("Starting program")
time.sleep(2)

##### Record the slow motion video #####
# '-w' sets the width  # '-h' sets the height
# '-fps' sets the frames per second (90 maximum - for slow motion)
# 't' sets the time in milliseconds (30000 = 30 seconds)
# '-o' sets the output filename

print("Recording started - 30 seconds")
os.system("raspivid -w 640 -h 480 -fps 90 -t 30000 -o vid.h264")

print("Recording complete. Please wait...")
time.sleep(2)

##### Convert the raw recorded video file to playable mp4 #####
# '-add' is the name of the raw video we want to convert
# The second filename is the output mp4 file
# (we use the same name followed by '.mp4')

print("Converting video. Please wait...")
os.system("rm -f vid.mp4")
os.system("MP4Box -add vid.h264 vid.mp4")

print("Video conversion complete")
time.sleep(2)

print("Closing program")
time.sleep(2)
```

LIAM FRASER

Liam is the creator of the RaspberryPiTutorials YouTube channel. He is currently studying Computer Science at the University of York and has a special interest in embedded systems.
liamfraser.co.uk

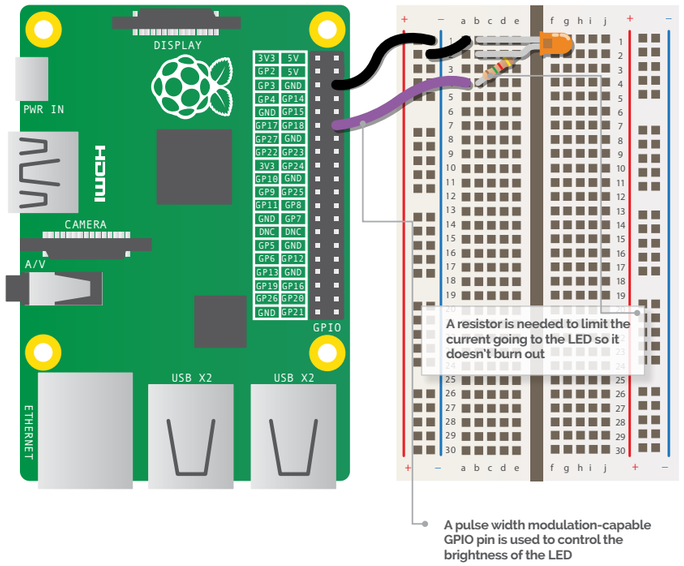
MAKE A PWM CANDLE LANTERN

Set a romantic mood with your Raspberry Pi by simulating a flickering candle effect using pulse-width modulation…

You'll Need

- A coloured LED
- Breadboard
- Female-to-male jumper cables
- Resistor (100 ohm to 330 ohm)

This tutorial is intended as a gentle – not to mention romantic – introduction to GPIO (general-purpose input and output) pins on your Raspberry Pi, and how to control them in Python. We'll be creating our romantic candle-like mood lighting using a random number generator to make an LED flicker at different intervals. In addition, its brightness will be varied using a technique called PWM (pulse-width modulation), which effectively controls what percentage of the time the LED is turned on. We will also take a look at the output of the pins on an oscilloscope, so that we can see how the code translates to the electrical signals that make things tick.



>STEP-01

Pick a resistor for your LED

A resistor will limit the current that flows through the LED. Different colour LEDs have different current limits, so you'll need to check the specifications where possible. 100 ohm or 220 ohm will definitely work, though your LEDs might end up being dimmer than usual. The equation for working out resistance is as follows:

R = (3.3V – LED VOLTAGE) / LED CURRENT

Our yellow LED needs a voltage of 1.8V – 2.2V and has a typical current of 20mA, so: **R = (3.3V – 2.0V) / 0.02** (which is 65 ohms). A resistor with a value between 65 and 130 ohms is ideal here, but a lower value will make your LED brighter.

>STEP-02

Setup the breadboard

Unplug your Pi and follow the breadboard illustration setup. Make sure you use the same GPIO pin we have, as only a couple are capable of pulse-width modulation (on the B+). We're using GPIO number 18 for PWM, which is described as **PCM_CLK/PWM0**.

The circuit path, as shown in the illustration, is **GPIO 18 > resistor > LED positive**. Finally, the LED negative leg goes into ground. The positive leg of an LED is usually longer. The negative side will have a flat edge rather than a circular one.

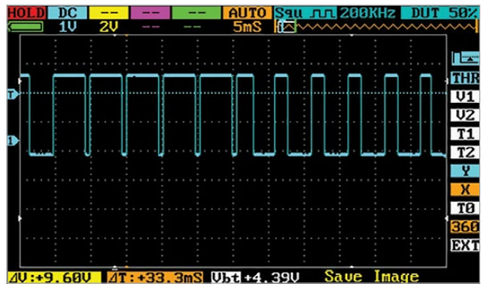
>STEP-03

Get coding!

Once you've wired up the project, power up your Pi and begin coding using an editor of your choice (or opening a terminal and typing **`nano candle.py`** will do).

Once we've imported the libraries we need, the **`setup`** function organises our program and starts PWM for us. The **`flicker`** function sets a random brightness by calling the **`set_brightness`** function, then sleeps for a random time. This function is then wrapped up in an infinite **`while`** loop within the **`loop`** function, which handles the all-important cleanup of the GPIO library when **CTRL+C** is pressed by the user.

Below This oscilloscope trace illustrates how the brightness of the LED is controlled



>STEP-04

Test your creation

Exit your editor and run the code by typing **`sudo python2 candle.py`** into your terminal (you need root privileges to access the GPIO pins). Now that you've tested it, you can exit with **CTRL+C** and we'll make it run at boot. This way, the Pi can run headless and not need any user interaction.

At the terminal, type: **`sudo nano /etc/rc.local`**, then add the following line: **`python2 /home/pi/candle.py &`** (but make sure you put this in the line before **`exit 0`**). Don't forget to save the changes.

The ampersand means the script will go to the background and let the boot process continue. Notice how sudo isn't required because **`rc.local`** is executed as root. Reboot the Pi with **`sudo reboot`** to verify that it works.

>STEP-05

Packaging it up

Now that the script is started when the Pi boots, you could package it up into a nice container using a portable phone charger as a power supply. Arts and crafts are out of the scope of this tutorial, but there are plenty of candle holders that can be fashioned out of paper if you search the internet. Paper is ideal, especially with lots of holes in, since the LED probably isn't throwing out much light.

>STEP-06

Presentation, presentation, presentation

The candlelight project is surprisingly effective, but presentation is key in matters of the heart, so you may want to spruce up your project before you use it on a loved one. Pretty lanterns are available very cheaply from most department stores; just make sure you select one that obscures the view of the interior. If the lantern isn't big enough to fit the Pi and breadboard, solder the resistor to the LED and hide the Pi behind it.

Below A candle lantern that obscures the view of the inside is perfect for disguising the LED and hiding your Pi

Candle.py

Language >PYTHON

```
import RPi.GPIO as GPIO
import time
import random

# Set the PWM output we are using for the LED
LED = 18

def setup():
        global pwm

        # GPIO uses broadcom numbering (GPIO numbers)
        GPIO.setmode(GPIO.BCM)
        # Set the LED pin as an output
        GPIO.setup(LED, GPIO.OUT)

        # Start PWM on the LED pin at 200Hz with a
        # 100% duty cycle. At lower frequencies the LED
        # would flicker even when we wanted it on solidly
        pwm = GPIO.PWM(LED, 200)

        # Start at a brightness of 100%
        pwm.start(100)

def set_brightness(new_brightness):
        # Sets brightness of the LED by changing duty cycle
        pwm.ChangeDutyCycle(new_brightness)

def flicker():
    # We want a random brightness between 0% and 100%.
    # Then then we'll hold it for a random time
    # between 0.01 and 0.1 seconds to get a nice flicker
    # effect. Play with these values to make the effect
    # suit your liking
    set_brightness(random.randrange(0, 100))
    time.sleep(random.randrange(1, 10) * 0.01)

# The wrapper around the flicker function makes sure the
# GPIO hardware is cleaned up when the user presses CTRL-C

def loop():
    try:
        while True:
            flicker()
    except KeyboardInterrupt:
        pass
    finally:
        GPIO.cleanup()

# setup the hardware
setup()

# start the flickering
loop()
```

DAVID HUNT

David has been making projects for the Raspberry Pi since the early days. These include a Camera Controller, TimeLapse Rail, Focus Stacker, and even a Bark-Activated Doggy Door Opener. Oh, and let's not forget the PiPhone!
DavidHunt.ie

WATER DROPLET PHOTOGRAPHY

Have you ever wanted to capture those split-second photographs of water droplets colliding? Now you can with a Raspberry Pi-controlled solenoid and camera trigger!

You'll Need

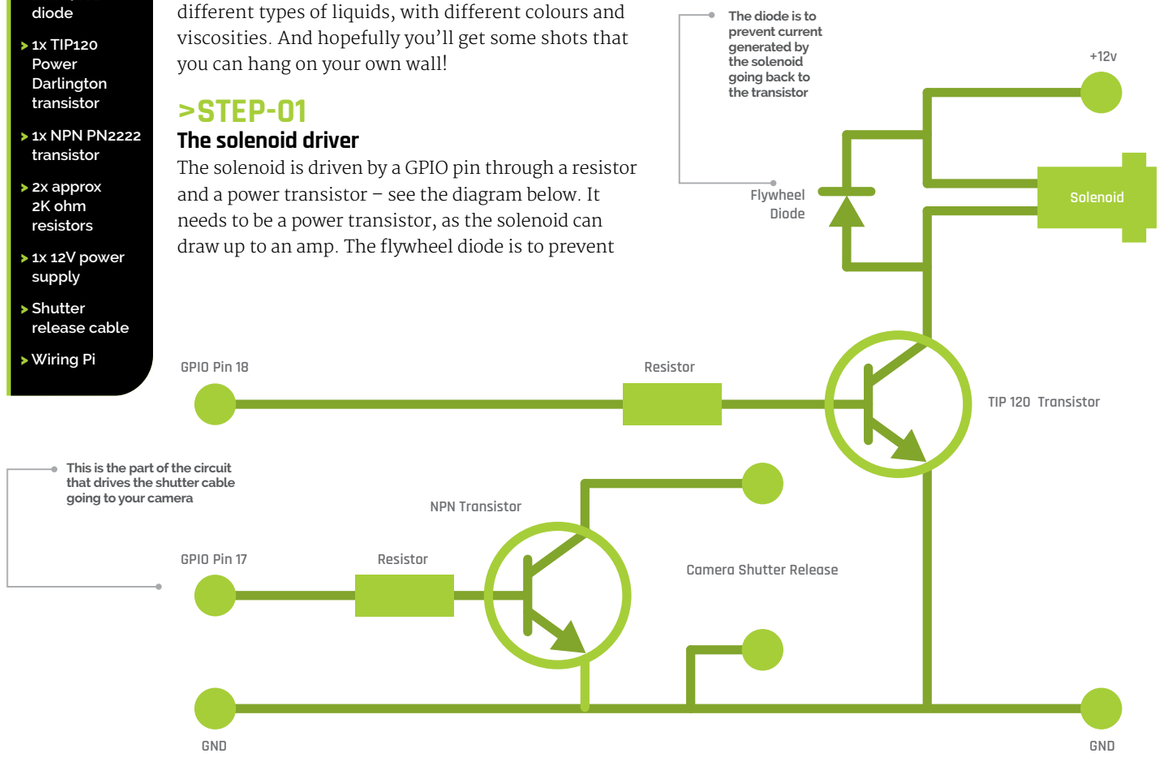
- Solenoid Valve
- 1x IN4001 diode
- 1x TIP120 Power Darlington transistor
- 1x NPN PN2222 transistor
- 2x approx 2K ohm resistors
- 1x 12V power supply
- Shutter release cable
- Wiring Pi

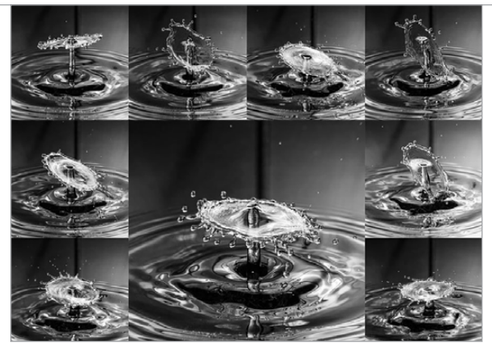
This tutorial shows you how to build a project that will allow you to capture those beautiful, carefully-timed photographs where water droplets are colliding. From assembling the hardware with a solenoid, to writing the code to drive it, you'll be doing your own droplet collision photography in no time. After that, you can have all sorts of fun using different types of liquids, with different colours and viscosities. And hopefully you'll get some shots that you can hang on your own wall!

>STEP-01

The solenoid driver

The solenoid is driven by a GPIO pin through a resistor and a power transistor – see the diagram below. It needs to be a power transistor, as the solenoid can draw up to an amp. The flywheel diode is to prevent any current generated by the solenoid from going back into the NPN transistor. Once the GPIO pin goes high, the current can flow from 12V to GND, enabling the solenoid to open the valve and allowing the liquid to pass through. We only open the valve briefly, just enough to allow a drop through at a time.





Above An example of the type of image that can be achieved

>STEP-02

The camera shutter driver

The camera shutter is triggered by a low-power NPN transistor. DSLRs usually have a shutter release input which is shorted to ground, causing the camera to take a picture. In this project we're using a signal transistor to cause that (usually 3.3V) input to short, so we can get the camera to take a picture from the Python script on our Pi. You'll need to get the correct shutter release for your camera, but they can be sourced on eBay for under £5.

>STEP-03

Setting up the solenoid

This is the messy part! A drinks bottle with a small opening is ideal for attaching to the input of the solenoid. This type is often used for sports drinks, and can usually be pushed onto the solenoid input without any leaks. You can cut the bottle in half for easy top-ups. Apply 12V to the solenoid and you should get a stream of liquid through the valve; remove power and the valve should close. Attach it to the circuit you built in step 1.

>STEP-04

Trigger the camera

Now connect up your camera circuit and test it with the Python code. You will need to adjust the timings to get the camera to trigger at the right moment. But initially, you should hear two clicks of the solenoid and one click of the camera. You can adjust the timing in two ways: by changing the Python code, or altering the distance between the solenoid and the liquid container. In the code provided, the timings were good for a 50cm fall.

>STEP-05

Get the lighting right

You'll need to use a flash to freeze the movement of the liquid. Otherwise you'll get blurred images, even if your camera is on a tripod. An off-camera flashgun triggered by a sync cable is a really good idea, as it allows you to move the flash into all kinds of interesting positions. Oh, and keep the flash power low for shorter flash durations, giving you sharper images. And you can always use two or three flash units at lower power for shorter flashes still.

Drop.py

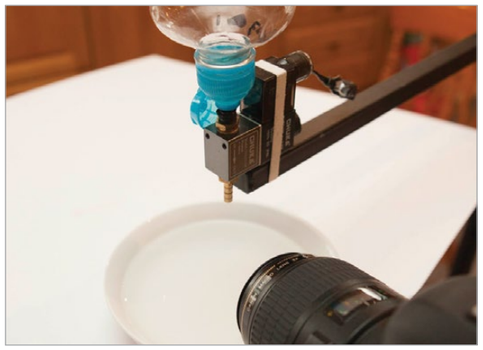
Code Language >PYTHON

```
# Import the relevant Modules
import wiringpi2 #Learn more about this library at wiringpi.com
from time import sleep

# Set up the GPIO Pins
gpio = wiringpi.GPIO(wiringpi.GPIO.WPI_MODE_GPIO)
shutterpin = 17
solenoidpin = 18
gpio.pinMode(shutterpin,gpio.OUTPUT)
gpio.pinMode(solenoidpin,gpio.OUTPUT)
wiringpi.pinMode(shutterpin,1)
wiringpi.pinMode(solenoidpin,1)

# Release a drop of liquid
gpio.digitalWrite(solenoidpin,gpio.HIGH)
sleep(0.06)
gpio.digitalWrite(solenoidpin,gpio.LOW)

sleep(0.1)

# Release a second drop
gpio.digitalWrite(solenoidpin,gpio.HIGH)
sleep(0.05)
gpio.digitalWrite(solenoidpin,gpio.LOW)

# Wait for the droplet to hit the liquid container
sleep(0.12)

# Trigger the camera (which is set to manual mode)
gpio.digitalWrite(shutterpin,gpio.HIGH)
sleep(0.1)
gpio.digitalWrite(shutterpin,gpio.LOW)
```

>STEP-06

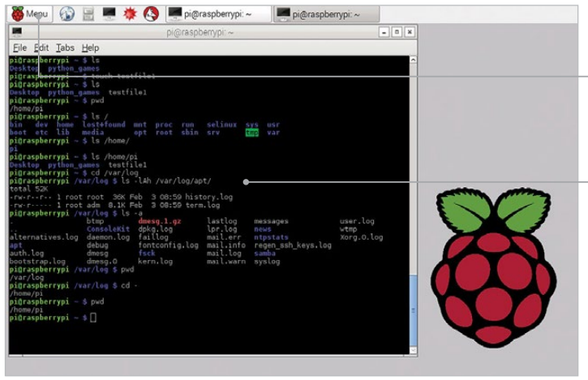
Adjust the camera settings

You should be shooting on manual setting, with a shutter speed as high as your camera will allow for flash. For Canons this is about 1/160th of a second, and maybe 1/250th of a second for Nikons. Use ISO 100-400 and then adjust your aperture till you get a decent exposure. You can then tweak the flash power down to get shorter flash durations, which will tend to freeze the motion of the liquid more. Open up the aperture more if you need to, but be aware that your depth-of-field will be reduced.



RICHARD SMEDLEY

Having found words often better than pointing at things, Richard stuck with the command line whence all around had fled.
twitter.com/RichardSmedley



COMMAND LINE Pi PART 1: DON'T PANIC

Richard Smedley presents a cut-out-and-keep guide to using the command line on the Raspberry Pi. In part one, we take a look around and discover things aren't as strange as they might appear…

READ THE MANUAL

Help is included, with man(ual) pages, but they can be a little overwhelming. At least use them to check out some extra options beyond the switches like **-a** we used here. To read the ls man page, type **man ls**.

Not a throwback to the past, but a quick and powerful way of getting your Pi to do what you want, without all that RSI-inducing menu chasing and icon clicking. The command-line interface was a great step up from manually toggling in your instructions in octal (base-8), using switches on the front of the machine! Graphical user interfaces (GUIs) brought friendly visual metaphor to the computer, losing some power and expressiveness. With the Pi, you can get the best of both worlds by knowing both – follow our introductory series and you'll soon be as comfortable at the command prompt as you are at your desktop.

When you boot up your Pi with Raspbian installed, you arrive at the command line by default. You log in and type **startx** to get to the desktop. If you hold down the **ALT+CTRL** keys and press **F1** (the first function key on the keyboard), you'll see that the command line is still there. Press **ALT+F2** through to **F6** and you'll find five further virtual consoles waiting for you to log in.

You can drop into these any time you like, but for now press **ALT+F7** and you'll be back in mouse and menu land. The command line is also available through a program called a terminal emulator (often referred to as a term or xterm). You'll also find people referring to the shell, or Bash. Don't worry about that for now; just click on the icon at the top of the screen that looks like a black television screen, or go to Accessories>Terminal in the menu: the terminal now awaits your commands.

Look around

If you're used to looking at files and folders in a file manager, try to clear your mind of the icons and concentrate on the names. Type **ls** and press **Return** (see the 'Press Return' boxout on the next page). On a fresh Raspbian install you'll just see two directories: python_games and Desktop. Type **ls python_games** (also see the Lazy Completion boxout) and you should see a listing like **Fig 1** on the next page.

Commands like **ls** are not cryptic (at least not intentionally) but they are terse, dating back to a time when the connection to the computer was over a 110 baud serial line, from an ASR 33 teletype terminal. And if you think it's strange to be defined by 50-year-old technology, just remember that your QWERTY keyboard layout was both to stop mechanical typewriter keys jamming, and to enable salespeople to quickly type 'typewriter' using the top row!

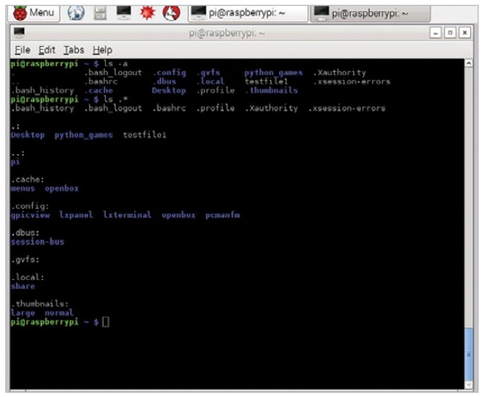


Fig 1 Switches modify behaviour in commands; ls -a shows (dot) files in your listing that are usually hidden from view

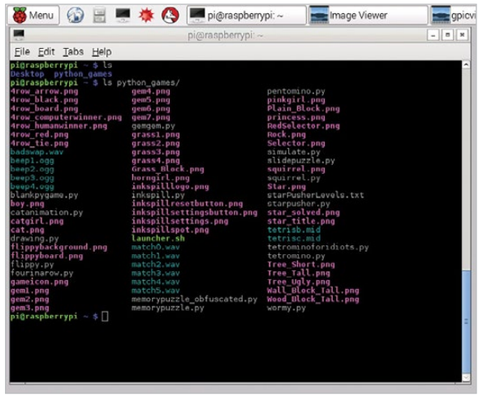


Fig 2 Who needs icons when you can fit a listing of 78 files into a small window? Coloured fonts indicate file types

PRESS RETURN

To save repeating it in the text, we'll say here – each time you type in a command, you need to hit the **Return** key at the end, to tell the Pi you've issued Bash with a command.

File path

You can list files and folders anywhere in your system (or other connected systems) by providing the path as an argument to your command. The path is the folder hierarchy – on a Windows computer, in a graphical file browser, it starts with 'My Computer'; on your Pi it starts at **/**, pronounced 'root' when used on its own as the root of your filesystem. Try it: **ls /** – again we get terseness; names like bin, which is short for binary, and is the directory where many programs are kept (**ls /bin** to see). **ls /dev** shows hardware devices in the Pi.

ls /home – see that 'pi'? That's you; you're logged in as user pi. If you've changed your login name, or created extra users, they'll all be listed there too – every user gets her own home directory; yours is the /home/pi folder we found ourselves in earlier. Before, with python_games, we used the relative path (the absolute path would be /home/pi/python_games) because we're already home – to check your location, type **pwd** (present working directory).

There's no place like ~

For any logged-in user, their home directory is abbreviated as **~** (the tilde character). **ls ~** and you'll see. There's apparently not much in your home directory yet, but Raspbian keeps a lot hidden from the casual glance: files and folders beginning with a dot, known as 'dot files', contain configuration information for your system and its programs. You don't need to see it normally, but when you do, just ask **ls** to show you all files with a command switch. You can do this with either the full switch **--all**, or the abbreviation **-a** like so: **ls -a ~**

Traversing the pathways of the directory hierarchy can be easier from the command line than clicking up and down a directory tree, particularly with all the shortcuts given. Your **ls -a** showed you **.** and **..** as the first two directories; these shortcuts represent the current and the parent directory respectively.

Try listing the parent directory – from /home/pi, **ls ../../** will show you two layers up. If you want to list the hidden files without the **.** and **..** appearing (after all, they're present in every directory, so you don't need to be told), then the switch to use is **-A**.

> Commands are not cryptic (at least not intentionally), but they are terse

Before we move on to other commands, let's look at chaining switches together: **ls -lAh ~**

-l gives you more information about the files and folders, and **-h** changes the units from bytes to KB, MB or GB as appropriate. We'll look at some of the extras the **-l** listing shows you in more detail later – particularly in parts two and three of this series.

Time for change

Now enough looking, let's start moving: **cd** is short for change directory, and moves you to wherever you want on the filesystem – try **cd /var/log** and have a look (**ls**, remember). Files here are logs: messages on the state of your system that are saved for analysis later. It's not something you'll often need to think about – Raspbian is a version of an operating system that also runs across data centres and supercomputers, where problem monitoring is very important – but useful to know, particularly if you have a problem and someone on a forum advises you to check your logs.

cd ~ will take you where you expect it. Try it, then **pwd** to check. Now try **cd -** (that's a hyphen, or dash), the '**-**' is a shortcut for 'wherever I was before I came here'. Now we've looked around, next time we're going to be doing things to our files.

LAZY COMPLETION

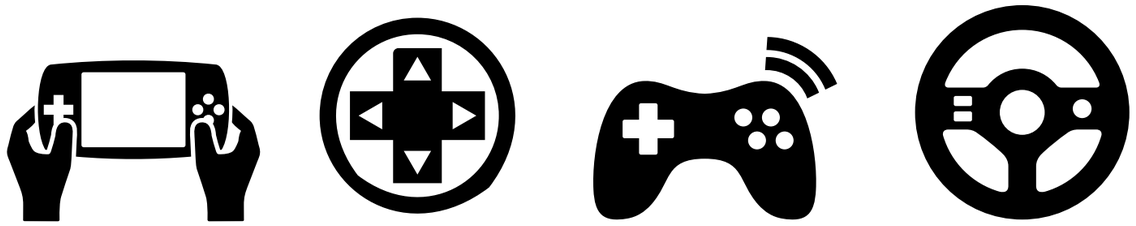
You don't need to type all of **ls python_games** – after **ls p**, hit the **Tab** key and it will auto-complete. If you've more than one file beginning with p, they'll all be listed and you can type more letters and hit **Tab** again.

PART 1

MAKE GAMES WITH PYTHON

SEAN M TRACEY

Sean is a technologist living in the South West of England. He spends most of his time making silly things with technology.
sean.mtracey.org



SHAPES & PATHS WITH PYGAME

In this new ten-part series, **Sean M Tracey** teaches us how to make a game on our Raspberry Pi from the ground up. In part one we learn the basics…

Hello, and welcome to this first part of ten tutorials in which we'll learn to make games on the Raspberry Pi with Pygame. Over the course of this series, we'll look at drawing, animation, keyboard and mouse controls, sound, physics, and maybe even the installation of a kitchen sink. Each part will add to our knowledge of Raspberry Pi game development, allowing us to understand the games we play and to create almost anything our imaginations can come up with.

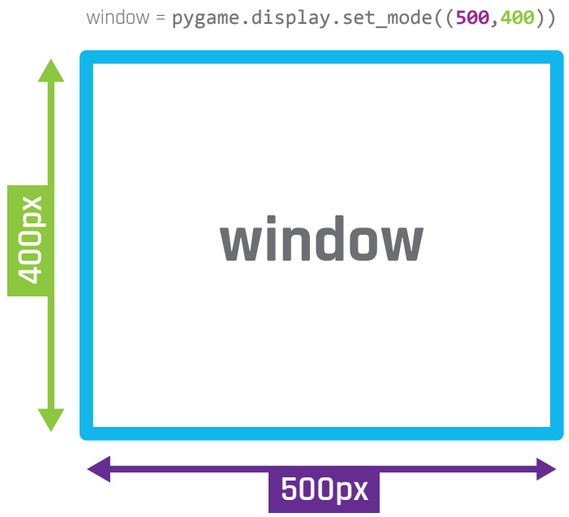
This series isn't for absolute programming beginners, but it's not far from it – we're going to guess that you've written some simple Python (or similar) programs in the past, and are able to do things like creating files and get around your Pi's filesystem without too much difficulty. If you've not set up your Pi and are a little lost on how to go about it, there are lots of easy-to-follow guides on the web which will help bring you up to speed. You could point your web browser to **raspberrypi.org/resources** to get started.

In this first part of the series, we're going to look at drawing and colouring various shapes in a window. This isn't quite *Grand Theft Auto V*, granted, but drawing shapes is the first step in building just about anything.

To start off, open your favourite text editor, create a new file, insert the following code into it and save it as **hello.py**:

```
import pygame

pygame.init()

window = pygame.display.set_mode((500, 400))

while True:

    pygame.draw.rect(window, (255,0,0),
                     (0, 0, 50, 30))

    pygame.display.update()
```


Below Here we can see how each variable in `window` affects the application window's shape and size. Width always comes before height



Let's run that code and see what it does. In your terminal window, enter **python hello.py**. If all has gone well, a new window will have opened showing you a red square on a black background in the top-left corner of the window. We've just created our first Pygame program – let's walk through it.

Understanding hello.py

The first two lines of our first program are very simple – all we've done is told Python that we want to use Pygame. **import pygame** loads all of the Pygame code into our script, so we don't have to write all of that code ourselves. Pygame is designed to make the creation of games and interactive software easy. **pygame.init()** tells Pygame that we're ready to start using it.

Let's look at the third line of code:

```
window = pygame.display.set_mode((500, 400))
```

window is the parameter we're going to use to tell our Pygame program about how it should look when it runs; each parameter affects the application window's shape and size – note that width always comes before height. **window** is also the parameter that we'll use to tell other lines of code the surface on which they should draw shapes and set colours. With **window**, we're calling the **set_mode** function of Pygame's display module – the latter is responsible for how the game window and surface (an informal term for the pixels we'll be manipulating) behaves. We're passing a tuple (which we can think of as a special list of things – in this case numbers) to **set_mode()** to tell it how big we want our game window to be. In this case, the application window is 500 pixels wide by 400 pixels tall. If you pass numbers that are bigger, the game window will be bigger; if we pass numbers that are smaller, the game window will be smaller.

The next few lines are where we make our program draw shapes on that window. When programs run, they execute their code, and when they're finished, they close themselves. That's fine unless, of course, you want your program to be interactive or draw/animate shapes over time (exactly what we need from a game). So, in order to keep our program from exiting, we make a **while** loop and put all our code inside. The **while** loop will never finish because **True** is always **True**, so we can keep running our program and drawing our shapes for as long as we like.

The first thing we do in our **while** loop is draw a rectangle. A rectangle is the simplest shape that we can draw in Pygame:

```
pygame.draw.rect(window, (255,0,0), (0,0,50,30))
```

> This isn't quite Grand Theft Auto V, granted, but drawing shapes is the first step

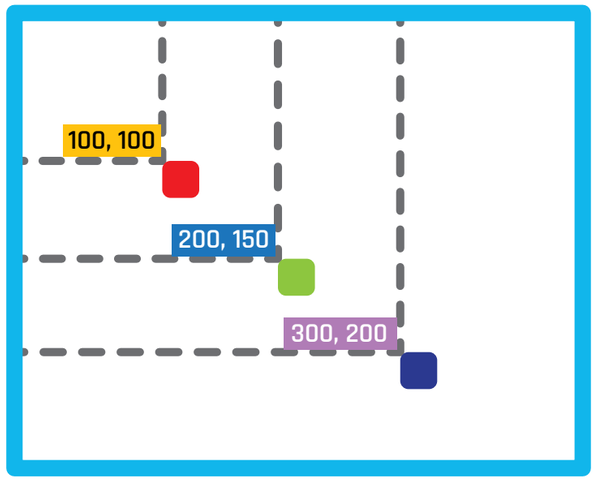
The parameters at the end are telling Pygame where we want to draw our rectangle, the colour we want our rectangle to be, how we want to draw it, and how big we want it to be.

In our **hello.py** program, we've told Pygame to draw a rectangle in our window – or, at least, the surface we create with our **window** parameter. Next, we told Pygame what colour we wanted our rectangle to be by passing it through a tuple (a special list of numbers) representing how much red, green and blue the final colour should have in it. Why red, green and blue? Those are the three colours your screen combines to create every shade you can see on it. 0 means that none of that colour should be used in the shape; 255 means that the maximum amount

PYGAME

Pygame is installed on Raspbian by default. Find documentation detailing all its features at **pygame.org/docs**

```
pygame.display.rect(window,(255,0,0),(100,100,50,50))
pygame.display.rect(window,(0,255,0),(200,150,50,50))
pygame.display.rect(window,(0,0,255),(300,200,50,50))
```



Left Here's a clear look at what each variable does to the shape we're drawing

Adding more shapes

We've successfully drawn one shape, so let's draw a few more. We'll draw some squares around the screen and mess around with their properties a little bit. There's no need to create a new file, so we'll stick with **hello.py** for now. Edit the **while** loop so it's the same as the following:

```
while True:

    pygame.draw.rect(window, (255,0,0),
                     (100, 100, 50, 50))
    pygame.draw.rect(window, (0,255,0),
                     (150, 100, 50, 50))
    pygame.draw.rect(window, (0,0,255),
                     (200, 100, 50, 50))

    pygame.display.update()
```

Now we should have three squares: red, blue, and green – nice and simple, but those squares are placed right next to each other. What would happen if they were to overlap? Let's find out. Change your code once more to the following:

```
while True:

    pygame.draw.rect(window, (255,0,0),
                     (0, 0, 50, 50))
    pygame.draw.rect(window, (0,255,0),
                     (40, 0, 50, 50))
    pygame.draw.rect(window, (0,0,255),
                     (80, 0, 50, 50))

    pygame.display.update()
```

This time we get two rectangles and a square, but that's not what we asked for! So what's gone wrong? When we execute our code, it works through line-by-line what it has to draw and where it has to put it. If one item is drawn and then another is over or on top of part of it, then we can no longer see what's beneath that second shape. The pixels of the shape drawn first are lost when we overlap it with another shape. If we change the order of our code, we can see this effect in action. Cut and paste the code for the second square so that it becomes the third square drawn, like so:

LINE WIDTH

When drawing a rectangle or ellipse, you have the choice of passing a line width. If you don't, the shape will be filled solid.

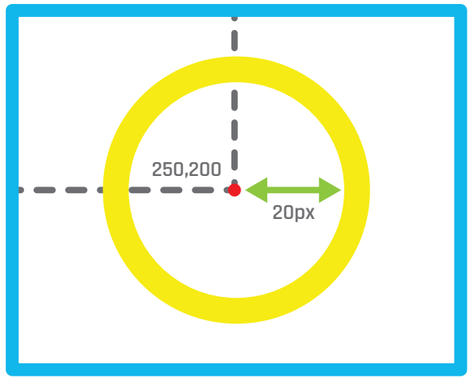
of colour should be in that shape. We told our rectangle that it should be the colour **(255, 0, 0)**, which is pure red. If we had told it to be **(255, 0, 255)**, it would have been a bright purple, because it's being drawn with the maximum amount of red and the maximum amount of blue. If we had told our rectangle to be coloured **(100, 100, 100)**, it would be a dark grey, because all of the colours would be equal.

After we've passed through a colour for our rectangle to be, we next tell it where it should go and how big it should be. We do this by passing a tuple of four numbers. The first number is an X coordinate – that is, how far from the left side of the window the left edge of our shape should be. The second number is a Y coordinate; this tells our shape how far from the top of our window the top edge of our shape should be. The third number is the width our rectangle should be, and the fourth number is how tall our rectangle should be. So, if we wanted our rectangle to be 50 pixels from the left of the window, 100 pixels from the top of our window, 20 pixels wide and 80 pixels tall, we'd pass **(50, 100, 20, 80)** to **pygame.draw.rect()**.

The order never changes. If you tell Pygame how big you want the rectangle to be when it's expecting a colour or vice versa, the program may crash, so take your time.

Our last line in **hello.py** is nice and simple: it tells Pygame that we're done drawing shapes for the moment and that it can now refresh the window. This saves our Pi having to draw and redraw the screen for every shape that we've created; instead, it can get them all drawn in one go.


```
pygame.draw.circle(window,(255,255,0),(250,200),20,1)
```



Above Here's how the variables affect the drawing of a circle

```
while True:

    pygame.draw.rect(window, (255,0,0),
                     (0, 0, 50, 50))
    #pygame.draw.rect(window, (0,255,0),
                     #(40, 0, 50, 50)) FROM HERE
    pygame.draw.rect(window, (0,0,255),
                     (80, 0, 50, 50))
    pygame.draw.rect(window, (0,255,0),
                     (40, 0, 50, 50)) #TO HERE

    pygame.display.update()
```

Now we get rectangle, square, rectangle. This is because the red and blue squares were drawn first and then the green square was drawn over the top of them. We still got squares but we just can't see all of them, so they look like rectangles.

But enough with rectangles. Rectangles are awesome and we can build many things with them, but Pygame lets us draw all kinds of other shapes too. We can draw circles, ellipses and paths (which are made up of many lines between multiple points).

Drawing circles

Drawing a circle is much like drawing a square except instead of passing a width and a height, we pass a radius and a point, around which we draw our circle. If we wanted to draw a yellow circle in the middle of our window with a diameter of 40 pixels, we would use the following code to replace the **while** loop in **hello.py**:

```
while True:

    #Just like before to help us remember
    #pygame.draw.circle(WHERE TO DRAW, (RED, GREEN,
BLUE), (X COORDINATE, Y COORDINATE), RADIUS, HEIGHT,
WIDTH)

    pygame.draw.circle(window, (255,255,0),
                       (250, 200), 20, 0)

    pygame.display.update()
```

Just like drawing a rectangle, we tell Pygame on which surface we want to draw our circle, what colour we want it to be, and where it should go. The radius is specific to drawing this particular shape. You might have noticed that we put a **0** after our radius; this is a value used to determine the width of the line that draws our circle. If we pass **0**, the circle is filled; but if we pass **2**, for instance, we get a 2-pixel-wide line with an empty centre:

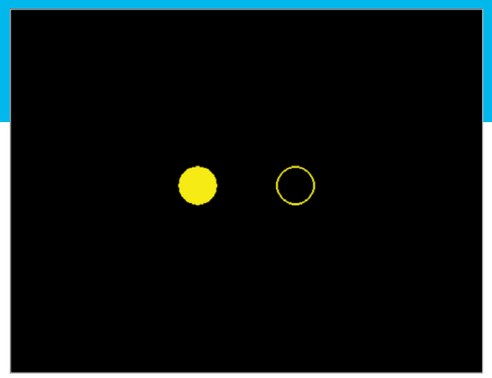
> Rectangles are awesome, but Pygame lets us draw all kinds of other shapes too

```
while True:

    #Filled
    pygame.draw.circle(window,(255,255,0),
                       (200, 200), 20, 0)

    #Not filled
    pygame.draw.circle(window,(255,255,0),
                       (300, 200), 20, 2)

    pygame.display.update()
```

What about ellipses? They are a slightly strange cross between drawing rectangles and circles. It's the same as drawing a rectangle: we pass an X coordinate, a Y coordinate, a width, and a height, but we end up with a circle(ish) shape. Let's draw an ellipse or two...

```
while True:

    pygame.draw.ellipse(window, (255, 0, 0),
                        (100, 100, 100, 50))
    pygame.draw.ellipse(window, (0, 255, 0),
                        (100, 150, 80, 40))
    pygame.draw.ellipse(window, (0, 0, 255),
                        (100, 190, 60, 30))

    pygame.display.update()
```

Right When drawing a circle, the last variable lets us know if the circle should be filled in or not



TUPLE

A tuple is like a list, but unlike a standard list, a tuple's contents can't be changed (it's immutable). **python.org/docs**

Just as before, run your code. You should now see three ellipses – one red, one green, and one blue, each a different size. If you wanted to visualise how these shapes were generated, you could draw rectangles using the same coordinates as you used to draw an ellipse and it would fit perfectly inside that box. Guess what? That means you can also make circles by using **pygame.draw.ellipse** if the width and height parameters are the same.

```
while True:

    pygame.draw.rect(window, (255, 0, 0),
                     (100, 100, 100, 50), 2)
    pygame.draw.ellipse(window, (255, 0, 0),
                        (100, 100, 100, 50))

    pygame.draw.rect(window, (0, 255, 0),
                     (100, 150, 80, 40), 2)
    pygame.draw.ellipse(window, (0, 255, 0),
                        (100, 150, 80, 40))

    pygame.draw.rect(window, (0, 0, 255),
                     (100, 190, 60, 30), 2)
    pygame.draw.ellipse(window, (0, 0, 255),
                        (100, 190, 60, 30))

    #Circle
    pygame.draw.ellipse(window, (0, 0, 255),
                        (100, 250, 40, 40))

    pygame.display.update()
```

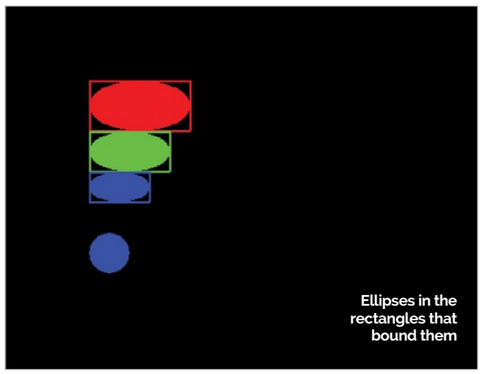
A new path

So that's rectangles, squares and circles, but what if we want to draw a triangle, a pentagon, a hexagon or an octagon? Are there functions for every single kind of shape? Well, no, but what we do have are paths. Paths let us draw irregular shapes by defining points in space, then joining them up with lines and filling in the space we've created.

This is a little more complex, so it's time to move on from our old friend **hello.py**. Create a new file, call it **paths.py**, and save it with the following inside:

```
import pygame

pygame.init()
window = pygame.display.set_mode((500, 400))

while True:
    pygame.display.update()
```

This is just our bare-bones Pygame app again, in fact. If you want to make a copy of this for experimenting without breaking anything, now would be a good time to do so.



Ellipses in the rectangles that bound them


```
pygame.draw.line(window,(255,255,255),(50,50),(75,75),1)
pygame.draw.line(window,(255,255,255),(75,75),(25,75),1)
pygame.draw.line(window,(255,255,255),(25,75),(50,50),1)
```

Right You can make a triangle from three separate lines

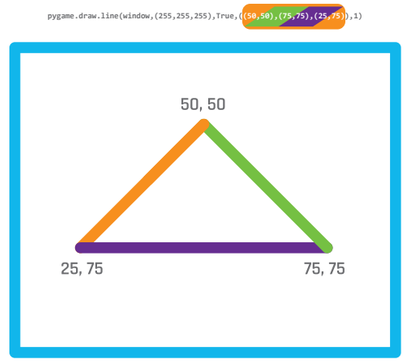
Each path is made of joined-together lines, but before we start joining things up, let's draw a couple of standalone lines to familiarise ourselves with them. We can do this with **pygame.draw.line()**. Edit **paths.py** so your **while** loop is like the following:

```
while True:

    pygame.draw.line(window, (255,255,255),
                     (0, 0), (500, 400), 1)

    pygame.display.update()
```

If you run this code now, you'll see a 1-pixel-wide white line going from the top left to the bottom right of our Pygame window. The parameters we pass to **pygame.draw.line** start off the same way rectangles and ellipses do. We first tell Pygame where we want to draw the shape and then we choose a colour. Now things change a little. The next argument is a tuple of the X and Y coordinates for where we want our line to start, and the third argument is a tuple with the X and Y coordinates for where we want our line to end. These are the two points between which our line will be drawn. The final argument is the width of the line that is being drawn in pixels.

With this, we can now create shapes by defining points in our window. Let's draw that triangle we talked about earlier:

```
while True:
    pygame.draw.line(window, (255,255,255),
                     (50, 50), (75, 75), True)
    pygame.draw.line(window, (255,255,255),
                     (75, 75), (25, 75), True)
    pygame.draw.line(window, (255,255,255),
                     (25, 75), (50, 50), True)

    pygame.display.update()
```



Above This triangle is made up of one line with multiple points. Follow the colours to see which variable is which

There! We have a friendly, white triangle with a 1px edge. But when we look at that code, it looks like a lot, doesn't it? So many things, like the colour or the width, are being written again and again just for the sake of it. There must be a better way, and indeed there is! All we need is **pygame.draw.lines()**. While **pygame.draw.line()** lets us draw a line between two points, **pygame.draw.lines()** enables us to draw a sequence of lines between numerous points. Each XY-coordinate point will be joined up to the next XY-coordinate point, which will be joined up to the next XY-coordinate point, and so on.

After running the code you'll see that it's exactly the same, except we did it in one line of code instead of three. You might have noticed that we didn't actually close the triangle – Pygame did it for us. Just before we pass the points for our shape to be drawn from, we can pass either a **True** of a **False** value that will let Pygame know that we want it to close our shapes for us. Change it to **False** and we get the first two lines of our shape, but not the third.

What if we want a more complex shape? We simply add more points like so:

```
while True:

    #pygame.draw.lines(WHERE TO DRAW,
COLOUR, CLOSE THE SHAPE FOR US?, THE POINTS
TO DRAW, LINE WIDTH)

    pygame.draw.lines(window,(255,255,255),
True, ((50, 50), (75, 75), (63, 100), (38,
100), (25, 75)), 1)

    pygame.display.update()
```

There you have it: a pentagon. If you want a hexagon or even a triacontagon, just add more points. Give it a go.

So that's how you draw shapes, lines and paths in Pygame. Already we know enough to make programs that could be used to display pixel art to our friends and family.

NEXT MONTH

In part two, things will get a little more complicated – we'll be animating all of the shapes we've learnt to draw so far. They'll bounce, stretch and spin, which is where the games begin...

RETRO GAMING GROUP TEST

On Test

- RetroPie **blog.petrockblock.com/retropie**
- PiPlay **pimame.org**
- Raspicade **sourceforge.net/projects/raspicade**

Get your rose-tinted glasses at the ready: here are three of the best emulator packages for the Raspberry Pi…

Emulation is a wonderful thing. Much like an old police box or a 1982 DeLorean, it can take us back in time to when we were driving in a Grand Prix, saving the world from the brink of destruction or simply creating lines out of falling blocks while waiting for the bus. We have a rich heritage of computer gaming, and with the help of the Raspberry Pi 1 or 2, it's beautifully preserved through the emulation software we're looking at today.

It's not all rose-tinted glasses, though. Emulation does have its issues, and chief among them is the legal grey area surrounding ROMs. A ROM is a dump of the game code contained in a file so ROMs represent your favourite games from yesteryear when loaded into your chosen emulator.

Of course, that game code is copyrighted software, created by companies with the specific goal of making money. So how can you play retro game ROMs legally? This is the grey area we face.

Obtaining retro-gaming ROMs is your own responsibility, so we'll leave you to make your own investigations online.

Legality aside, emulation provides a virtual museum of computing in your home, and will most likely be used to keep the knowledge of these times alive, long after the hardware has gone to silicon heaven. Let's get cracking…

PiPLAY

Best FOR FEATURES

PIPLAY WAS CREATED BY AUTHOR AND RETRO FIEND SHEA SILVERMAN, VIA A SUCCESSFUL KICKSTARTER THAT RAN IN MAY LAST YEAR

Above PiPlay offers a solid range of emulators in an easy-to-use package

PiPlay is based on the Raspbian operating system and comes with a custom interface that enables easy selection of the many consoles that can be emulated. It comes with a large choice of emulators, including NES, Game Boy, Atari 2600 and Commodore 64, to name a few. In our review, we tested three in particular – SNES, Mega Drive, and PlayStation.

PiPlay is a downloadable image that is copied to a blank SD card of 4GB capacity or greater. To copy the image, you can use the dd utility on a Linux machine, Win32DiskImager for Windows, and PiWriter for Mac OS X.

On first boot, PiPlay presents a quick introduction to the controls and interface which is best read and digested straight away, else it can be a little confusing later on. It comes with a great interface that can be navigated via keyboard or using a joypad. Setting up your ROMs is handled via a built-in ROM uploader, that can be reached using another device's browser and navigating to the IP address of the PiPlay Raspberry Pi. It's a pretty effective system that allows you to quickly upload your ROMs to the correct directory of your Raspberry Pi.

With the ROMs uploaded, the relevant emulators will be activated and loading it will trigger PiPlay to scrape the web for thumbnail images for your collection, creating a slick-looking library.

Configuring your joypad can be done as a default setting or per emulator, enabling you to create your perfect configuration. We used a USB SNES pad, which worked flawlessly. We also tried the Xbox 360 driver with an unofficial Xbox 360 pad, but during the configuration we found some buttons wouldn't map to the Xbox pad, which was a shame.

One of the few issues we encountered otherwise was related to audio output, which forced itself via the HDMI port no matter how often we changed the value in raspi-config. Not a major issue for home users, granted, but those building a cabinet, which normally uses an amplifier to boost sound clarity, might be disappointed.

We first tested the SNES emulator with *Super Mario World* and *Street Fighter 2*; both worked extremely well. Sadly, *Star Fox* wouldn't run, but this isn't terribly unusual. Next we tested Mega Drive emulation, firstly with *Sonic 1*. While that worked well, we had issues with *Sonic 2*, *3* and *Streets Of Rage 3*. Lastly, we tried the PlayStation emulator with *Crash Bandicoot* and while it ran fairly well, it was a little sluggish at times. Of course, it doesn't take long to exit PiPlay and use the raspi-config tool to overclock the Pi to 900MHz.

PiPlay is a lovely interface to many different emulators, but requires a few tweaks to make it better.

Final word

PiPlay offers a good collection of emulators that are easily configured to work with many games and controllers. However, there are definite issues when running some games, which are down to PiPlay.

Score 3/5

Above Mario was jumping for joy, but some other games didn't work so well

RETROPIE

Best OF THE BUNCH

RETROPIE IS THE DARLING OF THE EMULATION SCENE, THANKS TO THE MANY TOP-CLASS ARCADE BUILDS THAT ARE POWERED BY ITS SLICK INTERFACE

The RetroPie project is another emulation distribution that is based on Raspbian, and it provides a plethora of emulators. Raspbian may be the operating system, but RetroPie represents the glue that binds the OS to a beautifully simple interface called Emulation Station, a third-party themeable front end for emulation projects.

Copying RetroPie to an SD card is handled in the same manner as PiPlay, by transferring the image to the card using dd or a GUI application. On first boot, it asks the user to insert a controller to configure. We found this a little confusing, as the configuration only refers to the user interface and not the emulators contained therein.

To configure the controller for the emulators, you have to drop into the terminal and hack together a config script that covers all of the emulators, which is far from ideal for those new to emulation.

Since RetroPie emulates the same consoles as PiPlay, we tested the same SNES, Mega Drive and PlayStation emulators. First, the SNES with *Super Mario World* and then *Street Fighter 2* – both worked very well and were fluid to play. RetroPie was the only emulator to successfully load and play *Star Fox*. A quirk that we found with the SNES emulator was that it did not like ROMs saved as ZIP archives; rather, it preferred games to be unzipped before play.

For the Mega Drive we tested with *Sonic 1* and we were surprised to find that *Sonic 2*, *3* and *Streets Of Rage* all worked out of the box, putting RetroPie well ahead of PiPlay in the compatibility stakes.

Lastly, we tested the PlayStation emulator with *Crash Bandicoot* and encountered a slight stuttering during play, but nothing that a bit of overclocking couldn't solve.

Above RetroPie scrapes the net to get box art and details for games in your collection

RetroPie boasts a particularly slick user experience and comes with a ROM scraper tool, which prettifies your library of ROMs by downloading thumbnails and information from the web. You manage your ROMs via the Raspbian desktop, and RetroPie includes a great script that detects when a USB stick is inserted into the Pi. When detected, RetroPie creates a directory structure for ROM files that mirrors what is installed on the Raspberry Pi, so all you need to do is then put that USB stick into your desktop PC and copy the ROMs onto it (making sure to put them in the right folder). Now, when you put that memory stick back in your Raspberry Pi, RetroPie automatically puts them in the right directory, which is both rather clever and exceptionally useful.

Ultimately, RetroPie is a highly refined product, but one that's not aimed at the newcomer. If you're an enthusiast who wants to make their own cabinet or home entertainment solution, it's easily the best choice. It's a powerful, beautiful piece of software.

Above All the Mega Drive games we tested performed well

Final word

A seriously powerful and configurable experience that offers the best overall experience for those who know their stuff, or don't mind taking the time to learn.

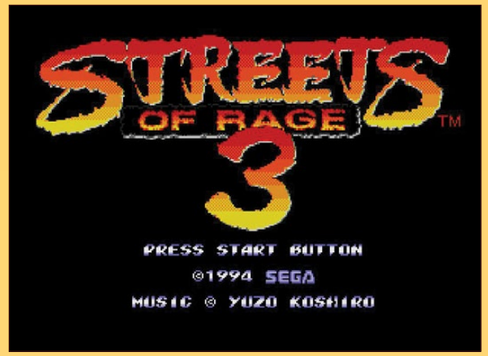
Score

RASPICADE

Best FOR NEWBIES

FINALLY, WE LOOK AT RASPICADE, WHICH IS DESIGNED TO PLAY WELL WITH THE SPECIFIC NEEDS OF ARCADE CABINETS AND INTERFACE WITH ARCADE-QUALITY JOYSTICK AND BUTTONS

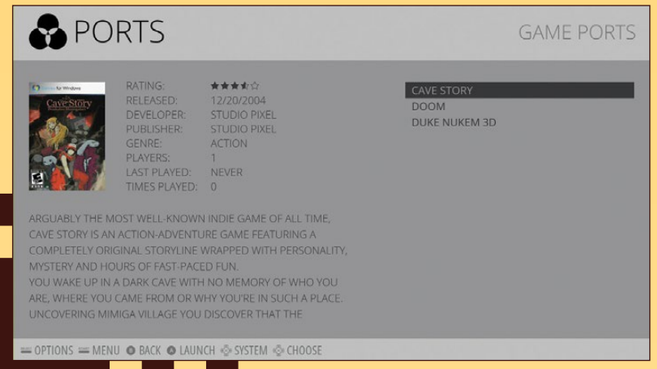
Right Raspicade rather confusingly comes with two Mega Drive emulators by default



Installation of Raspicade is equally as easy as the other two emulation offerings on test, and it took a grand total of ten minutes to get running from a standing start.

Raspicade is a little different to the others on test, inasmuch as it comes with a configuration script that runs every time the Raspberry Pi boots. Among other things, the script configures the audio output, enabling you to choose between the 3.5mm audio jack and the HDMI port, something lacking from the other solutions. The script also handles your IP address and, in another interesting twist, allows you to choose between three user interfaces: a simple Raspicade bespoke interface, Emulation Station 1 (which is an older interface but very light on resources) and Emulation Station 2, which comes as standard on RetroPie. Given the latter fact, we reviewed using Emulation Station 2 to ensure a fair test.

Below Raspicade and RetroPie share the same interface



First of all, we configured the joypad to work with Emulation Station, which seemed to go well, but found that there was no way to create a default configuration, so we needed to repeat the process for each emulator we tested. Elsewhere, we found two instances of 'Sega Genesis' in the interface; since the software used both PicoDrive and dgen Mega Drive emulators, we decided to test the latter, which is supported in all three options on test today.

As before, we tested all three *Sonic* games and *Streets Of Rage 3*. All played fluidly, with no issues to speak of. Next, the SNES emulator handled *Super Mario World* and *Street Fighter 2* with ease, but *Star Fox* refused to play along.

While we weren't terribly surprised at that, we were disappointed that Raspicade doesn't offer a PlayStation emulator, so we fired up the NES emulator and tried a built-in game called *Solar Wars*, which – unsurprisingly – worked really well.

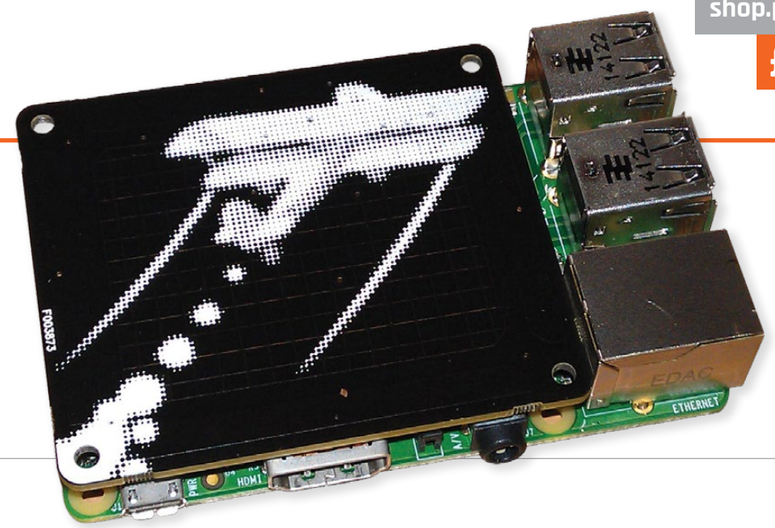
Final word

Overall, Raspicade is a good retro-gaming distro, and it provides a low barrier for entry for those wanting to dip their toes into emulation for the first time.

Score

shop.pimoroni.com

£16/$21



Maker Says

Control your Pi with a flick of the wrist

Pimoroni

SKYWRITER HAT

In our first new-look review, **Les Pounder** reaches for the sky to uncover the secrets behind the latest gadget from the 'Pirates of Sheffield'

How we interact with technology is constantly evolving – from the early days of clunky keyboards to the sensors we use to capture location and orientation automatically. What remains the most basic interaction is touch and, by extension, gestures. Gesture control can be achieved using technologies such as Microsoft's Kinect or OpenCV, but these are quite resource-intensive solutions. So a cheap, simple and resource-light solution is needed; step forward Skywriter.

Near-field sensing

The Skywriter is the latest board from Pimoroni, the Sheffield-based maker of a number of quality add-ons for the Raspberry Pi. Designed for the Pi Model A+ and B+, the Skywriter is a gesture controller built around the MGC3130, a 3D gesture recognition and tracking controller chip. It uses near-field sensing to locate the position of your hand in the air at a range of 5cm. The board can also detect touch input to a number of positions on its surface.

Physically, the Skywriter HAT measures 64mm wide by 56mm tall with a depth of 5mm. On the Raspberry Pi A+, it fits neatly over the top, and with a B+ it fits snugly in the space between the USB ports and the display slot (DSI).

Code control

Along with a neat hardware solution, the team have invested a great deal of time producing a Python API (application programming interface) for both Python 3 and 2.7, installable via the Pip package manager. Examples of how to use this board can be found in Skywriter's GitHub repository (**github.com/pimoroni/skywriter-hat**). We found the contents of **`test.py`** in the **python/examples** folder very interesting, since it puts out lots of interesting debugging data, as well as helping confirm it's working properly.

How can you use Skywriter HAT in your next project? Instantly, gesture-controlled gaming comes to mind. Using Skywriter as a gesture controller for a game of *Pac-Man* perhaps, controlling your Big Trak with just a flick of your hand or launching a rocket with a tap. We look forward to waving our hands around like Tom Cruise in *Minority Report* to take control of our next project, using nothing more than a gesture and some Python code.

Last word

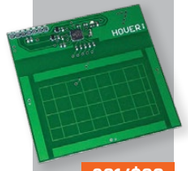
The Skywriter HAT is a small and well-constructed board that sits neatly on top of your Pi (as any HAT should). It's a really capable and versatile board that adds a novel form of input to any project and the easily understood Python API will benefit coders of all abilities.

★★★★☆

Related

HOVER

Using the same MGC3130 near-field controller chip as the Skywriter, Hover boasts the same I2C interface and compatibility with Raspberry Pi, Arduino and other single-board computers.



£31/$39

hoverlabs.co

anonymebox.com

£75/$115

Maker Says

An easy solution to anonymise you online

pi3g

ANONYMEBOX

With our every online move being monitored, can a Raspberry Pi-based appliance keep you and your family safe online? **Les Pounder** investigates...

Since George Orwell wrote *1984*, the fiction contained therein has become fact in our society. Big Brother is indeed watching you. Your ISP can throttle your speed and governments can get hold of your browser history. One of the tools enabling anyone to stay anonymous online is Tor (The Onion Router), which provides a series of relays that bounce your connection around the world via an encrypted network. Tor is configured per machine, so the Anonymebox's differing approach is more convenient.

Tor made easy

The Pi-powered Anonymebox connects to your home network so anyone can connect securely with any computer. It comes with an Ethernet cable to connect to your router, a Wi-Fi dongle and a USB-to-Ethernet adaptor. Building the Anonymebox is very easy too: insert the SD card containing the OS, connect the Ethernet cable to your router and then insert the Wi-Fi dongle before powering up.

Easy configuration

Once it's on, it can be configured via the browser of any device connected to your router. The web interface is sparse, with just an overview showing the devices connected to the Anonymebox, and a settings menu to change the admin password and configure the access point details. Your first post-setup activity involves updating the default password and, optionally, changing the default name of the access point.

Connecting to it over Wi-Fi is the same as connecting to any other router, but the Anonymebox will completely anonymise your online presence via Tor. Visiting a site such as **whatismyipaddress.com** will show that your location is in a completely different country.

Tor needs to be regularly updated to ensure protection, so the developer has created an easy-to-use method for updating. Just download an archive from its website, copy it to a USB flash drive and insert that into the spare port on your Anonymebox. The software is configured to act if an update is found. The Anonymebox is genuinely user-friendly, requiring little or no technical expertise to set up or use.

For those wanting to create their own Anonymebox, there's even a free download of the OS available from the website, along with instructions on which Wi-Fi dongle to purchase.

Last word

The Anonymebox is a good tool for those who don't have the skills to configure Tor on every machine they use, though its ease belies its true power.

★★★★☆

Related

ONION PI

It launched the Tor appliance trend, but is designed for the more technically minded of us comfortable working in the terminal.

Image courtesy of Adafruit.com

$85

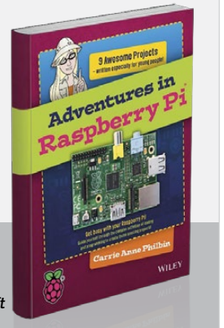
adafruit.com

RASPBERRY PI BESTSELLERS

Wiley's top three bestselling Pi books shouldn't be missed...

ADVENTURES IN RASPBERRY PI 2ND EDITION

Author: Carrie Anne Philbin
Publisher: Wiley
Price: £14.99
ISBN: 978-1118751251
tinyurl.com/oov562q



Nine projects – with encouragement, hints and tips – to take 11-15 year-olds through coding, games, *Minecraft* and music on the Pi, as well as GPIO-based projects, with accompanying videos on the website. Full review coming soon.

RASPBERRY PI USER GUIDE – 3RD EDITION

Authors: Eben Upton & Gareth Halfacree
Publisher: Wiley
Price: £14.99
ISBN: 978-1118921661
tinyurl.com/k4dc6zd

Updated for the Model B+, the official guide remains an invaluable introduction to all things Pi, particularly physical computing. Aimed at beginners, the enthusiasm and depth of knowledge give something to every reader.

ADVENTURES IN MINECRAFT

Authors: David Whale & Martin O'Hanlon
Publisher: Wiley
Price: £14.99
ISBN: 978-1118946916
tinyurl.com/oycg9qu

Spurred on by wanting to improve and customise their *Minecraft*, readers will swiftly pick up Python skills and integrate *Minecraft* with building electronic circuits. Recommended by teachers and young people alike.

PYTHON PROJECTS

Author: Laura Cassell & Alan Gauld
Publisher: Wiley
Price: £30.99
ISBN: 978-1118908662
tinyurl.com/kp9w24q

You've completed the Python tutorial – or beginner book, or MOOC – and you're ready to move on. If you learned programming in order to join or begin a particular project, no problem, but if you didn't, then where to go next to learn what you can really do with Python is a problem. Or, rather, it was before Cassell and Gauld produced this excellent compendium of ideas and projects to take you beyond beginner, and get learning by doing.

For Pi users, while physical computing is left to some pointers at the end of the book, there's still a wealth of material. Following a recap of core Python, it's straight into scripts to access the operating system, then managing data. Useful exercises, along with summaries of what's been learned, cement the education experience.

After desktop and web applications comes the tools you need for working on larger projects: testing, debugging, tuning, structuring, and releasing – all essential information. The book closes with a look at areas you may want to try next: SciPy, Pygame, drawing modules and animation, as well as the aforementioned physical computing. Plenty to inspire you to further Python adventures.

Score ★★★★★

BLACK HAT PYTHON

Author: Justin Seitz
Publisher: No Starch
Price: £23.50
ISBN: 978-1593275907
tinyurl.com/pts7hm6

Python is a popular choice in the field of information security, and penetration testing in particular. Seitz, a senior security researcher at Immunity, presents a broad range of security topics, touching on tools traditionally used, then pointing the way to build your own Python equivalents.

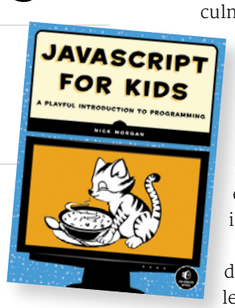
Like most good security books, it reveals what an insecure place our computer networks are, but provides you with the tools to do something about it, building Python replacements for many everyday tools like Netcat. This leads to stronger knowledge not just of the network security topics, but also where you can take Python. Along the way, Seitz respects the intelligence of the reader, but doesn't assume detailed networking knowledge – introducing information if it is necessary for progressing through the book. For example, SSH tunnelling is explained, but the reader is left to look up any extra information she may want on the Address Resolution Protocol (ARP).

The chapter on 'Web Hackery' will be of particular general interest. Anyone with a Joomla, Drupal, WordPress or similar site can feel justifiably alarmed about their security after a few pages of Python brute-forcing scripts, from discovering leftover files and scripts on the server to gaining admin login. A useful eye-opener.

Score ★★★☆☆

JAVASCRIPT FOR KIDS

Author: Nick Morgan
Publisher: No Starch
Price: £23.50
ISBN: 978-1593274085
tinyurl.com/pz8zyxx



The Pi is a great tool for learning Python, but also anything else about computing. For an even more take-anywhere, instant-results language, try JavaScript. It has a runtime – the web – that's familiar to everyone, and with a bit of HTML and CSS thrown in, children soon gain an understanding of all things web that will spur their creativity on to produce their own sites.

Coding is done straight into Chrome's JavaScript Console, which cuts out install hassles and time wasted on discussing editor choice! While pursuing games and animations, readers will learn objects, arrays, click events, flow control, Boolean operators, and HTML. This culminates in a graphical Snake game, using the canvas element, and suggestions of where to go next. Extra programming challenges, ending each chapter, balance interest and learning.

This is no dumbed-down guide – you can learn everything you need here at the same time as your children – but the material is well judged (Random Insult Generator etc), and well paced to teach JavaScript and programming skills in easy but satisfying steps to younger learners. Add in cute illustrations from Miran Lipovaa, of *Learn You a Haskell* fame, and you have a real winner. Recommended.

Score ★★★★☆

LAUREN IPSUM

Author: Carlos Bueno
Publisher: No Starch
Price: £11.50
ISBN: 978-1593275747
laurenipsum.org

Wow! The number of great coding books to appear in the last few years, aimed specifically at children, has been a very encouraging part of the movement to get young people coding. But coding – although fun, creative and rewarding – is only a part of computational thinking, which is a set of problem-solving skills including, but not limited to, algorithms, data modelling and logical thinking – invaluable in today's world.

A computer science book with no computers in, written in the grand tradition of Lewis Carroll, it follows the adventures and encounters of Lauren Ipsum, lost in Userland. From Recursion Junction to the Push & Pop Café, meeting characters like Hugh Rustic and the Wandering Salesman, Lauren's journey takes the reader through a history of ideas and logic.

Bueno has an engaging style and the lessons are so integral to the characters (Zeno's tortoise finds infinity in two inches of string) and stories (your kids will want to try the circle-drawing algorithms in Logo or Scratch) that there is no feeling of forced learning. Recommended for anyone of any age who wants to learn, this book would make a great introduction for schools grappling with the new curriculum – were it not too entertaining to be a textbook.

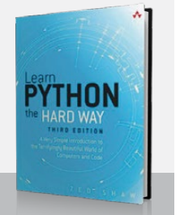
Score ★★★★★

ESSENTIAL READING: PYTHON

Python makes a great first language, but choose a book that matches your learning style...

Learn Python the Hard Way (Third Edition)

Author: Zed Shaw
Publisher: Addison Wesley
Price: £24.99 (free online)
ISBN: 978-0321884916
learnpythonthehardway.org



The 'hard way' is typing it all in until you absorb the syntax and spot mistakes. Works well, but Shaw doesn't cover Python 3.

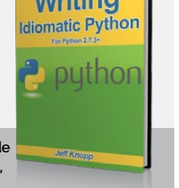
Program Arcade Games: With Python and Pygame

Author: Dr Paul Vincent Craven
Publisher: CreateSpace
Price: £21.99 (free online)
ISBN: 978-1500825966
programarcadegames.com

Balances games and programming exercises to keep the learner going. Very popular: available in several languages on the website.

Writing Idiomatic Python 3.3

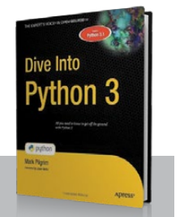
Author: Jeff Knupp
Publisher: CreateSpace
Price: £13.97
ISBN: 978-1482374810
tinyurl.com/bgj8zq9



Get Pythonic from the start. Concise guide to idiomatic code; best after another text, but suits some brave learners.

Dive Into Python 3

Author: Mark Pilgrim
Publisher: APress
Price: £35.49
ISBN: 978-1430224150
diveintopython3.net



Dives straight into code, then the explanations follow. A concise but comprehensive start that will appeal to independent study types.

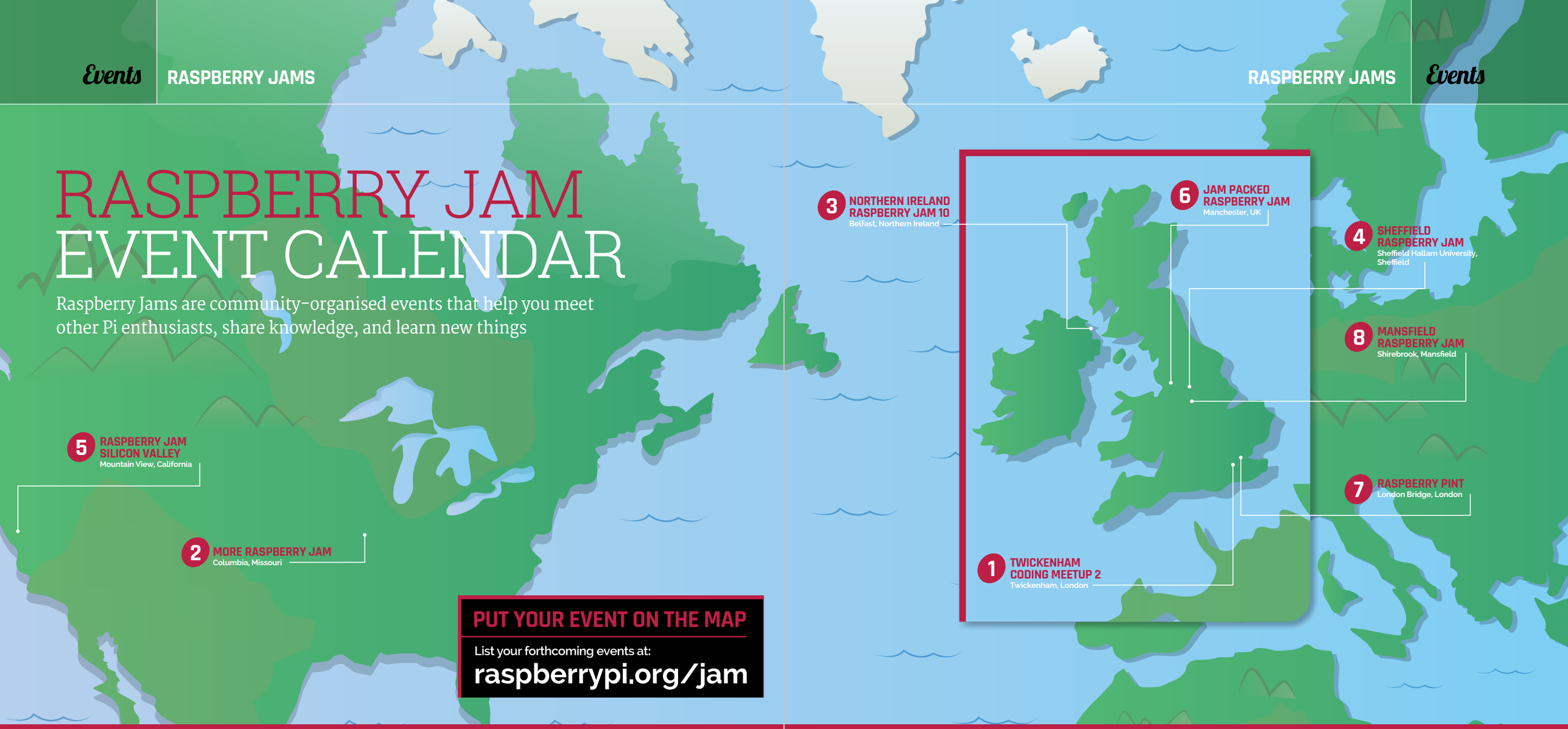
Learning Python – 5th Edition

Author: Mark Lutz
Publisher: O'Reilly
Price: £43.50
ISBN: 978-1449355739
tinyurl.com/mehbnpo

Comprehensive doorstop (1,600 pages); great for programmers new to Python and object orientation. Covers Python 2.7 and 3.3.

RASPBERRY JAM EVENT CALENDAR

Raspberry Jams are community-organised events that help you meet other Pi enthusiasts, share knowledge, and learn new things



PUT YOUR EVENT ON THE MAP

List your forthcoming events at:

raspberrypi.org/jam

1 TWICKENHAM CODING MEETUP 2

When: Friday 13 March
Where: 13 Richmond Road, London, TW1 3AB
bit.ly/1EUUhBq
An informal event for teachers and volunteers to promote coding in schools.

2 MORE RASPBERRY JAM

When: Saturday 14 March
Where: MOREnet, Columbia, Missouri 65203
on.fb.me/1zSgjAq
This jam is hosted by The Missouri Research and Education Network (MOREnet).

3 NORTHERN IRELAND JAM 10

When: Saturday 14 March
Where: Farset Labs, Belfast, BT12 5GH
bit.ly/1DA9tm0
A regular gathering for everyone from complete beginners to advanced enthusiasts.

4 SHEFFIELD RASPBERRY JAM

When: Thursday 19 March
Where: Sheffield Hallam University, S1 1WB
bit.ly/1CPcX50
The event starts with light refreshments at 17:30 and closes at 19:30.

5 RASPBERRY JAM SILICON VALLEY

When: Saturday 21 March
Where: Computer History Museum, CA
bit.ly/1IKPPZ1
The Silicon Valley Jam takes place on the third Saturday of every month.

6 MANCHESTER RASPBERRY JAM

When: Saturday 28 March
Where: MCA, Manchester, M40 8NT
bit.ly/1FtHq6Z
This get-together is part of a Jam Packed two-day event organised by Alan O'Donohoe.

7 RASPBERRY PINT

When: Tuesday 31 March
Where: The Miller, London Bridge, SE1 3SS
meetup.com/Raspberry-Pint-London
Meet other makers and educators who love open source hardware and software including the Raspberry Pi, in the centre of London.

8 MANSFIELD RASPBERRY JAM

When: Saturday 18 April
Where: Shirebrook Academy, Mansfield, NG20 8QF
bit.ly/1y609Cu
Join children and adults of all ages for a fun day of computing, talks, and workshops.

DON'T MISS: RASPBERRY PINT

When: Tuesday 31 March **Where:** London Bridge

Fancy a pint? Meet other makers, technologists, educators and people-in-sheds engaged in using, or simply interested in, the new wave of affordable and accessible technology led by the Raspberry Pi.

Raspberry Pint takes place on the last Wednesday of every month at The Miller, 96 Snowsfields Road, London, SE1 3SS and was formed to inspire the use of open source hardware and software in projects for fun, education and creation through technology. Learn more at **meetup.com/Raspberry-Pint-London**

In association with PIMORONI

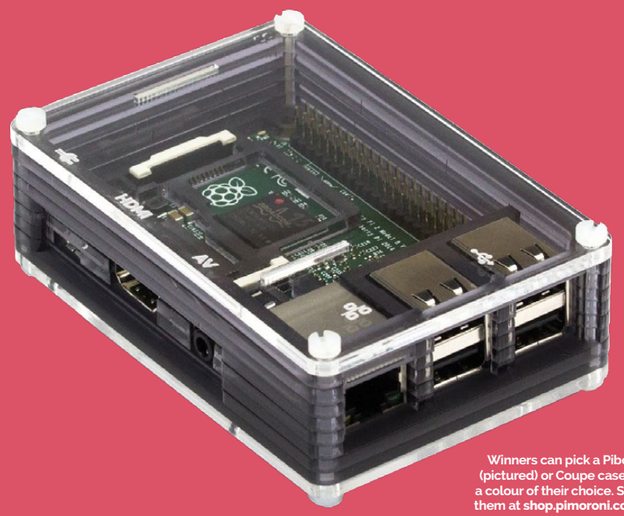
5 Pi 2s must be WON!

WHAT WOULD YOU DO WITH A RASPBERRY PI 2?

Let us know for your chance to win!

How to enter:

For your chance to win a Raspberry Pi 2 with a Pimoroni case of your choice, tell us what your dream Raspberry Pi project is. Simply email **competition@raspberrypi.org** with no more than 100 words detailing your perfect project, hack, or make.



Winners can pick a Pibow (pictured) or Coupe case in a colour of their choice. See them at **shop.pimoroni.com**

Terms & Conditions

Competition closes 30th March 2015. 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.

MATT RICHARDSON

Matt is Raspberry Pi's US-based evangelist. He's the co-author of *Getting Started with Raspberry Pi* and was a contributing editor at *Make:* magazine.

THE FIRST OF MANY STEPS

One of the newest members of the Raspberry Pi team, **Matt Richardson**, closes the issue with memories of his first experiences with the world's favourite credit card-sized PC...

Since this is the start of a new era for *The MagPi*, I wanted to use it as an opportunity to reflect on my beginnings with Raspberry Pi. Admittedly, when I first heard of it before it was released, I was sceptical. "A small Linux computer for only $35? I'll believe it when I see it," I thought. I predicted it would either go up in price or down in history as another example of vapourware. I'm glad that I got it so wrong.

I was shocked when the Foundation started accepting pre-orders for the Model B. The hype was reaching fever pitch and it forced me take this scrappy little computer a lot more seriously. At that time, Arduino was making hardware development more accessible than ever. What if all the powerful tools developed for Linux could be blended with what people were creating with Arduino? For example, writing or reading a file with Arduino was possible, albeit difficult. The same was true for networking, timekeeping, video display, and multitasking. I knew exactly what I wanted to do with Raspberry Pi, but I didn't think much further than that.

The hallelujah moment

I boned up on how Linux controls and reads the GPIO pins in anticipation of getting my first Raspberry Pi in the spring of 2012. One of the first things I did with it was to get an LED to illuminate by typing commands into the Linux command prompt. It was glorious. I often talk about this as one of my 'angels-singing-hallelujah' moments!

So of course I was mostly excited about physical computing projects that would allow me to wield this new power that was just bestowed upon me. I experimented with making web-based controls for the pins with Flask, the Python-based web framework. I used computer vision libraries to create a photo booth with an obligatory digital moustache superimposed on the subjects. With a small, battery-powered projector, I created a dynamic bicycle headlight, which shows the speed that I'm riding — among other possible data points — in the beam on the ground in front of me.

Beyond physical computing

Physical computing projects were only my initial motivation to try Raspberry Pi. Since then, I've started learning how to work with Wolfram and Mathematica, which is included with Raspbian. I've also used Sonic Pi to experiment with synthesizing electronic music. I used a Raspberry Pi to set up a network server for my 3D printer. And now I'm learning low-level assembly programming by following one of the 'bare metal' resources available for Raspberry Pi. It's teaching me a lot about how computers and operating systems work. After that, perhaps I'll hook a Pi up to a TV and use it as a media player.

What's important is that the Raspberry Pi is so versatile that there's not just one possible first step with it. Whether you want to use it to learn basic Linux, acquaint yourself with Python, or you need a local file server for your family, Raspberry Pi fits the bill quite nicely. And if you're anything like me, after Raspberry Pi gets a tiny toehold in your consciousness, soon you won't stop thinking of ways that you can use it. Clearly there's a host of possible motivations for using Raspberry Pi. I encourage you to think about yours as only a starting point for what you ultimately do with it.

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

Breakout Pi Plus

The Breakout Pi Plus is a useful and versatile prototyping expansion board for the Raspberry Pi

ADC Pi Plus

8 channel analogue to digital converter. I²C address selection allows you to add up to 32 analogue channels to your Raspberry Pi.

IO Pi Plus

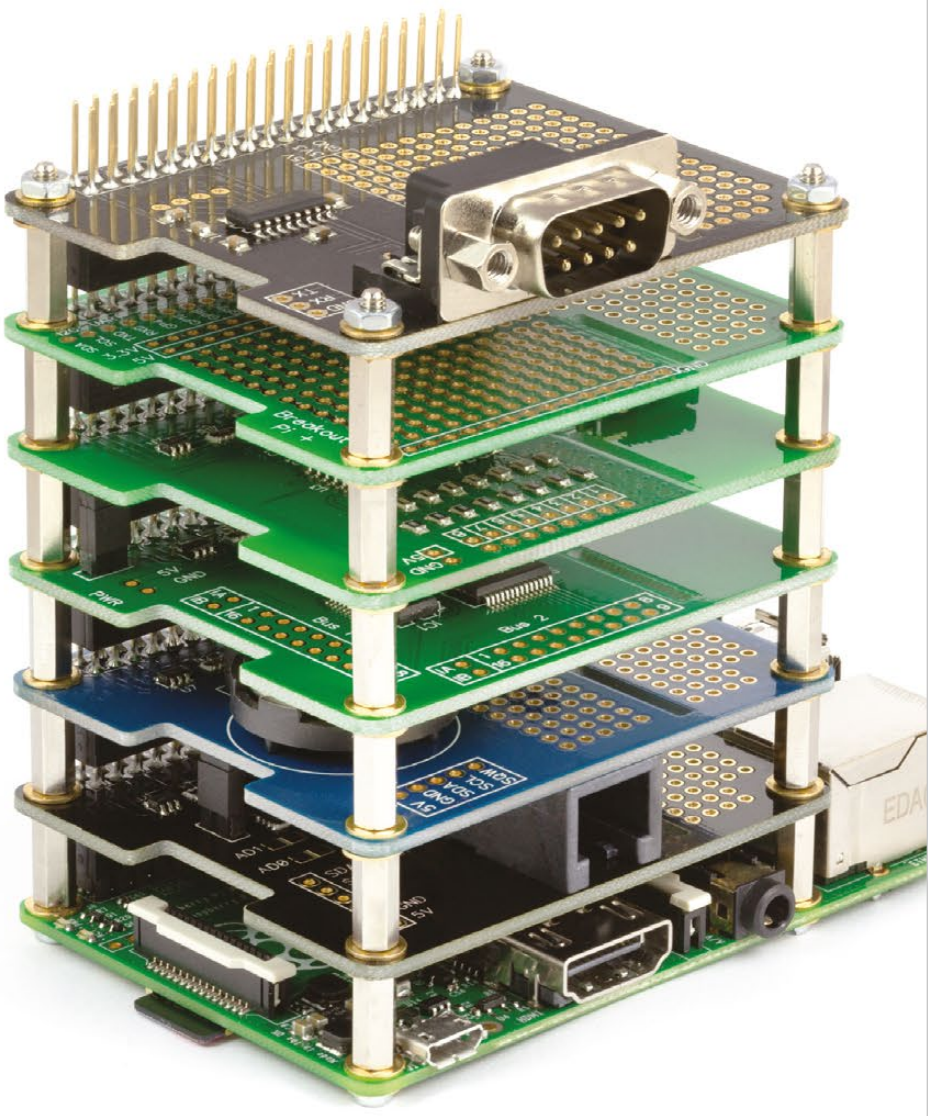
32 digital 5V inputs or outputs. I²C address selection allows you to stack up to 4 IO Pi Plus boards on your Raspberry Pi.

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.



We also stock a wide range of expansion boards for the original Raspberry Pi models A and B

AB electronics UK

www.abelectronics.co.uk