FREE RESOURCES FOR THE CLASSROOM

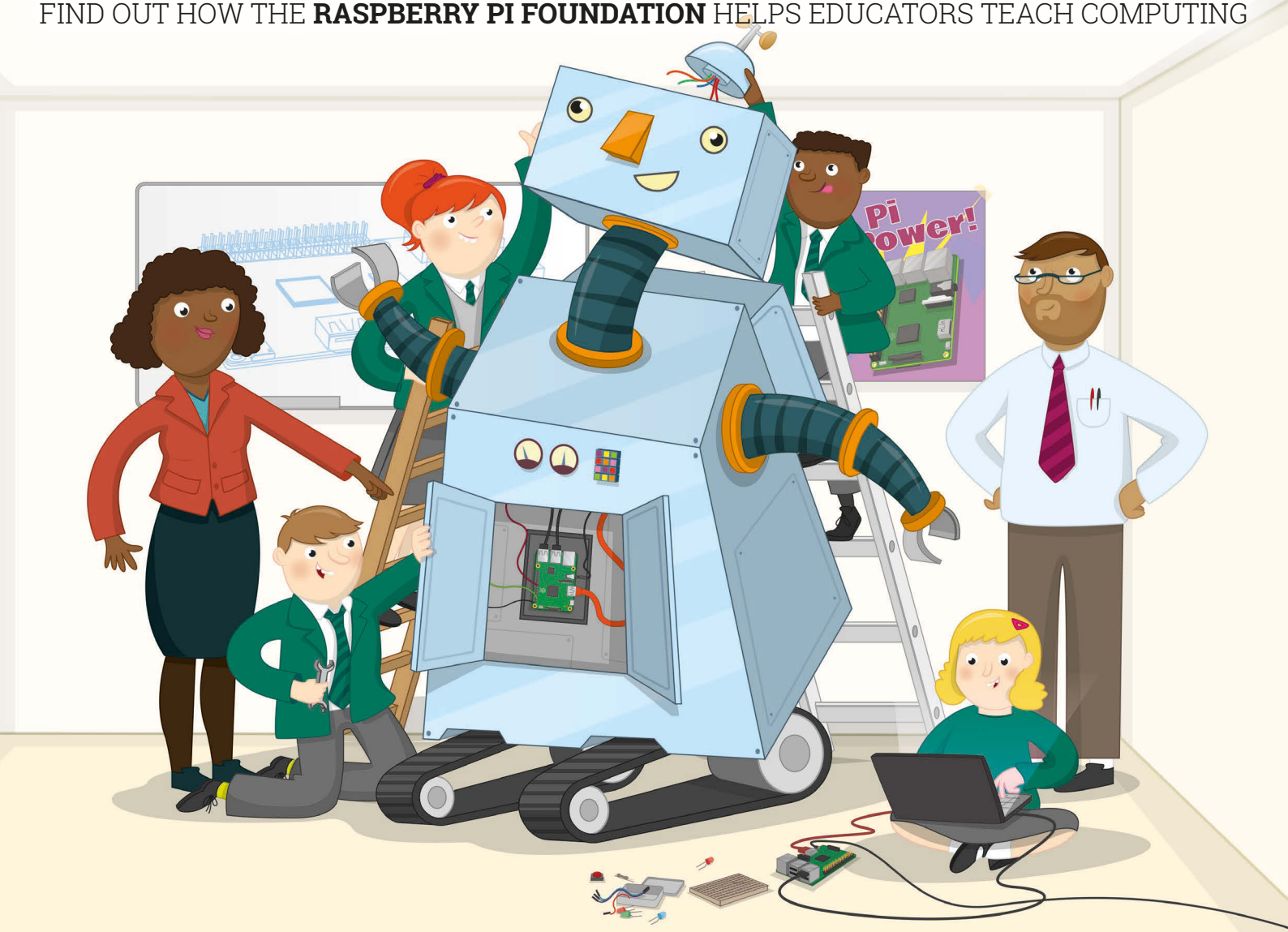
# The MagPi

EDUCATOR'S EDITION

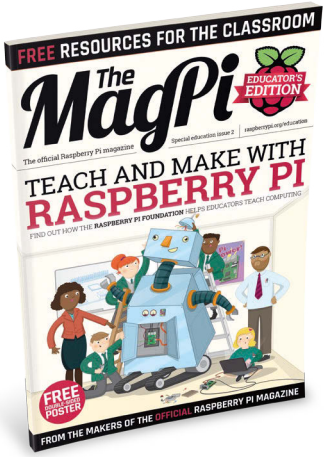
The official Raspberry Pi magazine | Special education issue 2 | raspberrypi.org/education

# TEACH AND MAKE WITH RASPBERRY PI

FIND OUT HOW THE **RASPBERRY PI FOUNDATION** HELPS EDUCATORS TEACH COMPUTING




FROM THE MAKERS OF THE OFFICIAL RASPBERRY PI MAGAZINE



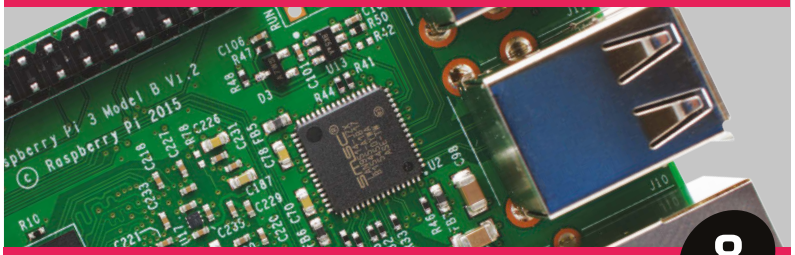

# Contents

Education special issue 2

raspberrypi.org/education



## GET STARTED WITH RASPBERRY PI



Get acquainted with the basics of setting up and using the world's favourite credit card-sized PC. It's easier than you think!



## THE DIGITAL MAKING REVOLUTION



Find out how the maker movement is revolutionising the way modern education is being taught around the world

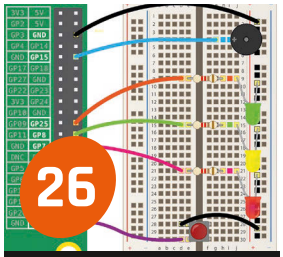
## START YOUR OWN CODE CLUB



{code club}

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

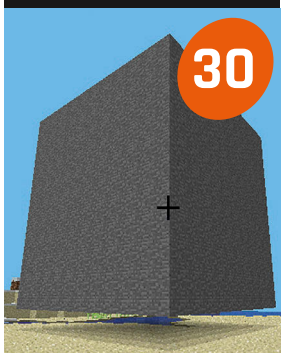
## FREE COMPUTING RESOURCES





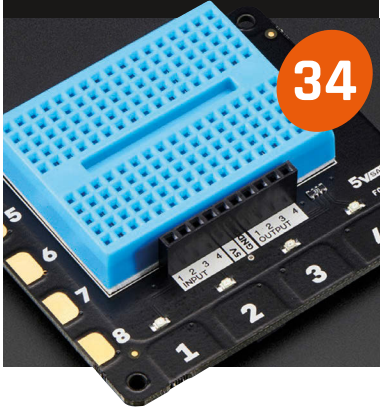
PHYSICAL COMPUTING WITH SCRATCH

CODE WITH MINECRAFT





GET STARTED WITH EXPLORER HAT PRO



# WELCOME TO THE RASPBERRY PI COMMUNITY

Find out how the Raspberry Pi Foundation supports a global community of educators and how you can get involved

The chances are that you've heard about Raspberry Pi, the low-cost, credit card-sized computer that was developed to encourage kids to learn how to code. We launched the world's first $35 computer in 2012. By the time of our fourth birthday in February 2016, we'd sold over eight million and helped kick-start a global movement to create the next generation of digital makers.

What's perhaps less well known is that the Raspberry Pi Foundation is much more than a computer company. We're a UK-based educational charity with a mission to put the power of digital making into the hands of people all over the world. One of the ways that we pursue this mission is by providing low-cost, high-powered computers, but it isn't all we do.

## A global community of educators

At the heart of Raspberry Pi is a global community of educators who are working inside and outside the classroom to inspire young people to get creative with technology. Our job is to provide that community with the support that they need. One of the ways we do this is by developing high-quality teaching resources and projects, many of which don't require a Raspberry Pi computer. They have all been designed by educators, and are available for free.

> "We're proud to be part of a movement which aims to empower people to shape their world"

## More than a computer company

Because we're a charity, we're able to use any and all profits that we generate from our commercial activity to invest in educational programmes and outreach, resources, training and support for educators, and building a global community that shares our mission.

Through our network of Code Clubs (see page 22), we're making sure that opportunities to get involved in digital making are as widely available as possible, mobilising a huge community of volunteers and educators in the process.

Through programmes like Astro Pi (see page 52), we're helping to make computing more relevant to young people who might not have thought that digital making was for them, but who are excited by human space exploration. We're doing the same with music, nature, and the arts, taking a deliberately cross-curricular approach to engage young people with very different interests.

Over the past three years, we've also trained hundreds of Raspberry Pi Certified Educators through our Picademy programme of free professional development. It has been amazing for us to see so many of those Certified Educators go on to support other educators to develop their practice, whether as CAS master teachers, by organising meet-ups, or by creating and sharing their own resources. This is the most exciting part of our work: seeing the community of educators grow and support each other. We're constantly inspired by what they do.

We're proud to be part of a movement which aims to empower people to shape their world through digital technologies. If you're not involved already, then I hope you'll be inspired to take the first steps.

**Philip Colligan,**
**CEO, Raspberry Pi Foundation**

# DIGITAL MAKING REVOLUTION IN EDUCATION

The digital maker movement, a mix of traditional artisan arts and crafts combined with computer programming and electronics, has been taking the world by storm. Its ethos of tinkering and inventing is being used in the classroom to inspire a whole new generation of makers

## HACKATHONS

Hackathons are events at which groups of individuals will build a digital product from scratch, often over a single 24-hour period. Fuelled by pizza and soda, as well as by their own enthusiasm, students can work together to build anything from internet-connected Christmas trees to the next great social networking app. The events are competitive but very supportive, and always lots of fun.

Hackathons are a great way to encourage creativity, problem-solving, and teamwork within the sphere of computing and digital making. There are plenty of student hackathons organised all over the country, and **mlh.io** is a good place to start if you're looking for an event near you.

Raspberry Pi computers, and other similar devices, are unlike the traditional computers you'd find in classrooms up and down the country. Rather than a hermetically sealed box, designed specifically to prevent a student from poking and prodding around with the internal components, you are presented with a single-board computer with all the parts exposed. This design decision is not an accident.

We want to demystify computers, to allow children to see that there's nothing to be afraid of, to show them exactly where the operating system can be found, and to let them experiment with controlling electronics using the Pi's General-Purpose Input/Output (GPIO) pins.

The design also poses questions for teachers who wish to deploy Raspberry Pi computers in their classrooms. Do you have them as fixed pieces of equipment, permanently attached to power supplies, keyboards, and mice? Do you use cases, and if so, which one do you choose? Do you give students their own SD card, or should they share? How do you keep the software up to date and ensure that students can always access their work?

These are all natural concerns, especially if you come from a traditional teaching background, but perhaps the first question you should be asking yourself is 'How do I make computing more engaging and relevant for my students?'

## Invent to learn

A recent Nesta report found huge positivity towards digital making: 82% of young people say they are interested in digital making and their parents are overwhelmingly supportive. As regards parents, 89% think digital making is a worthwhile activity for their children, and 73% encourage their children to make things with technology.

So how can you tap into this wealth of enthusiasm in your classroom? The first step is not to worry too much about the practicalities of using hardware such as the Raspberry Pi, Arduino, or BBC micro:bit. Things are going to get messy, no matter which platform you use, but that's part of the learning process for you and your students. In recent years, much has been made of the long-forgotten art of tinkering. Children have been encouraged to not break anything, but this is essential to discovering what works and what doesn't. The word 'fail' in digital making is used as an acronym for 'first attempt in learning'.

By bringing physical computing devices into the classroom, students gain from learning how to set kit up themselves, including monitors, keyboards, and mice. Playing with breadboards, buttons, and other electronic components teaches students to not be afraid of technology.

## Tackling the digital divide

Computing skills can provide opportunities for social mobility. A recent Naace report on Computing in the National Curriculum discusses this, noting that: "those who excel… are in high demand across large parts of the economy". The report advises that it's important to ensure that all pupils have the opportunity to study outside the classroom. Free and/or open-source software and low-cost hardware massively lower the barriers to participation in computing. Raspberry Pi Certified Educator James Robinson explains:

"We asked our GCSE class to purchase Raspberry Pi computers to use both at home and at school and, where appropriate, we were able to use pupil premium funds to support students. This meant that our entire cohort had their own general-purpose computer they could use for classwork, but more importantly for their own projects.

"Once all the students had access to identical hardware, we were able to set much more challenging, open-ended, and engaging tasks. We used some lessons to cover the basics of programming a Minecraft world while students worked on project-based homework. The results were fantastic: each student interpreted the brief differently and used varying techniques to solve the problem."


## RASPBERRY JAMS

Raspberry Jams are community-organised events with a focus on digital making and the Raspberry Pi computer. Jams are filled with talks, show-and-tell sessions, and workshops, all showcasing the wonderful, useful and often wacky projects that can be created with a Raspberry Pi, a little skill, and a lot of creativity. If you're interested in attending a Jam near you, then check out **raspberrypi.org/jam** for a list of upcoming events.

## How the Foundation supports makers

Bringing the maker movement into the classroom is one of the charitable aims of The Raspberry Pi Foundation. Our team of experienced educators write fun, engaging, and flexible learning resources, all of which are linked to curriculum objectives. They are published under a Creative Commons licence that gives you the flexibility to adapt them to suit your needs. We also provide free professional development for teachers worldwide through our Picademy programme. You can learn more about our training initiatives on page 58.

> "82% of young people say they are interested in digital making and their parents are overwhelmingly supportive"

We've sent two Pis and Sense HATs to the International Space Station as part of British ESA Astronaut Tim Peake's Principia mission. In 2015 we also gave away 1,000 Raspberry Pi weather stations worldwide for students to build, program, and collect data. We also work closely with developers and academics to build education-tailored applications like Sonic Pi, a cross-platform programming and music-making tool.

Keen to find out more and get involved? Visit our education webpage for access to news, events, free resources, our educator community, maker project articles, and much more! **rpf.io/edu**

FIND THIS & OTHER RESOURCES: rpf.io/learn

# ADVANCING COMPUTER SCIENCE WITH RASPBERRY PI

The Raspberry Pi Foundation produces free resources for learners and educators all over the world. Content and Curriculum Manager **Marc Scott** explains why and how...

One of the core goals of the Raspberry Pi Foundation is to advance computer science education, and one of the key ways in which we attempt to achieve this is in the publication of free educational resources. Our aim is to be the first port of call for educational materials, whether you're a learner who has just unboxed your first Raspberry Pi computer, or an educator teaching computer science to eager students of any age and level.

## SIMULATING WEIGHTLESSNESS

In this resource you can learn how to simulate the effects of weightlessness in space using Scratch. The resource teaches some fundamental computer science concepts such as sequencing, looping, variables, and conditionals. It's an ideal activity for those just starting out with a Raspberry Pi and wanting to find out a little bit more about learning to program in Scratch. See it at **rpf.io/weightless**.

## Teach, Learn and Make

Head on over to **raspberrypi.org/resources** and you'll find three categories of resources to choose from: Teach, Learn, and Make.

Our rapidly growing set of Teach resources are designed with educators in mind. There you can find complete sets of lesson plans, student worksheets, and resource files ready for you to print off, project, or for your students to consult. Some schemes require Raspberry Pis, such as the networking lessons. Some are cross-platform, such as the Sonic Pi lessons on making music. Some tackle tricky concepts in computer science, such as the sorting algorithm lessons. There are even some cross-curricular schemes, such as Sensing Space.

Our Learn and Make resources are designed to be accessed by learners, and do not necessarily require the support of an educator. They could also easily be used in lessons and are especially suitable for after-school or lunchtime clubs. If you've ever wanted to build a device to tell you how many astronauts are currently in space, predict the future with a digital Magic 8 Ball, or learn what it's like to float around in zero G, this is the place to go. Our Learn resources focus on developing new skills and knowledge through project-based learning, while the Make resources engage learners with fun and often hands-on software and hardware projects.

We add new resources to the site almost every week, so it's worthwhile checking back every now and then to see what new and exciting projects we've been dreaming up lately.

## FLAPPY ASTRONAUT

This resource is for the more advanced learner. It uses the Sense HAT hardware, along with some Python code, to produce a clone of Flappy Bird, in which the player navigates between pipes by vigorously shaking the Raspberry Pi to overcome gravity. It's a tricky game to master, but a lot of fun to play. Test it yourself at **rpf.io/flappy**.

### Free as in beer and speech

A key difference between the Raspberry Pi Foundation resources and those produced by many other organisations is that ours are free. That's free in both senses of the word!

Both time and money are finite, and we recognise that educators can spare little of either. That's why we've created these resources for you to use in any way you wish, and why you'll never be charged for downloading, printing, and sharing them. In fact, we would encourage you to do all those things.

We're big believers in open-source software at the Foundation, and our attitude to educational resources mirrors this belief. The licence under which we publish our work is very permissive. We use a Creative Commons Attribution and Share-Alike licence for all our educational materials, and encourage others to do the same. This means that if you want to make changes to our resources, you are welcome to go ahead and do so. If you want to alter our work so that it's more appropriate to your learners, better suited to your teaching style, or just plain better than the original, then that's not a problem. Copy the text, change it in whatever way you like, and then publish it yourself. All we ask is that you attribute us as the original authors, and that any derivative works you produce are shared under the same licence.

> "Both time and money are finite, and we recognise that educators can spare little of either"

### Get involved

We're a very small team here at the Raspberry Pi Foundation, and as such we rely a huge amount on our wonderful community of enthusiasts, tinkerers, and educators. Not only do they use our resources, they contribute as well, doing anything from correcting a few typos right up to producing full schemes of work for us to publish on our site. You could get involved too, and help to improve or create educational materials. For the technically minded, just head on over to our GitHub page at **github.com/raspberrypilearning**, where you can fork our resource repositories, make any improvements you like, and then submit a pull request for us to look over. If the idea of using Git is a little bit daunting, though, you can just use GitHub's reporting feature to raise an issue and explain what the problem is; we'll get the message and can make the changes for you.

If you want to have a go at making a resource yourself, then go ahead. Share it with us and if we like it, you could well find your work being proudly displayed on our website.

# GETTING STARTED WITH RASPBERRY PI

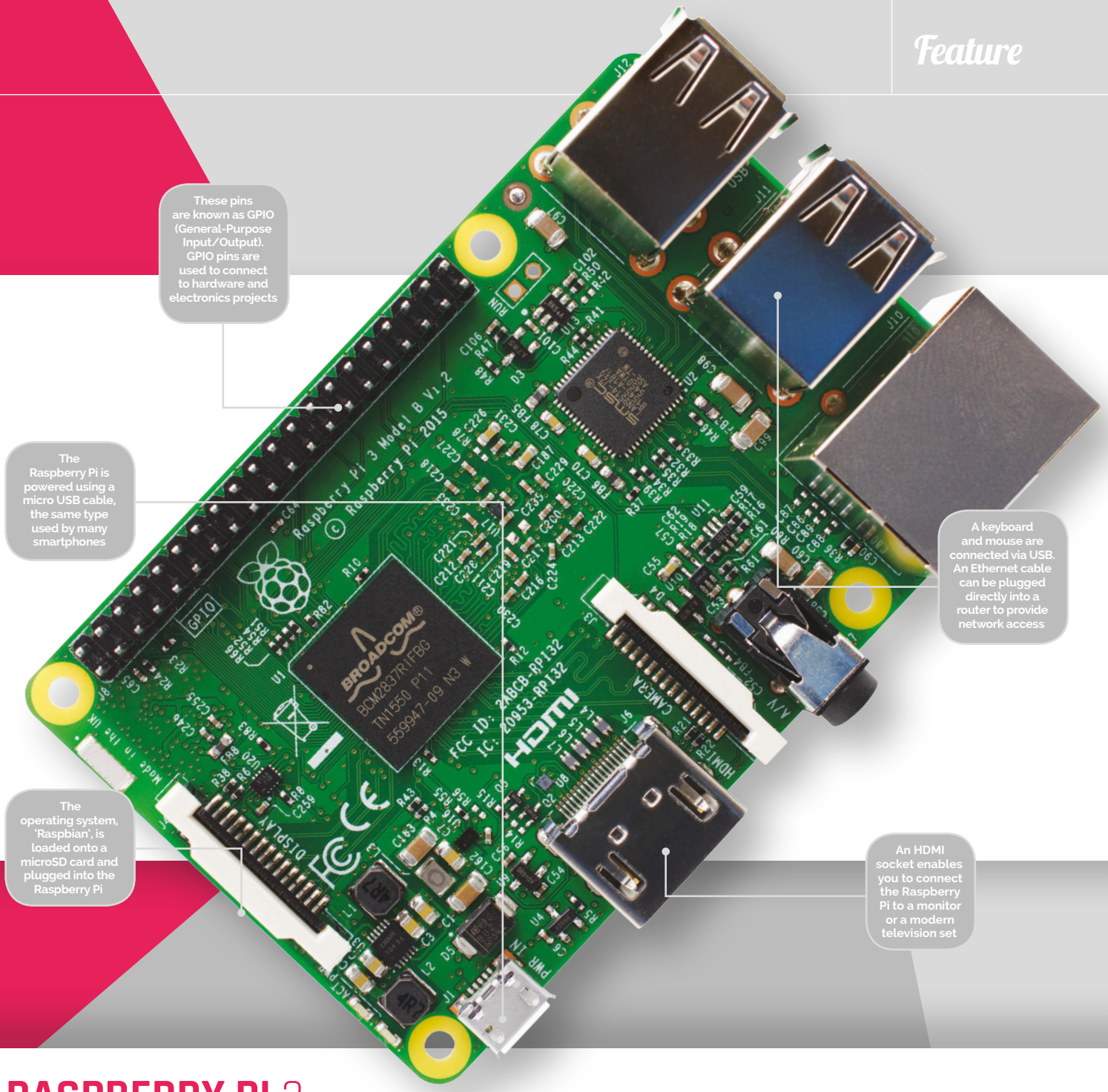
Creating amazing projects is easy with a Raspberry Pi, but first you need to plug it in and set up Raspbian, the default operating system. This guide will get you up and running in no time

The Raspberry Pi is a wonderful microcomputer that is brimming with potential. With a Raspberry Pi you can build robots, learn to code, and create all kinds of weird and wonderful projects.

Hackers and enthusiasts have turned Raspberry Pi boards into fully automated weather stations, internet-connected beehives, motorised skateboards, and much more. The only limit is your imagination.

But first, you need to start at the beginning. On picking your Raspberry Pi up for the first time, you're faced with a small green board of chips and sockets and may have no idea what to do with it. Before you can start building the project of your dreams, you'll need to get the basics sorted: keyboard, mouse, display, and operating system.

Creating projects with a Raspberry Pi is fun once you've mastered the basics. So in this guide, we're going to take you from newbie zero to Raspberry Pi hero. Grab your Raspberry Pi and let's get going...




# RASPBERRY PI 3

## The Raspberry Pi 3 is the latest model, and the version recommended for most newcomers

### SD card

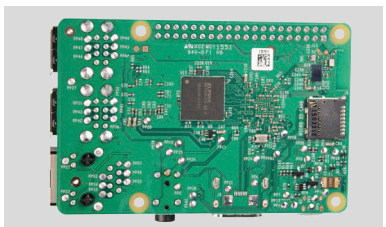
On the underside of the Raspberry Pi 3 board is the SD card slot. You preload the operating system onto a microSD card and use it to boot up the Raspberry Pi.

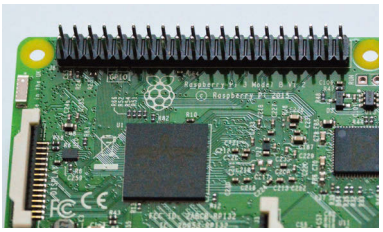
### Wireless network

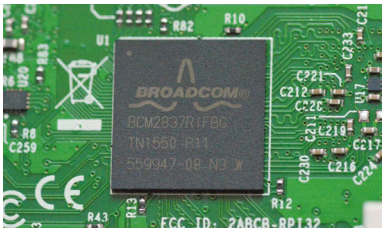
The Pi 3 is the first Raspberry Pi to feature built-in wireless LAN and Bluetooth. This enables you to connect to a wireless router and get online without using a WiFi dongle.

### 1.2GHz ARM CPU

Featuring the latest 1.2GHz quad-core ARM CPU (central processing unit), the Raspberry Pi 3 is faster than many smartphones, and powerful enough to be used as a desktop computer.







# EQUIPMENT YOU'LL NEED

## All the kit you need to get a Raspberry Pi up and running for the first time

You don't require much to get your Raspberry Pi started: a microSD card from an old camera, a smartphone charger, a recycled HDMI cable, and a keyboard and mouse are all you need.

Most items can be sourced from computer hardware around the house, or begged and borrowed from friends and family. If you're looking for the ultimate in low-cost computing; the Raspberry Pi is it.

You should be able to source, salvage, and scavenge most equipment you need to get a Raspberry Pi up and running. To get the most out of your Raspberry Pi in the long term, though, you should use high-quality components.

A good microSD card from a named brand will be faster and more reliable. Not all USB power adapters are born equal, either. A reliable branded adapter will provide a steady stream of power, even when you attach multiple devices.

The Raspberry Pi board isn't shy, and it'll work just fine naked, but a good case keeps the board safer and makes it easier to store. There's a huge range of cases available, and many offer unique features such as waterproofing, stackability, or wall mounting.

The official Raspberry Pi case is a slick piece of kit that's perfect for any Pi user. Made of five parts that click together, it enables you to quickly open the case and access the board and GPIO pins.

Any equipment you can't recycle can be picked up via the Raspberry Pi products page (**magpi.cc/2bnamFF**), bought from distributors like Element14 (**element14.com**), Allied Electronics (**alliedelec.com**), and RS Components (**magpi.cc/2bnapBl**).

## MICRO SD CARD

The microSD card acts as the hard drive for your Raspberry Pi. You install the Raspbian operating system onto the card, then all your documents, files, and projects are saved to it as you work.

Raspberry PI fan Jeff Geerling did a community favour by purchasing over a dozen different microSD cards and benchmarking each one. The results were pretty dramatic, with some cards running up to four times as fast as others. Samsung Evo+ and SanDisk Extreme are two popular brands worth looking out for, and both are fairly cheap. You can read more at **magpi.cc/2bncFs3**


## HDMI cable

An HDMI cable is the easiest way to connect your Raspberry Pi to a computer monitor or television. You don't need an expensive one, and most people recycle one from an old games console or DVD player.

## USB power

A good 2A or 2.5A power supply provides you with enough power to run a Raspberry Pi with all kinds of peripherals connected. You can buy an official Universal Power Supply (**magpi.cc/2a14pye**).

## Keyboard

Any standard USB keyboard can be used to give commands to your Raspberry Pi. You can use a Bluetooth keyboard with the Raspberry Pi 3, or any other Pi model with a Bluetooth dongle attached. A wired keyboard is easier to use when setting up your Raspberry Pi.


## Mouse

Any standard mouse will work with the Raspberry Pi, although ones with two buttons (non-Apple mice) work better. Like keyboards, a Bluetooth mouse will work once it's paired, but a wired mouse works as soon as you plug it in.

# INSTALLING RASPBIAN

Discover how to use NOOBS to quickly set up the Raspbian operating system on your Raspberry Pi

Before you start using your Raspberry Pi, it needs to have an operating system (OS). This is the software used to start the hardware, and open and close programs.

Many computers use a specific operating system tied to the hardware. You'll probably be used to Windows on a PC and OS X on a Mac computer.

Most Raspberry Pi owners use an open-source operating system called Raspbian, which is based on Linux. The current version is based on a version of Linux called Debian Jessie, hence the name Raspbian (sometimes you'll hear it called 'Raspbian Jessie').

Linux is like Windows and Mac OS X, but more fun because it's open-source, so anybody can view the source code and improve it.

You can install a range of different OSes on a Raspberry Pi, some based on other versions of Linux, others based on Windows, and even some unique environments like RISC OS.

Raspbian is the official OS and the one most beginners should start with. It's the simplest to install, the easiest to use, and most projects and tutorials use Raspbian as their base.

## Start with NOOBS

There are two approaches to installing Raspbian and other operating systems. Beginners should start with NOOBS (New Out Of Box Software). More advanced users may copy an image file containing a whole operating system directly to the SD card.

First, you must format your microSD card to use the Windows FAT 32 format. The easiest way to do this on a Mac or Windows PC is to use a program called SD Card Formatter (**magpi.cc/2bncvkm**).

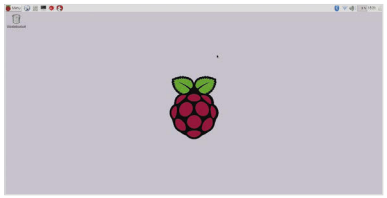
Connect your microSD card to a Mac or Windows PC, typically using a microSD-to-SD card adapter or a USB card reader, and use SD Card Formatter to erase the card.

Next, download the NOOBS ZIP file from **magpi.cc/2bnf5XF**. Extract the contents of the file and open the NOOBS folder. Copy the contents across to the root of the SD card. See the 'Setting up NOOBS' steps for more information.

## AVAILABLE OSES

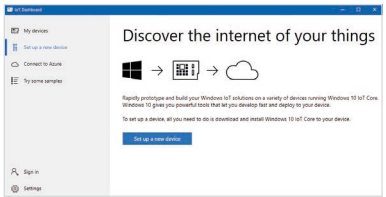
### Raspbian

The official operating system is the easiest to use, and the one beginners should start with. It works a lot like other popular operating systems.



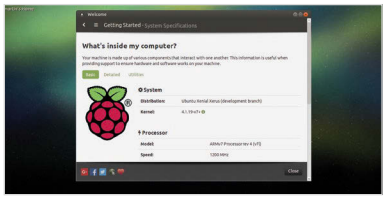
### Windows 10 IoT Core

Not the full version of Windows, sadly, but Windows 10 IoT Core enables programmers to run Internet of Things and embedded projects.

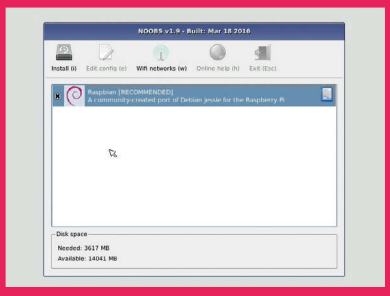


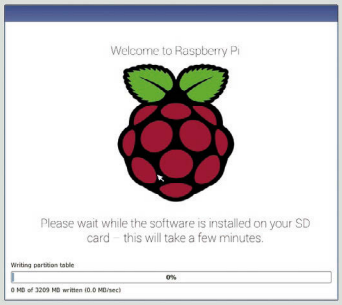

### Ubuntu MATE

Ubuntu is one of the world's most popular Linux operating systems, and Ubuntu MATE is a lightweight version that runs just fine on the Raspberry Pi.



NOOBS automates the process of installing Raspbian. Select the Raspbian option and click on Install to run it




NOOBS automatically copies all the files needed to run Raspbian onto your SD card

With the NOOBS files copied across, remove the microSD card from your computer and slot it into your Raspberry Pi. Now connect the keyboard, mouse, and HDMI cable. Finally, attach the USB power to boot up the Raspberry Pi.

The Raspberry Pi will boot, displaying the NOOBS installer. By default it only has one option, 'Raspbian [RECOMMENDED]'. Place a tick next to Raspbian and click Install. Click Yes in the Confirm alert to begin installing Raspbian.

Now you just need to wait while the Raspbian file system is extracted. When it's finished, you'll see the Raspbian desktop and the message 'OS(es) Installed Successfully'. Click OK to start using your Raspberry Pi.

## Installing image files

Installing an operating system from an image file is a slightly more complex procedure, but one that more advanced (and Pi Zero) users should learn. Image files are copied differently in Windows, compared to Linux and Mac computers.

In both systems, you format the microSD card to FAT 32 as usual, then you download the operating system as an image file, a large file ending in '.img'. This file is then copied bit by bit as an exact replica to the microSD card.

On a Windows PC, you will copy the image file using an app called Win32DiskImager (**magpi.cc/2bndEsr**). On Mac and Linux machines, most users copy the file using a command called 'dd' in the terminal.

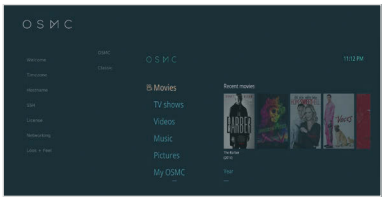
Full instructions for copying image files for Windows, Mac, and Linux can be found on the Raspberry Pi website (**magpi.cc/1V50j8E**).

A good alternative for Mac owners is a program called Apple Pi Baker (**magpi.cc/2bcD53z**). This program enables you to pick the image file and the SD card, and then handles the copying automatically.

Learning how to copy image files is essential if you want to use operating systems other than Raspbian. Beginners should stick with NOOBS to install Raspbian to start with, though. It's much easier and is the best operating system for beginners.

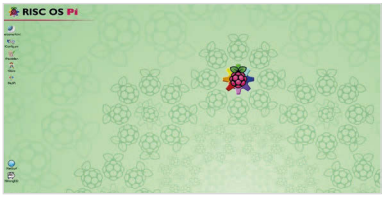
### OSMC

OSMC (Open Source Media Centre) is an easy way to transform your Raspberry Pi into a video and audio player.




### RISC OS

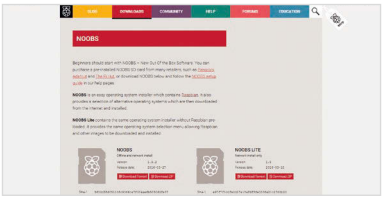
RISC OS is an operating system originally designed by Acorn Computers for ARM-based systems. It's very light and completely different.




# SETTING UP NOOBS

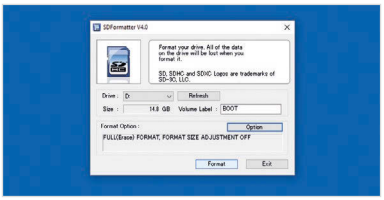
### Download NOOBS

In a browser, visit **magpi.cc/2bnf5XF**. Click Download ZIP to get all the files. Open your downloads folder and locate the NOOBS file: currently it's 'NOOBS_v1_9_2'. Right-click on a Windows PC and choose Extract All, then Extract. Just double-click the file on a Mac to extract it.



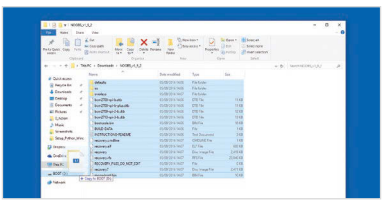

### Format SD card

Open SD Card Formatter and you'll see the card in the Drive letter. Change the Volume Label to BOOT so you can identify it later. Now click Option and change Format Type to Full (Erase). Ensure Format Size Adjustment is set to Off and click OK. Click Format, then OK. Click Exit to close SD Card Formatter when it's finished.




### Copy NOOBS files

Open the freshly extracted folder so you can view all the files. It should have folders called **defaults**, **os**, and **overlays**, and files including **bootcode.bin** and **recovery**. Select all of the files and drag them onto the BOOT icon in the sidebar. This copies all of the files inside the NOOBS folder to the root of the SD card. It's important to copy the files inside NOOBS, and not the NOOBS folder itself.



# USING RASPBIAN

## Getting to grips with the Raspberry Pi's official operating system

A Raspberry Pi can run many operating systems, but Raspbian is the official OS and the one that most newcomers will start with.

Raspbian is a Linux operating system based on the popular Debian distribution. Fully customised for the Raspberry Pi hardware, it's usually a trouble-free experience using a Raspberry Pi with Raspbian.

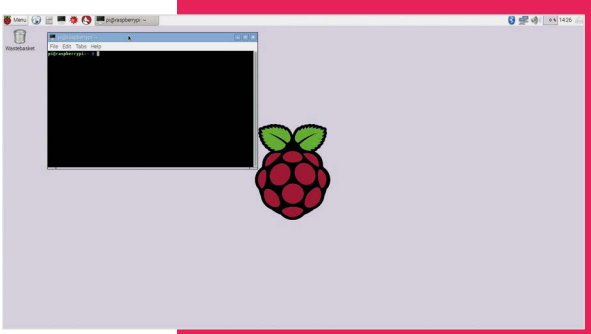
One aspect of Linux that will be new to Windows and Mac users is being able to choose from different graphical interfaces. Raspbian includes one called LXDE, which stands for 'Lightweight X11 Desktop Environment'.

This heavily modified version of LXDE enables you to use a Raspberry Pi as you would another computer. You have a Menu button, which offers access to most of the programs and apps installed. Programs open in windows, which you can switch between, minimise, maximise, and close using buttons.

Many users might be wondering why this is anything special. Well, computers didn't always have windows; instead, most users used a command-line interface and entered text commands to start programs.

It's possible to buy SD cards pre-formatted with the Raspbian software. This saves you from having to install the operating system

You'll learn how to use the terminal and control your Raspberry Pi computer using text commands



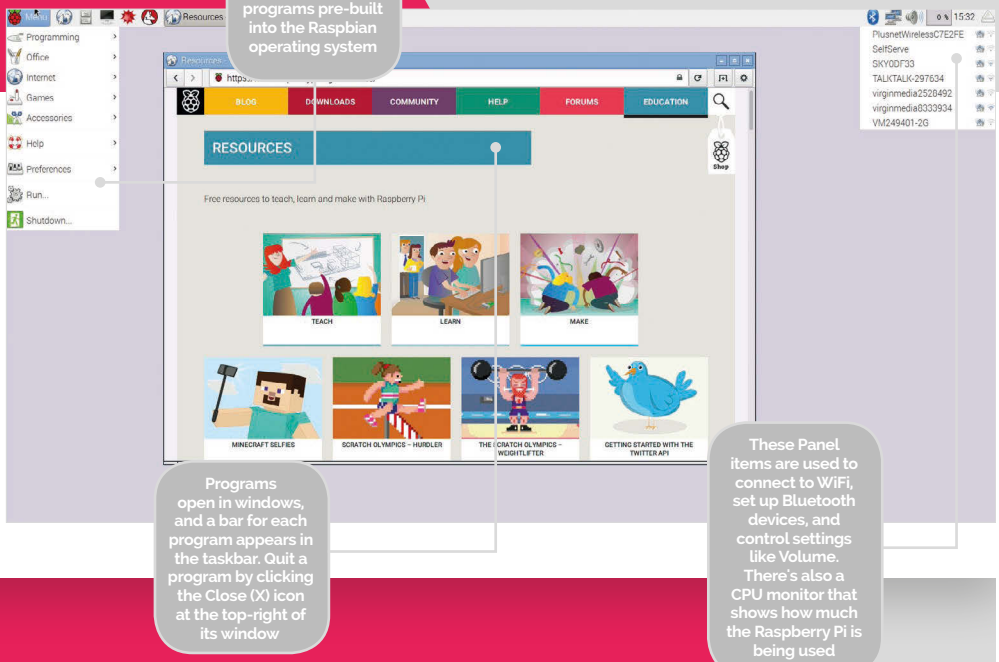
### Terminal velocity

In Raspbian, you'll probably spend some time working under the hood of the desktop in a command-line environment. Next to the Menu button is the terminal, a program that enables you to enter Linux text commands. Learning how Linux works, and how to create programs that run from the command line, is part of the joy of owning a Raspberry Pi. It's a return to classic computing where you need to learn how things actually work.

Raspbian is a great environment for learning to code. Along with easy access to the command line, you get all kinds of programming environments built in: everything from MIT's Scratch to Python and Java. You even get a full working version of Mathematica, a cool maths environment that normally costs £190 to buy, with access to real-world data.

### Office worker

It isn't just about programming, though. You can use your Raspberry Pi as a desktop computer, and the operating system comes with LibreOffice built in. This is a full office suite of programs, similar to Microsoft Office. Its programs include Writer (word processing), Calc (spreadsheets),




Impress (presentations), Draw (vector graphics and flowcharts), Base (databases), and Math (formula editing).

Raspbian connects to the internet, and has a built-in web browser called 'Epiphany'. You also get an email client called 'Claws Mail'. Both can be accessed under **Menu > Internet**.

The Raspberry Pi connects to the internet using Ethernet (a cable that runs from your Raspberry Pi to a modem/router) or WiFi. It's easy to connect to a WiFi network, and we'll look at setting up both WiFi and Bluetooth overleaf.

## Settings and software

You can adjust the settings for your Raspberry Pi in two ways: using the desktop interface or a terminal program called Raspi Config.

Choose **Menu > Preferences** to find a collection of different system settings. Add / Remove Software can be used to find and remove packages from the Raspbian system.

Appearance Settings, Audio Device Settings, Main Menu Editor, and Mouse & Keyboard Settings all adjust appearance and interaction with Raspbian. Most of the options are self-explanatory.

The Raspberry Pi Configuration choice provides more in-depth options. Here you can change your password (**raspberry** by default) and the hostname of the Pi on the network (**raspberrypi** by default). You can choose to boot to the desktop or the command-line interface (CLI), and enable and disable various hardware interface options.

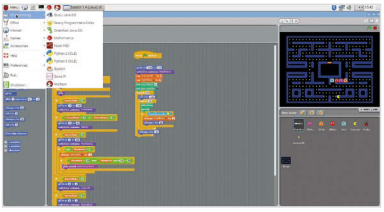
Raspi Config offers even more detailed options. Open a Terminal window and then enter **`sudo raspi-config`**. A blue screen with options in a grey box appears. Use the up and down arrow keys to move between options; press the right and left arrow keys to move into an option (and back to the main menu). More information on these options can be found at **magpi.cc/2bnfuJF**.

The important thing about Raspbian is not to worry about experimenting with different options and settings. Feel free to explore the menus, command line, and configuration settings. You can always reset your microSD card with NOOBS and start again.

# USING THE RASPBIAN INTERFACE

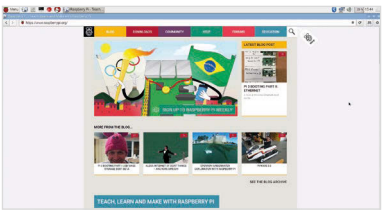
### Programming tools

Raspbian comes with a selection of coding tools, found under **Menu > Programming**. Scratch makes it easy to learn programming concepts, and popular languages like Python and Java are ready to use right out of the box.



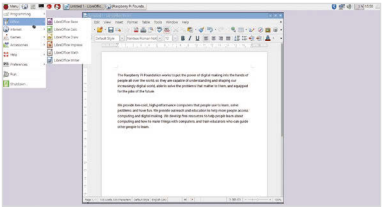
### Web software

A web browser called Epiphany is built into Raspbian, along with an email program called Claws Mail. There are links to Raspberry Pi Resources and *The MagPi* under **Menu > Internet**.



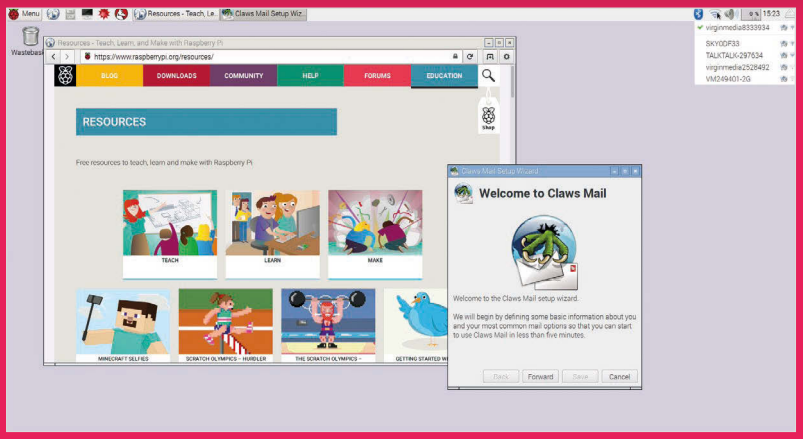
### Office suite

Raspbian features powerful LibreOffice programs like Writer and Impress. These are the equivalent of Microsoft Office apps and enable you to create documents on your Raspberry PI.



# SETTING UP THE INTERNET

Get online wirelessly and quickly, with this guide to setting up wireless LAN on your Raspberry Pi




A wireless internet connection enables you to get help online and set up apps like Claws Mail

The Raspberry Pi is best when connected to the internet. You can use it to browse the web, play online videos, and send and receive emails. More importantly, you can get the latest updates and install the software packages you need for any project.

To do this, you'll need to get online. This is easier than ever with the Raspberry Pi 3, because it now has a wireless antenna built into the board.

Other models of Raspberry Pi, including the Pi Zero, require a WiFi dongle connected to a spare USB port.

With wireless added to your Raspberry Pi, it's easy to get online. Boot into the Raspbian desktop and look for the WiFi Networks icon in the Panel (on the top-right of the display).

Click WiFi Networks and you'll see a list of all the local wireless networks. Choose your network and (if you have one) enter your password, also called the 'Pre Shared Key'. The Raspberry Pi connects to the wireless network, enabling you to get online. In this respect it's pretty much like any other computer that connects to WiFi; it will even remember the password for next time.

Once you're online, you can use the Epiphany browser to fetch webpages. Click Web Browser in the Launch Bar.

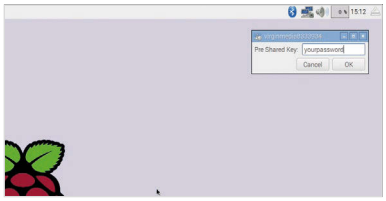
## CONNECTING TO A WIRELESS NETWORK

### Check for networks

Click on the Wireless Networks icon in the Panel. Raspbian will display a list of all the wireless networks available in your local area. Click on the one that's yours.

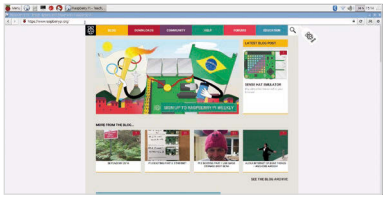
### Enter your password

Enter your WiFi password in the Pre Shared Key field and click on OK. The network symbol will switch to a wireless symbol and you'll be connected.



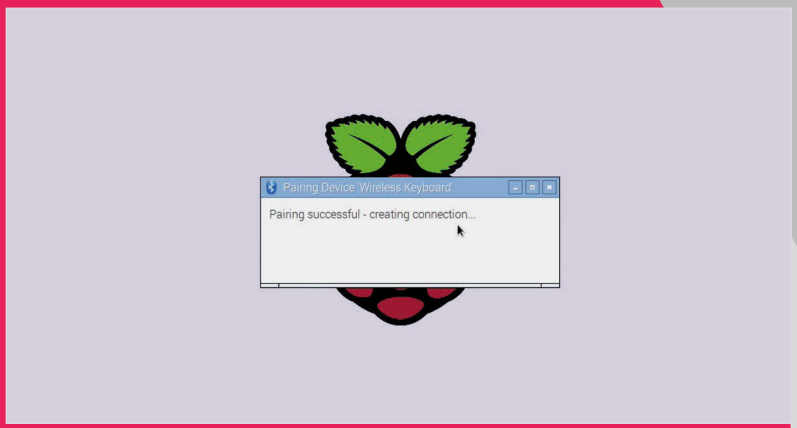
### Test your connection

Test your internet connection by opening a webpage. Click on Web Browser in the Launch Bar and enter **www.raspberrypi.org** in the URL field. Press **RETURN** to load the page.



# SETTING UP BLUETOOTH

## Connect wirelessly to nearby devices with Bluetooth technology




Devices connected by Bluetooth work wirelessly with your Raspberry Pi

Bluetooth is another piece of technology that has been added to the Raspberry Pi 3 board. With Bluetooth you can connect wireless devices, such as mice and keyboards, directly to your Raspberry Pi.

As with wireless LAN, if you own an older Raspberry Pi model or a Pi Zero, you'll need to attach a USB dongle to use Bluetooth devices.

With Bluetooth hardware on your Raspberry Pi board, it's easy to connect to a device wirelessly, a process known as 'pairing'.

You can pair wireless gaming controllers, like a PlayStation joypad, or Android smartphones. Many Raspberry Pi projects make use of Bluetooth, enabling the Raspberry Pi to communicate with nearby electronic components and devices.

The easiest way to test out Bluetooth is to set up a wireless mouse or keyboard; both are fairly easy devices to come by.

In some ways, the process is similar to connecting to a WiFi network, but the Bluetooth device you want to connect to must be set to pairing mode first. This is also known as making the device 'discoverable'. Putting a device into pairing mode varies by device; holding down the power button until an LED flashes is fairly commonplace, but check with the instructions for your device.

You then use the Bluetooth icon in the Raspbian desktop Panel to connect to the device: choose **Bluetooth > Add Device**.

It's possible to put your Raspberry Pi into pairing mode by choosing **Bluetooth > Make Discoverable** from the Panel. Then you can connect to your Raspberry Pi from other Bluetooth devices like mobile phones.

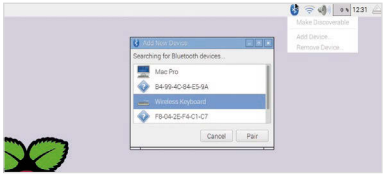
## SETTING UP A BLUETOOTH DEVICE

### Pairing mode

Start by putting your Bluetooth device in Pairing / Discoverable mode. We're using an Apple wireless keyboard. Hold down the power button until the LED flashes. Click Bluetooth in the Panel and choose Add Device.


### Add new device

The Add New Device window opens and will scan for nearby Bluetooth devices. Some will have names, others just identifying numbers (check on the device). Choose a device from the list and click Pair.




### Enter code

The Pi now attempts to pair with the Bluetooth device. You'll be asked to enter a code on the keyboard; press the buttons and **RETURN**. You can now start using the Bluetooth device with your Raspberry Pi.

# GETTING TO KNOW GPIO

Discover the joy of electronics by hooking up components, wires, and hardware to the pins on a Raspberry Pi board

One of the most powerful and fun features of the Raspberry Pi is the row of pins at the top. Known as 'GPIO' (General-Purpose Input/Output), these pins enable you to hook up the Raspberry Pi to additional hardware and electronics.

There are lots of hardware attachments for the Raspberry Pi that connect directly to the GPIO pins. Many are known as HATs (Hardware Attached on Top). These connect directly to the GPIO pins and sit on top of the Raspberry Pi. More importantly, HATs are designed to work as soon as you connect them to the Raspberry Pi, so hardware branded as a HAT is easier to set up.

The real joy of GPIO isn't using pre-made hardware, but building your own electronics projects. You can connect the GPIO pins to all kinds of electronic circuitry and control it using the Raspberry Pi. With the right cables, you can hook the GPIO pins up to switches, buttons, sensors, buzzers, and all manner of electronic gizmos and widgets. These are used to learn all about electronics hardware and circuit building.

> "You can connect the GPIO pins to all kinds of electronic circuitry and control it"

While it's possible to wire parts directly to the GPIO pins, most tinkerers place electronic components in a breadboard and connect this to the Raspberry Pi.

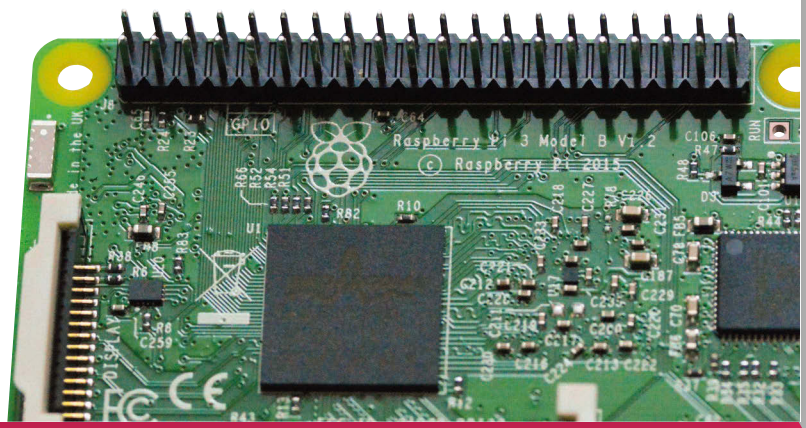
## BREADBOARDS AND BREAKOUTS

Electronic components are plugged into the holes on the breadboard, and components in adjacent holes are linked. In this way, you can build up a test circuit without having to actually solder components together.

If you follow the instructions, connecting directly to the GPIO pins on a Raspberry Pi is safe, but randomly plugging wires and power sources into the Raspberry Pi may cause bad things to happen, especially plugging in devices that use a lot of power (like motors).

Because of this, many electronics enthusiasts use a device known as a 'breakout cable' between the Raspberry Pi and breadboard. The breakout cable plugs into the GPIO pins, and into the breadboard.

There are also devices like the Explorer HAT that combine a breakout with a breadboard and enable you to create prototype circuits.




Unlike the type used for cutting bread, an electronic breadboard is a plastic slab with lots of holes in it.

Wiring a breadboard (or circuit) directly to the GPIO pins is generally safe, as long as you avoid circuits with external power sources. Most tinkerers invest in a breakout cable to go with the breadboard (see 'Breadboards and breakouts').

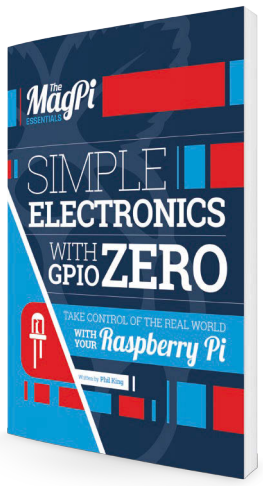
With your circuit set up, you then control the GPIO pins in a programming environment like Python or Scratch. GPIO pins are set to input or output mode. GPIO outputs are easy because the pin is switched on or off (known as HIGH or LOW in computing terms). When the GPIO pin is HIGH, voltage flows through the GPIO pin, lighting up an LED or buzzing a buzzer. Set the pin to LOW and the LED goes out, or the buzzer goes quiet.

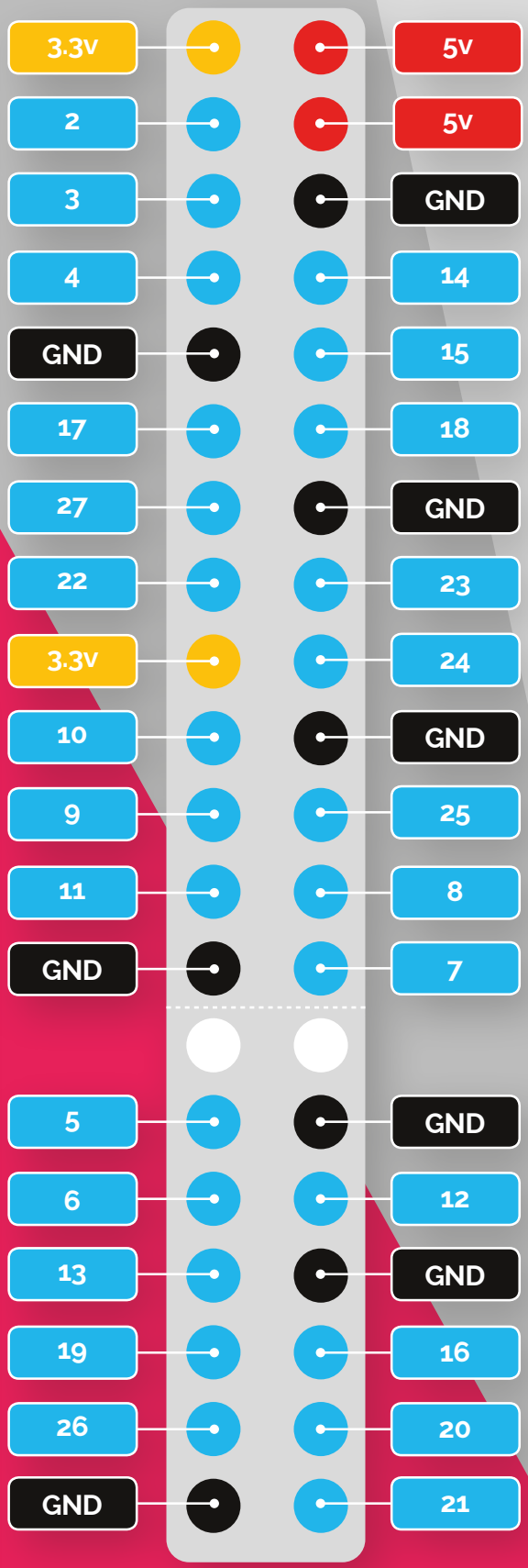
GPIO input is a bit more tricky. In this case, the GPIO pin is set to HIGH or LOW and responds to a change from a circuit. A button (or other electronic component) can change the circuit from LOW to HIGH, or HIGH to LOW, with the Raspberry Pi coded to respond accordingly. This is often referred to as 'pull up' or 'pull down'. Don't worry: if this all sounds complicated, you can get started by using GPIO Zero to make programming much easier.

Never underestimate the pure fun you can get from a little computer, a bunch of pins, and a handful of electronic components. Discovering how to use GPIO is a great way to spend your time.

## GPIO ZERO ESSENTIALS

Learning to use the GPIO pins is the route to having real fun with a Raspberry Pi. It's a big subject, with lots of tricks and tinkering to discover. Our *GPIO Zero Essentials* book teaches you the basics (and beyond) of using the GPIO port with the GPIO Zero Python library. See **magpi.cc/GPIOZero-book** for more information.




There are 40 GPIO pins, each with a specific function. Use this image as a handy guide whenever you're programming electronics

# CASE STUDY: CREATIVE COMPUTING AT EASTWOOD ACADEMY

Computing is a new and exciting subject, but with so many potential options it can be worth looking at how other schools are doing things to get ideas of what's possible. Some teachers have dived straight into the new subject and are already seeing impressive results, as this case study shows. **Oliver Quinlan** investigates...

It's nearly two years since computing became a subject for all children in England to study, and we're now seeing some amazing work to bring opportunities for digital making into schools. Recently, I visited Eastwood Academy in Southend-on-Sea, where teacher Lucas Abbott has created a digital making room, and built a community of young programmers and makers.

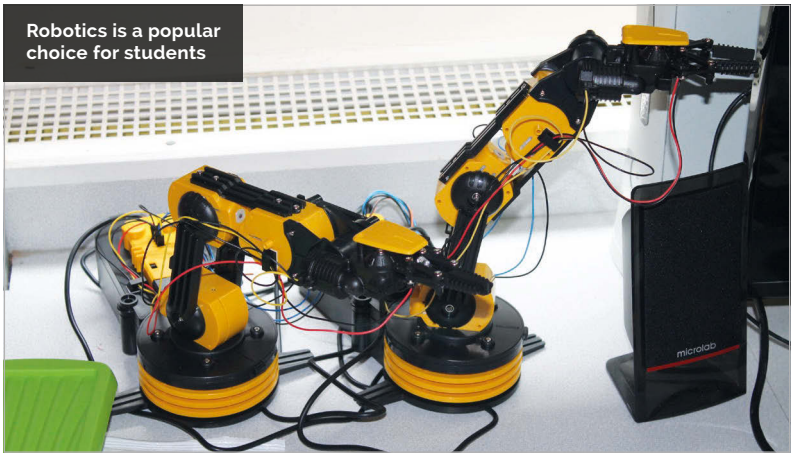
Lucas trained as a physics teacher, and got hold of a Raspberry Pi for projects at home back in 2012. His head teacher heard about his hobby, and when the curriculum shifted towards all children learning programming, Lucas was approached to take up the challenge of developing the new subject of computing in the school. With the help of friends at the local Raspberry Jam, a Linux user group, and other programming meet-ups, he taught himself the new curriculum and set about creating an environment in which young people could take a similarly empowered approach.

## The curriculum

In Year 7, students start by developing an understanding of what a computer is; it's a journey that takes them down memory lane with their parents, discussing the retro technology of their own childhoods. Armed with knowledge of what they're working with, they then tackle programming with the Flowol language, progressing to Scratch, Kodu, and the BBC micro:bit. In Year 8 they get to move onto the Raspberry Pi, firing up the 15 units Lucas has set up in collaborative workstations in the middle of the room. By the time the students choose their GCSE subjects at the end of Year 8, they've experienced programming a variety of HATs, hacking Minecraft to run games they have invented, and managing a Linux system themselves.

## AN ARCADE TABLE

The maker club at Eastwood Academy are currently part way through their most ambitious project yet. While some students are programming their own Minecraft games or experimenting with HATs, one group of students is working with the teachers to create an arcade game table. Following a tutorial from *The MagPi*, they have begun using a cheap IKEA coffee table and two old PC monitors to create an arcade machine in a table. The games will run on a Raspberry Pi computer, powered by the RetroPie emulator. They're just getting started, but this shows the ambitious projects that students and teachers can aim for when they collaborate.




Robotics is a popular choice for students

The classroom is buzzing during the maker club that happens every single lunchtime


Raspberry Pis are integrated into the classrooms; these older units were kindly donated to a project in Vietnam

## Pis for collaboration

Fifteen Raspberry Pi computers have been set up in the centre of the room, at stations specifically designed to promote collaboration. While the PCs around the edges of the room are still used, it was the Pi stations where pupils were most active, connecting things for their projects and making together. Clever use of ceiling-mounted sockets has allowed these new stations to be set up at a low cost.

The teaching is based on building a firm foundation in each area studied, before giving students the chance to invent, build, and hack their own projects. I spent a whole day at the school; I found the environment to be entirely hands-on, and filled with engaged and excited young people learning through making. In one fabulous project, two girls were setting up miniature space rockets, propelled using compressed air with a computer-based countdown system. Problem-solving and learning through failure are part of the environment, too. One group spent a session trying to troubleshoot a HAT-mounted display that wasn't quite behaving as they wanted it to.

## Making at lunchtimes

Lessons were impressive, but the daily lunchtime making club was even better. About 30 young people rushed into the room and got started with projects ranging from figuring out how to program a robot Mr Abbott had brought in, to creating the IKEA coffee table arcade machines from a recent tutorial in *The MagPi*.

I had a great conversation with one student who told me how she had persuaded her father to buy a Raspberry Pi, and then taught him how to use it. Together, they were inspired to create a wood-engraving machine using a laser. Lunchtime clubs are often a place for socialising, but there was a real sense of purpose here too, of students coming together to achieve something for themselves.

## A computing community

Since 2014, most schools in England have had lessons in computing, but Eastwood Academy has also been building a community of young digital makers. They're linking their ambitious lessons with their own interests and aspirations, building cool projects, learning lots, and having fun along the way.

We'd love to hear from other schools taking such an ambitious approach. Contact Raspberry Pi Foundation Research Manager Oliver Quinlan at **oliver@raspberrypi.org** to let us know about digital making in your school.

## MAGPI TUTORIALS

This edition of *The MagPi* has been put together specially for educators, but there's also a monthly edition packed full of inspiration and tutorials for Raspberry Pi projects. At Eastwood Academy, Lucas has been using tutorials from *The MagPi* to inspire his students at the lunchtime maker club. Some projects are suitable for all, while others will stretch your most eager and able students. They range from robots to home automation and camera projects, and will appeal to young people with a wide variety of interests. You can buy printed copies of *The MagPi* in newsagents across the UK, but you can also download the PDF version for free, so there's no shortage of inspiration for your students to get started with amazing digital making projects.

Download free PDFs of *The MagPi* at: **magpi.cc**

Above Esther Devonport has built her coding confidence by running her Star Club at Fritchley Primary School in Derby

# CODE CLUB STAR CLUBS

Meet the members of Code Club's 'Star Clubs' network, run by volunteers and teachers who share our passion for getting children excited about digital making

## VISIT A STAR CLUB NEAR YOU

If you're thinking about volunteering and would like to see a Code Club in action, you can arrange a visit to your nearest Star Club. Expect a friendly welcome from the adults and kids bursting with enthusiasm. You can take the visit at your own pace; feel free to sit at the side and observe, or get stuck in by assisting the club leader or making alongside the children. Make sure that you let the teacher and volunteer carry on with their club as normal; save up your questions for after the children have left.

Visit the Code Club website to get in touch with a Star Club and arrange your visit: **codeclub.org.uk/star-clubs**

Across the UK and around the world, enthusiastic and passionate people are volunteering their time to run free after-school Code Clubs for children aged 9–11.

In 2014, Code Club introduced a new initiative in the UK to profile some of the community's outstanding coding clubs, called Star Clubs. These clubs welcome visits from potential volunteers and help to showcase some of the fantastic benefits that running a Code Club can bring to children, schools, and community venues, alongside volunteers and teachers.

There are currently 35 Star Clubs across the UK, and we spoke to some of the people who run them to get an insight into their experiences volunteering with Code Club.

### Experiences from the Star Club network

Jill Sim runs a Star Club at Rosebank School in Dundee. Before starting her club, Jill had very little experience with computing; although she considered herself to be fairly computer-literate, the idea of coding seemed daunting.

"When I first started my Code Club, I was a bit apprehensive, until I saw how well the children took to it. Within a few weeks of getting the club up and running, I found that the children all learnt

Rosebank Primary School's Star Club won the Dundee Games Jam this year, coding a three-level game in just two hours

at their own pace." Now, after running the club for two years, Jill is much more confident; she knows that the children are comfortable to follow the projects through and help each other when they get stuck.

"We cover all three of Code Club's project curriculums – Scratch, HTML, and Python – and the children are free to choose what they want to work on. Some prefer to stay with Scratch, others move on and dabble with HTML or Python; that's where I get a bit lost, but it doesn't worry them."

With the children's coding skills improving, the club has also had the opportunity to showcase their skills. This year, the club won the Dundee City Schools' Code Jam, a competition held between twelve other primary schools from across the city to create, code, and present their own games.

"Code Club has given us a sense of shared achievement as a school," Jill adds. "The club members know they have raised the profile of the school and are especially proud to be a Star Club: they like welcoming visitors and explaining their creations."

Esther Devonport is another teacher who runs a Star Club at her school in Derby. She used her Code Club to improve her own confidence with coding, initially running the club with the help of a volunteer before taking the reins herself. She believes that the Code Club has brought some significant benefits to her rural village school: "When we had our OFSTED inspection a few years ago, computing came out as an outstanding feature and that's down to Code Club. The children's maths and problem-solving abilities have all improved alongside their teamwork and resilience. Getting something wrong isn't a bad thing any more: it's just a problem to be solved."

Moreover, Esther sees that Code Club has given the children at her school some great experiences. "One term, our chair of governors joined the club. She was in her sixties, but a keen mathematician, and wanted to see how it all worked. It was fab to see her learn a new skill and show her work alongside the children's. The children loved to show their work to her; they explained what they had done so simply and they were all beaming with pride."

# 10 TOP TIPS FOR RUNNING YOUR CODE CLUB

We crowdsourced some advice on how to run a Code Club from our experienced Star Club community. Here are their top tips:

- If you're new to running a Code Club, start with a small number of children and learn alongside them. You can then increase the number of club members as you become more confident.
- Following a regular format for each session can be beneficial, so the children get familiar with how the Code Club is run – this can also help maximise the time, as an hour is not long!
- Don't worry if the kids don't follow the projects exactly. They learn the most through experimenting.
- Try letting everyone work at their own speed each week, picking up where they left off at the end of the previous session.
- Save paper by using the Code Club projects online: children will soon get used to switching tabs in their web browser.
- Try paired programming; encouraging children to work through the projects together can improve concentration and effectiveness.
- Don't show the children their mistakes. Instead, point them in the right direction and let them find bugs themselves.
- Doing a planned show-and-tell session two or three times a term is good for the children to be proud of their work.
- Giving out certificates at the end of term is great: the kids love them.
- Find a local Code Club meet-up where you can talk to fellow volunteers and exchange helpful advice.

# START A CODE CLUB IN YOUR SCHOOL

Are you a teacher interested in setting up a Code Club in your school? Find out how simple it is to get started

It's easier than you think to run a Code Club yourself: you don't need existing coding skills, just a can-do attitude to get stuck in learning alongside your students for an hour a week!

If you haven't yet heard about Code Club, it's a UK-based non-profit organisation offering free learning materials and support for teachers, volunteers, and parents running after-school coding clubs for children aged 9-11.

Code Club's specially designed projects offer structured and fun content for the clubs. The projects are step-by-step guides for children to follow to create animations, games, websites, and much more. Children build up their programming skills as they move through the projects. There are also challenges to provide opportunities to apply what they've learnt.

Caroline Harding, a Year 4 teacher who helps to run a Code Club at her school in Croydon, told us about the benefits the club has brought the children. "Making Code Club available to the children in our school has helped tremendously with their confidence and engagement in coding and computing in general," Caroline says. "It taps into their problem-solving skills and enables them to develop critical thinking skills. Programming and coding is an area of the curriculum that many staff can find intimidating. Knowing that the children have some experience of the program can help ease some anxieties and enables that 'have a go' attitude!"

By starting a club at your school you'll be joining a huge community of teachers who do the same thing: around 50% of Code Clubs are run by teachers.

If you're considering getting a Code Club started, we have come up with a few tips to help you.

**Below** Code Club allows children to experiment and invent, using different languages to create their own games, animations, and websites

## Register your club online

To access Code Club's project materials, you will need to register online. You can sign up as a Code Club Host by visiting **jumpto.cc/teachers**, making sure to use your school email address so we can validate you as a teacher.

Once you've entered your details, you'll be able to select the option to run the club yourself. Your club will then be automatically activated and you'll have immediate access to all Code Club's online resources.

Code Club have projects in three different coding languages: Scratch, HTML/CSS, and Python. Beginning with Scratch is recommended, as this visual

**Left** There's no cost involved in starting a Code Club. It's free for schools and the kids who attend

block-based language provides a great introduction to key programming concepts. If your pupils are already experienced with Scratch, though, you may wish to get started with HTML/CSS or Python. There are twelve Code Club projects in each language to keep your club occupied for a full term.

## Your first Code Club

It's worth preparing for your first Code Club session by working through the project in advance, so that you're aware of all the instructions and the places where pupils could possibly get stuck.

Code Club is fun and it offers children (and their teachers) the opportunity to get creative with coding. It's a chance to experiment and invent, helping children to learn an important skill for their future, while engaging with technology and creating things that they can get excited about.

The model can be adapted to suit different venues and educational needs. Most clubs run through one coding project per week, but some children like to spend longer perfecting their designs. Some clubs have pairs of students sharing computers, and many clubs also like to experiment with physical computing. You can customise your club to suit you and the children.

## Code Club in practice

There are thousands of teachers running their own Code Clubs across the country, and around the world. We spoke to Matthew Cave, assistant head teacher at West Town Lane Academy in Bristol, who told us about his club.

Beginning with Year 5 and 6 students, Matthew and his team introduced Code Club's Scratch projects for all Key Stage 2 children. Now, they have a whole-school approach, with ScratchJr introduced for Key Stage 1, and they have invested in new technology including Lego WeDo and My Romo.

Matthew says, "We've been running our Code Club for over a year now, with 40 children attending. The club is in high demand."

Code Club's fun approach has provided other benefits: "It's amazing to see the sense of achievement the children get when they finish their projects. We can really see them starting to persevere with the tasks in Code Club, using analytical thinking to troubleshoot."

What advice does Matthew have for teachers who are thinking of starting a club? "It's dead easy, so take the plunge! The children will run with it, so don't worry about not being an expert."

## TACKLING THE COMPUTING CURRICULUM

Running an after-school Code Club can help you to develop confidence to teach the computing curriculum, and to integrate computing into your everyday lessons.

If you and your colleagues are keen to get some additional, more formal training, you may be interested in Code Club's Teacher Training courses. There are three modules on offer, focusing on 'Computational Thinking', 'Programming and Networks', and 'The Internet'.

Many of the sessions are now free for teachers, so if you're interested, you can make an enquiry with the Code Club team by emailing **hello@codeclubpro.org.**

# PHYSICAL COMPUTING WITH SCRATCH AND THE RASPBERRY PI

With Scratch on the Raspberry Pi, you gain access to the GPIO pins and the world of physical computing

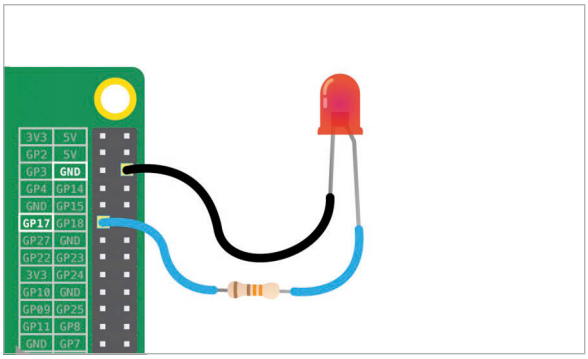
One powerful feature of the Raspberry Pi is the row of GPIO pins along the edge of the board. GPIO stands for General-Purpose Input/Output. These pins are a physical interface between the Raspberry Pi and the outside world.

The GPIO pins allow the Pi to control and monitor its environment by being connected to electronic circuits. The Pi is able to turn LEDs on and off, read values from components like light sensors or motion sensors, and even run motors to drive robots. We refer to this as physical computing.

There are 40 pins on the Raspberry Pi, and they provide various different functions.

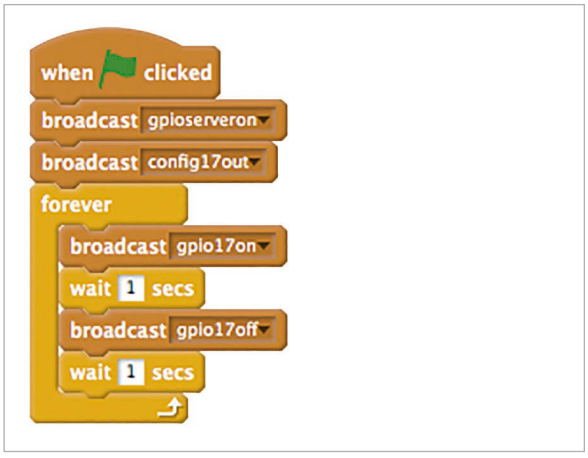
## Blinking an LED

Making an LED blink on and off using Scratch is easy with the Raspberry Pi. You'll need an LED (any colour will do), a resistor (anything above about 47Ω), and three female-to-female jumper leads. You can get all of these components from online electronics suppliers, or even from eBay. An LED has a long leg and a short leg. The long leg is called the anode and needs to be connected to a positive pin on the Raspberry Pi, in this case the one called GP17. The short leg, called the cathode, needs to be connected to a ground pin. The resistor is there to protect the LED from receiving too much current. Without the resistor, the LED may burn itself out or even pop.



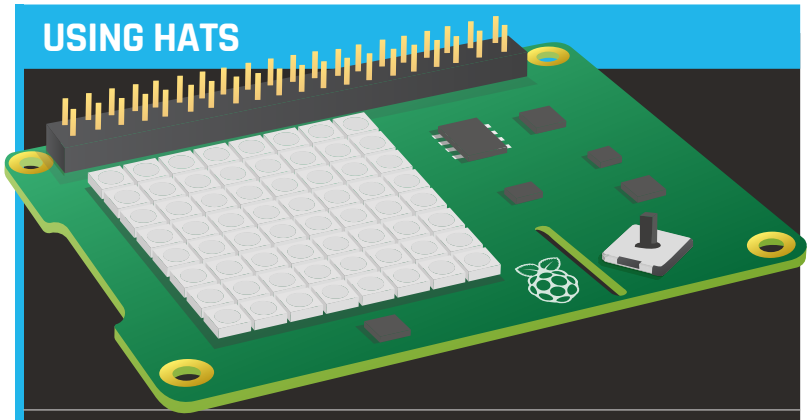

A single LED connected to the Raspberry Pi using jumper leads and a resistor

Now, with a little bit of Scratch magic, you can make the LED blink. The script at the top of the next page does several things. Firstly, it switches on the GPIO server that allows Scratch to control and read the GPIO pins. Then it configures pin 17 to be an output pin. This means that Scratch can either make the pin output 3.3V or 0V. Think of it as a light switch that can be on (so your light bulb gets 230V) or off (so the light bulb gets 0V). Lastly, using an infinite loop, the pin is switched on and then off again with pauses in between, making the LED blink.




A simple script to make an LED on pin 17 blink on and off

## USING HATS

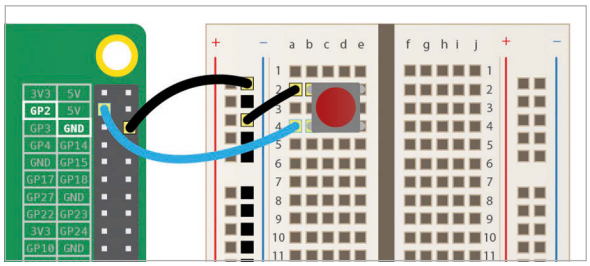


As well as using general-purpose electronics, the version of Scratch on the Raspberry Pi can be used in conjunction with many different add-on boards. The Sense HAT, for instance, can be used with Scratch to enable your students to measure temperature, humidity, and atmospheric pressure. It can even be used as an alternative way of controlling a game, with its accelerometer and built-in joystick.

Other boards can be equally useful. The Explorer HAT Pro, for example, can be used to add capacitive touch to your students' projects, and can be safely used to drive up to two motors.

## Using a button

You can also use Scratch to detect signals from an input device, the simplest of which would be a button. A button works by either making or breaking a circuit. In this example, the button is a push-to-make button, so when it's pressed, a circuit is completed. This example also uses a breadboard to help hold the components in place. This is not essential, but your circuits will become easier to manage as they become more complicated if you use a breadboard. As you can see in the diagram, the button is connected to GPIO pin 2 and also a ground pin (GND).



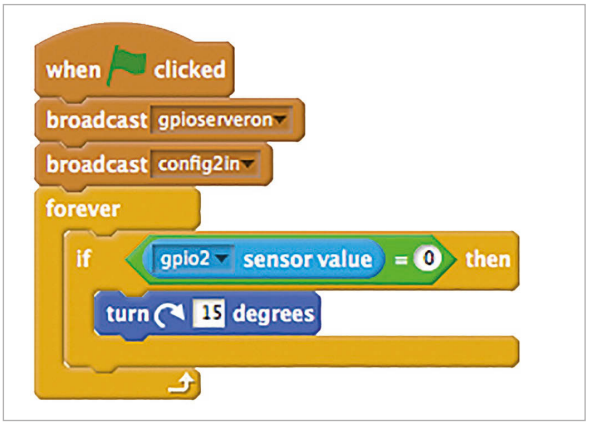

A single button wired to a breadboard to control a Scratch sprite

Before Scratch can react to your button, it needs to be told which pin is configured as an input pin. Once it knows this, you can begin to use the value of the pin in your programs. Set up the script as shown, and then click the green flag to run the program.


A Scratch script that can be used to initialise an input component

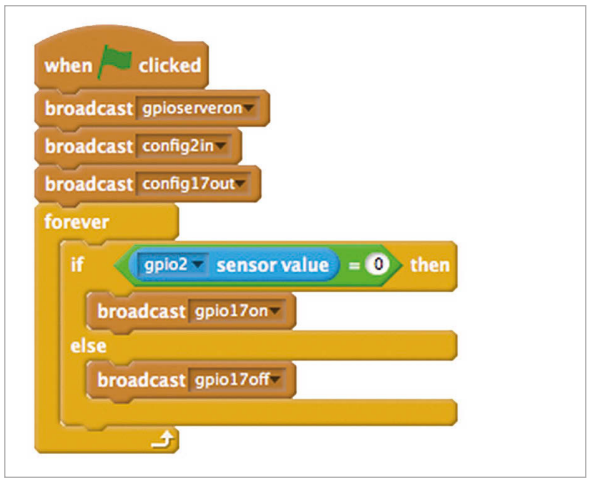
Afterwards, you should be able to find 'gpio2' in the Sensing blocks, in the drop-down menu of the **`slider sensor value`**.

Now you can use the value of the sensor to control a sprite on the screen. In this script, while the button is being pushed, the cat will spin on the spot.




A Scratch script to make a sprite spin when a button is pushed

Buttons and LEDs can be used together in the same script, of course. With the LED attached to pin 17 and the button attached to pin 2, this script will make the LED blink each time that the button is pushed.

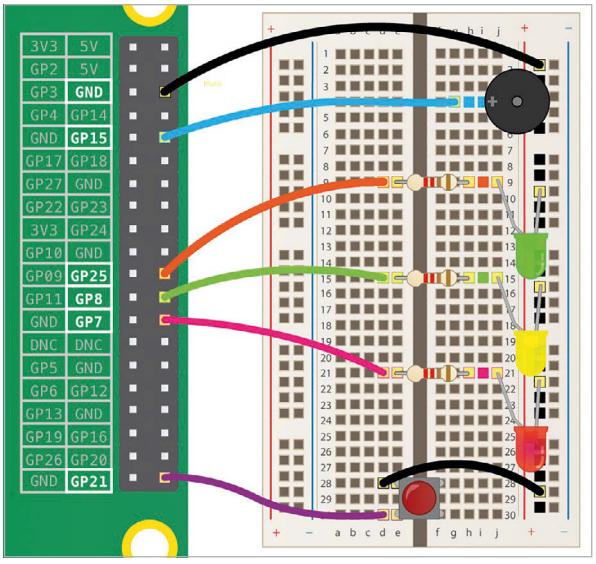



A Scratch script to light an LED whenever a button is pushed

## Making traffic lights

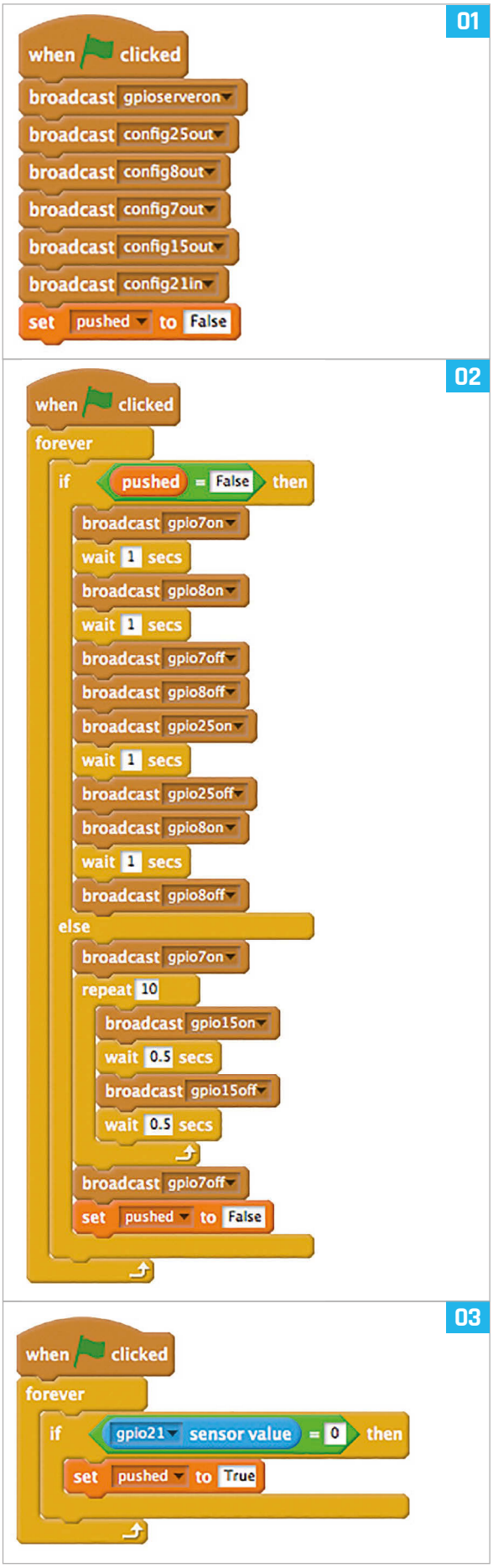
Traffic lights are a great way of exploring physical computing and computational thinking with your students. Most children will be familiar with the concepts of traffic lights and the basic sequence of the lights, but scripts to make the lights work can range from the simple to the enormously complicated.

The first step is to set up the circuit. You'll definitely want to use a breadboard with this circuit, to keep all the components and cables tidy. In the circuit diagram overleaf, the LEDs have been attached to pins 25, 8, and 7. A button has been attached to pin 21 and there's a buzzer on pin 15. All the components are sharing a single ground pin that has been connected to the breadboard.




A circuit containing three LEDs, a button, and a buzzer, to simulate traffic lights

Fig 1-3 The full traffic lights program




Next, you need to configure all the GPIO pins so that they are set to be inputs and outputs. The buzzer works in exactly the same way as an LED, so it's just another output. Once you have done this, clicking the green flag will give you access to the sensor on pin 21.


Initialising the LEDs, button, and buzzer for a traffic light simulation

The standard traffic light sequence is red, red/amber, green, amber, and then back to red. A first simple step would be to get your students to figure out how to use a loop and some **`wait`** blocks to get this sequence to run. This can be done in a separate script from the initial configuration script.

The next logical progression would be to get the script to pause on red for a minute or so when the button is pushed. This simulates a pedestrian pushing a button to cross the road. Next, you can get them to make the buzzer sound while it's safe to cross.


Flashing a traffic light sequence

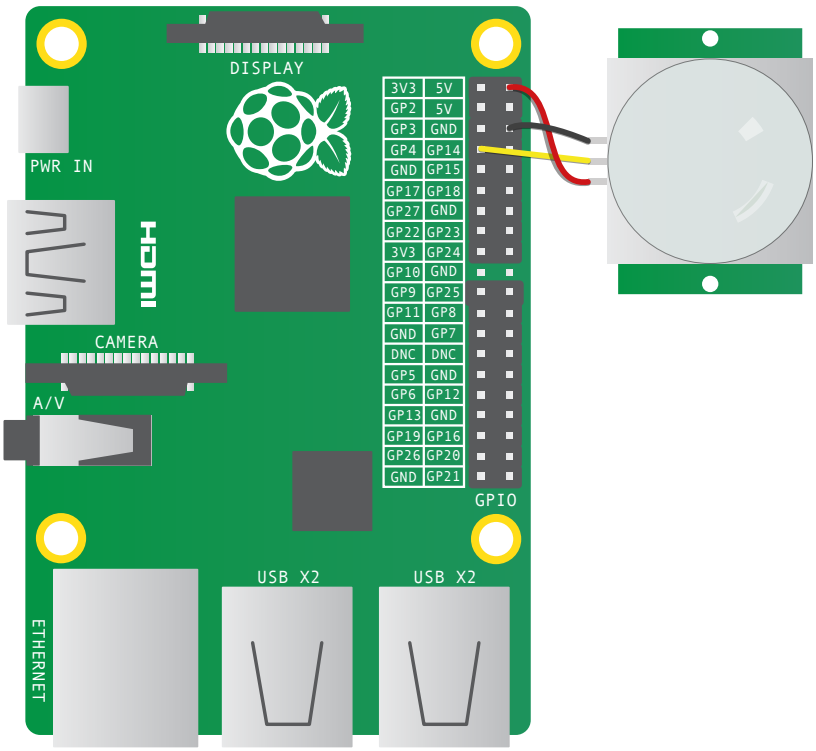
## LEDS AND RESISTORS

Not all LEDs are made equal. Some LEDs are more delicate than others, often depending on the quality of the manufacturer. If you are in any doubt about the size of resistor to go for, it's better to verge on the side of caution and use one that's fairly large (a few hundred ohms). The only adverse effect will be to cause the LED to be a little dimmer. If you want to test whether an LED is working or not without having to rely on any programming, then you can connect it through a resistor to the 3.3V pin on the Raspberry Pi.

## Using a PIR sensor

Humans and other animals emit heat all the time. A passive infrared (PIR) sensor detects changes in the amount of IR radiation (heat) it receives. When there is a change, a pulse is triggered. This means that a PIR sensor can detect when a human (or any animal) moves in front of it. Unlike the other components you've used in this article, the PIR requires 5V to work. There are three pins on the PIR; they should be labelled VCC, GND, and OUT. If these labels aren't clear, they are sometimes concealed beneath the Fresnel lens (the white cap), which you can temporarily remove to see the pin labels. Don't assume that your PIR has its pins in the same order as the one shown in the diagram.

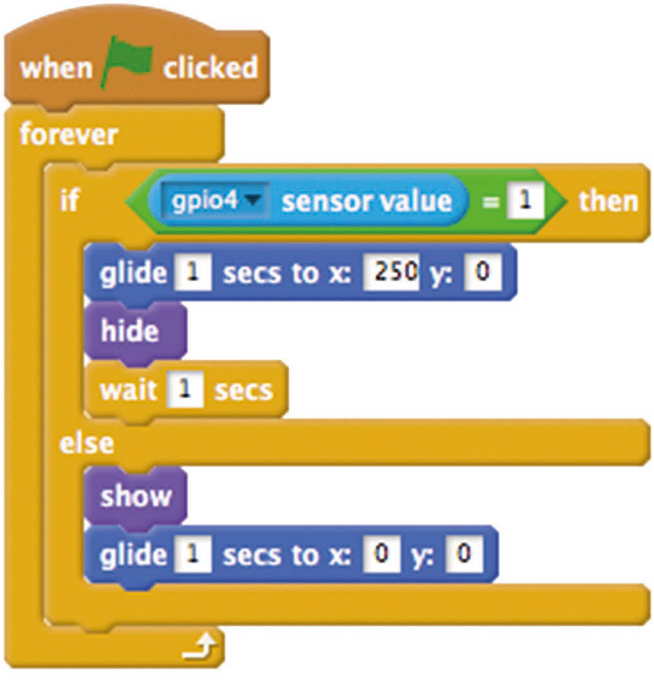
As shown above, the VCC pin needs attaching to a 5V pin on the Raspberry Pi. The GND pin on the PIR sensor can be attached to any ground pin on the Pi. Lastly, the OUT pin needs to be connected to any of the GPIO pins. Here, the OUT pin has been connected to pin 4. With your PIR circuit complete, you are now ready to use Scratch to sense motion.




**Above** How to wire a PIR sensor up to a Raspberry Pi

Just like with the button, you'll need to set pin 4 to be an input pin, and start the program to give you access to the sensor block.

With the script below, you can make the sprite run away and hide every time motion is detected, and come back again when everything is calm.


**Below** Make the cat hide whenever somebody moves

## BUYING ELECTRONICS

In general, electronic components are fairly cheap to buy. It's best to buy in bulk because they work out much cheaper that way. LEDs can cost as little as half a penny each, and even a PIR sensor can cost as little as 50p. Another good source of electronic components is your science and D&T departments. You may find that they have dozens of breadboards, hundreds of resistors, and an assortment of jumper leads lying around in cupboards somewhere.

# TEACHING COMPUTING WITH MINECRAFT

Minecraft is a popular sandbox open-world building game that's extremely popular with children of all ages, so it's an amazing way to introduce kids to the world of computing

When Minecraft Pi (found in **Menu > Games**) has loaded, click on **Start Game**, followed by **Create new**. You're now in a game of Minecraft! If you have played first-person perspective games before, you should find Minecraft fairly easy to play. You use the mouse to look around and the following keys to control movement and actions:

| Key | Action |
|---|---|
| **W** | Forward |
| **A** | Left |
| **S** | Backward |
| **D** | Right |
| **E** | Inventory |
| **Space** | Jump |
| **Double Space** | Fly / Fall |
| **Esc** | Pause / Game menu |
| **Tab** | Release mouse cursor |

You can select an item from the quick draw panel with the mouse's scroll wheel (or use the numbers on your keyboard), or press **E** and select something from the inventory. These objects are the building blocks of the world. Take a walk around, hack things, and build things!

With the sword in your hand, you can click on blocks in front of you to remove them (or to dig). With a block in your hand, you can right-click to place that block in front of you, or left-click to remove a block.

## Hacking the world with Python!

With Minecraft running and the world created, bring your focus away from the game by pressing the **TAB** key, which will free your mouse. Open Python 3: **Menu > Programming > Python 3 (IDLE)**.

To create a new file, go to **File > New File** and then **File > Save**. You'll probably want to save this in your home folder or a new project folder. Now move the windows so they're side by side.

To start controlling the world using Python, you need to import the Minecraft library. This will create a connection to the game that you can test by posting the message 'Welcome to Minecraft Pi' to the screen:

```
from mcpi.minecraft import Minecraft

mc = Minecraft.create()

mc.postToChat("Welcome to Minecraft Pi")
```

Save the file with **CTRL+S** and run with **F5**. When your code runs, you should see your message on screen in the game.

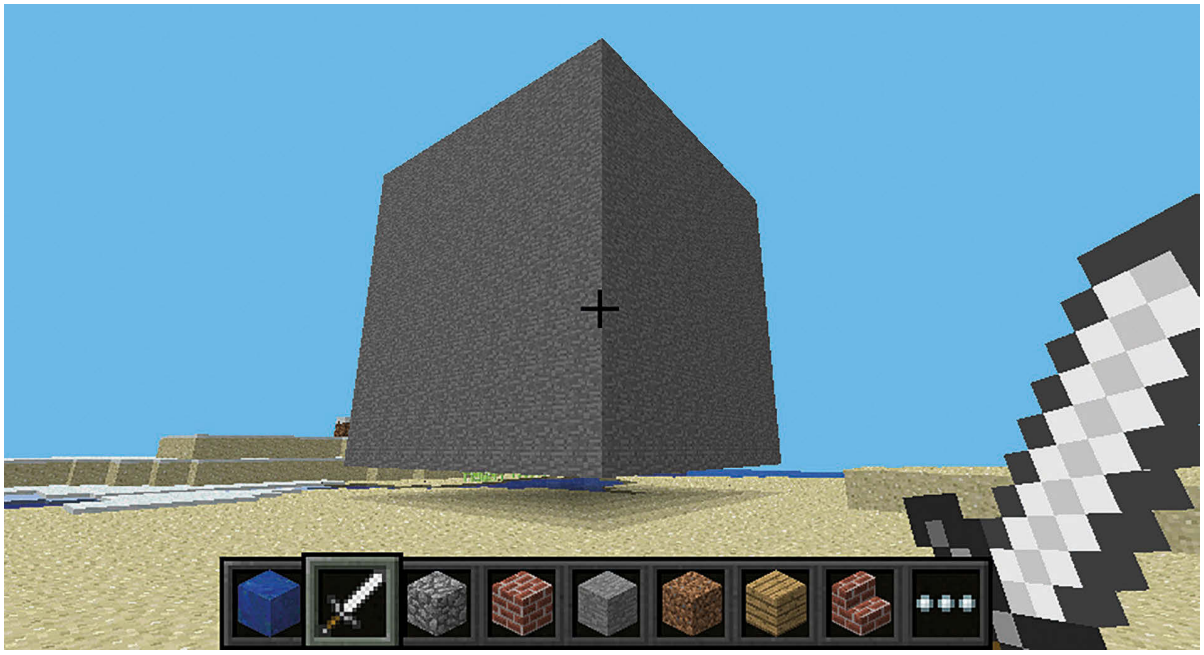
## Where in the world are you?

To do anything in Minecraft Pi, it's useful to know your player's location. This will then allow you to build and remove blocks, relative to your current position. With a single line added to your script, you can find your **x**, **y**, and **z** coordinates, and then print them out.

```
x, y, z = mc.player.getPos()
print(x, y, z)
```

In the world of Minecraft, **x** and **z** are the walking directions – north/south and east/west – and **y** is up/down.

Note that **getPos()**returns the location of the player at the time, and if you move position you have to call the function again or use the stored location. This is a great way of introducing the concept of variables to your students, and maybe even looking at the concept of coordinates.



**Left** A giant cube of stone with only a single additional line of code

## Teleporting

As well as finding out your current location, you can specify a particular location to teleport to. Add this line to the bottom of your script, then save and run your code.

```
mc.player.setPos(x, y+100, z)
```

This will transport your player to 100 spaces in the air. You'll teleport to the middle of the sky and fall straight back down to where you started, so don't forget to look down as you plummet to the ground.

Try teleporting to somewhere else by adding values to your **x** and **z** coordinates or even using negative numbers. Students will often push this to the extreme, and try to teleport a million spaces into the air, or deep underground. This kind of experimentation will let them test the limits of the game and will encourage them to play with different values in their code.

## Building things

You can place a single block at a given set of coordinates with **mc.setBlock()**. You might want to create a new save file for this next exercise.

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

x, y, z = mc.player.getPos()
mc.setBlock(x+1, y, z, 1)
```

This will place a block one square in the **x** direction from where you're standing. Don't forget that the Python API doesn't know which way you are facing, so you'll have to have a look around to find the block that has been placed.

You'll notice that the block placed is made of stone. This is because you passed the value **1** as the last argument passed to **setBlock()**. The Minecraft API treats **1** as a stone block, but there are plenty of other blocks that you can try, many of which have unusual properties.

| Block | Code |
|---|---|
| AIR | 0 |
| STONE | 1 |
| GRASS | 2 |
| DIRT | 3 |
| COBBLESTONE | 4 |
| WOOD PLANKS | 5 |
| WATER (FLOWING) | 8 |
| LAVA (FLOWING) | 10 |
| SAND | 12 |
| GOLD ORE | 14 |
| BED | 26 |
| WOOL | 35 |
| FLOWER (YELLOW) | 37 |
| TNT | 46 |
| OBSIDIAN | 49 |
| FIRE | 51 |

Again, it's worth letting your students experiment with different values, so that they can figure out which values represent which blocks. They can also try placing blocks at different locations, or write simple scripts to build structures.

## Setting multiple blocks

As well as setting a single block with **setBlock()**, you can fill in a volume of space in one go with **setBlocks()**:

```
x, y, z = mc.player.getPos()
mc.setBlocks(x+1, y+1, z+1, x+11, y+11, z+11, 1)
```

This will fill in a 10×10×10 cube of solid stone. You can create bigger volumes with the **setBlocks()** function, but it may take longer to generate and the speed at which the blocks are placed will depend on which Raspberry Pi you are using to run the code. With the **setBlocks()** function, students can start to build some fairly complex structures. After all, a house is little more than a cube of bricks with a second cube of air placed inside it! More cubes of air can be added to place holes for windows and doors. Your students will have to plan carefully to figure out where to place their cubes, and perform some basic mathematics to calculate exact coordinates and sizes.

## Dropping blocks as you walk

Now you know how to place blocks, you can play with continuously updating your location and placing blocks as you walk. This is a fantastic way of introducing loops to students.

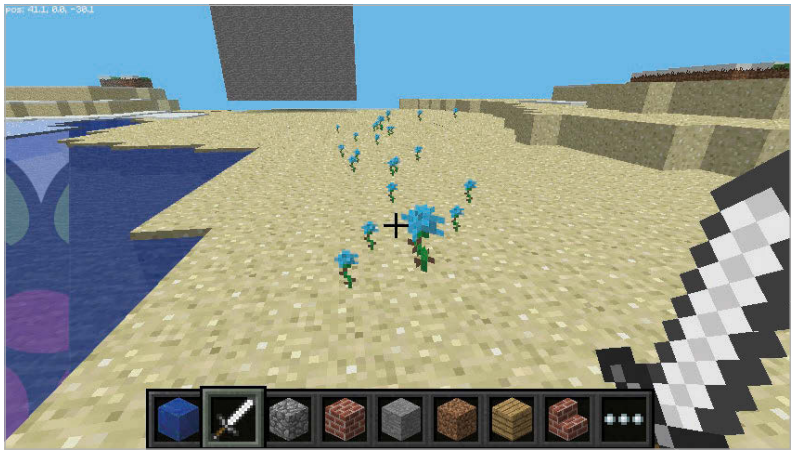
The following code will drop a flower behind you wherever you walk. You'll notice that the variable **flower** is set to **38** and then used in the **mc.setBlock()** function. This makes your code a little more readable.

```
from mcpi.minecraft import Minecraft
from time import sleep

mc = Minecraft.create()

flower = 38

while True:
    x, y, z = mc.player.getPos()
    mc.setBlock(x, y, z, flower)
    sleep(0.1)
```

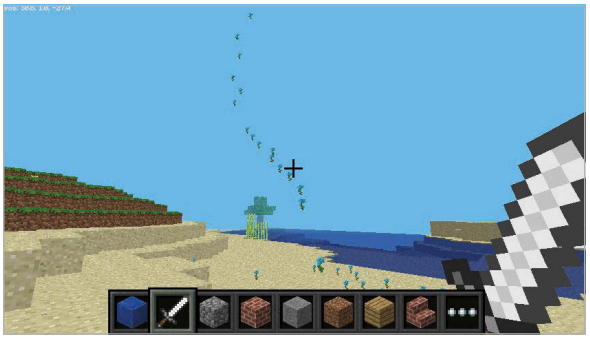
Now walk forward for a while and turn around to see the flowers you have left behind you.

As you used a **while True** loop, this will go on forever. To stop it, press **CTRL+C** in the Python Shell window (or click its 'x').

Try flying through the air and see your flower trail.



**Below** Leaving a trail of flowers on the ground



What if you only wanted to drop flowers when the player walks on grass? You can use **getBlock()** to find out what type of block is directly below you.

```
x, y, z = mc.player.getPos()
# player position (x, y, z)
block_beneath = mc.getBlock(x, y-1, z)
# block ID
print(block_beneath)
```

This tells you the ID of the block the player is standing on. Test this by running a loop to print the block ID of whatever you're currently standing on:

```
while True:
    x, y, z = mc.player.getPos()
    block_beneath = mc.getBlock(x, y-1, z)
    print(block_beneath)
```

Now you can use an **if** statement to choose whether or not we plant a flower:

```
grass = 2
flower = 38

while True:
    x, y, z = mc.player.getPos()
    # player position (x, y, z)
    block_beneath = mc.getBlock(x, y-1, z)
    # block ID

    if block_beneath == grass:
        mc.setBlock(x, y, z, flower)
        sleep(0.1)
```

Perhaps next you could turn the tile we're standing on into grass if it isn't grass already:

```
if block_beneath == grass:
    mc.setBlock(x, y, z, flower)
else:
    mc.setBlock(x, y-1, z, grass)
```

Now you can walk forward and if you walk on grass, you'll leave a flower behind. If the next block is not grass, it turns into grass. When you turn around and walk back, you now leave a flower behind you. This

type of activity is an excellent way of introducing the power of conditional selection, and allows students to play with different and increasingly complex statements to change their world, depending on the blocks that are around them.

## Playing with TNT blocks

Another interesting block is TNT! To place a normal TNT block, use:

```
tnt = 46
mc.setBlock(x, y, z, tnt)
```

However, this TNT block is fairly boring. The TNT block you've just made is no different from any of the other standard blocks in Minecraft. You could safely use it to build a TNT house! By passing some extra data to the function, you can tell Minecraft Pi that the TNT should be explosive.

```
tnt = 46
mc.setBlock(x, y, z, tnt, 1)
```

Now use your sword and left-click the TNT block: it will be activated and will explode in a matter of seconds.

Now try making a big cube of TNT blocks:

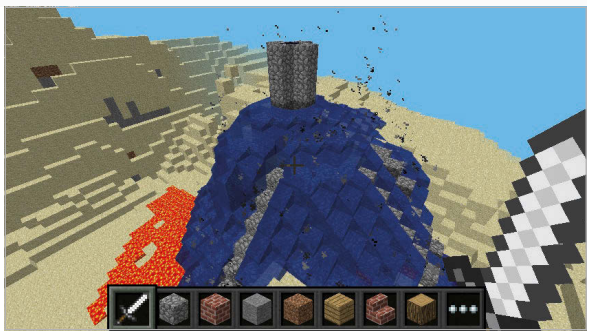
```
tnt = 46
mc.setBlocks(x+1, y+1, z+1, x+11, y+11, z+11, tnt, 1)
```

Now you'll see a big cube full of TNT blocks. Go and activate one of the blocks, and then run away to watch the show! The speed at which the explosion will render will again depend on the type of Raspberry Pi you're using. There's quite a lot going on in the game when a large cube of TNT explodes so, don't expect a real-time explosion.

## Fun with flowing lava

One block that's a lot of fun to play with is flowing lava.

```
from mcpi.minecraft import Minecraft

mc = Minecraft.create()
x, y, z = mc.player.getPos()
lava = 10

mc.setBlock(x+3, y+3, z, lava)
```



### FOR LOOPS

You can introduce `for` loops to your students to get them to build towers in the sky. Using the `range()` function in Python, this simple script will build a tower 100 blocks high.

```
from mcpi.minecraft import Minecraft

mc = Minecraft.create()
x, y, z = mc.player.getPos()

for i in range(100):
    mc.setBlock(x, y+i, z, 1)
```

Find the block you've just placed, and you should see lava flowing from the block to the ground. Some blocks in Minecraft obey the force of gravity, and some blocks in Minecraft can generate other blocks. Flowing lava does both of these things.

The other cool thing about lava is that when it cools down it becomes rock, just like in real life. You can get lava to cool really quickly with the addition of a little water. Move to another location in your Minecraft world and try the following code:

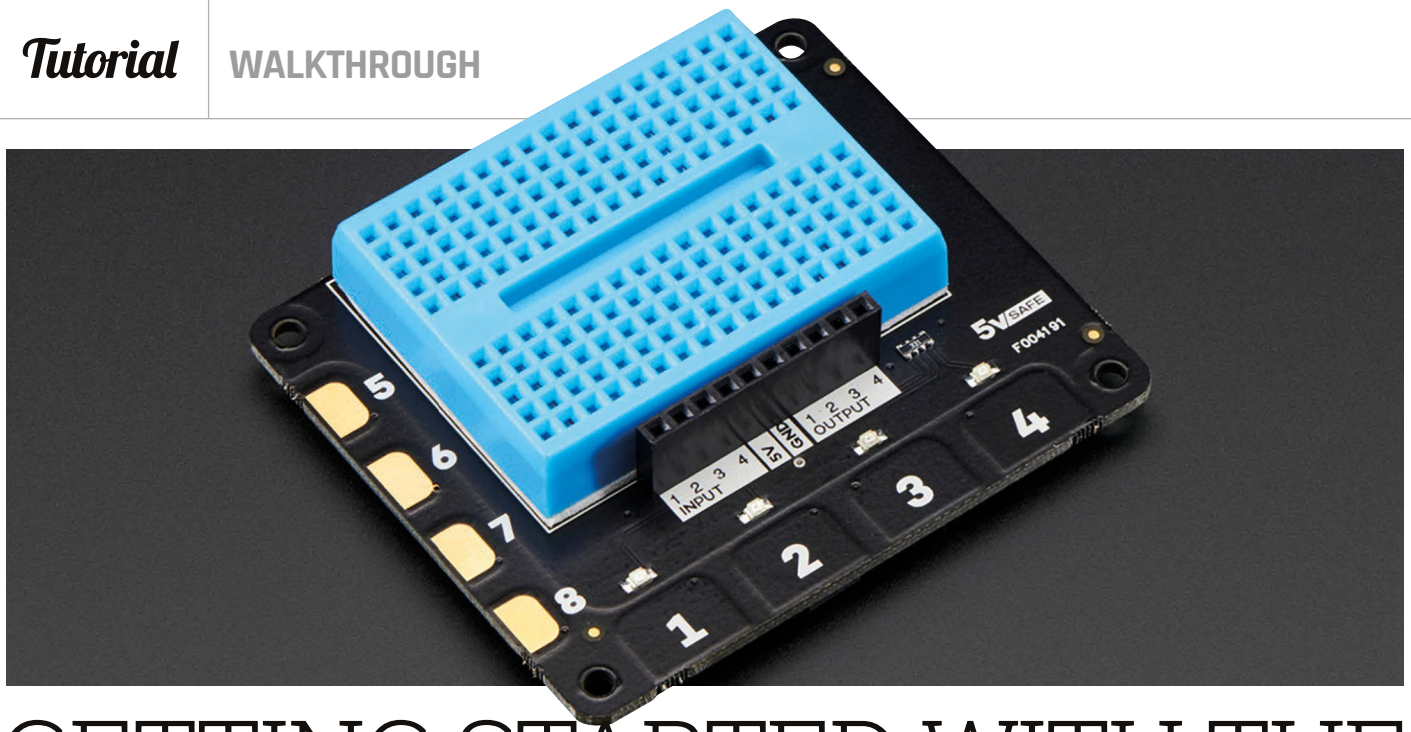
```
from mcpi.minecraft import Minecraft
from time import sleep

mc = Minecraft.create()
x, y, z = mc.player.getPos()
lava = 10
water = 8
air = 0

mc.setBlock(x+3, y+3, z, lava)
sleep(20)
mc.setBlock(x+3,y+5, z, water)
sleep(4)
mc.setBlock(x+3, y+5, z, air)
```

You can adjust the **sleep** parameters to allow more or less lava to flow.

## Moving on

There's plenty you can do now that you know your way around the Minecraft world and how to use the Python interface. See if you can write an algorithm to build a house or other building, for instance. At an even more advanced level, you could try building a house that follows you around. Or why not try adding in a little bit of physical computing? You could use buttons to trigger events in the Minecraft world, such as automatically placing cubes of TNT to help you quickly blast through mountains. Or you could use a mixture of LEDs to indicate the type of blocks that are beneath your feet.

Have a look on the Raspberry Pi website and try out this resource to make a Whac-a-Mole game: **magpi.cc/2ca208j**.




# GETTING STARTED WITH THE EXPLORER HAT PRO

If you want to play around with motors, capacitive touch, or analogue inputs with your Raspberry Pi, there's no simpler solution than using the Explorer HAT Pro

## You'll Need

- ExplorerHAT Pro
- LED
- 47Ω and 330Ω resistors
- Push button
- LDR
- 5V motor

Although you can drive motors using diodes and transistors, set up capacitive touch using a Darlington pair, and use capacitors and resistors to take measurements from analogue sensors such as a light-dependent resistor, you'll need to have access to all the right components and a fair bit of experience with electronics and programming. The Explorer HAT Pro abstracts away all the complexity of performing these tasks in an easy-to-use HAT, with a well-supported library of software that's just a simple pip3 install away. Moreover, the Explorer HAT Pro inputs and outputs are all 5V-safe, so you don't need to worry about accidentally breaking your Raspberry Pi when using 5V circuits.

We love physical computing at the Raspberry Pi Foundation, and are frequently amazed by the varied and exciting creations our community come up with. These projects can appear daunting to someone who is making their first foray into physical computing, though, and even following some of the simplest tutorials in online videos can lead to growing shopping lists of equipment and the technical know-how of a professional software developer.

With the Explorer HAT Pro, these problems disappear. Learners can rapidly and easily prototype their physical computing projects, making anything from autonomous robots to touch-sensitive musical instruments, with little more than a box of craft materials and a dozen or so lines of code.

In the rest of this article, we'll show you how to begin using the Explorer HAT Pro, and get up and running with physical computing in a matter of minutes.

## HANDLER FUNCTIONS

Handler functions are functions that are designed to do a certain task. So, in the examples provided in the section on using motors (overleaf), for instance, four handler functions were created, each of which drives the motors in a specific way. Handler functions can then be triggered by what are known as events. In the example, the events are button pushes. The program is listening in the background for the button pushes to occur, and these then call the handler functions.

## Getting ready to explore

If you want to use the Explorer HAT Pro, then you have two options available: it can be controlled using either Python or Scratch. For the purposes of this article, we'll focus on using Python.

First, you'll need to make sure I2C is enabled. To do this, open up the Raspbian **Menu**, go to **Preferences**, and then **Raspberry Pi Configuration**. You'll see I2C in the Interfaces tab, with a radio button that needs to be pressed next to the word Enabled.

To install the Python library so that you can use the HAT, you'll want to open a terminal window and type:

```
sudo apt-get install python3-smbus
sudo pip3 install explorerhat
```

Next, you'll want to shut down your Raspberry Pi and then attach the Explorer HAT Pro, before starting it back up again.

It's always best to ensure that everything is working as it's supposed to, so in the terminal you should type:

```
python3
>>> import explorerhat
```

You should see 'Explorer HAT Pro detected'. This means everything is working as planned, so you can close the terminal and get ready to explore.

## The built-in LEDs

The Explorer HAT Pro has four small LEDs that can easily be controlled with software. Open up **Python 3 (IDLE)** from **Programming** in the **Menu** and type the following directly into the Shell. (Don't type the > characters – they'll be added automatically.)

```
>>> import explorerhat
>>> explorerhat.light.red.on()
>>> explorerhat.light.blue.on()
>>> explorerhat.light.green.blink(1, 0.2)
>>> explorerhat.light.yellow.pulse(0.5, 2, 0.1, 1)
>>> explorerhat.lightoff()
```

It's worth playing around with the values passed to the **blink** and **pulse** calls to see what they do. Playing around with the numbers is the best way to learn about features of the library.

## Using other LEDs (or output components)

You can use the digital outputs on the Explorer HAT Pro to switch on LEDs, buzzers, or any other digital components, and the built-in breadboard makes connecting them easy. The example illustrated in **Fig 1** uses a 5mm red LED and a 47Ω resistor. You might notice that, unlike wiring an LED directly up to a GPIO pin, here you use the 3V output of the Explorer HAT, and the input on the HAT is acting as a switchable ground. The code is as simple as using the built-in LEDs. Again, you can write this directly in the Shell if you like, but the example shown has been written in a new file, which can be run by saving the file and then pressing **F5**.

```
import explorerhat
from time import sleep
explorerhat.output.one.on()
sleep(3)
explorerhat.output.one.off()
sleep(3)
explorerhat.output.one.fade(0,100,10)
```

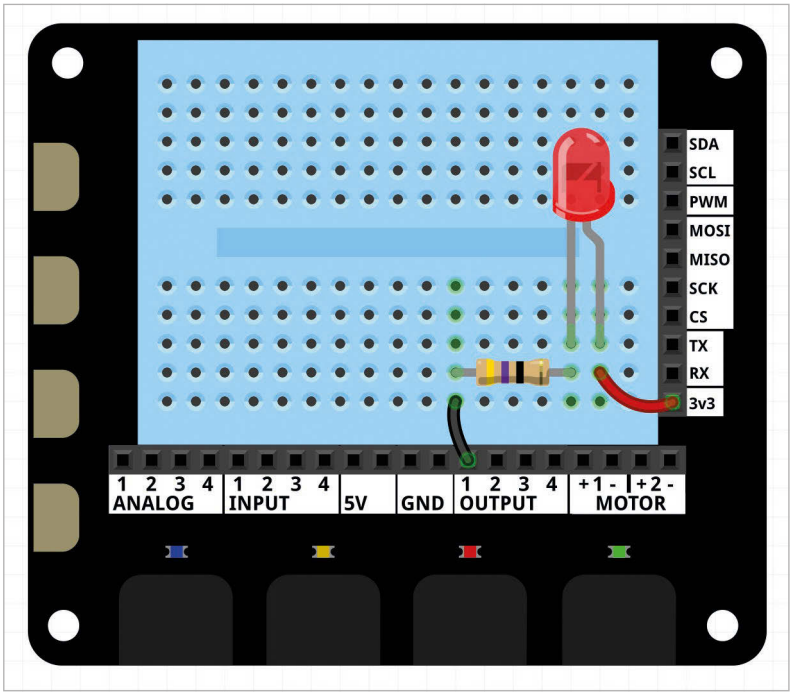



**Fig 1** Using the built-in breadboard to wire up an LED to a digital output, using a 47Ω resistor

Once again, it's worth having a play with the values to understand what the functions are doing. Can you figure out what the three values passed to **fade** do? You can also use all the previous functions on the built-in LEDs with your output components. What do you think **fade** would do if you used it with a buzzer?

## Buttons and capacitive touch

There are four touch-sensitive buttons on the Explorer HAT Pro, and a further four capacitive touch inputs. The touch-sensitive buttons are great for triggering events when testing out bits of code, and the capacitive touch buttons can be used either directly on the HAT itself, or wired up to any conductive surface using crocodile clips. The code overleaf shows how the buttons and capacitive touch inputs can be used to control the built-in LEDs with a few handler functions, which is by far the simplest way to keep the program listening out for the button presses.

### CAPACITIVE TOUCH

Although you can touch the capacitive touchpads directly, it's good fun to connect them up to conductive surfaces, and then trigger them by touching those surfaces. What do we mean by a conductive surface? Anything made of metal is a good start. Tinfoil works well, but then so do door handles, chair legs, and drawing pins. Most organic material is fairly conductive as well. How about making a musical instrument out of slices of fruit, or an alarm that's triggered if somebody tries to touch your lunch?

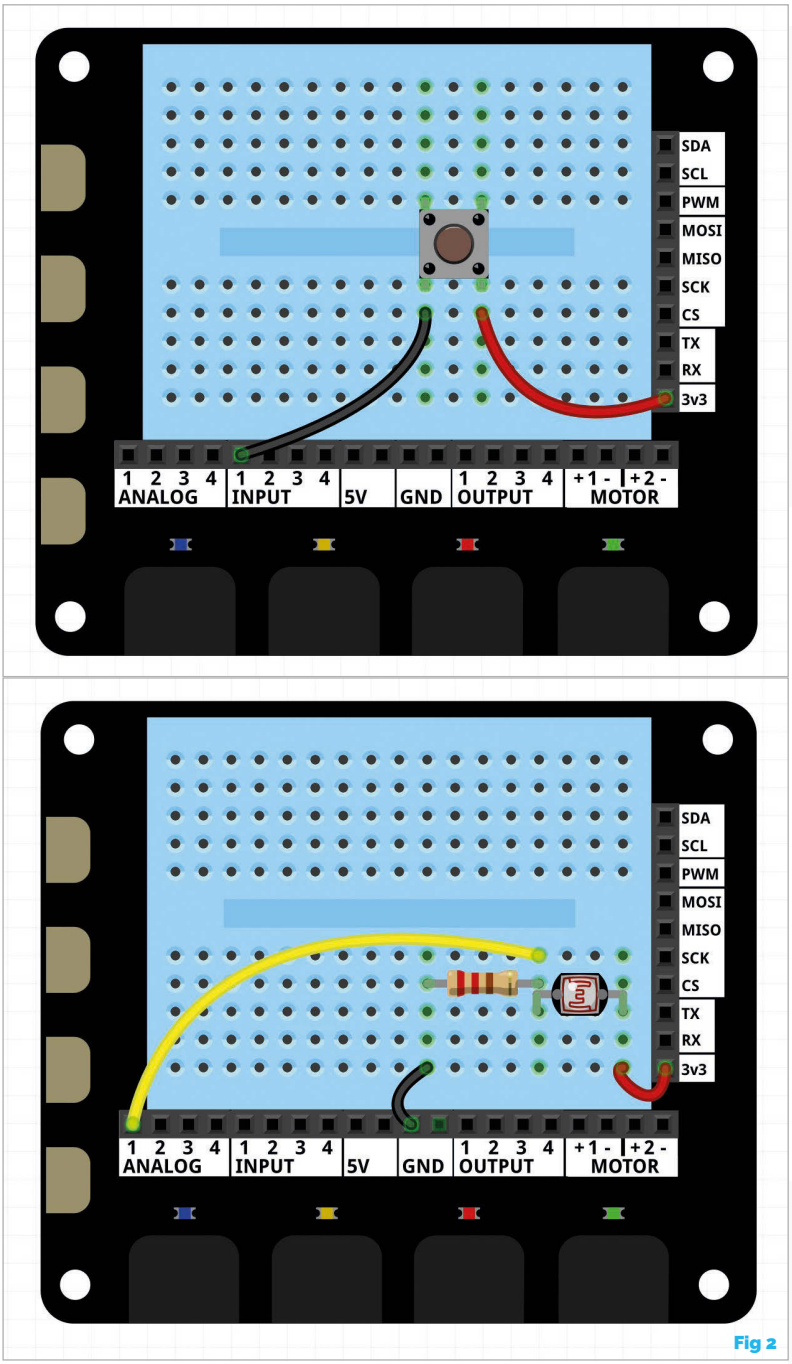



Fig 2

**Above top** Using the built-in breadboard to wire up a button to a digital input

**Fig 2, Above** Using the built-in breadboard to wire an LDR up to an analogue input, using a 220Ω resistor

## IMAGINATIVE USES OF MOTORS

Most often used to turn wheels to drive a vehicle, motors can be employed in much more imaginative ways. We've seen them used to wind pulleys, spin pointers, turn merry-go-rounds, and hurl ping-pong balls. Using physical computing to drive motion in one form or another can turn seemingly dull and boring code into an exciting and stimulating vehicle for learning. You'll be amazed at what children can come up with when you show them something as simple as a spinning wheel, and then give them free rein to design their own projects.

```
import explorerhat

def lightson(channel, event):
    explorerhat.light.on()

def lightsoff(channel, event):
    explorerhat.light.off()

def lightspulse(channel, event):
    explorerhat.light.pulse(1,0.1,1,0.1)

explorerhat.touch.one.pressed(lightson)
explorerhat.touch.two.pressed(lightsoff)
explorerhat.touch.eight.pressed(lightspulse)
```

Again, there's no reason why you couldn't use other output components instead of the built-in LEDs. How about using four different buzzers to create different sounds when the buttons are pushed?

## Analogue inputs and sensors

Trying to get readings from analogue components like a light-dependent resistor (LDR) can be tricky when using a Raspberry Pi on its own. With the Explorer HAT Pro it becomes so much easier, because of the four analogue inputs.

The example shown in the diagram (**Fig 2**) is using an LDR, which changes resistance depending on the amount of light hitting its surface. You could easily swap in another component, though, such as a thermistor for detecting changes in temperature. The code shown here will change the state of the built-in LEDs depending on the amount of light hitting the LDR. You might need to tweak the numbers a little to get the desired effect.

```
import explorerhat

while True:
    value = explorerhat.analog.one.read()
    if value < 0.4:
        explorerhat.light.on()
    else:
        explorerhat.light.off()
```

## Driving motors

Motors can be difficult things to use at times. Without using a motor controller chip or lots of transistors, resistors, and diodes, you run the risk of the motor not working and even damaging your Raspberry Pi. The Explorer HAT Pro makes driving motors forwards and backwards extremely easy, and you can wire your motors straight into the HAT without worrying about damaging the Raspberry Pi. Up to two motors (5V and up to 200mA) can be used with the HAT.

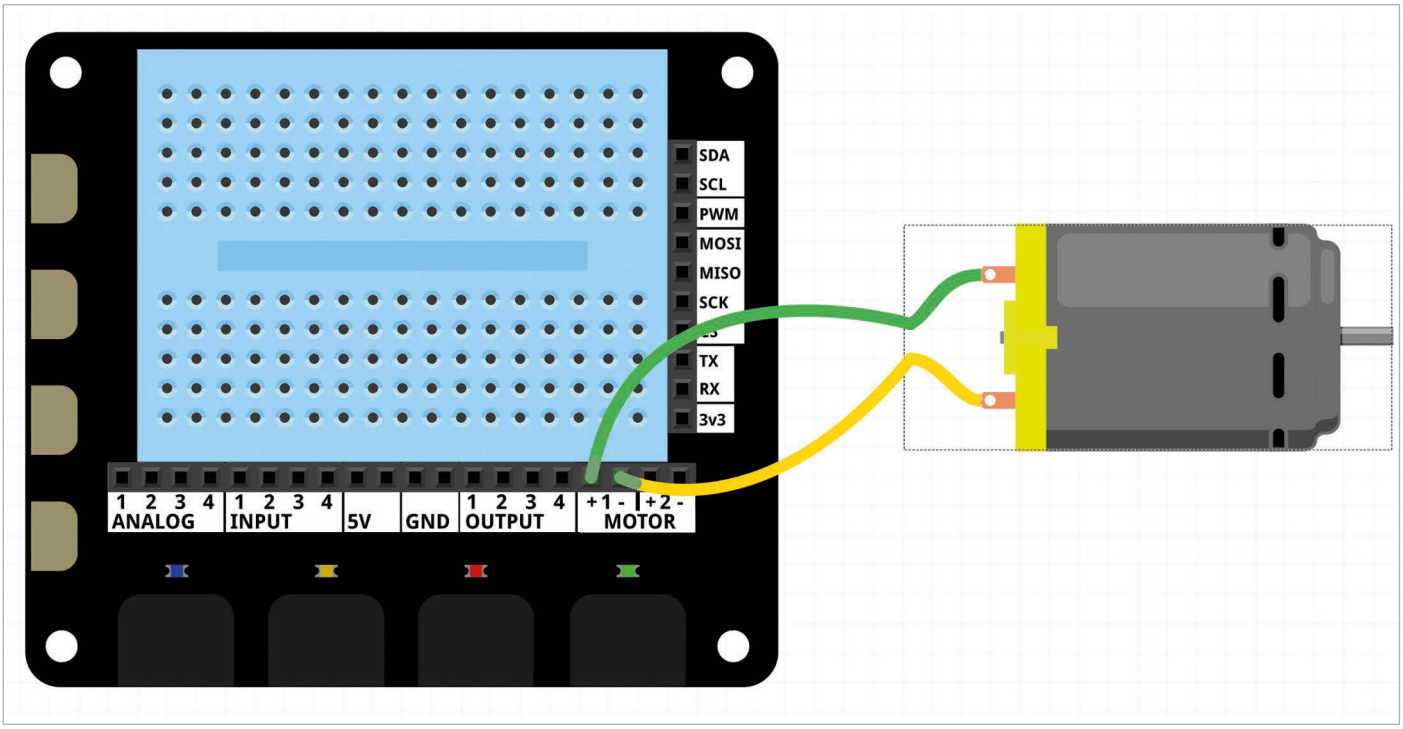



Fig 3 Wiring a motor directly to the 5V motor output pins on the Explorer HAT Pro

The example shown in **Fig 3** would be a simple setup to create a small robot driven by two motors, and using the HAT's buttons to control its motion using handler functions.

```
import explorerhat

def forward(channel, event):
    explorerhat.motor.one.forward(100)
    explorerhat.motor.two.forward(100)

def backward(channel, event):
    explorerhat.motor.one.backward(100)
    explorerhat.motor.two.backward(100)

def turn(channel, event):
    explorerhat.motor.one.forward(100)
    explorerhat.motor.two.backward(100)

def stop(channel, event):
    explorerhat.motor.one.forward(0)
    explorerhat.motor.two.forward(0)

explorerhat.touch.one.pressed(forward)
explorerhat.touch.two.pressed(backward)
explorerhat.touch.three.pressed(turn)
explorerhat.touch.four.pressed(stop)
```

## Unleashing the power of making

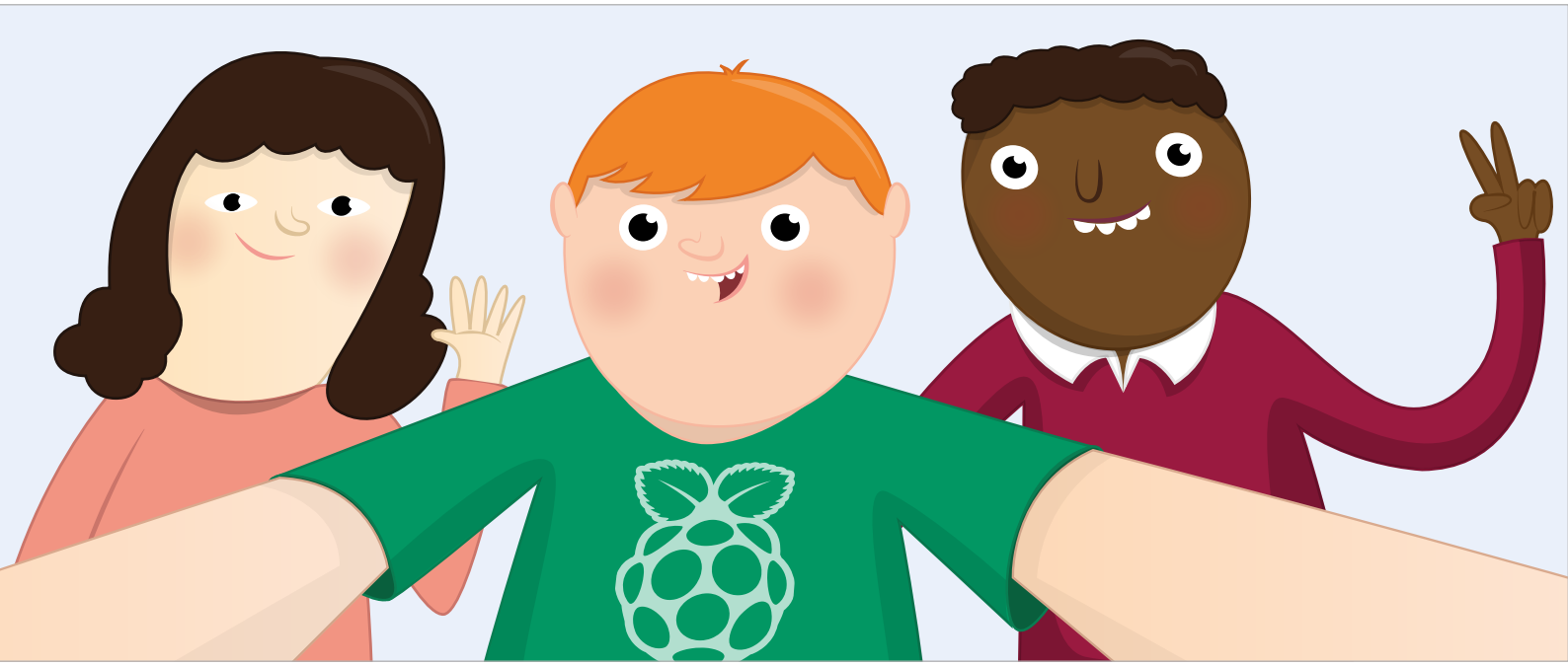
While everything you can do with the Explorer HAT Pro can be accomplished with a handful of electronic components and some electronics and programming know-how, what we love is the ease with which learners can go from knowing next to nothing to producing their own robots, humane mousetraps, and fruit-based musical instruments. You'll be amazed at what learners come up with once they've been shown a few of the basics, and then unleashed on a box full of craft equipment purchased from a local pound shop.

> "Learners can go from knowing next to nothing to producing their own robots"

The Explorer HAT Pro is also great when combined with other libraries. If you want a simple way to integrate some physical computing into Minecraft, for instance, then the Explorer HAT Pro is the perfect way to achieve this. Learners can play around with building gigantic structures at the touch of a button, or destroying whole mountains whenever the room gets a bit dark (sensed by an LDR). They can even use the LEDs to indicate how far away they are from their house, or maybe what block their character is standing on.

## And much, much more

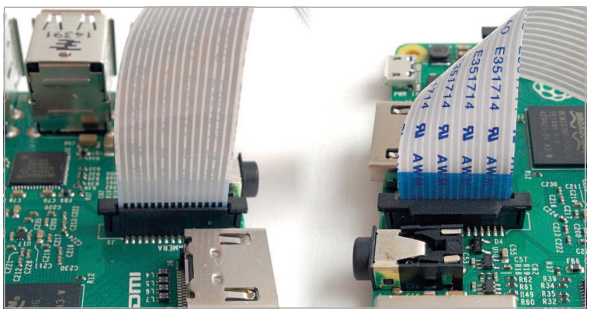
What we've tried to show you in this article is only a taster of what the Explorer HAT Pro is capable of. There's much more information in the Pimoroni documentation over on GitHub (**github.com/pimoroni/explorer-hat**), and if you don't yet have an Explorer HAT Pro then you can head on over to the Pimoroni website (**shop.pimoroni.com**) and pick one up now.



# CAPTURING INPUTS USING THE PI CAMERA MODULE & SENSORS

Capturing images and video using a Raspberry Pi Camera Module is a great way to get started with programming and making computing tangible: students can use button presses, sensors, and more to trigger their camera

Before you can delve into programming your Raspberry Pi camera, you're going to need to connect and test it. Whichever official Camera Module you are using, they all connect in the same way. Firstly, ensure your Raspberry Pi is disconnected from its power source and then locate the CAMERA connector (between the HDMI and headphone socket).



**Above** Make sure you securely connect the camera, with the blue tab facing the headphone socket

Next, lift the plastic flap slightly and insert the camera's cable with the blue tab facing the headphone socket. Finally, push the plastic flap down to secure the CAMERA connector, then power up your Raspberry Pi so that you can enable and test the camera.

By default, the camera is not enabled and so you'll have to enable it. You can do this by opening up a terminal window and typing `sudo raspi-config nonint set_camera 1` followed by the **ENTER** key. When you reboot, your camera will be enabled and ready to use (you only need to enable your camera once, not every time you use it).

Once rebooted, you can test your camera is enabled by taking your first image using the built-in command-line tool. Open a terminal window and type `raspistill -o test.jpeg`. This will start a camera preview before waiting five seconds and then taking a picture. You can then open and check your picture in the File Manager from the desktop.

## Capturing images using Python

Taking pictures in this way is really useful, but controlling the camera from a Python program is far more powerful. You can incorporate the camera into your own projects and customise its behaviour in great detail.

To get started, open up Python 3 (IDLE) from the desktop **Menu** – you'll find it under **Programming**. Once it's open, create a new program file by clicking **File > New File**.

The first program you're going to write will simply replicate the camera behaviour you saw earlier and take a picture after five seconds. Type the code below into your new empty file, then save it as **selfie.py**.

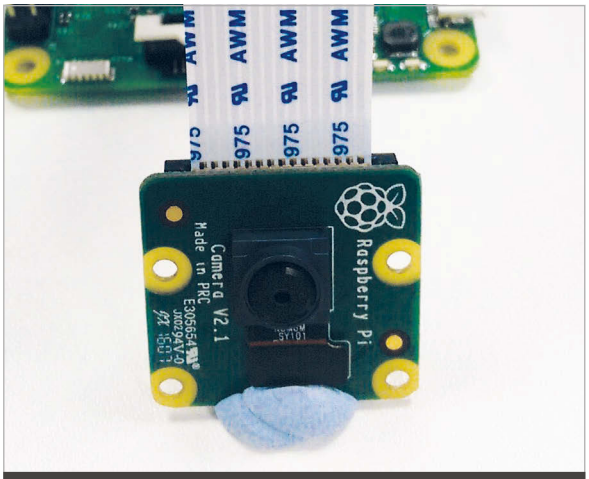
```
# selfie.py - takes a single picture after 5 seconds
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.start_preview(alpha=192)
sleep(5)
camera.capture("/home/pi/image.jpg")
camera.stop_preview()
```

- The first line, beginning with a **#** symbol, is a comment or annotation. It serves no purpose other than for the reader.
- The following two lines add two extra libraries (or abilities) to your project – in this case, controlling the camera and sleeping (adding pauses).
- The **`camera = PiCamera()`** line connects to the Pi camera and names that connection 'camera'. This enables you to use the camera functions.
- The final four lines start an image preview (which is transparent), pause, capture an image, and then stop the preview.

To run your code, press **F5**. You should see the image preview appear and, five seconds later, an image will be captured.

So far, your Python program only replicates what you could already do with **`raspistill`**. However, you could make some simple adaptations such as the length of the pause before a picture is taken, or disable the preview by adding a **#** before the two lines that mention the preview.

If you run this program more than once, you'll find that the image that is taken each time replaces the previous **image.jpg**. One of the first questions students will ask is how they can capture multiple images. In order to do this, you will need to add a loop to your program. Let's first look at a finite loop that captures five pictures.




### TOP TIPS

- It's a good idea to find some way to hold the camera still, and also away from the Raspberry Pi. If the camera touches the GPIO pins, it can cause some damage to the camera. You can buy camera mounts and cases, but we've found that a blob of Blu-Tack works just as well.
- The standard 15cm camera cable is great for most projects, but sometimes you want something a little longer. Most Raspberry Pi suppliers stock longer cables (up to 2m) for a couple of pounds.
- If your camera is mounted upside down, you can rotate the image by adding the following line to your code:
  **`camera.rotation = 180`**

Rather than rewriting what will be a similar program, you could adapt your **selfie.py** code to make it capture multiple pictures. First, click **File > Save As** and rename it as **images.py**.

You need to make a couple of changes to your selfie code, as follows.

```
# selfie.py - takes a single picture after 5 seconds
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.start_preview(alpha=192)

for i in range(5):
    sleep(5)
    camera.capture("/home/pi/image{}.jpg".format(i))

camera.stop_preview()
```

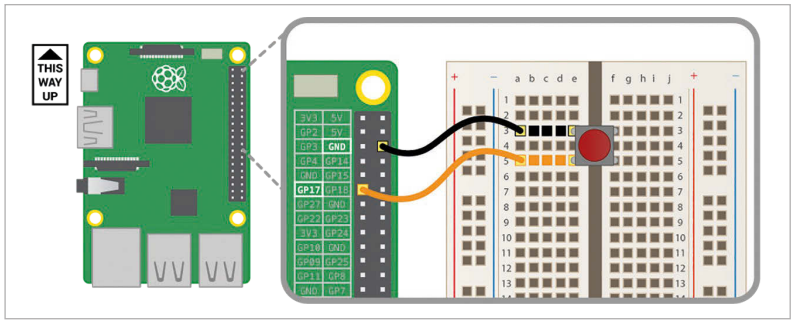



**Fig 1** Connect your button to pin 17 and a ground pin. If you don't have a button, make one by tapping your two bits of wire together

The new code includes the line **`for i in range(5):`**. This begins your loop and will make **`i`** count from 0 to 4 and stop when it reaches 5. The two lines (now indented) below will be repeated each time the loop runs. There is one final change, which is to the file name: the new code inserts the value of **`i`** instead of the **`{}`** within the file name. When you save (**CTRL+S**) and run (**F5**) your new code, you will end up with five numbered images, from **image0.jpg** to **image4.jpg**.

By experimenting with the **`sleep`** time and number of images, you and your students can find a number of applications for this code. For example:

- Capture a set of time-lapse images to monitor the behaviour of a plant every minute for 24 hours.

```
for i in range(1440):
    sleep(60)
    camera.capture("/home/pi/image{}.jpg".format(i))
```

- Capture ten images within a second to monitor a rapid event such as a photo finish.

```
for i in range(10):
    sleep(0.1)
    camera.capture("/home/pi/image{}.jpg".format(i))
```

## Triggering the camera with sensors

Once you've captured your first images with Python, there are loads of projects you could do. One simple idea is to trigger the camera with a button press. For this you will need a few components:

- 2 × Male-to-female jumper wires
- Breadboard
- Push button

Using these components, you should build the circuit shown in **Fig 1**, connecting a ground (GND) pin to pin 17 via a push button.

When the button is pressed, it will complete the circuit and we can use Python to watch pin 17 for this press and take a picture. You could use any GPIO pin for this, or even multiple buttons on different pins triggering different events.

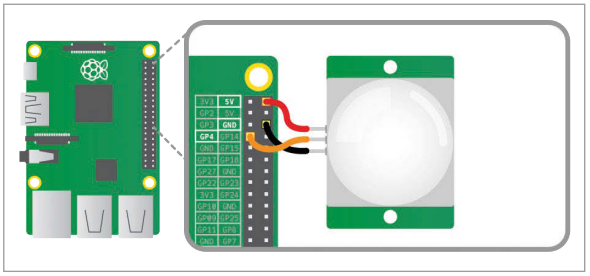
You'll need to make some adaptations to your **selfie.py** code to make it respond to a button. First, **File > Save As** and rename it as **button.py**.

```
# stopframe.py - captures a series of
pictures in response to a button press
from picamera import PiCamera
from gpiozero import Button
from time import sleep

camera = PiCamera()
mybutton = Button(17)

camera.start_preview(alpha=192)
for i in range(5):
    mybutton.wait_for_press()
    camera.capture("/home/pi/button.jpg")
camera.stop_preview()
```

A button is a great way to trigger an event from the real world, but it is not the only possibility. We could use any number of sensors to make this happen. For this next example, you'll need a passive infrared (PIR) sensor, like the kind of thing you'd find in a burglar alarm. These sensors can detect movement, and can be easily connected to your Raspberry Pi using three female-to-female jumper wires.

There are three pins on a PIR sensor: VCC (power), GND (ground), and OUT. You may have to remove the plastic dome on top in order to see the pin labels.


Connect the VCC to the 5V pin, and GND to any ground pin of your Pi, then connect the OUT pin to any GPIO pin. In the diagram below, pin 4 has been used.



Now all you need to do is to make some slight adjustments to your **button.py** code. Instead of a button on pin 17, you need a motion sensor on pin 4. First, **File > Save As** and rename it as **motion.py**.

```
## motion.py - captures a picture in
response to a PIR sensor trigger
from picamera import PiCamera
from gpiozero import MotionSensor
from time import sleep

camera = PiCamera()
pir = MotionSensor(4)

pir.wait_for_motion()
camera.start_preview(alpha=192)
sleep(5)
camera.capture("/home/pi/motion.jpg")
camera.stop_preview()
```

Save your code (**CTRL+S**) and run (**F5**) it to test your motion-sensitive camera.

## Video and image effects

The PiCamera library also lets you capture video in much the same way as stills. To do so, you use the command **camera.start_recording**. You also need to set the frame rate and length of the video.

Here's some example code:

```
##video.py - Captures a 15 second video
from picamera import PiCamera

camera = PiCamera()
camera.start_preview(alpha=192)
camera.framerate = 24
camera.start_recording('my_video.h264')
camera.wait_recording(15)
camera.stop_recording()
camera.stop_preview()
```

To play the captured video with the Pi's built-in omxplayer, you need to convert it using the avconv tool. Download the latter from a terminal window with **sudo apt-get install libav-tools**. Then convert the video with the following command:

```
avconv -r 24 -i my_video.h264 -vcodec copy
my_video.mp4
```

The 24 is the frame rate you recorded in, and you'll also need to replace the input and output file names to match whatever you recorded. To play your video back, enter the following command into a terminal window.

```
omxplayer my_video.mp4
```

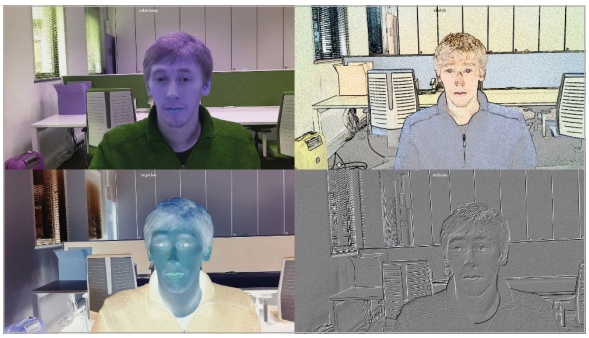
Another nice feature of the PiCamera library is the ability to apply a camera effect to whatever is being captured, using the **image_effect** property. For example, to capture a negative image you would add **camera.image_effect = 'negative'** to your code, anywhere after the line **camera = PiCamera()**.

## READ THE DOCS

We'll only scratch the surface of what the Pi camera can do in this guide. However, the PiCamera Python library is really well documented. To find out more about the options available when using the library, as well as some further recipe examples, visit : **picamera.readthedocs.io**

```
# effect.py - takes a single negative
picture after 5 seconds
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.image_effect = 'negative'
camera.start_preview(alpha=192)
sleep(5)
camera.capture("/home/pi/image.jpg")
camera.stop_preview()
```

Below, Ben Nuttall showcases the effects colorswap, sketch, negative, and emboss. Find a full list of effects in the PiCamera documentation (see 'Read the Docs').



In this guide, you have captured images and video, and connected with physical sensors to trigger events. Given some simple starting points like these, your students can start to devise their own projects and get creative with computing!

## FURTHER PROJECTS

If you enjoyed working with the camera and want more ideas for your students, browse our free learning resources at **rpf.io/learn**.
A few highlights include:

**TWEETING BABBAGE – magpi.cc/2c8PWnz**
Capture images using a camera and motion sensor to then post to Twitter.

**MICROBIT SELFIES – magpi.cc/2c8PPbW**
Connect a BBC micro:bit to your Raspberry Pi and use its buttons to trigger the camera.

**INFRARED BIRD BOX – magpi.cc/2c8P2rr**
Use an infrared camera and LED light to capture wildlife images at night.

# BLINKY LIGHTS WITH GPIO ZERO

GPIO Zero makes controlling hardware child's play: using this Python library and a Raspberry Pi, your students can take their first steps into physical computing

## Why GPIO Zero?

At the time of writing, it's included in the standard Raspberry Pi image, but what exactly is GPIO Zero? We spoke to Ben Nuttall, one of its developers.

"GPIO Zero provides a simple interface to everyday physical components," Ben tells us. "Looking at a 'Hello World' program in Java, it's about five, six, seven lines long and there's lots of stuff that you wouldn't really explain; you just have to write it. That's something you don't get with the 'Hello World' of Python, but you do if you're trying to make an LED flash. Rather than importing the GPIO library and performing complicated setup at the start, it should just be that you have an LED and you turn it on."

GPIO Zero is built upon the already existing RPi.GPIO library, but abstracts away some of the details of dealing with hardware. Previously, beginners had to perform a number of setup steps before switching on the LED. In fact, a typical program to flash an LED would have taken ten lines of code – with GPIO Zero, your students can do it in three.

Of course, at some point students may want to understand what's going on behind the scenes, and may need that full level of detail. However, GPIO Zero allows beginners to make something happen first and choose at what level they need or want to understand it.

In this simple tutorial, you will make an LED flash using Python, which can be extended and adapted to a whole range of projects.

## Hardware setup

Whenever you work with physical components, whether it's robots, sensors or the humble LED, you're introducing something else you need to be able to debug. For this reason, setting up your hardware and testing it works is always a sensible step before jumping into programming.

The first step in making an LED flash is to connect it to your Raspberry Pi and test it. To do this, you'll build the simplest of circuits, which includes a power supply, LED, and a resistor. **Fig 1** shows a simple circuit diagram, and **Fig 2** is how you would build this with your Raspberry Pi.

In this setup, the Pi is supplying the power (3.3 volts) and acting as the positive terminal on a battery. The circuit is completed by connecting to a ground pin

**Below** A simple circuit including a cell, resistor and LED, built using a breadboard

Fig 1

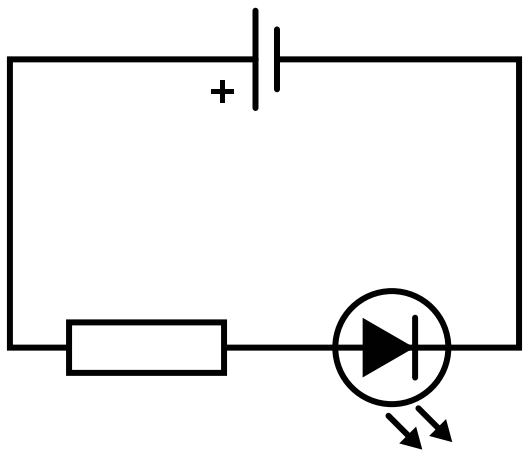


Fig 2

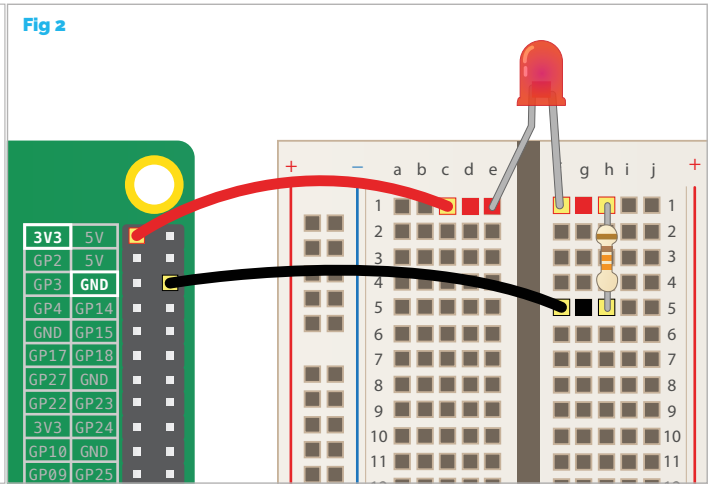


Fig 3

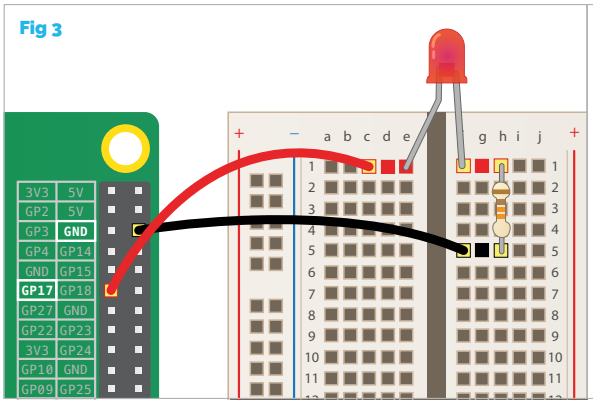
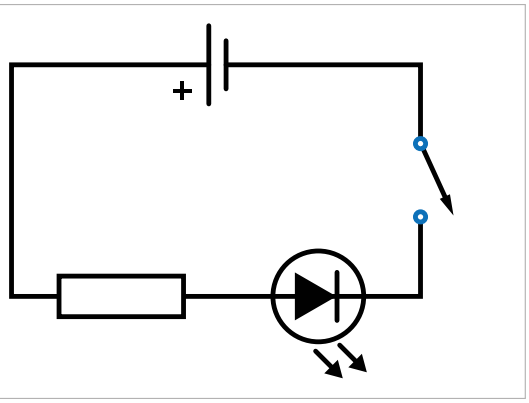



Fig 4



**Left** Two simple LED circuits, one with a switch and one without

acting as the negative terminal. If when you connect the circuit, the LED doesn't light up, then check the LED, as it has to be the correct way round.

Once you're happy that the LED is working, you should now disconnect the wire going to the 3.3V pin and move it to a GPIO pin; in this example pin 17 is being used (**Fig 3**). You'll probably find that when you do this the LED turns off, as the GPIO pin acts like a switch in our circuit (**Fig 4**). At the moment that switch is off, but you can control it using Python.

## Flashy flashy

Now comes the fun bit: you can write some code that will make the LED flash. The first thing you'll need is a Python window to write your code in. To load Python, click on the main **Menu**, followed by **Programming**, and finally **Python 3 (IDLE)**.

IDLE is the default environment for writing Python code and initially loads a Python shell (it will have a >>> prompt). What you want is an empty file, so create one by clicking **File > New File**. You now have an empty program window waiting for your code. Begin by importing the **LED** class from the GPIO Zero library, and the **sleep** function from the **time** library:

```
from gpiozero import LED
from time import sleep
```

Next, create a new LED in your program and assign it a variable name. This could be anything; in the example it's called **my_led**.

```
my_led = LED(17)
```

The number 17 corresponds to the GPIO pin being used; if you've used a different pin, then replace 17 with the appropriate number. Next, you need to begin a loop that will switch the LED on and off with pauses in between.

```
while True:
    my_led.on()
    sleep(1)
    my_led.off()
    sleep(1)
```

The **while True:** line creates an infinite loop and all the indented lines below are inside that loop; the indentation is important to how Python understands the code. Your final code should look like this:

```
from gpiozero import LED
from time import sleep

my_led = LED(17)

while True:
    my_led.on()
    sleep(1)
    my_led.off()
    sleep(1)
```

To test your code, you first need to save it; press **CTRL+S** to save and call it something like **led.py**. To run the code, you can now press **F5**; this should switch focus to the shell window and you should see your LED flashing. Well done!

Hang on: didn't we promise a program that flashes an LED in three lines? This last program was eight lines long! This is a really good example of abstraction, a key concept for learners to understand. Abstraction is all about hiding details, and GPIO Zero has a number of these built in. A really good example is **blink**, a function which does the same as the loop you just wrote and includes the **sleep** functionality. This means your above program can be rewritten as below:

```
from gpiozero import LED

my_led = LED(17)

my_led.blink()
```

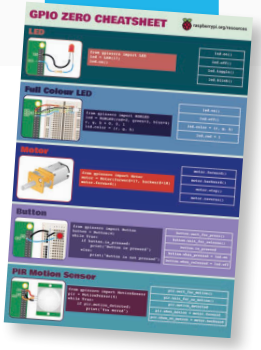
You can also adjust the timing of the blinks by passing some values into the **blink** function; for example, **my_led.blink(2,0.5)** would switch the LED on for 2 seconds and then off for 0.5 seconds.

**blink** is just one example of the many built-in functions within GPIO Zero, which can be explored further using the included poster or by visiting **gpiozero.readthedocs.io**.

### GPIO ZERO CHEATSHEET

To help you and your students get started with GPIO Zero, you'll find a handy poster included with this magazine (which can also be downloaded as a PDF from **raspberrypi.org**).

It shows you how to connect a number of components including LEDs, motors, and motion sensors. It also includes code snippets to get you started with the basic functionality of each device.



**SAM AARON**

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

Sonic Pi

# LIVE CODING EDUCATION

## You'll Need

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

Join Sonic Pi creator and live performer **Sam Aaron** as he introduces us to the joys of live-coding music on the Raspberry Pi...

The lights sliced through the mist as the rhythms moved the crowd, the atmosphere ripe with a heady mix of synths. However, something wasn't quite right. Projected in bright colours above the performer was futuristic text, moving, dancing, flashing. This wasn't fancy visuals; it was merely a projection of Sonic Pi running on a Raspberry Pi. The musician wasn't spinning discs – she was writing, editing, and evaluating code. Live. This is live coding.

It may sound like a far-fetched scene from a futuristic music festival, but is actually a description of a pupil performing at their primary school assembly. Coding music like this is a growing trend and is often described as live coding. However, this approach to manipulating code isn't just useful for performance; it's also a fabulous way of engaging a new generation of coders in the classroom.

## Engaging a new generation of coders

Across the educational landscape, it's becoming clear that teaching programming is an increasingly important aspect of our curriculum. What is not obvious, however, is exactly how we go about engaging students in order to effectively teach the core fundamentals of programming. One approach to solving this problem has been to ask expert programmers what excited them when they started learning. This has yielded responses ranging from sorting algorithms to binary arithmetic. Whilst these may be deeply interesting and engaging topics for computer scientists, it's not clear that they are effective topics for exciting a broad and diverse range of students in introductory computing. This is especially significant considering that most students will not embark in a career in programming in either industry or academia.

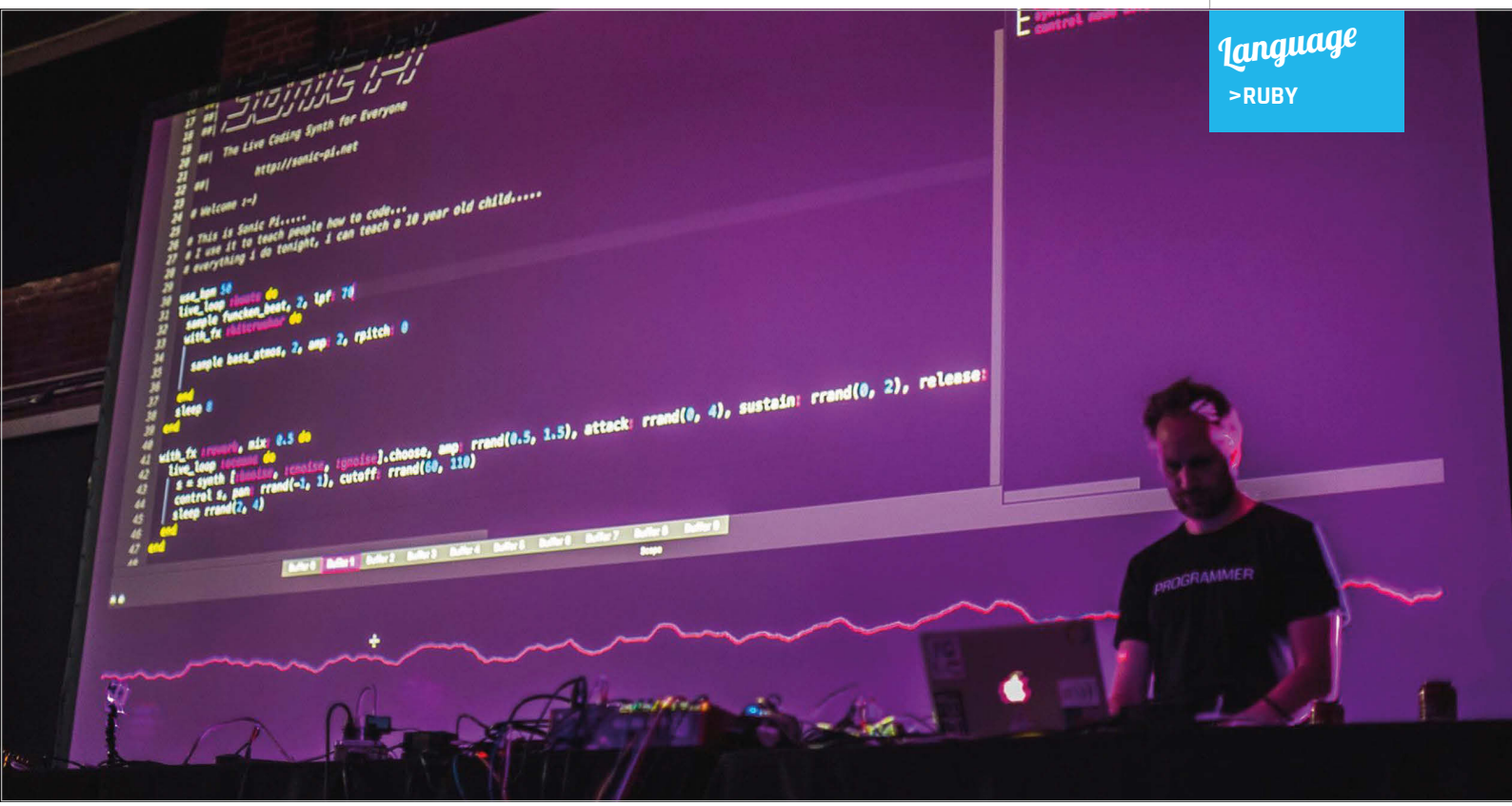
Sonic Pi takes a different approach. It turns code into a powerful new kind of musical instrument with a focus on fast feedback and iterative learning. It enables students to code the kinds of music they're typically used to listening to. If this claim sounds a little ambitious, take a look at this quotation from the *Rolling Stone* magazine (**magpi.cc/2ciPcr**) covering a recent Sonic Pi performance at Moogfest USA:

"[The set] – which sounded like Electric Café-era Kraftwerk, a little bit of Aphex Twin skitter and some Eighties electro – was constructed through typing and deleting lines of code. The shadowy DJ sets, knob-tweaking noise and fogbank ambient of many Moogfest performers was completely demystified and turned into simple numbers and letters that you could see in action. Dubbed 'the live coding synth for everyone', it truly seemed less like a performance and more like an invitation to code your own adventure."

## Powerful yet simple

Of course, there's very limited educational value in a system that's powerful yet incredibly difficult to learn. However, Sonic Pi was conceived and built within classrooms in close collaboration with teachers and has benefited from many design iterations based on hundreds of hours of student observation. Simplicity has even been baked into

Language
>RUBY




**Above** Sonic Pi creator Sam Aaron performing at Moogfest 2016

the core design philosophy by only allowing features that may be easily understood by and taught to a ten-year-old child.

For example, in order to get started, you only need to learn two extremely simple commands: **play**, which allows you to play different notes, and **sleep**, which enables you to choose how long to wait before playing the next note. Once you've mastered **play** and **sleep**, the next command to learn is **sample**, which gives you the ability to play any pre-recorded sound. Sonic Pi includes a large number of built-in recordings to use, such as drums, guitars, and atmospheric noises. However, the real fun starts when you record your own sounds to play back and manipulate. Finally, you also have access to a full set of professional studio effects, such as echo and reverb, to manipulate your sounds. For instance, playing a sample of a guitar with reverb is as simple as:

```
with_fx :reverb do
  sample :guit_harmonics
end
```

## Explore the full computing curriculum

Considerable care and attention has been placed to ensure that Sonic Pi allows educators to deliver all of the core concepts in the UK's new computing curriculum. It is an ideal follow-on language to Scratch due to its simple block-like, text-based syntactic structure. It is already seeing extensive use worldwide in all levels of education, from primary through to university. There are even primary schools which encourage their pupils to give performances with Sonic Pi during assemblies.

There is already a lot of support for educators who wish to use Sonic Pi in the classroom. You can easily use Sonic Pi to teach basic computing concepts such as sequencing, iteration, selection, functions, data structures, and algorithms. However, rather than teaching these ideas in a distant, abstract way, you are invited to deliver them using a simple, engaging musical narrative which gives the constructs extra meaning and motivation. For example, instead of sequencing, you could teach melodies; instead of iteration, repeating rhythms; instead of lists, bass lines or riffs. This is possible because each and every musical idea in Sonic Pi is immediately represented by one or more core computer science techniques. You can even go much further than the basic curriculum and explore concurrency, determinism, and even live coding. Before we dive into these advanced topics, let's take a quick look at how Sonic Pi brings new meaning to the basic concepts of sequencing, iteration, and selection.

> "You can easily use Sonic Pi to teach a range of basic computing concepts"

## Sequencing melodies

From the outset, Sonic Pi encourages the learner to code sequences of instructions. This is because the most simple programs consist of sequencing calls to **play** and **sleep** to create simple melodies:

```
play 70
sleep 1
play 75
sleep 0.5
play 82
```

What's exciting here from a musical perspective is that by sequencing **play** and **sleep** like this, you have access to most of Western music. In other words, with just two commands you can recreate any known melody or rhythm, and of course you're entirely free to compose your own.

## Iterating rhythms

Iteration is also very simple to represent and work with. For instance, imagine we wish to play a bass drum five times followed by a cymbal three times:

```
5.times do
  sample :bd_haus
  sleep 0.5
end

3.times do
  sample :drum_cymbal_open
  sleep 1
end
```

In the example above, all the code between the first **do** and **end** lines is repeated five times, and the code within the second block is repeated three times. We therefore play the **:bd_haus** sample and wait for half a second repeatedly five times, before finishing by playing the **:drum_cymbal_open** sample and waiting for one second repeatedly three times.

**Below** It's amazing what interesting sounds can be created with just a few lines of code


## Selecting sounds

Selection is also easy to represent. Let's use some randomisation to repeatedly choose whether to play a cowbell or a snare drum with a one in two chance of playing either percussion instrument:

```
loop do
  if one_in 2
    sample :drum_cowbell
  else
    sample :drum_snare_hard
  end
  sleep 0.125
end
```

In this example we use an infinite **loop** to repeatedly call the function **one_in** with the parameter **2**. This will return **true** or **false**, with a 50% chance of either. We then use this value to select one of two possible samples to play. If **one_in 2** returns **true** we hear a cowbell, otherwise we hear a snare drum. We then wait for **0.125** seconds before repeating. Executing this code creates an interesting non-repeating rhythm and is the basic building block of a probabilistic drum sequencer.

## A brief introduction to live coding

In order to get a taste of what Sonic Pi can do, let's now dive straight into the deep end and take a look at one of the most exciting aspects on offer: live coding. This is a new style of programming that provides many new learning opportunities in the classroom due to its immersive nature and incredibly fast feedback cycle. The basic idea is simple: in addition to the traditional start/stop model of programming, we can also continually tweak and modify the program as it runs. This allows us to rapidly explore the effects of small tweaks to the program, which both encourages tinkering and increases engagement. For example, once you have mastered the live coding workflow, it is easy to get into a flow-like state for long periods of time, modifying and remodifying the code.

## The live loop

The key to live coding is mastering Sonic Pi's unique programming construct, the **live_loop**. At a quick glance, it looks no more complicated than a 'forever' block in Scratch. Let's look at one:

```
live_loop :beats do
  sample :bd_haus
  sleep 0.5
end
```

There are four core ingredients to a **live_loop**. The first is its name. Our **live_loop** above is called **:beats**. You're free to call yours anything you want. Go crazy. Be creative. I often use names that

communicate something about the music I am making to the audience. The second ingredient is the **do** word, which marks where the **live_loop** starts. The third is the **end** word, which marks where the **live_loop** finishes, and finally there is the body of the **live_loop**, which describes what the loop is going to repeat – that's the bit between the **do** and **end**. In this case we're repeatedly playing a bass drum sample and waiting for half a second. This produces a nice regular bass beat. Go ahead, copy it into an empty Sonic Pi buffer and hit **Run**. Boom, boom, boom!

## Redefining on-the-fly

OK, so what's so special about the **live_loop**? So far it just seems like a glorified **loop**! Well, the beauty of **live_loop**s is that you can redefine them on-the-fly. This means that whilst they're still running, you can change what they do. This is the secret to live coding with Sonic Pi. Let's have a play:

```
live_loop :choral_drone do
  sample :ambi_choir, rate: 0.4
  sleep 1
end
```

Type in the code above and then press the **Run** button or hit **ALT+R**. You're now listening to some gorgeous choir sounds. Now, whilst it's still playing, change the rate from **0.4** to **0.38**. Don't hit **Stop** but instead hit **Run** again. Whoa! Did you hear the choir change note? Change it back up to **0.4** to return it to how it was. Now, drop it to **0.2**, down to **0.19**, and then back up to **0.4**. See how changing just one parameter on the fly can give you real control of the music? Now have a go at playing around with the rate yourself. Choose your own values. Try negative numbers, really small numbers, and large numbers. Most importantly, have fun!

## Sleeping is important

One of the core lessons about **live_loop**s is that they need rest. Consider the following **live_loop**:

```
live_loop :infinite_impossibilities do
  sample :ambi_choir
end
```

If you try running this code, you'll immediately see Sonic Pi complaining that the **live_loop** did not sleep. This is a safety system kicking in! Take a moment to think about what this code is asking the computer to do. That's right, it's asking the computer to play an infinite amount of choir samples in zero time. Without the safety system, the poor computer will try to do this and crash and burn in the process. So remember, your **live_loop**s must always contain a **sleep**.

**Above** Sonic Pi can be used alongside traditional musical instruments

## Combining sounds with concurrency

Music is full of things happening at the same time. Drums at the same time as bass at the same time as vocals at the same time as guitars.... In computing we call this concurrency, and Sonic Pi provides us with an amazingly simple way of writing concurrent code. Just use more than one **live_loop**!

```
live_loop :beats do
  sample :bd_tek
  with_fx :echo, phase: 0.125, mix: 0.4 do
    sample  :drum_cymbal_soft, sustain: 0, release: 0.1
    sleep 0.5
  end
end

live_loop :bass do
  use_synth :tb303
  synth :tb303, note: :e1, release: 4, cutoff: 120, cutoff_attack: 1
  sleep 4
end
```

Here, we have two **live_loop**s: one looping quickly making beats, and another looping slowly to make a crazy bass sound.

One of the interesting things about using multiple **live_loop**s is that they each manage their own time. This means it's really easy to create complex polyrhythmical structures and make very interesting music with only a few lines of code.

## Use Sonic Pi in your classroom today

This ends our whistle-stop tour of live-coding music with Sonic Pi. If you are excited about this new approach to engaging students with computing, you can download the app from **sonic-pi.net**. It is completely free and cross-platform, working identically on Windows, Mac, and the Raspberry Pi. It comes with a complete built-in tutorial which assumes you know nothing about either code or music. There is also an accompanying (free) book published by *The MagPi* (**magpi.cc/1VGIOux**) which includes many ideas and invitations to experiment and learn. Finally, there is a full scheme of work targeting the new computing curriculum, which is free to download from the Raspberry Pi website.

FIND THIS & OTHER RESOURCES: rpf.io/learn

# MAKE A DIGITAL MAGIC 8 BALL

Bring the time-honoured tradition of shaken-not-stirred fortunes to the Sense HAT, making use of those motion sensors

## You'll Need

- Sense HAT **magpi.cc/1TGGUt5**
- Sense HAT Python library **magpi.cc/1RKRoqc** (pre-installed with Raspbian Jessie)

In this activity you will build your own Magic 8 Ball using your Raspberry Pi, a Sense HAT, and some Python code. A Magic 8 Ball is a toy to which you verbally ask a closed question. You then shake it, and it will give you a prediction.

In this tutorial you will use IDLE 3, the Python development environment, to write some code for the Sense HAT. This means you can test your code and fix it as you write it. To open IDLE 3, click on the main Raspbian **Menu** (at the top-left of the desktop), followed by **Programming** and then **Python 3 (IDLE)**.

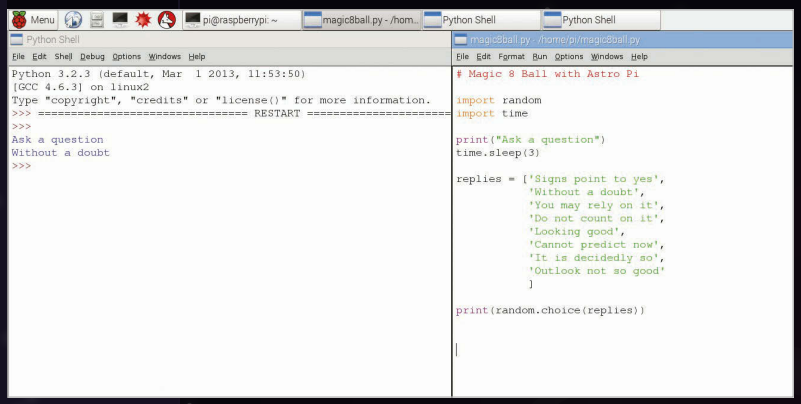



**Fig 1** Test the program first in a Python shell window

Once the Python Shell window has loaded, click on **File** and **New File**. This will open a text editor window in which you can write, save, and test your code. Save the blank file as **magic8ball.py** by clicking on **File** and **Save As**.

## Printing replies to the screen

A good way to start your Magic 8 Ball program is to first create a text version of how it works. Let's think about what a Magic 8 Ball does. First, you ask it a question, then you shake the ball, turn it over, and read a reply that it has randomly selected. Therefore, you will need a list of replies and a way of randomly choosing one from the list and displaying that answer on the screen.

To start, you need to import the **random** and **time** libraries. Type the following into your **magic8ball.py** text file:

```
import random
import time
```

Using the **print** function, you can print text to the screen for the person using your program to read.

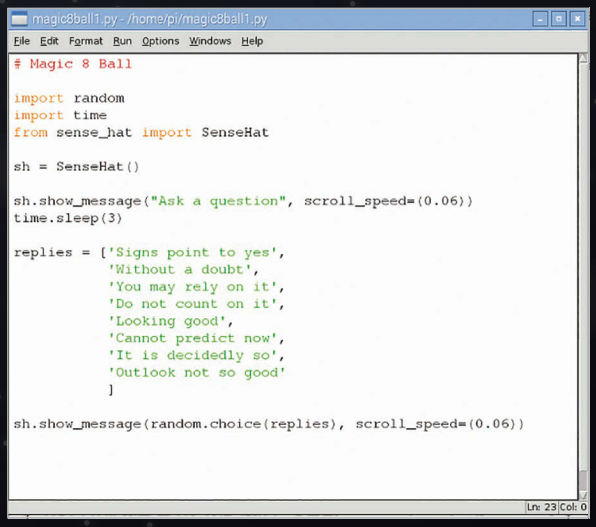



**Fig 2** You can alter the speed of the scrolling text

Type:

```
print("Ask a question")
```

Then there needs to be a pause before the program responds, so that the user can ask a question verbally or mentally. You can use the **time** library to ask the program to sleep for a set amount of time, like this:

```
time.sleep(3)
```

The program will pause for three seconds. You can change the value to make the pause longer or shorter. Now, create a list of replies that the program could give to the question. Lists can be named in much the same way as variables; for example, **number = [1, 2, 3, 4]**. This list called 'number' has four items in it. Your list will contain strings of text that will be displayed on the screen; these strings will be quite long. To create your list, type:

```
replies = ['Signs point to yes', 'Without a doubt', 'You may rely on it']
```

Add as many replies to your list as you like. Make sure that you separate each reply with a comma. You can break up your list onto multiple lines, as shown below, to make it easier to read. However, this is not required for your program to work:

```
replies = ['Signs point to yes',
        'Without a doubt',
        'You may rely on it',
        'Do not count on it',
        'Looking good',
        'Cannot predict now',
        'It is decidedly so',
        'Outlook not so good'
        ]
```

Finally, an instruction is needed to select an item from the list at random and then display it on the screen. You can use the **random** library to do this, by typing:

```
print(random.choice(replies))
```

Save your code by clicking on **File** and **Save**, then run your program to test it works by clicking on **Run** and **Run Module**. Your code should look similar to that shown in **Fig 1**.

## Display text on the Sense HAT

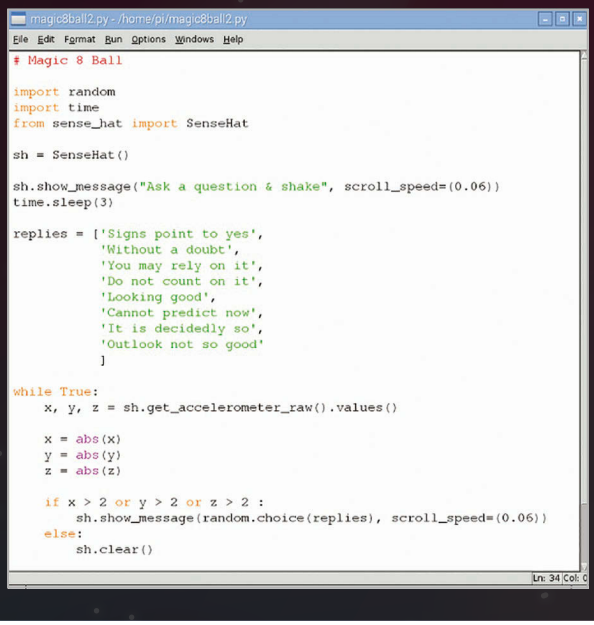
Now that you have text outputting to the Python 3 Shell window, let's change the code so that the text scrolls across the LED matrix on your Sense HAT. To do this, you will need to use the **SenseHat** library and replace the 'print' functions with a 'show message' function. Underneath the 'import modules' section of your code, add the following lines:

```
from sense_hat import SenseHat
sh = SenseHat()
```

Next, replace **print** with **sh.show_message** in your code. There are two places where you will need to do this. To test the code, save your program by pressing **CTRL+S** on your keyboard, then run it by pressing **F5** to check that it works on the Sense HAT.

You may find that the text scrolls rather slowly across the LED matrix on your Sense HAT. To speed the text up, you can add **scroll_speed=(0.06)** to your text strings, as in **Fig 2**.

Normal Magic 8 Balls activate by being shaken up after asking a question. How do you think you could make that happen, using the Sense HAT's motion sensors? That's your next programming challenge!




**Left** A little hint for how to use the Sense HAT's motion sensors so you can shake your Magic 8 Ball

# CODING A MICRO:BIT WITH THE RASPBERRY PI

The BBC micro:bit is an excellent way of teaching some aspects of physical computing, and the great news is that you can program it from your Raspberry Pi

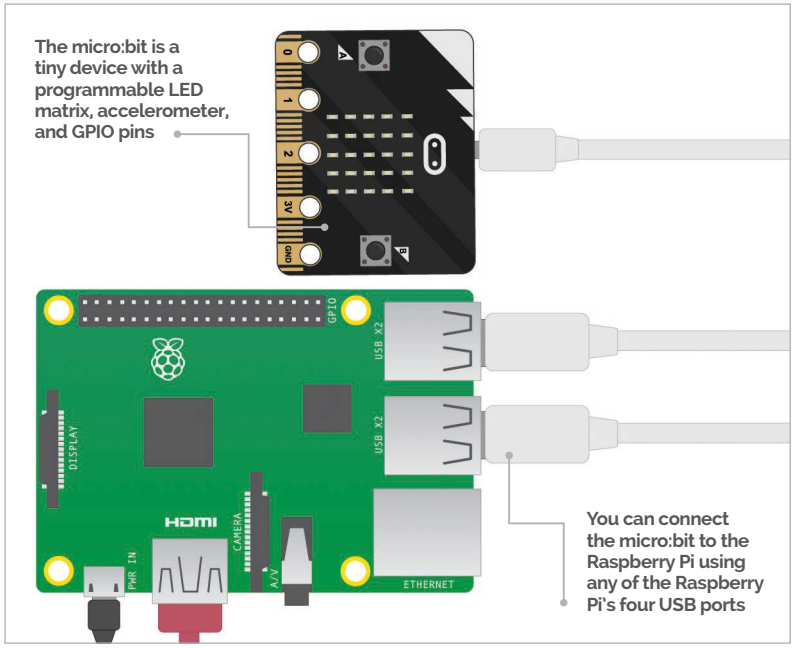
**You'll Need**

- A BBC micro:bit
- A USB A-to-micro-B cable

Thousands of micro:bits were handed out to children all over the country by the BBC in 2016, and hopefully you've managed to keep a few for your department. What better way is there to teach students about physical computing than to combine the power of the Raspberry Pi with the fun and simplicity of the micro:bit?

MicroPython is a small but very fast version of Python 3 that has been specially designed to work on microcontrollers, such as those found on the micro:bit. To start writing MicroPython code on your Raspberry Pi, it's helpful to have an IDE to make things easier for you. Luckily, Mu is an open-source editor designed especially for children, and it can run on your Raspberry Pi. You can get Mu by typing the following into a terminal:

```
sudo apt-get update && sudo apt-get install Mu -y
```




## Using Mu

The Mu software has been designed with young learners in mind. It has a very easy-to-use interface, and most of the menu items should be self-explanatory.

- The **New** button will open a new file. In Mu this is done in a new tab. Have a go at opening a few new files, and then closing them again.
- The **Load** button is for opening existing code that you've written.
- The **Save** button saves any work you have in the visible tab.
- The **Flash** button will push your code onto the micro:bit. You'll learn more about this later on.
- The **Repl** button opens an interactive shell. This is covered in the next section.
- The **Zoom** buttons will alter the size of the text in the window.
- The **Theme** button switches between light and dark themes. You can choose your preference.
- The **Help** button will open the Epiphany web browser and take you to the help pages.
- The **Quit** button will close Mu.

## Using the REPL

The REPL is an interactive shell, running on the micro:bit itself. Here you can write code and instantly see it running on your micro:bit.

- Click on the **Repl** button and wait for the interactive shell at the bottom to open up.


**Above** Mu is an integrated development environment (IDE) for the micro:bit and MicroPython with a very child-friendly interface

- You can click into the REPL and start writing your code straight away. Try the following two lines:

```
from microbit import *
display.scroll('Woop, woop')
```

- Did you see the text scrolling across the LED matrix of the micro:bit? If not, you can type the second line again to scroll the message a second time:

```
display.scroll('Woop, woop')
```

- The REPL is a great place to write single lines of code to test them out, but for larger scripts you'll need to use files.

## Writing and pushing code

- Click on the **Repl** button again to close the REPL.
- In the main window, you can now write a simple script to use the micro:bit's buttons:

```
from microbit import *
while True:
    if button_a.is_pressed():
        display.scroll('A PRESSED')
    if button_b.is_pressed():
        display.scroll('B PRESSED')
```

- Now you can save this script by clicking on the **Save** button. Call the file **what_pressed.py**.
- Next, you need to use the mysterious **Flash** button. Press the button and a dialog box should appear.
- The amber LED on the underside of your micro:bit should also flash. This is because the file is being loaded onto your micro:bit.
- Have a go at pushing the buttons on the micro:bit to see the text scrolling across the LED matrix.

## More features of the micro:bit

Some of the cool features on the micro:bit are the GPIO pins, the accelerometer, and compass.

- If you have access to an LED, a resistor, and some leads, you can connect up the micro:bit to the components as shown in **Fig 1**. If you don't, there's no need to worry: the program you'll write also uses the LED matrix.
- You're going to write some code that will light an external LED, and some of the LEDs on the matrix, when the micro:bit is shaken. Have a read through the code below, and then use Mu to push it to the micro:bit.

```
from microbit import *

shake = False
while True:
    if shake:
        pin0.write_digital(1)
        display.show(Image.SQUARE)
    else:
        pin0.write_digital(0)
        display.clear()
    if accelerometer.was_gesture('shake'):
        shake = not shake
        sleep(500)
```

- Flash the code to your micro:bit and then give it a good shake. Do you see the LEDs changing?

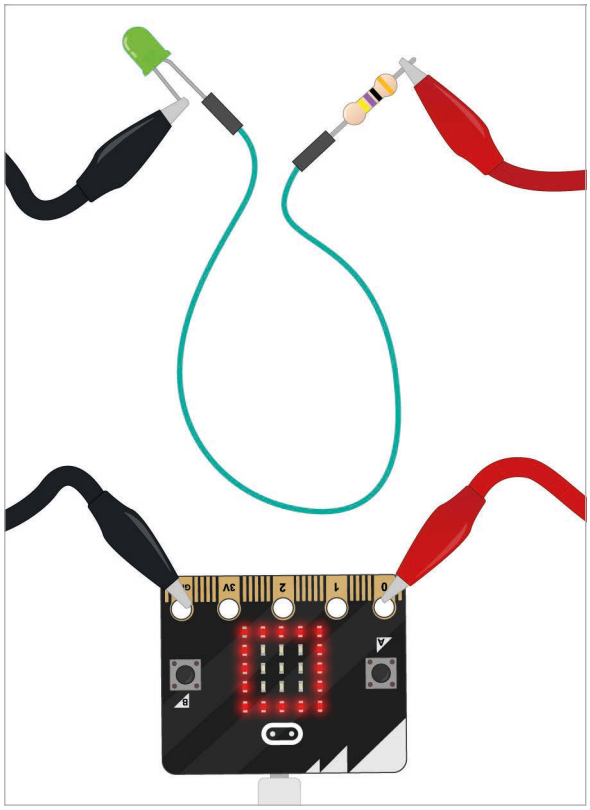


**Fig 1** The GPIO pins in the micro:bit allow students to control physical components such as LEDs

### USING BLUETOOTH

When using Mu and writing MicroPython code on the micro:bit, you can't access the Bluetooth functionality of the micro:bit. If you want to play with Bluetooth, you'll need to use TouchDevelop.

### ALL THE DIALOG BOXES

Every time you flash a new script to the micro:bit, it will disconnect and reconnect to the Raspberry Pi. That's why the dialog box keeps popping up. Don't worry – you can just cancel each time.

# WHAT NOW FOR ASTRO PI?

Now that **Tim Peake**'s mission is over and he's safely back on the ground, what does this mean for our intrepid space-travelling Raspberry Pis?

Back in 2014, Raspberry Pi joined forces with the UK Space Agency, the European Space Agency, and the UK Space Trade Association to run a national competition for schools. This gave students in the UK the chance to devise computer science experiments for British ESA Astronaut Tim Peake to run aboard the International Space Station (ISS).

Two augmented Raspberry Pi computers (called Astro Pis) were launched to the ISS as the platform on which to run these experiments. They each feature a special sensor board, a Raspberry Pi peripheral called the Sense HAT. This can measure the environment inside the ISS, detect how the station is moving through space, and measure the Earth's magnetic field.

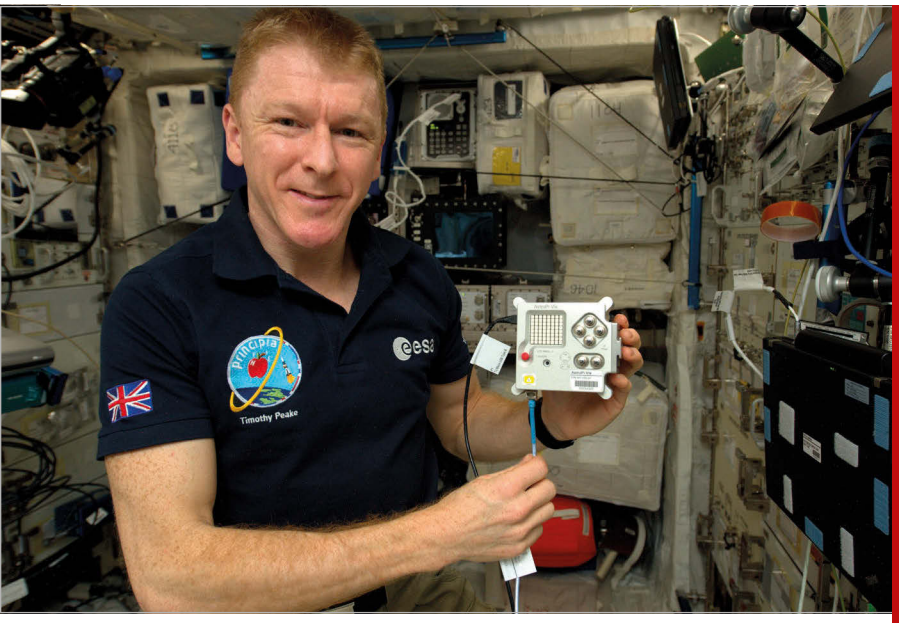
The competition ran from January to July of 2015, and produced seven winning experiments and four highly commended ones that were all launched into space a few days before Tim arrived at the station. Between February and April, these experiments were run on board the ISS under Tim's supervision. Tim recorded a brilliant video explanation of the Astro Pi project which you can watch online here: **vimeo.com/157627149**.

We also ran an extension to this competition in early 2016 involving two music-based challenges, where the students had to either reprogram the Astro Pis to behave like an MP3 player, or write some electronic music for Tim to listen to. Check out Tim's video message to the students that took part: **vimeo.com/167392781**.

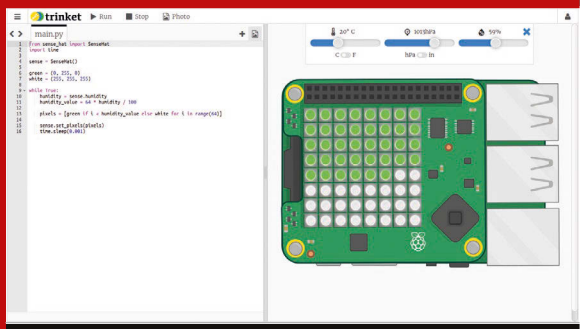
Tim's flight is now over, but the Astro Pi computers are staying on board the ISS until 2022. That's when their hardware clock batteries technically expire, although they'll probably last much longer. At that point we will get them down from space, retire them to a comfortable museum somewhere, and everyone can go visit them!

## MORE BRITS IN SPACE?

Later this year, ministers from ESA member states will meet in Lucerne, Switzerland, to approve the next phase of national space agency projects. The UK will have to decide whether to continue funding the European human spaceflight program at the level that gave us Tim Peake's Principia mission. Without this funding, it's uncertain whether he would fly again or if new British astronauts would be recruited by ESA, so let's all hope they make the right choice.




## SENSE HAT EMULATOR



The Raspberry Pi Foundation recently teamed up with Trinket to give everyone on the planet a free Sense HAT to code on! See the official announcement at **magpi.cc/2bscvQk**. The emulator – which can be found at **trinket.io/sense-hat** – provides a Python code editor on the left and an interactive Sense HAT on the right, which shows the LEDs, joystick, and sensors of the device.

In addition to allowing anyone to experiment with the Sense HAT for free, the new emulator will be central to future Astro Pi competitions that we run in Europe. With the emulator, students can enter these contests regardless of whether or not they own a Sense HAT.

## Ground control to Ed and Izzy?

In the meantime, though, we have worked with ESA to connect the Astro Pi payload to the Joint Station LAN (JSL) on board the ISS. This is significant because it allows ground-to-space remote access to the Astro Pi payload. To accomplish this, a VPN is used along with a Ku band radio link bounced off a number of satellites in orbit. Firstly, ground control open a VPN into the ESA private network and from there they connect to a computer called the MPCC ground node in Munich, Germany. That machine can access the NASA KUIPS service (Ku-Band Internet Protocol Service), and provides connectivity to everything on the ISS Joint Station LAN. All that for a £25, credit card-sized computer!

During most of Tim's flight we didn't have the certification for this, so every time we wanted to download experiment results, Tim would have to shut down the Astro Pi, remove the SD card, take it to a crew support laptop, and manually copy the files across. That requires about 20 to 30 minutes of crew time, which is expensive. With the JSL connection, ESA can have unlimited access, and can remotely deploy new experiments and retrieve results without having to bother the crew. This means we can potentially do much more with the payload, and that means running more code in space written by your students!

## What happens now?

The European Space Agency are planning to roll out Astro Pi educational resources to all ESA member states. That's 22 countries! Each ESA member state has what's known as an ESERO office which provides free resources, support, and information for teachers to enhance STEM subjects using space as a context. We're working with ESA to distribute hundreds of boxed Astro Pi kits through these ESERO offices around Europe. They will have a range of supporting educational materials translated into the local languages, so they could be coming to a school near you! Find your local ESERO office here: **magpi.cc/2cdi0Xh**.

ESA has Astro Pi educational plans for the next two astronauts that are going up: French ESA astronaut Thomas Pesquet, launching in November this year, and veteran Italian astronaut Paolo Nespoli, who launches in May 2017. So be sure to keep an eye out for new Astro Pi competitions in the coming months, as there may be new opportunities for your students to get their code running in space!

ESA operations are also planning to use the Astro Pi payload for a number of utilitarian, non-educational applications. These plans may represent a more long-term use for the Astro Pi payload for when there's no ESA crew member on board the ISS. One plan is to use them as delay-tolerant networking download servers for the JSL. Downloading large amounts of data to the ground usually ties up a laptop but because the Astro Pi operating system is a derivative of Debian Linux, they can easily be configured for this task. They would have a large USB hard disk connected, which would be shared over the JSL using Samba, and anything copied over by the crew would be slowly synchronised to the ground via the NASA ION networking protocol. This means the payload would be contributing to the daily operations of the ISS, and could pave the way for more space-based Raspberry Pi hardware in future.

# SKYCADEMY: MISSION ACCOMPLISHED

High-altitude ballooning sees educators taking their students to space (well, nearly)

Hitching a ride on a Soyuz rocket isn't the only way to get a Raspberry Pi into the stratosphere. High-altitude ballooning involves sending a small electronic device or 'tracker' into the upper atmosphere, using a meteorological balloon. That's what our Skycademy programme is all about: a CPD course designed to show educators how to deliver their own near space project.

James Robinson, who leads the project, explains: "Having run a [HAB] launch myself in school, I'd seen how engaged and inspired my students were, seeing that reaching near space was within their grasp. We want to share that experience with other educators; while the project involves several disciplines, getting to grips with each is not too difficult." Our initial goal has been simple: give some enthusiastic educators the hands-on experience and confidence to do their own supervised launch, and then support them and their students in running their own projects.

## Teacher training

In August 2015, four teams of six educators from around the UK met to begin their training. Over three days the teams learned how to build, configure, and track their high-altitude payload. On the second day, the teams launched their own payloads, following a demo launch by seasoned professional Dave Akerman. Our educators then had the task of tracking their flight by car, in the hope of recovering their payload as it landed. The payloads reached heights of around 35km, capturing images throughout their flight. At the apex, people could view the curvature of the Earth and the blackness of space!

## Skycademy continues

Skycademy wasn't just a three-day event. It's a long-term programme, and our attendees have gone on to lead some great launches. James explains:

"The event marks the beginning of a year-long project where we've funded each of the newly experienced educators to run a launch themselves with their students. It's really important to us that it's not just a bit of fun for a few days. It's a great opportunity to explore the power of project-based learning using a range of disciplines including science, technology, engineering, and computer science."

We've since seen a number of launches in our first year, including a flight for the Women of the World festival bearing a physical hashtag (**youtu.be/9PW3pTf7w8w**), entirely student-led flights, long-term projects, and some great images.

Our second round of flights saw 30 educators participating in the Skycademy 2016 event held at the beginning of August. We're looking forward to seeing what the rest of our educators do over the coming year.

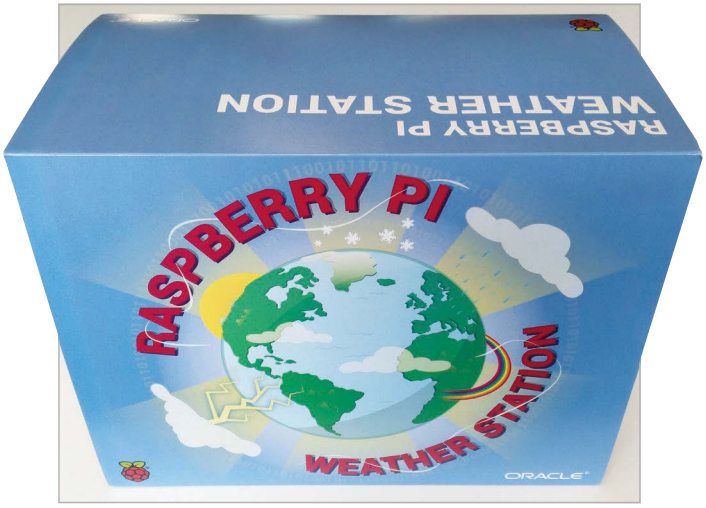
To find out more about Skycademy, visit **rpf.io/skycademy**.

**Left** High above the Earth, our intrepid weather balloons take shots of the world below

## ANATOMY OF A FLIGHT

- A large helium balloon lifts a small (<500g) payload up to 35km, where the balloon bursts.
- After the burst, the payload falls, using a parachute to slow its descent.
- The payload contains a flight computer (Raspberry Pi) connected to a GPS antenna, a radio transmitter, and camera.
- The payload continually collects GPS data and images and transmits them via its radio.
- On the ground, listeners can receive the GPS data and use it to predict the landing site.

# RASPBERRY PI ORACLE WEATHER STATION




We sent 1,000 free Raspberry Pi-powered weather station kits to schools around the world. Here's what happened next...

Stand next to any British person long enough and they will start to talk about the weather. It's the law. But wherever we are in the world, we all live in a big bag of weather. We talk about it, read about it, watch it on TV, and we study it at school. We all have weather in common. So it wasn't a coincidence that two years ago we chose the weather as the topic of our next project. More specifically, with the help of a generous grant from Oracle, we decided to make a Raspberry Pi-powered weather station kit and send it to 1,000 schools around the world.

Earlier this year we shipped the first kits to 150 schools in the UK, closely followed by a further 500 to EU countries and the US. The rest of the kits were shipped in July 2016. The response has been fantastic, and in the past few months we've loved seeing the weather stations appear in the wild, and students and teachers using them in a learning environment.

## Education and collaboration

The weather station is an educational tool aimed at teaching computing, engineering, and data handling. The Raspberry Pi at the heart of the kit sucks up data on wind speed and direction, temperature, humidity, air pressure, and rainfall. This data is logged on the Pi but can also be uploaded to a central Oracle Apex database, where it can be shared. We're looking forward to seeing schools collaborate, wherever they are in the world, as more of the stations come online.

We've written a scheme of work for teachers based around the weather station, which explores the sensors and explains how to interact with them. We're also writing new learning resources that will include material based around the Java language, along with lessons on data handling, analysis, and visualisation.

## What next?

We've been visiting schools to see their weather stations in action, and we've been delighted by the excitement and engagement the kits have generated. Students are really proud to have built a fully functional weather station and brought it to life. We'll be visiting more schools over the 2016-17 academic year, so if you or your school has a weather station and would like us to visit, then please get in touch via our forums or email **weather@raspberrypi.org**.

Finally, we do hope to sell the kits in future. We'd love to see a massive network of weather stations in schools the world over, creating genuinely useful real-world data, and being used in subjects such as computing, science, maths, and geography. Watch this space...

## WHY A KIT?

Because kits are fun and contain learning. By building the hardware yourself, you gain a better understanding of how the sensors and other components work. By installing the software manually you learn all sorts of useful skills, from using the Linux command line to basic programming. The challenge is the whole point! We also wanted students to be able to hack the weather station, perhaps by adding a new sensor, or customising the cases.

# RASPBERRY JAM

Raspberry Jams are community events for everyone to learn about digital making with Raspberry Pi, and to share knowledge with others

Raspberry Jams are events organised by the community to share knowledge, learn new things, and meet other Pi enthusiasts. They're a great way to find out more about the Raspberry Pi and what you can do with it, and to find like-minded people. Raspberry Jams have been running since mid-2012; starting in the UK, they quickly spread around the world. There are over 30 regular Jams in towns and cities all over the UK, and we're aware of Raspberry Jams in Australia, Canada, France, Germany, India, Malaysia, Nigeria, Spain, Taiwan, and the USA.

Anyone can run a Jam, and it can take any format you want it to. Some Jams are small and casually organised; others are large and formally structured. Small Jams tend to focus on people bringing their own Pis along to learn the basics or work on projects, building things with the support of other attendees. Larger Jams tend to involve talks or demonstrations from people who have projects to show. Organised workshops for children can go down really well at Jams, as they give kids a chance to learn something new, which they can continue working on at home.

You don't need to be an expert to attend a Raspberry Jam. In fact, Jams are the ideal environment to get started with Raspberry Pi, and to learn from keen and experienced community members. Whether you're a parent, teacher, or curious hobbyist, you'll be welcomed with open arms. If there's anything specific you want to find out about, just ask the organisers and they'll point you in the right direction.

We feature Jams on the Raspberry Pi website: we have a map and calendar of all the events, so people can find a Jam near them. There are also some case studies and tips for getting started running your own events.

## Raspberry Jam for teachers

If you're a teacher, a Raspberry Jam is the perfect place to learn about what can be done with a Raspberry Pi, and how you can apply it to your own teaching practices. You can meet people who do programming in their jobs, or who build projects at home, and even meet other teachers and educators who use Raspberry Pi and other technologies in the classroom.

## RASPBERRY JAM RECIPE: A LIBRARY

A library could make a perfect venue for a Raspberry Jam: there's probably space for hacking and letting people showcase their projects, and maybe you can find a room for running workshops. If the library has any PCs, maybe you could use the monitors, mice, keyboards, and even the wired or wireless internet connection if they have one. Make sure you clear this with the library staff first and help them understand what you're doing with their equipment. You should also ensure the PCs are set back up when you're done.

Many local councils are keen for residents to make use of their libraries, and welcome more people to see what's happening there. They are likely to have a method of spreading the word about your event, and you can boost this yourself by sending flyers to local schools. Libraries are great for hyperlocal events as they're often very easy to access for people living nearby.

Jam workshops are a great way to give people the chance to learn in a safe and helpful environment

Photo: Cat Lamin


The Northern Ireland Raspberry Jam focuses on engaging kids in digital making

Photo: Andrew Mulholland

As a teacher, Jams are a great opportunity to observe. See how the workshops are run, see how kids react to people's projects, see what's possible with Raspberry Pi, think about what would inspire your own kids, and find interesting ways of getting through to them based on their own interests.

Many teachers of computing and other STEM subjects attend Jams as they're a great way to network with other leaders in education, and work towards your continuing professional development. Keep up-to-speed on what's going on in the digital making world, and take the opportunity to live on the cutting edge of computing education, using your special skills as a teacher to find the best ways to deliver the curriculum to your own classes and prepare your children for the ever-changing digital world.

> "You can use the ideas from the Jam to fuel creativity of your classroom projects"

If you teach a creative subject, you'll find plenty of great projects on show at a Jam. You can use these ideas to fuel the creativity of your classroom projects, and to encourage kids by showing them what's possible.

## Start your own Jam

Anyone is allowed to start their own Jam, and there are no rules. All you need to run a Jam is a venue, and a date and time. When you've pinned those things down, you can invite people to your Jam! You can hold a Jam anywhere. Some of the bigger Jams are held in huge venues like universities, while others are just a few people sitting around a table. We've seen Jams held in lecture theatres, conference centres, schools, cafes, hackspaces, garages, living rooms, libraries, and even pubs. Most Jams take place at the weekend, but they don't have to: an after-school club or evening gathering could work, too. If you want children to attend your Jam, you'll need to find a good time that works for everyone.

It's helpful to use an online service like Eventbrite to manage ticketing. This allows you to manage registrations, so you know how many people to expect on the day (and to limit numbers to the capacity of your venue). You also get a webpage and the means to promote your event online.

## RASPBERRY JAM RECIPE: A SCHOOL OR UNIVERSITY

Like libraries, schools and universities can be ideal venues for a Jam. They usually have PC suites which you can use for workshops, and classrooms full of tables of chairs can be a great space to let people hack on their own projects. Just be sure to provide power to the tables.

If you want to put on a track of talks, then there's no better space than a university lecture theatre. Try to invite people to speak, and be welcoming to anyone who wants to present (especially keen young people).

Depending on the activities you're putting on, you may need to provide equipment. If you're running workshops for participants, you'll need to provide Raspberry Pi workstations – including the Pi, mouse, keyboard, and monitor – along with any additional hardware for the activity. Your venue may be able to help with this.

If you want to start a small Jam in your area, don't worry about putting on any talks or workshops at first. Just arrange a date and invite people to come. You can use the first event to work out what people want, and begin to form a plan. A Jam is not a one-person show: it takes many people to make the event a success. Make the Jam about the attendees and what they want from it.

To help to get the word out about your Jam, be sure to submit your event to the Raspberry Pi website so that it will appear on our map and calendar: **raspberrypi.org/jam**

# PICADEMY: TEACHING THE TEACHERS

Picademy is the Raspberry Pi Foundation's FREE teacher training initiative, which aims to give educators the skills they need to begin teaching computing with confidence

**MANY USEFUL IDEAS**

"Picademy's Continuing Professional Development was excellent and it gave me many useful ideas to go back to school with, particularly the use of Minecraft Pi. It was a great experience to spend time with like-minded people who could see how the Pi can be of benefit in supporting the computing curriculum. I use the Pi in my school, and I teach workshops on Minecraft Pi and Python programming."

**Sarah Zaman**

**Primary Computing Teacher and Raspberry Pi Certified Educator**

Getting started with the computing curriculum can be daunting. There's a host of hardware and programming languages, and a baffling array of technical terminology. However, the barrier to getting started rarely has much to do with technical proficiency or programming knowledge. Confidence is typically the main issue, and early setbacks can have a huge impact on this.


Picademy aims to give teachers the skills and knowledge they need to get creative with computing, no matter what their level of experience. It's a two-day course that allows educators to experience some of what can be achieved with a little help and a lot of imagination. We are regularly blown away by what educators create.

## What's involved?

Day one involves running through a series of workshops introducing our teachers to physical computing, using the General-Purpose Input/Output (GPIO) pins on the Raspberry Pi to control electronic components like LEDs and buttons. All programming is done either in Scratch or Python. There are also sessions based around using add-on boards like the Camera Module or Sense HAT, coding music with Sonic Pi, and terraforming the world of Minecraft.

On day two, the teachers have free rein to come up with their own project ideas. Previous projects have included killer robots, Christmas jumpers with twinkling LEDs, bespoke games in Minecraft, and more. They reinforce the skills they learned on day one, and also practise decomposing problems, testing, debugging, and building resilience. Our attendees leave as Raspberry Pi Certified Educators, ready to share their newfound enthusiasm with other educators and children.

Since Picademy's launch in 2014, courses have taken place at locations around the UK, and even in the USA. Teachers can now apply for the next Picademy, to be hosted at the Glasgow Mitchell Library, comprising four two-day courses in October and November. For details, and to find out about future Picademy events, visit **rpf.io/train**.

# READ US **ANYWHERE**


**FREE: DOWNLOAD ALL 30 ORIGINAL ISSUES!**

The MagPi Magazine

## Available now

for smartphones & tablets

Available on the App Store

GET IT ON Google Play

## *Subscribe from*

**£2.29** or **£26.99**

rolling subscription — full year subscription

### Download it today – it's free!

- Get all 30 legacy issues free
- Instant downloads every month
- Fast rendering performance
- Live links & interactivity

# SUBSCRIBE **TODAY**

# & SAVE UP TO 40%

Subscribe to the official Raspberry Pi mag at our exclusive educator's rate

## Subscription benefits

- Get it first (before stores)
- Free delivery to your door
- Never miss a single issue

## Pricing

**Quarterly Direct Debit:**
**£10.50**

**Six issues:**
**£23**

**12 issues:**
**£45**

NEW TEACHER OFFER!


## How to subscribe:

- Call on **01202 586848** and ask us about our **UK** educator rates


**Find us on your digital device by searching for 'The MagPi'**

Available on the App Store

GET IT ON Google Play