WHAT'S INSIDE

AMAZING MICRO:BIT MAKES, NEWS, REVIEWS AND MORE!

micro:mag

Issue 5
August 2019

Links to useful **GUIDES**

10 MICROPYTHON **PROJECTS**

Explore MicroPython

Explore the essentials of MicroPython and the BBC micro:bit.

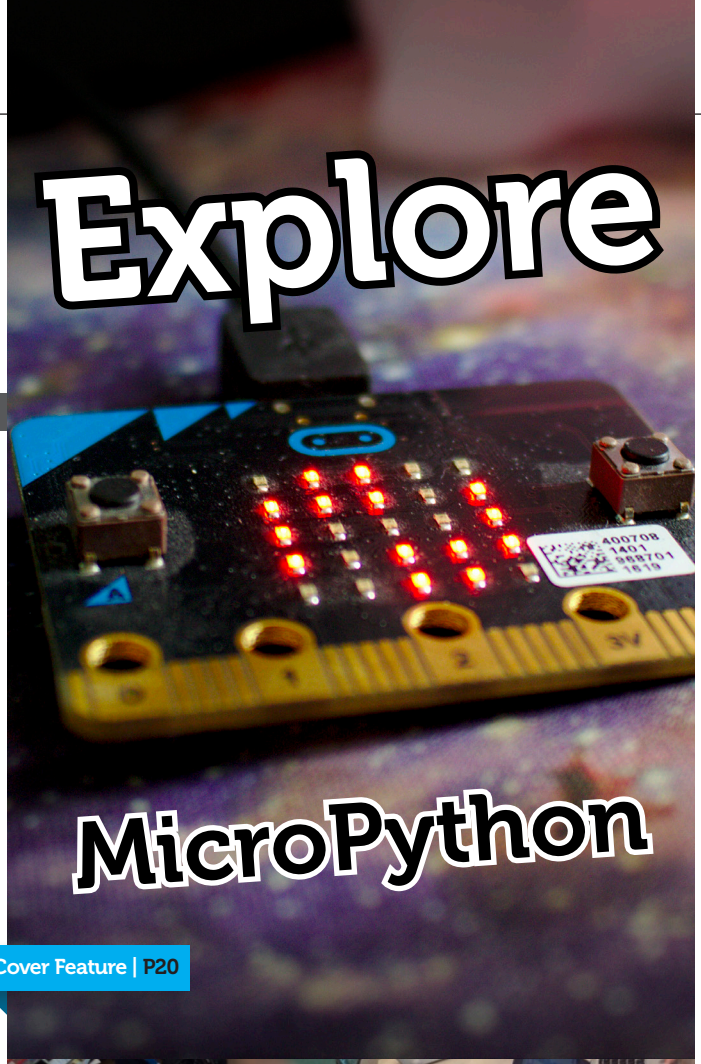
MicroPython **MAKES**

micro:hit **PYTHON SPECIAL**

THE UNOFFICIAL MICRO:BIT COMMUNITY MAGAZINE

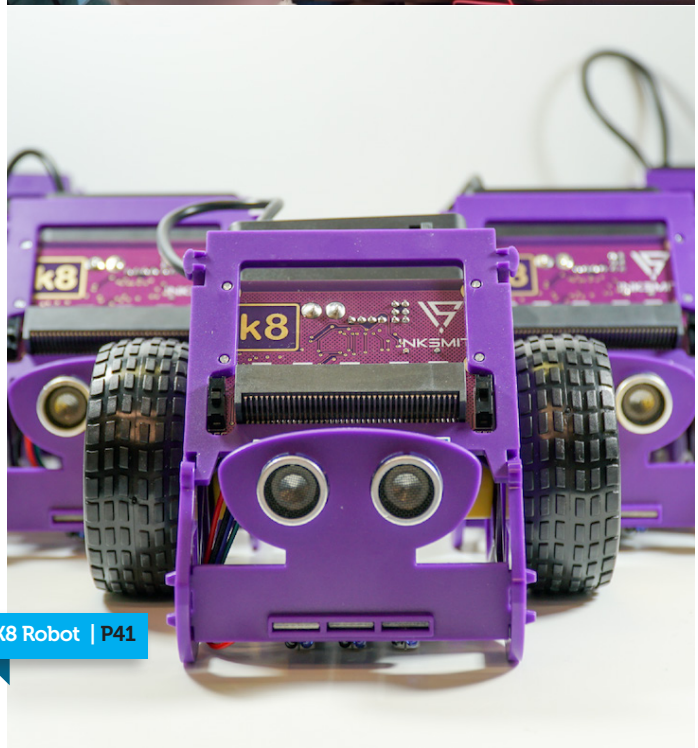
micromag.cc

Contents



Cover Feature | P20

micro:bit In Schools| P14



K8 Robot | P41

Hello World

Joshua Lowe
Senior Editor

It's a pleasure to be able to welcome you to the all new micro:mag. Our team has been at work for months to bring you a new design and layout for the magazine. This is a step to making micro:mag a better magazine and it's only possible because of our amazing readers. Special thanks to LGFL who kindly sponsored our move to Adobe Creative Cloud which gave us access to Adobe InDesign (the industry standard for magazine creation). Another special thanks must go to Chris Burkill who designed the magazine for us.

We have another amazing Issue in store for you. Issue 5's cover feature is "MicroPython on the micro:bit". We've collated some great MicroPython powered articles for you as well as a special cover feature that has 10 MicroPython examples to get you started. We've also got a special micro:hit for you that shows you how to power a string of LEDs with MicroPython & the micro:bit.

Alongside that, we have tons of other great articles in Issue 5 including news about the official micro:bit conference, the launch of do your :bit, a look at the world's first WiFi add-on for the micro:bit as well as a tutorial on how to build a timing gate, just to name a few.

On behalf of the whole micro:mag Team, enjoy your printed copy of Issue 5!

Meet the team

Kerry Kidd
Editor In Chief

Kerry is a freelance programmer/educator who enjoys writing tutorials and tinkering with the micro:bit

Follow me:
@RaspiKidd
raspikidd.com

Joshua Lowe
Senior Editor

Josh is a young coder & creator of the Edublocks tool for micro:bit He has delivered lots of workshops & talks around the world.

Follow me:
@all_about_code
edublocks.org

James Bastone
Copy Editor

James runs a local Makerspace and joined the team because he genuinely enjoys proofreading articles.

Follow me:
@NJBastone

Contributors

- Fair Chance Learning
- Robin McKean
- Martin Woolley
- Sam Watson
- Phil Hall
- Les Pounder
- Q Misell
- Adriano Parracciani
- Robert Wiltshire
- Christopher Roscoe
- Su Adams
- Katie Henry
- Dexter Industries
- Chris Burkill
- Kitronik
- LGFL

A brand new challenge has been announced by the Micro:bit Educational Foundation for the Global Goals.

do your :bit

ABOVE: do your :bit supports the UN Global Goals

Following on from the success of the 2018 Global Challenge, the Micro:bit Educational Foundation have announced their brand new challenge, "do your :bit" in partnership with Arm & World's Largest Lesson.

The new challenge focuses on Global Goals 14,"Life below water" and 15 "Life on Land". The challenge is built to test the imagination of children and teens to build projects with the micro:bit to help solve these 2 goals. This could be a device that will reduce the amount of plastic used in schools or something that encourages people to recycle. The sky's the limit! The challenge is open to young people ages 8-14 anywhere in the world. To enter, entrants will need to write a written entry describing the project and why they have created it, a prototype of the project in order to show how the idea will work, a code file for the micro:bit written in any language of your choice and also some extra videos and photos for the judges. Winners will win lots of micro:bit kit and also a trip to a special event in London! One cool thing about this new challenge is that the micro:bit foundation have released some resources and lesson plans plus some inspiration to get you started, so this challenge is great to do as a club in school. Entries open 16th September and close on 28th February.

For more information see: microbit.org/do-your-bit/

"This challenge will focus on Global Goals 14 and 15. We want you to come up with innovative ways to use your micro:bit to help protect life on land and below water."

MakeCode Update

The MakeCode editor from Microsoft is the most popular way to program the micro:bit and is packed full of features that make it easy to use. Recently the MakeCode team have released a new update with some awesome new features. Let's see what's in store.

Functions with Parameters is ine of the exciting new features in this update. According to Microsoft, this is one of the most requested features as of lately. Now when you make a function, you have the option to add Text, Boolean or Number type parameters.

Green Screen is a cool new feature that will allow you to overlay your webcam onto the MakeCode workspace. This allows you to demonstrate code and hardware at the same time!

The new My Projects view – Also one of the top requested features gives you the ability to see all your projects, and then open, delete, and duplicate the projects.

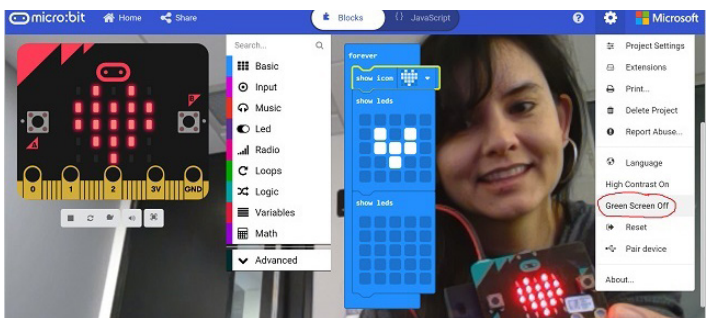
Tutorials now save your progress so you can jump back into where you left off.

The Servo Blocks have been updated, they now help you calibrate your servos and give you better control.

Last but not least, the new Convert to text block allows you to convert different data types into text!

We love seeing these new features being added to Make-Code as it opens up a range of brand new possibilities. You can see the update at

makecode.micro.bit.org



Learning with micro:bits and robotics in classrooms across Canada

BY Fair Chance Learning

Fair Chance Learning and InkSmith have teamed up to bring the micro:bit-powered robot, k8, and curriculum connected professional learning to Canadian classrooms.

About the Author

fair chance learning
what every student deserves

FCL is a Canadian professional learning services provider for educators. We inspire ideas, empower educators, and transform learning in the classroom to help give every student a fair chance to reach their learning potential. Learn more at www.fairchancelearning.com

Twitter: @FCLedu

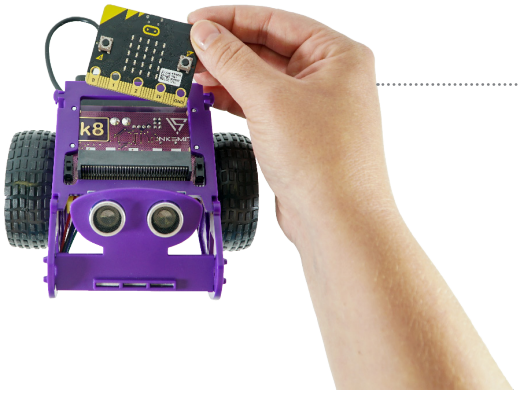
Facebook: @FCLedu

Fair Chance Learning and InkSmith have partnered to expand student achievement in coding and robotics across Canada by delivering more of the micro:bit-powered robot, k8 Modular Robotics Kit, to classrooms and providing enriched professional development and curriculum resources for educators. Fair Chance Learning will be providing educators and students across Canada with curriculum-connected workshops and training to equip students with future-ready digital skills and robotics knowledge.

To the right:

Fair Chance Learning team members Titus Ferguson and Barb Seaton team up with InkSmith's Jeremy Hedges, Doug Braden and Andrew Brumwell at CONNECT Conference in Niagara Falls, Ontario, Canada.

Created by Education Technology company, InkSmith, the k8 Modular Robotics kit was created with the goal of providing a fun and interactive way for students to learn and explore the world of coding and robotics. The k8 robot is a modular kit that is fun and easy to build. Combined with the capabilities



To the left:

The K8 robot uses the micro:bit to enable students to learn beyond their traditional boundaries.

of the micro:bit, k8 can be programmed to do all sorts of activities including line following, sonar sensors, remote control and more!

"This Fair Chance Learning and InkSmith partnership is another reason to look to Canada for models in building fundamental digital skills - including coding - in today's students," says Hal Speed, Chief of Global Engagement at The Micro:bit Foundation. "They're harnessing of the power of the micro:bit to enhance student learning with robotics".

InkSmith and Fair Chance Learning are excited about the opportunity to expand the possibilities of the micro:bit through the k8 Modular Robotics kit. The k8 robot will give both students and educators a whole new way of learning and coding with their micro:bit. Equipped with 2 sonar sensors, 3 infrared sensors, and the ability to radio control with another micro:bit, the k8 robot provides a tangible way for students to interact with their code in the real world.

This partnership is a collaboration of two like-minded organizations that share a desire to support educators in order to prepare students with future-ready skills. Fair Chance Learning will enhance InkSmith's own professional development and curriculum offerings by collaborating as the exclusive training partner for the k8 Modular Robotics kit.

"We have been eager to include robotics-driven learning into our service offerings, but until now we hadn't found a robot or robotics partner that fit our criteria for use in education," says Dustin Jez, Co-Founder of Fair Chance Learning, "In keeping with the spirit of the micro:bit, the InkSmith k8 Modular Robotics kit breaks down barriers to integrating coding and robotics into the classroom with its affordability and ease of while enhancing learning for all students."

"Our team is really excited to partner and work closely with Fair Chance Learning in the years to come," said Jeremy Hedges, Founder and President of InkSmith. "It's always been very clear to me that we share a common vision for how we want to see education grow to support kids here in Canada and around the world".

By combining robotics with the micro:bit, the k8 breaks down barriers for introducing robotics in their classroom. It's affordability and ease-of-use allows educators and students to focus on learning the curriculum with technology – not learning the technology.

Where To Buy

You can buy the K8 from Fair Chance Learning for 79.99 Canadian Dollars (£47.58)

Click the link to buy

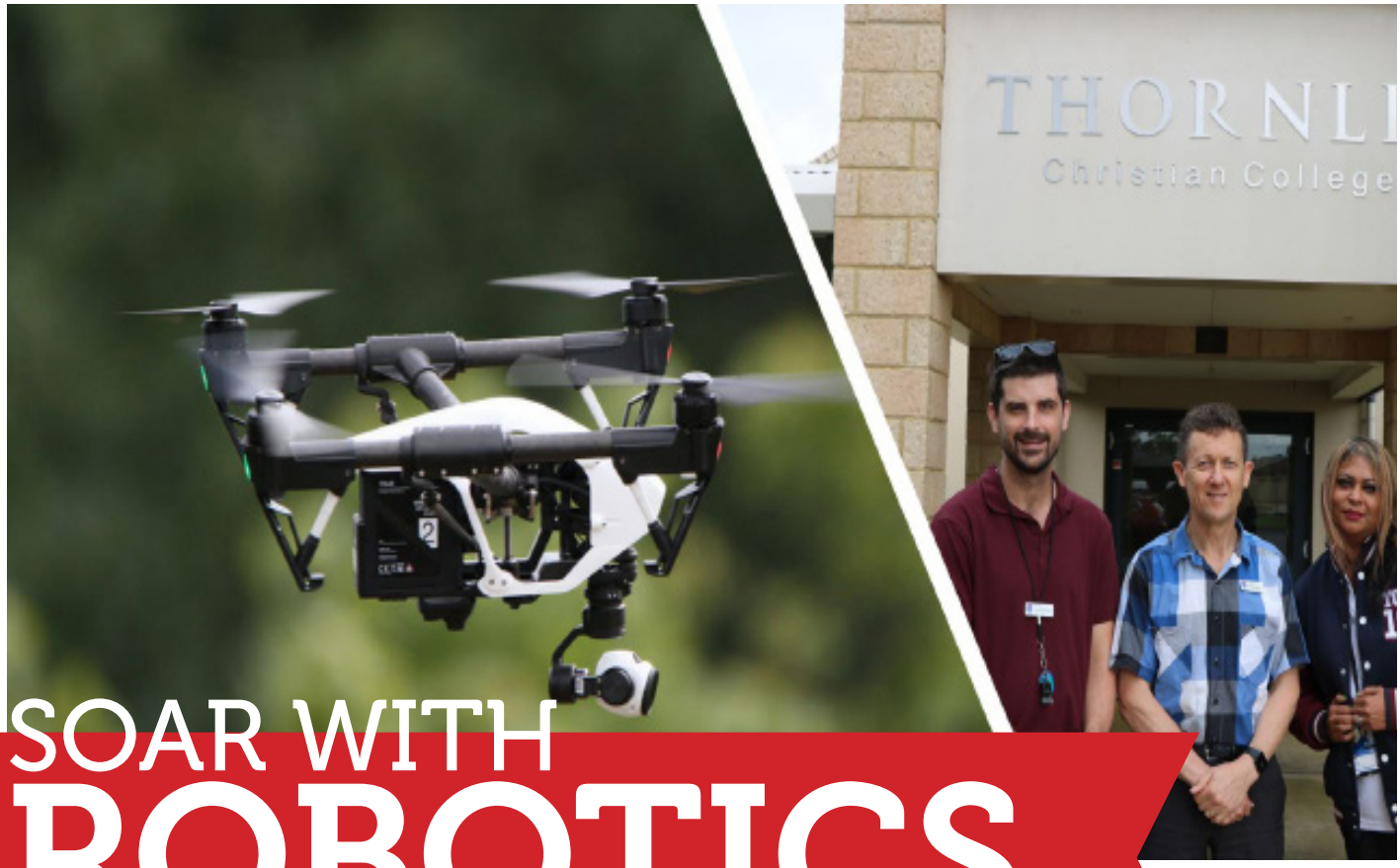
https://www.fairchancelearning.com/product-page/k8-robotics-kit

Also See

Our review of the K8 By Joshua Lowe on Page 41

Above:

When properly integrated, EdTech tools like the micro:bit and K8 can increase students' learning potential and achievement

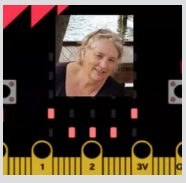


SOAR WITH ROBOTICS

Working with trees and bees using Drones, Apps, and micro:bits.

By **Robin McKean**.

About the Author



Digital Technologies Project Officer with at the University of Adelaide; Millennium Kids Inc Council - working shoulder to shoulder and face to face with teachers and students to integrate digital technologies for a sustainable future.

Twitter: @rmckean

Drone Pilot Training, with simulated helicopter rescues and environmental flight missions, has 'taken off' with micro:bits, Game Controllers and Inventor's Kits from the National Lending Library of the Computer Science Education Research Group (CSER) at the University of Adelaide. Students are SOARing with Robotics as they undertake sustainability assignments, and as young inventors - design, build and test working models of quadcopters and submersibles. Trainees are earning their drone pilot wings as they work through squadron ranks from Safety and Junior Flight Officer to First Officer, Captain and the highest-ranking Top Gun – Engineer.

Squadron Leaders from TCC- Thornlie Christian College, have joined forces with the CSER Digital Technologies Project Officer from the University of Adelaide to design and facilitate a hands-on FUN middle years extracurricular program for students interested in using digital technologies to code, create and connect with the real world. The TCC SOAR with Robotics program has been inspired by the range of unique digital solutions generated by children around the world who participated in the micro:bit Global Challenge. It emulates similar actions undertaken by our local Millennium Kids (MK Inc) – young problem solvers, innovators and change agents. MK Kids are SOARing with their #1000actionsfortheplanet, Green Lab Projects and the MK Kids On Country Project. When On Country, Kids from Coolgardie, with their traditional elders and MK volunteers, travel to remote areas in the Great Western Woodland to learn about their traditional language and culture while monitoring the wellbeing of the bushland using both traditional and newer drone and micro:bit technologies.

CSER Lending Library Kits contain a selection of accessories, including Mini Buggy and Kitronik

andServo:lite

Inventor's kits, bulldozer, Bumper and Servo:lite modules, and 12 micro:bit GO modules that have the capacity to provide a multitude of programmable digital solutions. When working in combination with multi-functional and versatile Unmanned Aerial Vehicles (UAVs), Autonomous Underwater Vehicles (AUVs), First Person View (FPV) and Remotely Piloted Vehicle (RPV) technologies, the drone pilot trainees have access to the CSER Project Officer, training opportunities and Lending Library kit in order to SOAR with passion projects to suit their needs.

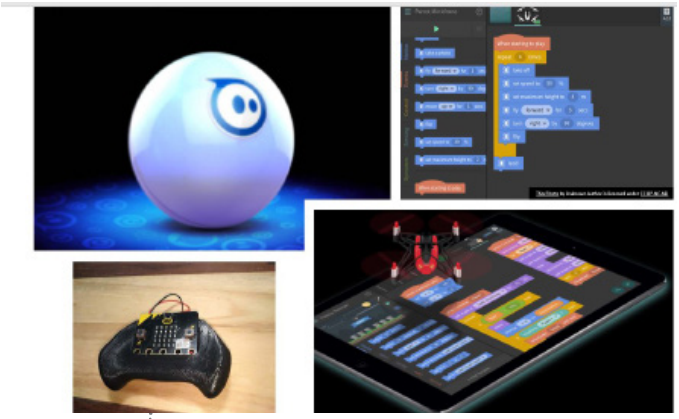
All program participants are encouraged to work through their pilot training and level up in order to SOAR to new S- safety, O- operations, A- action learning and R- research heights.

Above:
Millennium Kids Inc project volunteers and Ngadu Kids

S – Safety Officer Training and Can I fly there? drone safety is the mandatory level for entrance to the SOAR program and must be completed before Junior Flight Officer training can begin. All Safety Officers work extensively within the Civil Aviation Safety Authority drone guidelines to develop a personal checklist of safety protocols and procedures for future operations.

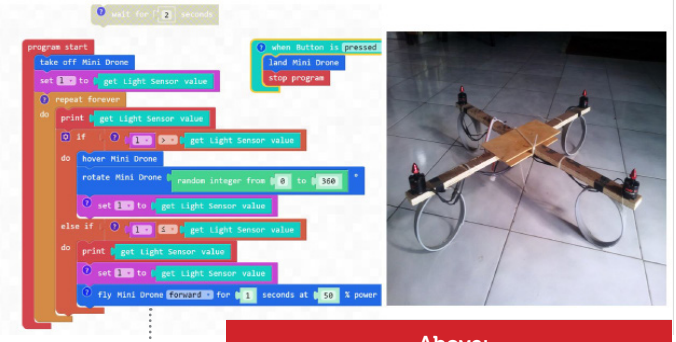
O – Operational Levels begin with Junior Flight Officers who are required to program the Kitronik GAME controller for the micro:bit and use these devices to operate simulated helicopter rescues. A JFO must use their micro:bit and Controller to demonstrate mastery of piloting skills for simulated rescues, evacuations and firefighting assignments. To become a fully-fledged Flight Officer, pilots in training will use the Tickle app to program a micro:bit for rigorous ground and water manoeuvres using devices including Sphero, Dash, Mbot and any available Lego systems such as WeDo.

A - Action Learning is demonstrated by trainee First Officers, as they complete the Tickle App AirShow as an Hour of Code and choreograph and execute drone precision aerial biomimicry. First Officers are successfully able to control a drone with a micro:bit so that it will react to the environment the way an animal would i.e. a moth or bat reacting to light and heat.



Above:
Flight Officers use the Tickle app to program a micro:bit and successfully.

R– Research is carried out in order to become a Captain. Trainees can Fly High, under the command of a Squadron Leader or Top Gun. They capture and chart field data using accelerometer, magnetometer and temperature and data logging technologies from micro:bit's internal sensors over Bluetooth. They level up by using micro:bits to choreograph a bee waggle dance and negotiate the design and completion of Global Learning and Observations to Benefit the Environment (GLOBE) activities or other project-based environmental impact studies such as the use of micro:bits and drone technology to monitor and calculate the carbon offset of urban or woodland canopy



Above:
Drone biomimicry uses micro:bit sensors to demonstrate a moth's reaction.

Top Guns then continue to 'Fly High' and connect with many disciplines including science, technology, engineering and mathematics (STEM), as well as film, media, and journalism. Top Guns are aircraft engineers with a license to fly solo, design, build and equip quadcopters, rovers or submersible FPV vehicles for completion of environmental missions for this planet or beyond.

Thermal Pollution Monitoring with Bitty Blue

BY Martin Woolley

How technology could help address environmental issues in Saudi Arabia

Above:
Bitty Blue from Bitty Software

Thermal pollution is an environmental issue that we don't hear a lot about. It involves the artificially induced, rapid changing of the ambient temperature of the sea, lakes or rivers and it's a serious problem which can have a devastating effect on fish and other organisms as they succumb to thermal shock.

The most common source of thermal pollution is industry. Power plants and factories which dump heated wastewater directly into the sea can cause a rapid increase in water temperature and the death of aquatic life.

I didn't know anything about thermal pollution until a few months ago. In fact, I'd never heard of it and I'd like to share with you how it is that I came to learn about the subject, all thanks to two schoolgirls, Nouf and Mawwadah who live in Saudi Arabia.

Nouf and Mawwadah contacted me by Twitter, asking for some guidance on how to approach a school project involving micro:bit. Their teacher had asked them

About the Author

Martin loves to code and make magic things happen with Bluetooth wireless communications and was lucky enough to be involved in creating the micro:bit as the team's Bluetooth specialist.

Twitter:

@bittysoftware

Find out more...

You can learn more about Bitty software and Bitty Blue on Martin's website:

bittysoftware.com

to think about thermal pollution and how technology might help prevent it, and to build a working proof of concept.

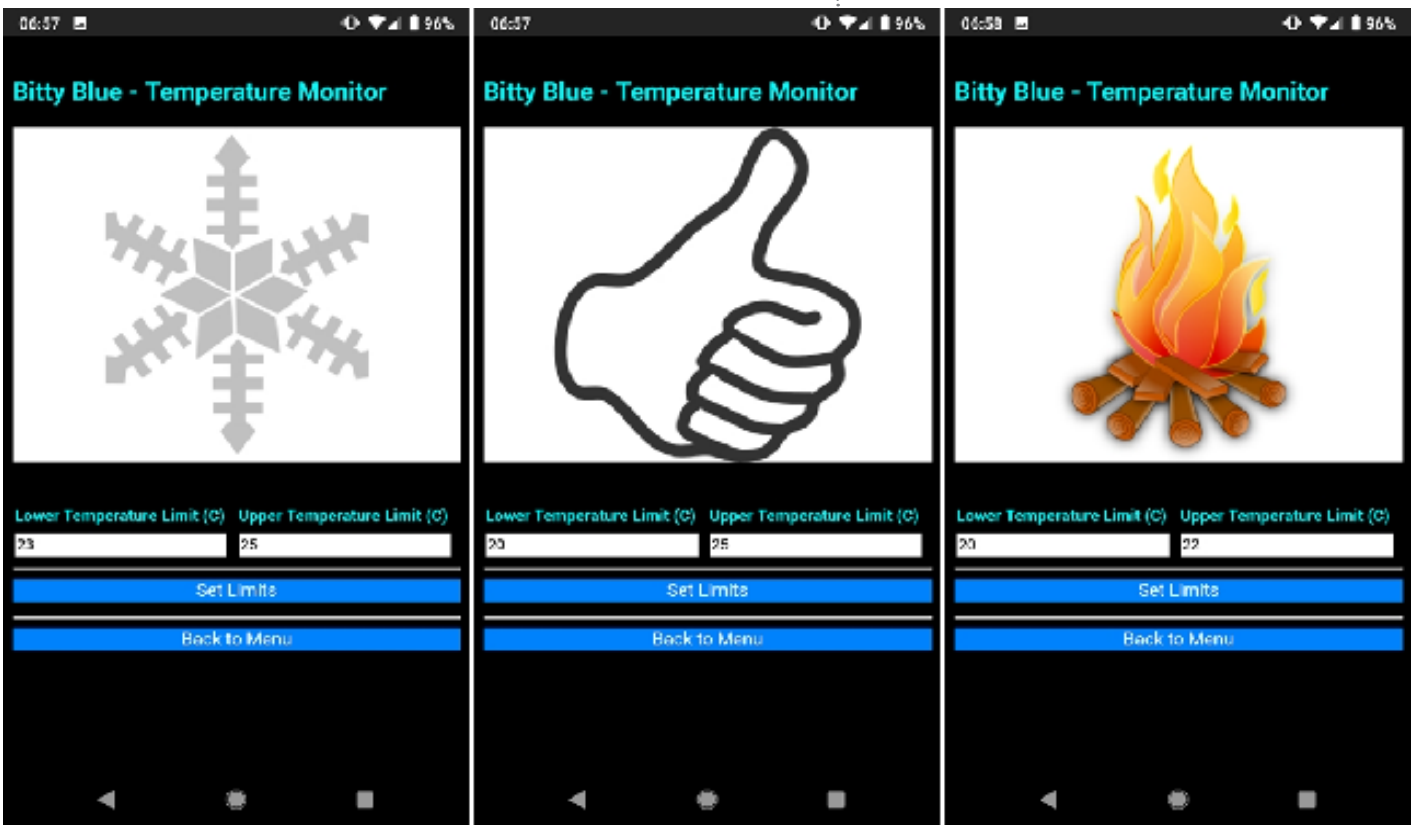
The project idea goes like this; imagine the wastewater pipes from every factory by the sea were fitted with special devices which could monitor their temperature. If the temperature got too hot or too cold, risking thermal shock, the device could inform a computer system wirelessly, raising the alarm that the factory was breaking the law and causing thermal pollution.

Nouf and Mawwadah discussed ideas for the project with me and I explained how Bluetooth, one of the most commonly used wireless communication technologies and which these days has a range of over a kilometre, could be an important ingredient in the solution. We talked about the basic logic required to periodically measure the temperature, check it and report any breaching of allowed temperature limits. We talked about the programming concept of an "event", how micro:bit "events" can be used to trigger Bluetooth communication with an application on another device and what the transmitted messages would need to contain.
With a little guidance, Nouf and Mawaddah went on to use MakeCode to create a micro:bit project which did exactly what it needed to. It looped, measuring the temperature every second or so and then sent a Bluetooth message containing a number which meant "OK" (0), "Too Cold" (1) or "Too Hot" (2).

But what was their micro:bit sending these messages to?

Bitty Blue is free of charge mobile application from Bitty Software. It has a variety of features, all centred around micro:bit and its Bluetooth capabilities. One of them, it so happens is a temperature monitoring application, precisely what Nouf and Mawwadah needed. Depending on the content of the Bluetooth messages it receives from a micro:bit, it displays a big, clear, graphical indication of whether the temperature being monitored is too hot, too cold or within acceptable limits.

Below:
Bitty Blue temperature monitoring



The first ever official micro:bit conference, micro:bit live is set to take place in Manchester this October!

micro:bit LIVE

ABOVE: micro:bit stand at BETT 2019.

Micro:bit LIVE is a two-day conference all about the micro:bit, where you will be able to discover some of the new teaching and learning resources first hand.

Learn about what is new within software and hardware as well as learn how the micro:bit is being used around the world and broadening the participation in computing for all children.

What is going on over the two days?

Over the two days, you can participate in workshops, talks, panel discussions, poster sessions as well as network until your heart is content with fellow micro:bit enthusiasts and micro:bit partners and don't forget to come and find the micro:mag team.

The event is being held at MediaCIty in Manchester with tickets being £22 each for the 2 days and lunch is included. The foundation are giving all the ticket money to local Salford schools so they can have access to micro:bits. Tickets are being released in waves and they're selling fast so don't miss your opportunity to attend the first (and hopefully annual) micro:bit conference.

micro:bit LIVE will be taking place at the BBC MediaCityUK, Greater Manchester, England on October 4-5.

Tickets: go.micromag.cc/mblive

micro:bit Live 2019

OVER **40** micro:bit ACCESSORIES by

Kitronik

Kitronik.co.uk/microbit

MICRO:BIT IN SCHOOLS

What I did when I taught Year 2 at my mum's school some micro:bit coding using the 4tronix Bit:Commander. By **Sam Watson**.

About the Author

Sam loves to code in his spare time. Sam originally started coding using a micro:bit and loves showing others how to get started too.

Featured Products

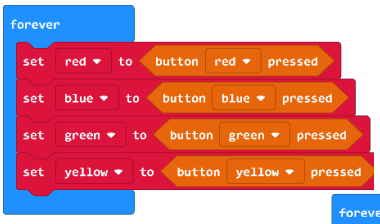
The bit:commander and bit:bot can both be purchased at:

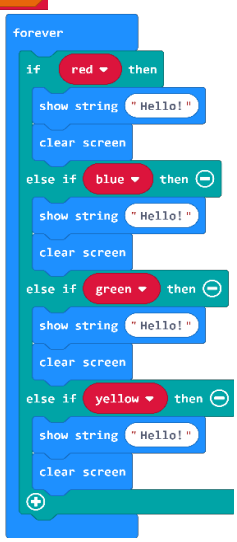
4tronix.co.uk

As I had a teacher training day at my school but my mum didn't I decided to go with my mum to work and teach some Year 2s coding with the micro:bit alongside the 4tronix Bit:Commander.

I had been asking my Mum for ages if i could go into her school with to do some coding with her class. Finally, in June I had a teacher training day but her school didn't, so I went along with her for the day! When I arrived at my Mum's school I got the Chromebooks out and set up a table with them and the micro:bits. First I went to the micro:bit MakeCode Editor on the 8 Chromebooks. I then installed the Bit:Commander package for this editor.

Next, I used the following code as a template for the children to customise it to how they liked. I wrote the code in the MakeCode micro:bit editor.

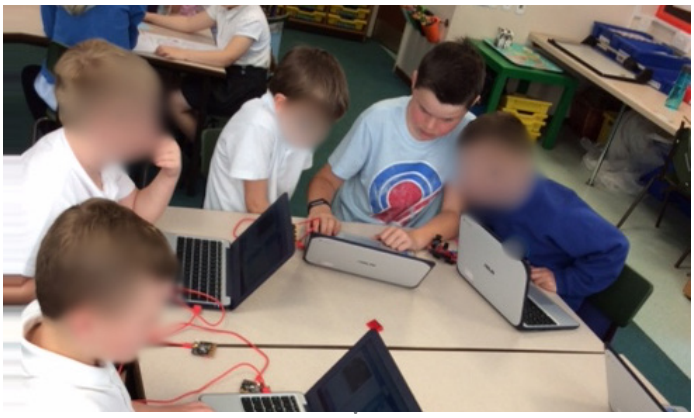




The Code

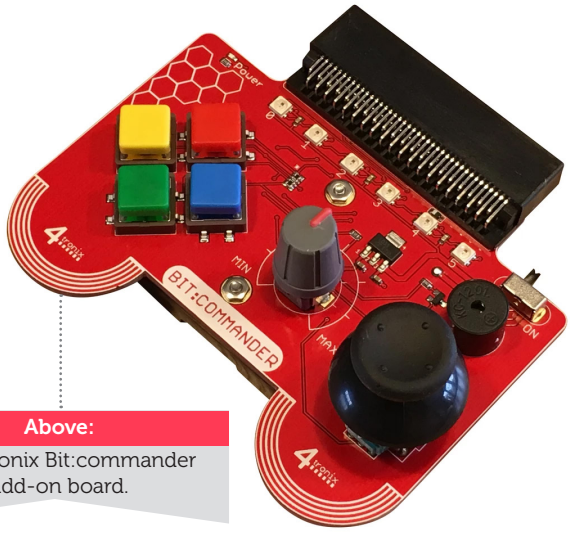
This code was written using the MakeCode micro:bit editor with the 4Tronix Bit:Commander extension.

My Mum chose 8 children at a time to come and work with me using the code on the last page, the children could edit the 'Show LEDs' and 'Show String' blocks to how they wished. Each child chose their own pattern or message.



Above:
Group of children all using the micro:bit

Next, I downloaded the code onto the children's individual micro:bit and then one by one I inserted the micro:bit into the Bit:Commander to let the children run and test their code on the Bit:Commander. Some children used characters from games such as Super Smash Bros. or Super Mario Bros. in their code. Others used the 'show string' blocks to state something such as their and their best friends names. After each child had finished their code I reset the code in the editor to the template code while they tested their code.



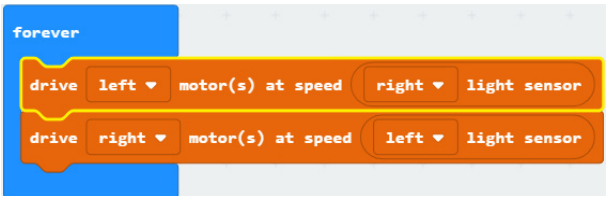
Above:
The 4Tronix Bit:commander add-on board.

I worked with different groups of children all morning before I went and had my lunch. After lunch, I did the same again but with the other Year 2 class. Again I taught small groups and all the children really enjoyed experimenting with the micro:bits.

RIGHT:
The 4Tronix bit:bit Robot.

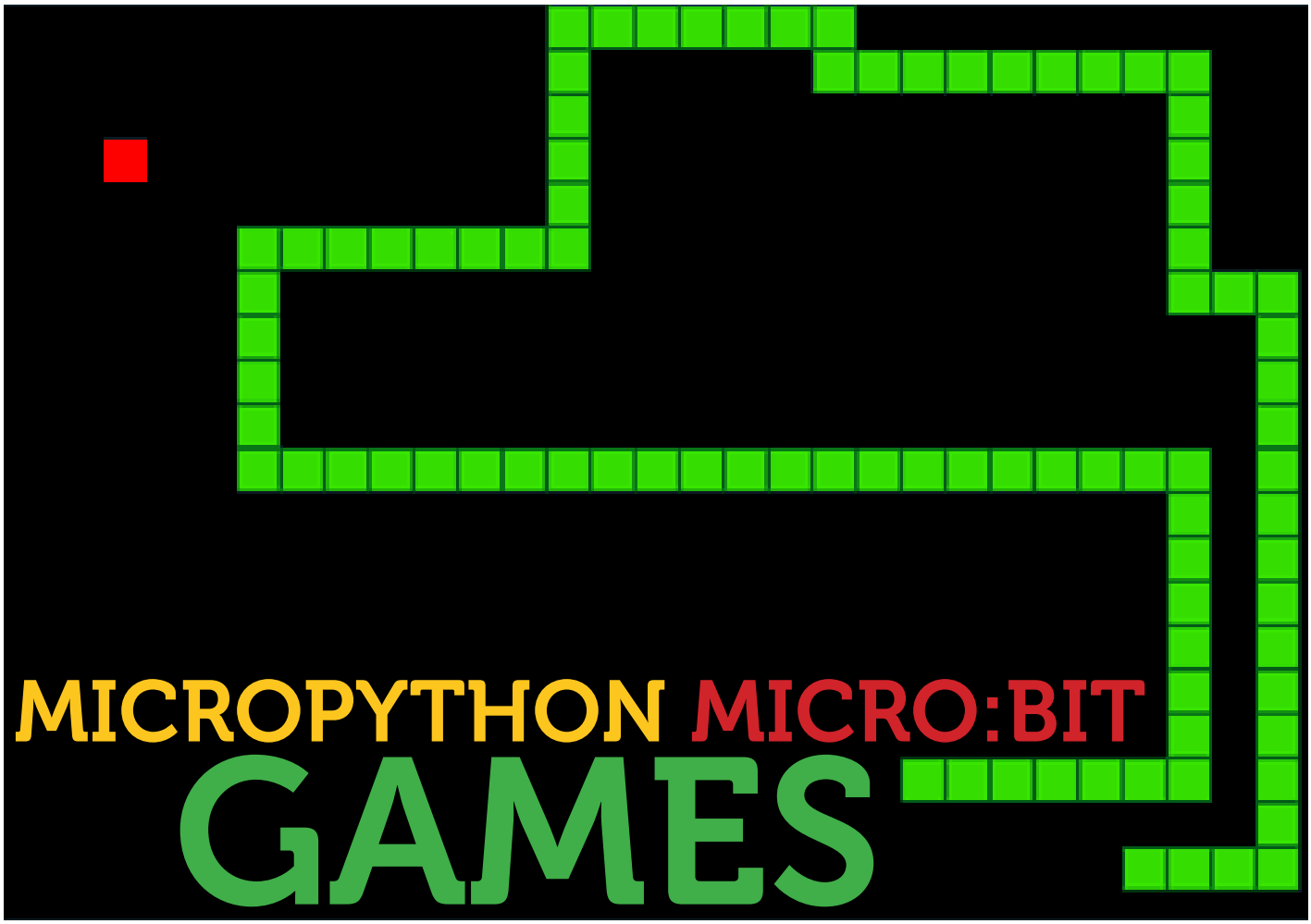
Lots of children saw the Bit:Bot and were interested in what it was and what it could do. I told them that I might be able to show them later. So at the end of the day after I had worked with around 60 children, I coded the Bit:Bot. The children all sat round in a circle eager to watch! I demonstrated the bit:bot inside the circle.

They were all surprised and amazed and were in awe of it! Then I told them that it could follow a torch beam, so mum put all the classroom blinds down while I coded the bit:bot with the following:



Unfortunately, we were in a real rush as it was time for the children to go home, and I used the wrong Bit:Bot package, so the code didn't work! Even though it didn't work, the children were still impressed that it was a possibility, and also that the Bit:Commander had controlled the Bit:Bot. I told them all how great micro:bits are and that if they want to get into coding they should definitely buy one!

I had a great day but was really tired and my feet were killing by the time we got home. Hopefully some of the children will be inspired to learn to code as it could lead on to play a major part in their lives like I hope it will mine!



MICROPYTHON MICRO:BIT GAMES

Phil Hall has ported two classic games to the micro:bit using MicroPython and has a view to improve them. Check them out!

About the Author

Phil has a Degree in Computer Science and have been tinkering with computers and electronics for the last 35 years. I now live in Yorkshire, England.

Phil is known as @rhubarbdog on many sites including Slack, IRC, Stack Exchange, Linux, Raspberry Pi and MicroPython forums

Firstly you'll need to download the games from GitHub (https://github.com/rhubarb-dog/microbit-games). Click the "Clone or / Download" button, click "Download ZIP" from the dropdown. Follow your operating system's usual instructions to extract the files. These games are written in MicroPython, there is also .hex files which can easily be copied to a micro:bit.

There's nokia classic snake.py and a maze.py game implementing a viewport. Concepts like high score tables and game timers will be discussed. These games are basic skeletons there's a whole load of opportunities to re-use this code to create your own games. The idea of providing people with these games i've created is so that they can learn from the code but also help me by improving them. If you do something cool with the games, please do share them with me using one of the platforms listed in my author bio.

Let's take a look at each of the games.

You will need

- A BBC micro:bit
- A computer with internet connection
- Software you may find useful:
  - microfs, to see the micro:bit filesystem
  - Mu Editor, a simple python editor

snake.py

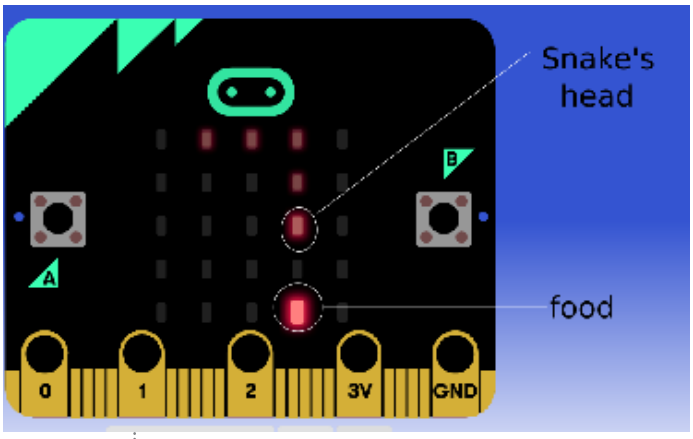
This is a port of the Nokia 3310 classic snake. Using the micro:bit's accelerometer you tilt the micro:bit to steer the snake around the screen. Eat food to score points and grow. Avoid eating your tail, or you'll die and that's game over.

The snake stores its tail's location in a python list. Adding the current position before moving and deleting the oldest coordinates if no food was eaten.

moving and deleting the oldest coordinates if no food was eaten.

The game is quite hard due to the lack of space a 5x5 screen gives. Expanding the micro:bit with a 8x8 or bigger LED matrix makes it much more playable.

Try adjusting the variable JOY_TOLLERANCE to improve steering. Make the game easier by only adding to the snakes tail for every 3 blobs of food eaten.



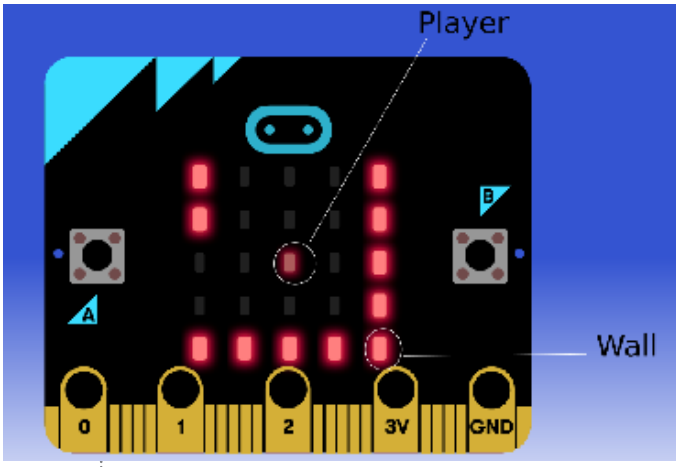
Above:
A simulation of what snake.py looks like

maze.py

The first thing this game does is introduce a playing area much larger than the micro:bit screen. The screen is used as a viewport into this much larger world. Like snake you tilt the micro:bit to move the player through the maze trying to find the exit. The player is the blinking pixel, the exit the pulsating one.

The limit with this game is the memory constraints of running a MicroPython program on a micro:bit. More features are possible in your games if you use Microsoft MakeCode. Extensions i've experimented with is escaping the maze in the smallest number of steps or quickest time. Placing a collectable token in every space, you win by collecting them all. To add enemies like the ghosts in Pac-Man wasn't possible in MicroPython and I'm yet to complete the task in Microsoft MakeCode. It's quite easy to create a different maze or place the player and exit in random locations. If selecting random locations ensure they aren't in a wall or the exit is in the same location as the player.

There are two things any game will benefit from thats a high score table and a clock to let you know how long you've actually been playing.



Above:
A simulation of what maze.py looks like

hi-score.py

Code snippets to include high scores are covered in this file. It uses the file system on the micro:bit, so high scores are maintained when the micro:bit is reset or has no power. It's only possible to have high scores in MicroPython programs Microsoft MakeCode doesn't support accessing the micro:bit's file system.

clock.py

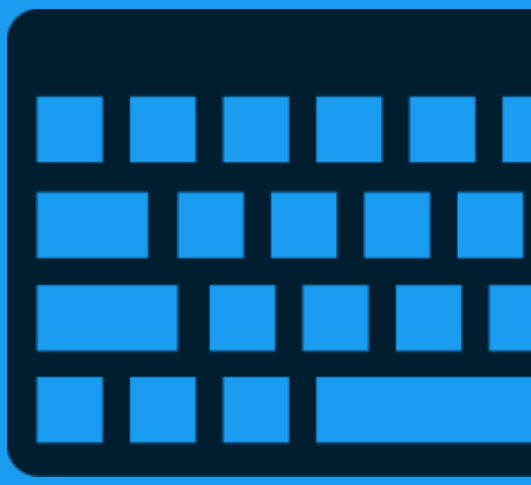
This file defines a single class Clock. This is a by the second timer with no wrap around under MicroPython. If you wish to make a Microsoft Makecode implementation it will handle times upto 12 days. Don't use this clock to record intervals accurately it is upto a second slow.

The class Clock has 5 methods

start – start the timer
stop – stop the timer
reset – reset it to 0 and keep going if the clock wasn't already stopped
time – The number of elapsed seconds since it was last reset
sleep(milli seconds) – This method should be used in place of the microbit.sleep function or either the time.sleep or time.sleep_ms function As it handles the updating of the number of seconds.

CALL FOR CONTRIBUTIONS

WRITE AN ARTICLE FOR THE NEXT MICRO:MAG



We want to hear about your awesome micro:bit powered project, event or story

FILL IN THE CONTRIBUTIONS FORM AT MICROMAG.CC/CONTRIBUTE

Even if you've never written for a magazine before, our team are here to help you get your article in micro:mag.

SUBSCRIBE TO EMAIL UPDATES

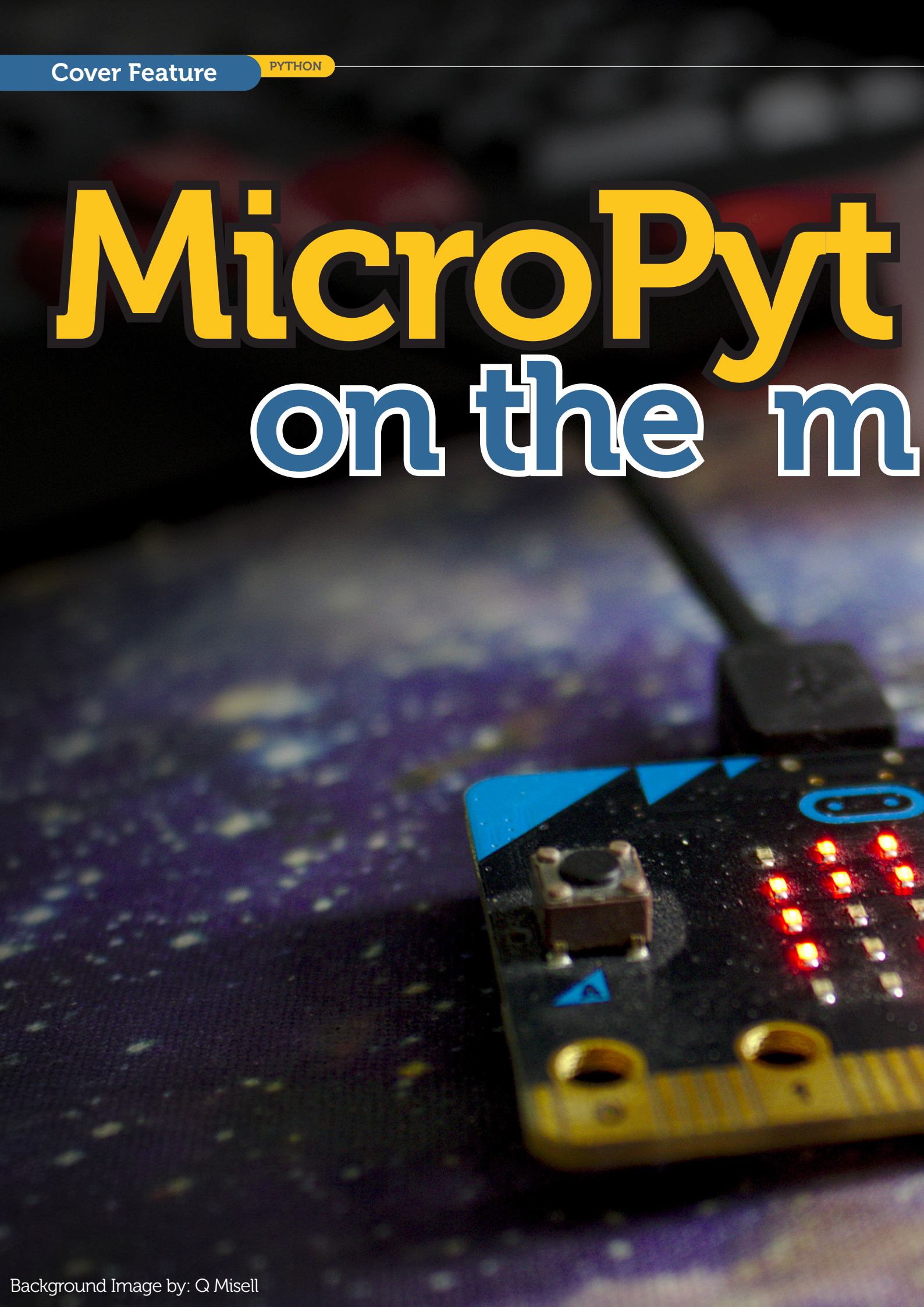
GET FRESH COPIES DIRECT TO YOUR INBOX!

Never miss an Issue and be the first to find out when a new Issue get's released by subscribing.

SIGN UP FOR FREE USING THE FORM AT MICROMAG.CC/EMAIL

We'll only ever contact you a few times a year about new Issues and other cool micro:mag related things.

MicroPyt on the m



Background Image by: Q Misell

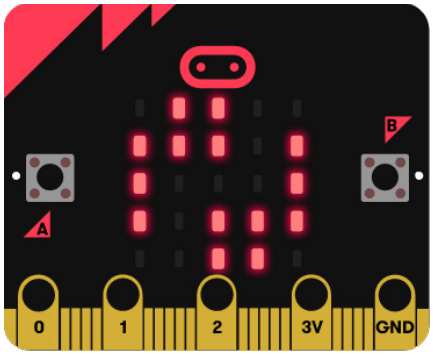
This issue, we explore the power and capabilities of MicroPython on the micro:bit. Cover Feature Collated by **Joshua Lowe**

hon icro:bit

Started by Guido Van Rossum 29 years ago in 1990 as a small project, Python is now the worlds fastest growing programming language and it's not hard to see why. Within the Python Community, there is a saying "Python is the second-best language for anything" which means that no matter what, you can pretty much guarantee whatever project you're trying to make, you can use Python to power it.

In recent years, there has been a large uptake in Python on Hardware. This is down to a port of Python called MicroPython which is a stripped-down version of Python made specifically to run on small microcontrollers like the micro:bit and the ESP32. Back when the original micro:bit project was in the works by the BBC, a team of people from the PSF which included Nicholas Tollervey and Damien George (The original creator of MicroPython) created the port of MicroPython for the BBC micro:bit.

This amazing project continues to live on and we'd really like to see more people using MicroPython on the micro:bit. So, we have compiled this Cover Feature which showcases some cool MicroPython guides, some projects and also a special micro:hit 9 step MicroPython challenge!



Getting Started

Below are two popular MicroPython editors. Find out more about them and choose the one you like best.

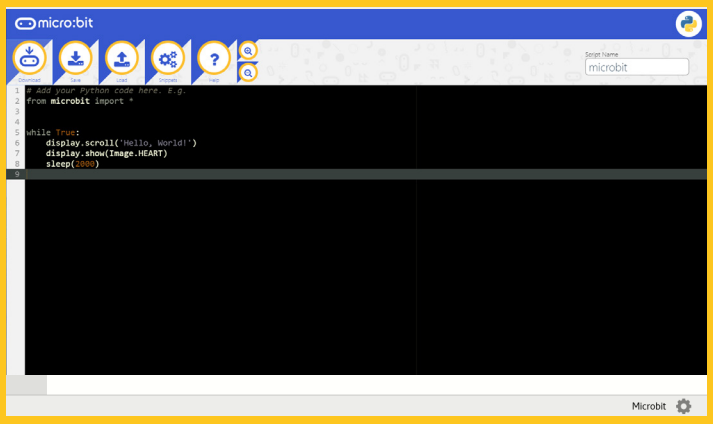
Meet the **Online Editor**

Micro:bit Foundation's official MicroPython online editor.

The official MicroPython editor from the micro:bit Educational Foundation is probably the quickest way to get up and running with MicroPython on your micro:bit as it's web-based and accessible from most web browsers.

The online editor is perfect to use within schools or anywhere where you don't have control over your network as you don't need to install any additional software. The online editor has a snippets button at the top which allows you to reuse commonly used blocks of code very easily without typing them out repeatedly.

Some really cool features are coming soon to the online editor like a Repl, direct flashing and also



the capability to add libraries and other files to the micro:bit easily via a new menu in the editor. You can find the link to the stable version of the editor below but you can also get the BETA at python.microbit.org/v/beta.

python.microbit.org

Meet the **Mu Editor**

The easy to use, multipurpose, Python editor.

Mu is a Python code editor for beginner programmers which was originally built for the BBC's micro:bit rollout. Since then, Mu has grown into a fully-fledged Python editor which supports multiple Python modes like CircuitPython, Python 3 & PyGame zero, just to name a few. It does, however, still retain support for the micro:bit.

Mu is a downloadable editor that works on Windows, macOS, Linux & Raspberry Pi and is visually similar to the web python editor but with lots more features, let's take a look at some of them. In Mu, it's easy to access the filesystem so you can use python libraries. This is handy if you want to use add-ons. You have access to the REPL which, with a click of a button, you can send commands

to your micro:bit and also see a serial output. There's also a checker that will check for any errors before you run your program and last but not least, you can also easily flash your micro:bit with one button instead of drag and drop. Below is the website where you can download Mu.

codewith.mu

Useful Links

There are plenty of places that are super useful when getting started with MicroPython on the micro:bit that you should take a look at!

MicroPython Read the Docs

The original team that built MicroPython on the BBC micro:bit created a handy Read the Docs page with explanations about each MicroPython feature and some code examples to go alongside it. For more advanced developers, the docs also include an API reference guide which lists all the commands and parameters for each micro:bit feature.

go.micromag.cc/mpyrtd

Python coding on the micro:bit Book

For those of you who prefer a physical book, there's a great one that's been written by James Gatenby and it explains the features of MicroPython with simple language. There's also several examples and projects to help you get started. You can buy it on Amazon and at CPC.

go.micromag.cc/mpybook

Awesome micro:bit List

Carlos from the Micro:bit Educational Foundation has created an "Awesome List" for micro:bit related projects and resources. There's loads of cool micro:bit stuff to check out and one of the sections on the list is stuff related to MicroPython. You should definitely use this list to get help and inspiration for using MicroPython.

go.micromag.cc/mpylist

MultiWingSpan

An invaluable resource for micro:bit lovers and especially MicroPython users is the MultiWingSpan wesbite. This features some great tutorials and examples for getting started with MicroPython. There's content for using add-on boards with the micro:bit using Python, some tutorials on how to interact with components and much more. Some of the content is quite advanced but there is stuff on there for everyone to enjoy.

go.micromag.cc/mpymws

EduBlocks

A great way to get started with MicroPython if you're coming from using MakeCode or Scratch is EduBlocks, created by one of our editors, edublocks provides a drag and drop block based programming environment with Python text on the blocks allowing you to see what the Python code is whilst you're coding. By using blocks, you're learning Python but not having to worry about the syntax. There are also some great resources too.

go.micromag.cc/mpyeb

10 Mini Projects

We've created 10 mini projects in MicroPython to help you explore the possibilites of MicroPython on the BBC micro:bit.

Projects 1 & 2

Project 1

Random Number Generator

This project will create a Random Number generator.
The random number generator will generate a number between 1 and 5 and dsiplay it on the micro:bit. Lets get coding!

Github Code Link:

go.micromag.cc/mpyp1

```
from microbit import *
import random
display.scroll("Shake Me")
while True:
    if accelerometer.was_gesture('shake'):
        shake = random.randint(1, 5)
        if shake == 1:
            display.show(1)
        if shake == 2:
            display.show(2)
        if shake == 3:
            display.show(3)
        if shake == 4:
            display.show(4)
        if shake == 5:
            display.show(5)
```

Project 2

Spirit Level

This project will create a Spirit Level.
The Spirit Level program. Great for checking that surfaces are level! Let's get coding!

Github Code Link:

go.micromag.cc/mpyp2

```
from microbit import *
while True:
    reading = accelerometer.get_x()
    if reading > 20:
        display.show("R")
    elif reading < -20:
        display.show("L")
    else:
        display.show("-")
```

10 Mini Projects

Projects 3 & 4

Project 3

Countdown Timer

This project will create a Countdown Timer. The timer will count up to 20 when button A is pressed and the timer will start counting down from what ever number you chose between 1 and 20 when button B is pressed. Lets get coding!

Github Code Link:

go.micromag.cc/mpyp3

```
from microbit import *
seconds = 0
while True:
    if button_a.is_pressed:
        if seconds < 20:
            seconds += 1
            display.scroll(seconds)
    if button_b.is_pressed:
        for i in range (seconds):
            display.scroll(seconds)
            sleep(1000)
            seconds -= 1
        display.show(Image.NO)
```

Project 4

Simple Dice

This project will create a simple dice This is a great little project if you want to play a board game, but the dice seems to have been misplaced. Let's get coding!

Github Code Link:

go.micromag.cc/mpyp4

```
from microbit import *
import random
while True:
    dice = random.randint(1,6)
    if accelerometer. is_gesture('shake'):
        display.show(dice)
```

10 Mini Projects

Projects 5 & 6

Project 5

Coin Flip

This project will create a Coin Flip Game. When you press the A button, the micro:bit will choose from True or False and display a happy or sad image depending on the choice.

Github Code Link:

go.micromag.cc/mpyp5

```
from microbit import *
import random
while True:
    if button_a.is_pressed():
        if random.choice([True, False]):
            display.show(Image.HAPPY)
        else:
            display.show(Image.SAD)
```

Project 6

Hello World

When learning a new programming language this will typically be the first program you learn to create, but this literally takes about a minute with a micro:bit. So I have decided to integrate the buttons and display some images too.

Github Code Link:

go.micromag.cc/mpyp6

```
from microbit import *
display.show(Image.HAPPY)
while True:
    if button_a.is_pressed():
        display.scroll("Hello World")
    elif button_b.is_pressed():
        display.show(Image.HEART)
```

10 Mini Projects

Projects 7 & 8

Project 7

Step Counter

This project will create a step counter. Everytime the micro:bit is shaken it increases the number of steps by 1. Lets get coding!

```
from microbit import *
steps = 0
while True:
    display.scroll(steps)
    sleep(50)
    if accelerometer. is_gesture('shake'):
        steps += 1
```

Github Code Link:

go.micromag.cc/mpyp7

Project 8

Temperature Sensor

This project will create a thermometer
Grab the temperature of the micro:bit to get an approximate temperature reading when you shake the micro:bit.

```
from microbit import *
while True:
    if accelerometer.is_gesture('shake'):
        temp = temperature()
        display.scroll(temp)
```

Github Code Link:

go.micromag.cc/mpyp8

10 Mini Projects

Projects 9 & 10

Project 9

Sending Messages

This project will use 2 micro:bit's over radio to send messages to each other. You'll need to upload this code to 2 micro:bit's for it to work.
Let's get coding!!

Github Code Link:

go.micromag.cc/mpyp9

```
from microbit import *
import radio
radio.on()
while True:
    if button_a.is_pressed():
        radio.send("Hello World")
    incoming = radio.receive()
    if incoming = "Hello World":
        display.scroll("Hello World")
```

Project 10

Touch a pin

When you touch a pin on the micro:bit, this project will scroll a message on the micro:bit display saying which pin you touched.

Github Code Link:

go.micromag.cc/mpyp10

```
from microbit import *
while True:
  if pin0.is_touched():
    display.scroll("You touched Pin 0!")
  elif pin1.is_touched():
    display.scroll("You touched Pin 1!")
  elif pin2.is_touched():
    display.scroll("You touched Pin 2!")
```

PRINT EDITIONS NOW AVAILABLE

GRAB YOUR PRINT EDITION OF MICRO:MAG TODAY!

Each Issue will now be available in print for a month after it's release date for £5.99 each!

BUY YOUR PRINT COPY ON OUR STORE AT MICRO:MAG.CC/STORE

You can still download the free digital edition from our website. View the catalogue at micromag.cc/issues

micro:bit

Introduction to MicroPython

Use MicroPython to create a simple project that flashes some LEDs. By **Les Pounder**

About the author

Les is a maker and trainer who has worked with the Raspberry Pi Foundation and the BBC to deliver computing training

@biglesp

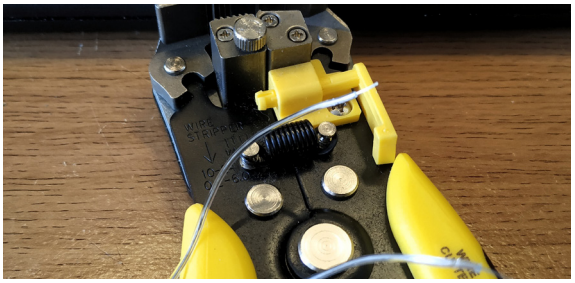
bigl.es

You will need...

- BBC micro:bit
- USB Cable
- Mu (codewith.mu)
- 2x Crocodile Clips
- LED lights powered by 2 x AA batteries (Pound / Euro / Dollar stores sell these)
- Wire Strippers

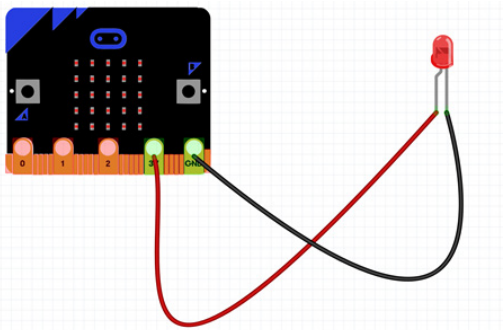
For this issue we take a look at using Micro Python on the micro:bit to create a simple project that flashes LEDs when the micro:bit is shook. For this project you will need to download and install Mu from https://codewith.mu/

Step 1 - Strip the wires



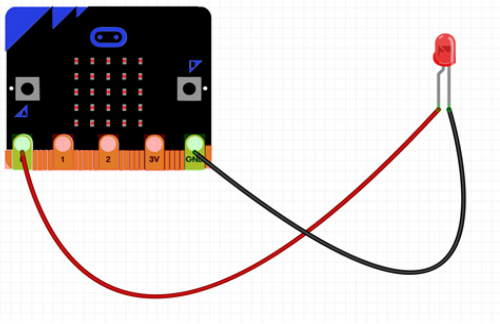
To use our LEDs we need to snip the wires from the battery pack, and then strip the wires. To snip and strip we can use any wire cutters.

Step 2 - Connect & Test



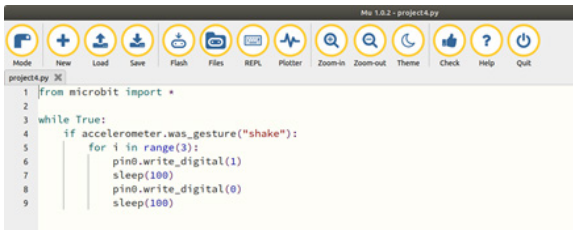
Using crocodile clips connect the wires of the LEDs to the 3V and GND pins. Do they light up? If not swap the wires. This is a hardware test, to make sure our electronics work.

Step 3 - Connect & Test



With the LEDs tested successfully we now need to move the crocodile clip from 3V to pin 0. This will cause the LEDs to turn off as we have not written the code to control it.

Step 4 - Using Mu

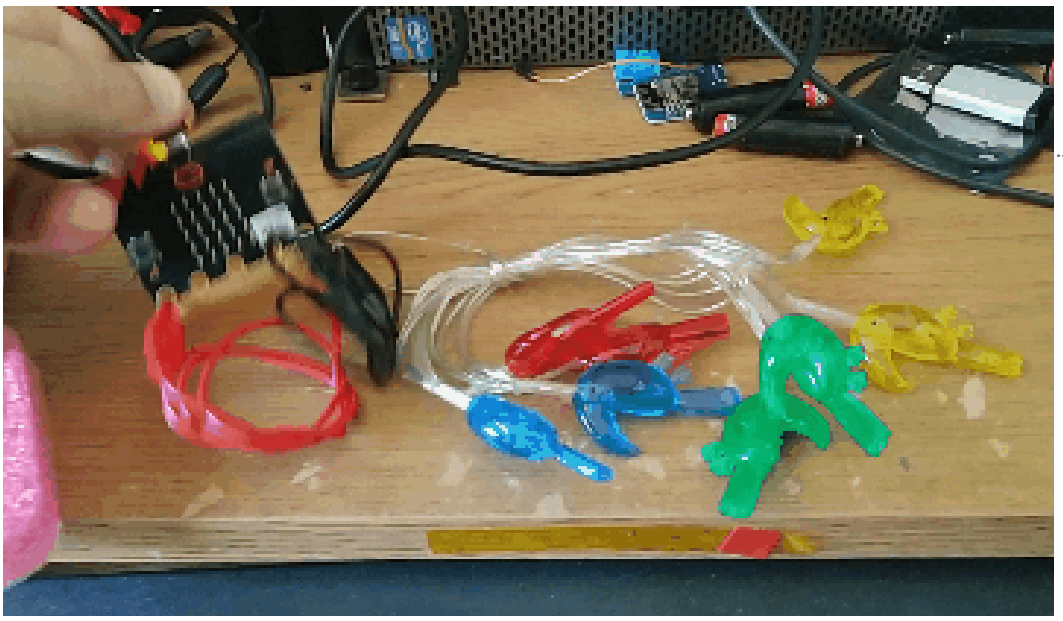


With the LEDs tested successfully we now need to move the crocodile clip from 3V to pin 0. This will cause the LEDs to turn off as we have not written the code to control it.

Step 5 - Imports + Loops

```
1  from microbit import *
2
3  while True:
```

Our first lines of code import the library enabling us to use the micro:bit hardware, then we create



To the left:
The micro:bit all wired up to the LEDs.

a loop that will run as long as the micro:bit is powered on.

Step 6 - If the micro:bit is shook!

```
4       if accelerometer.was_gesture("shake"):
```

Our micro:bit has an accelerometer which can be used to detect movement. Here we use it to detect when it has been shaken.

Step 7 - Writing a loop

Our LEDs need to flash three times and for that we use a for

```
5           for i in range(3):
```

loop and a range. Setting the range to 3, the loop starts at 0 and every time it goes round it counts up to 3.

Step 8 - Turning the pin on and off

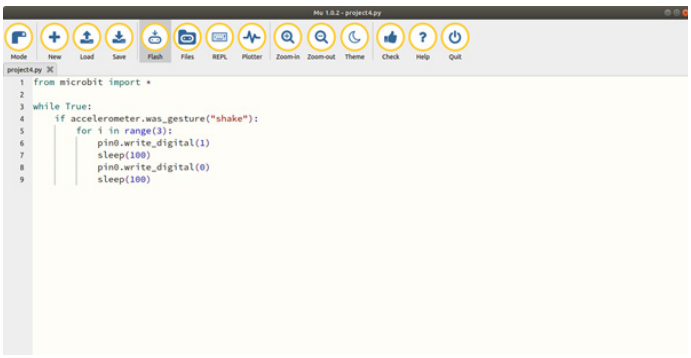
With the LEDs tested successfully we now need to move the

```
6               pin0.write_digital(1)
7               sleep(100)
8               pin0.write_digital(0)
9               sleep(100)
```

crocodile clip from 3V to pin 0. This will cause the LEDs to turn off as we have not written the code to control it.

Step 9 - Flash & Test

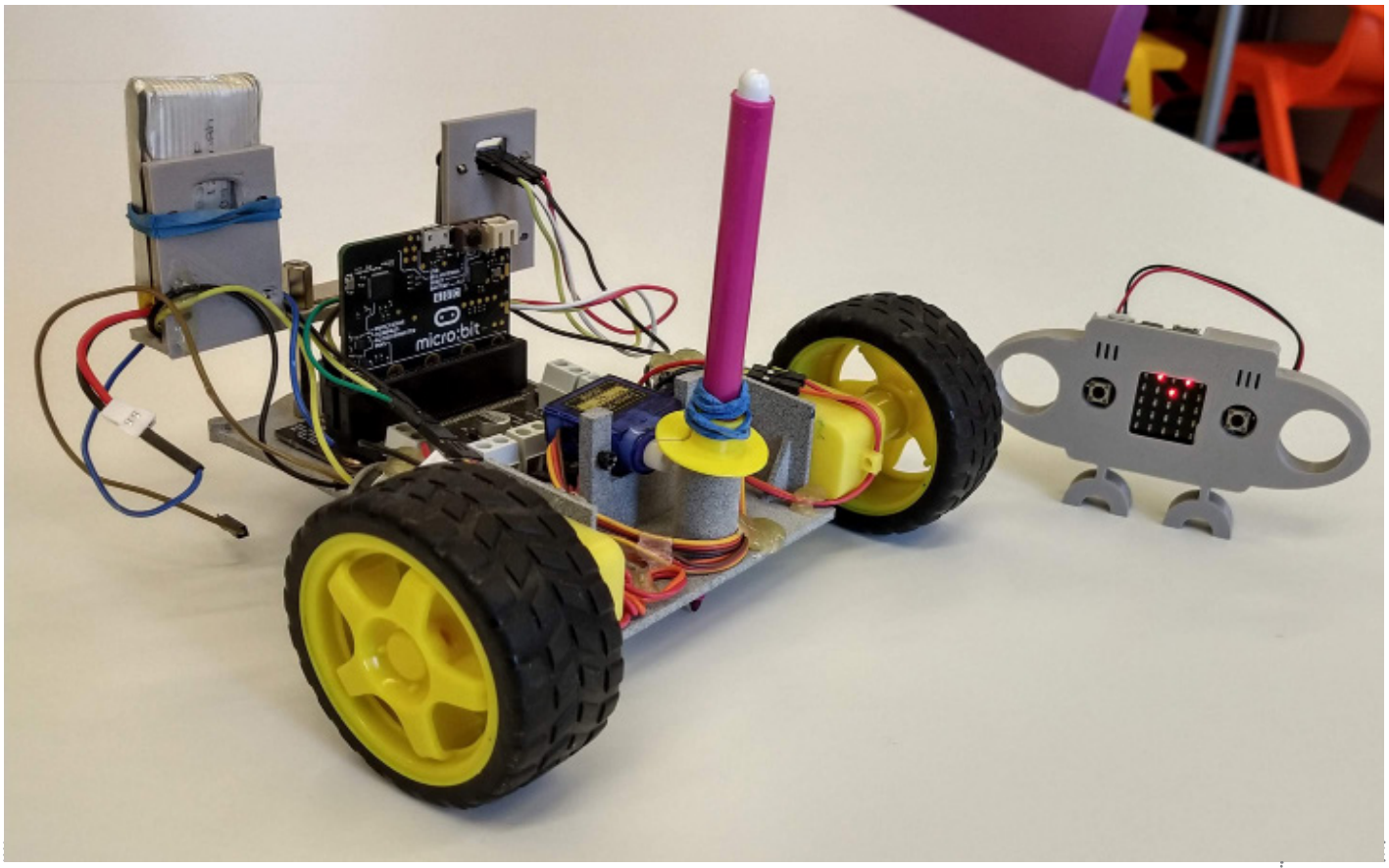
To copy the file to the micro:bit click on Flash and then wait for the yellow LED on the back to stop flashing. Now shake the micro:bit and watch the LEDs flash!



Full Code Listing

```
from microbit import *

while True:
    if accelerometer.was_gesture("shake"):
        for i in range(3):
            pin0.write_digital(1)
            sleep(100)
            pin0.write_digital(0)
            sleep(100)
```



Above:
ScaraBot EDU Base and Remote Controller

ScaraBot Edu

BY Adriano Parracciani

How to build a DIY drawing robot that can be remote controlled, or programmed.

About the Author

As a STEM Educator Adriano designs and runs one-shot workshops to annual technology laboratories, based on Creative Learning. Adriano also likes making things with hands and mind and likes tinkering with different materials.

Twitter: @CyberParra

ScaraBot EDU is a DIY drawing robot based on the micro:bit, which will allow you to explore: generative art with coding, math and geometry, and the art of doodle design by hand-control. Designing and building ScaraBot EDU has been a great learning experience to me, encompassing: electronics, 3D modelling, 3D printing and making. Once my ScaraBot EDU was ready, it was amazing to start making and drawings through coding or via remote control. I hope it will be the same for you!

FEATURES

The ScaraBot EDU is formed from two main parts, both based on the micro:bit:
ScaraBot EDU Base
ScaraBot EDU Remote Controller

You can make drawings in two ways:

Remote Controlled Mode

ScaraBot EDU can be remote-controlled by a second micro:bit through radio frequency communication. The ScaraBot Radio Controller is like a joypad and depending on how you move it, it sends specific radio commands to the ScaraBot EDU Base.
For example: if you tilt the ScaraBot EDU Radio Controller down, it sends the “forward” message to

the ScaraBot EDU Base.

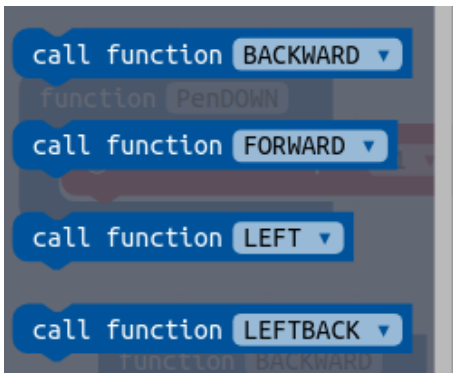
In order to use the ScaraBot in the radio-controlled mode you need to:

1. Upload the ScaraBotEDU.hex code into the ScaraBot EDU Robot
2. Upload this ScaraBotController.hex code into the ScaraBot Radio Controller

Programming Mode

You can create and upload a specific program onto the micro:bit to generate the desired drawings. I have created a library of motor commands to reduce your coding time and make it easier for you. You don't need to think about which PIN has to be set on/off to obtain a specific movement, rather you can just drag the block function you need.

In order to use the ScaraBot in programming mode, you can import the motor library hex code into the MakeCode Editor. Use the motor code blocks to create your drawing projects; experiment and have fun!



HOW TO MAKE

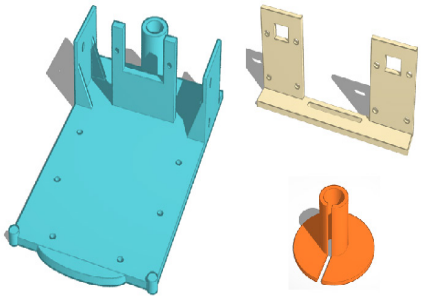
The Wiring

Follow the wiring diagram in Figure 1 to create the right connections between the micro:bit's pins and the motors

The OLED Display (optional)

Optionally, you can mount an OLED display to your ScaraBot EDU to show messages and data. You can modify the program and decide which messages and data to display and when. The OLED display connects to the micro:bit using four wires. You can find the wiring diagram in Figure 2.

In order to program the OLED, you must add



a package to MakeCode.
Go to the Add Package menu inside the MakeCode editor and look for tinkeracademy-tinker-kit. When this package is installed a new OLED category appears with several new coding blocks inside it.

3D Part: The Chassis

I have designed the ScaraBot's chassis with Tinkercad.

You can download the 3D file (STL) from the GitLab repo listed on the right, it's ready to be printed.

3D Part: The Pen Ring

In order to lift the pen on and off, the pen should have a thicker part in the middle that can be touched by the servo-motor's arm. To do that, I have designed a cylindrical ring with a larger disk on the base. The marker pen has to be inserted inside the ring; a rubber band around the ring will allow it to adhere to the pen.

Here is the Pen Ring 3D file [https://gitlab.com/cyberparra/ScaraBot-edu/blob/master/penring.stl]

3D Part: the Holder (optional)

The OLED display can be mounted on this holder which can also hold the LIPO battery.

Here is the Holder 3D file [https://gitlab.com/cyberparra/ScaraBot-edu/blob/master/OLEDholder.stl]

CONCLUSIONS

Can you guess why I called it ScaraBot EDU?

Because scarabocchi is the Italian for scribbles, and making a scribbling machine lets you learn a lot and have lots of fun!

You Will Need

2x Right Angle Gear Motors
2x Wheels for Gear Motors
1x Steel Ball Caster
2x micro:bits
1x Motor Driver Board
1x 4.5-6v Power Supply
1x Servo Motor
1x DIY Chassis and fixings

Download the files:
go.micromag.cc/scarabot

Diagrams

Figure 1

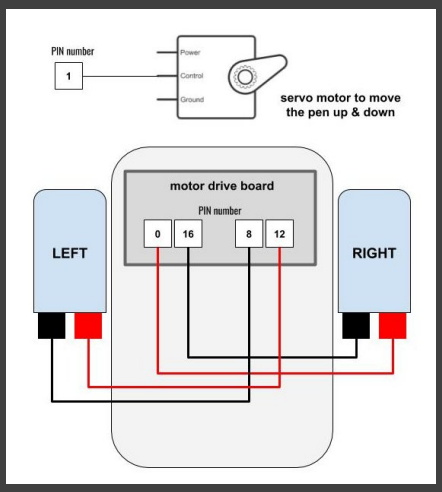
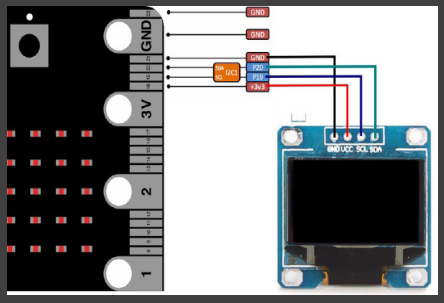


Figure 2



Toy Cars Timing Gate

BY Robert Wiltshire

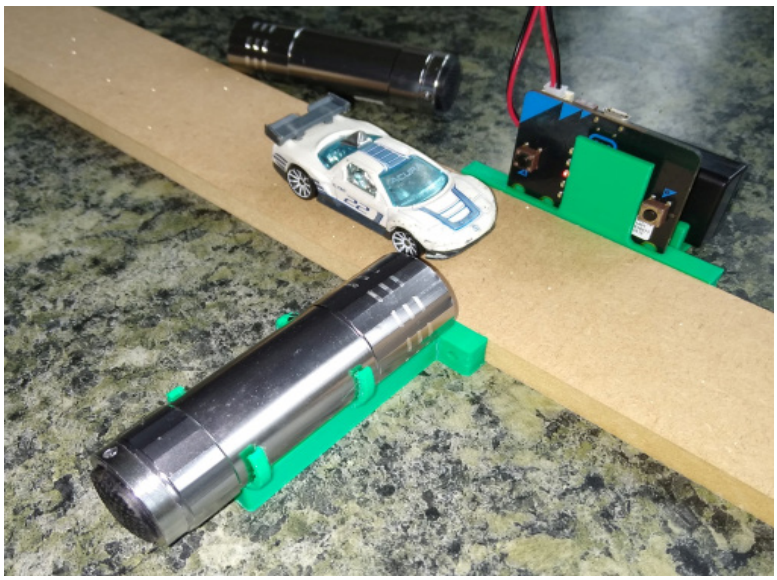
In this project, Two micro bits are used as timing traps. A torch shines a light across the track onto a micro:bit. The micro:bit LEDs are sensitive to light and when a car's shadow passes in front of the LED the drop in light intensity signal can be used to trigger a timer. One micro:bit to start, a second to stop. A third micro:bit then receives the timing signals, reads the times then calculates and displays the speed for each run.

About the Author

Robert is part of a two-man team from Software Cornwall that runs school workshops, assemblies, careers fairs and work experience weeks, with a Mission to Mars, to engage Cornish students in the local tech industry. I am an old school maker turned digital. My aim is to attract as many students to the possibilities of the local tech industry as a career.

Twitter:

@SwCornwall



Left: One of two micro:bit gates with torch and 3D printed holders

The Timing Gates

This is the code for the two-timing gates. The only difference between them is the "Start" here is replaced with a "Stop". Flash a start and stop version of this code onto two micro:bits.

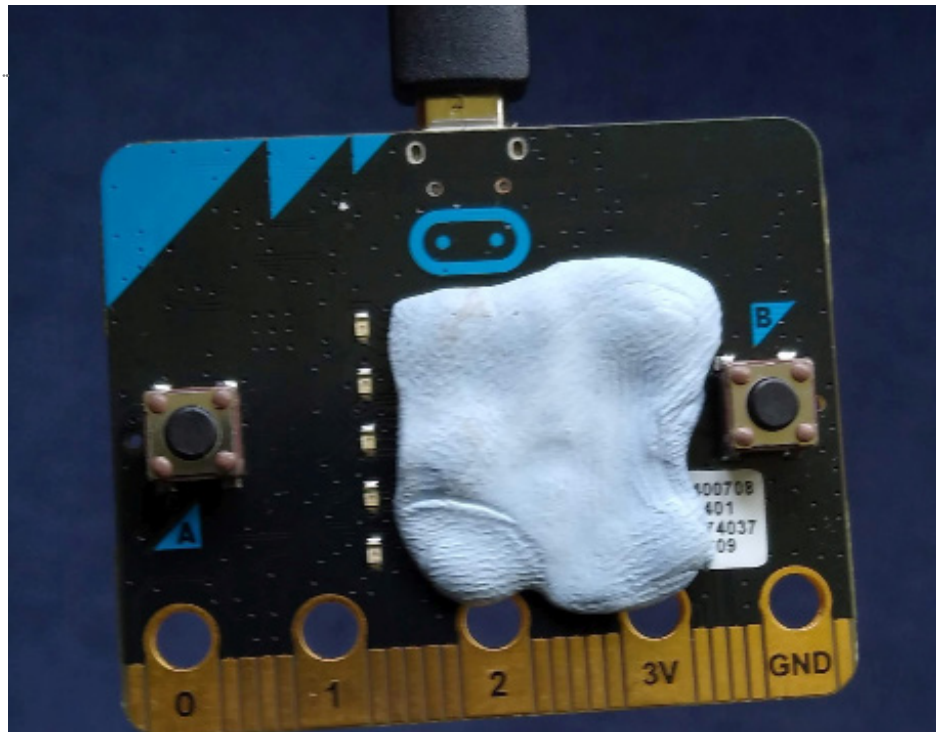
```
from microbit import *
import radio

radio.on()
torch_light = 100

while True:
  light_level = display.read_light_level()
  if light_level < torch_light:
    radio.send("Start")
    sleep(1000)
  sleep(10)
```

This code only sends a message, which is the word "Start" or "Stop", when the light drops below the torch_light value.

Stick a piece of tape or Blu Tack over the LED on the front of the two other micro:bits leaving just the left side clear. This makes for a nice sharp timing gate. With battery packs connected to the other two micro:bits. Place the Start and Stop timing gates on the side of the track with their torches. Place them a measured distance apart from LED to LED. We used 250mm or 0.25m for our tests.



Right:
mcro:bit with masked LED]

The Timer

The micro:bits can count, in milliseconds, the amount of time that has passed since they are switched on. By recording the time at the point of receiving the two messages and subtracting the lower start time from the larger stop time the elapsed time is found. If you then know the distance between the two LED on the track the speed can be calculated.

The final piece of the code is this code below for the receiver. Change the LED_Distance value to yours (in metres). So half a metre would be 0.5.

```
import radio
from time import ticks_ms

radio.on()
print("Ready")
start_time = 0
stop_time = 0
LED_distance = 0.25

while True:
  message = radio.receive()
  if message == 'Start':
    start_time = ticks_ms()
    print("Start timing at " + str(start_time))
  elif message == 'Stop':
    stop_time = ticks_ms()
    print("Stop timing at " + str(stop_time))
    time_difference = stop_time - start_time
    car_speed = (LED_distance / time_difference) *
1000
    print("Speed of car " + str(car_speed) + " m/sec")
```

Flash this code and then, assuming you're using Mu, click REPL. The Python chevron >>> will appear. Restart the micro:bit by pressing the button on the rear. The "Ready" message will show the code is running and waiting. Now let a car drive down in front of the gates.

If necessary adjust the positions of the lights to get a better shadow or the torch_light value might need lowering.

Here is the display showing two times and the speed.

```
10 while True:
11     message = radio.receive()
12     if message == 'Start':
BBC micro:bit REPL
>>>
MicroPython v1.9.2-34-gd64154c73 on 2
Type "help()" for more information.
>>> Ready
Start at 4865
Stop at 5179
Speed of car 0.796178 m/sec
```

So which of your cars is the fastest, or slowest!

Full instructions and 3D STL files are available at : https://go.micromag.cc/timing_gate

TEXTING WITH MICROPYTHON

Can micro:bit send text messages with only two buttons using MicroPython? By **Christopher Roscoe**

About the Author

Christopher is a father of two (Oliver & Isla) and a self-taught programmer since about 1981

You will need...

Two micro:bits

A computer

MicroPython editor

Two children who want to send silly messages to each other.

Code files

go.micromag.cc/mptexting

When I wrote this a year ago, my children were pestering me for phones so that they could send text messages to their friends. I wondered whether it would be possible to make something with their micro:bits that would work at short range. I had been reading the MicroPython manual and discovered the radio section. That left me with only one problem: how could they write messages with only two buttons?

Two buttons is not a lot to be playing with but, for me, the main strength of the micro:bit is the creative effort required to make this little marvel do what you want. So we have Button A and Button B but we also have the two buttons together. I could have used the shake gesture for a reset. Let's get into it and code our own micro:bit controlled texting system. You'll need to load up **python.microbit.org** for this tutorial.

Load the modules

```
from microbit import *
import radio
import music
```

Set up the radio channel

Having loaded up the modules we need to set up the radio system to ensure that both micro:bits are using the same 'radio group'. This is like setting two walkie-talkies to the same channel so that they can hear each other when messages are sent.

```
12 radio.config(group=1)
13
14 # We will use the default radio
15 radio.on()
```

Set up the variables

This next section sets up all the variables we need to make the code work. The key discovery here for me was the idea of putting all the characters you want to send in the same character string. You can add to this if you want. My son uses semicolons in his text messages so I had to include that. I also added several spaces to speed things up. You can change the order of any of the characters in the string.

```
# this variable is used to hold a single character at a time.
character = ""
# this is an integer that is used to select one
```

```
  character from charList
  charNum = 0
  # all available characters - edit this list as required.
  charList = "? ABCDEFG HIJKLM NOPQRST UVWXYZ .,:;*'?!1234567890"
# auto-detects the length of the above
numberOfChars = len(charList)

# ensures the message is blank at the outset.
message = ""

# this is the message received so should be blank to start with.
incomingMessage = ""

# The doneButtons variable is used to make sure that pressing BOTH buttons will prevent the code for buttons A OR B from operating.  When the if statement has run, the variable sets to 1 which prevents the other code from running. Each IF statement will only work if the doneButtons is 0.
doneButtons = 0
```

The Main Loop

Although I dare say a lot of this could be rewritten as functions, I haven't got my head around them yet so I stuck to tried and tested methods that I could make work but you could rewrite this as a challenge. I annotated it as I was writing it so that I knew what each section did. A year on I can still understand what I was trying to do but I am not always this fastidious.

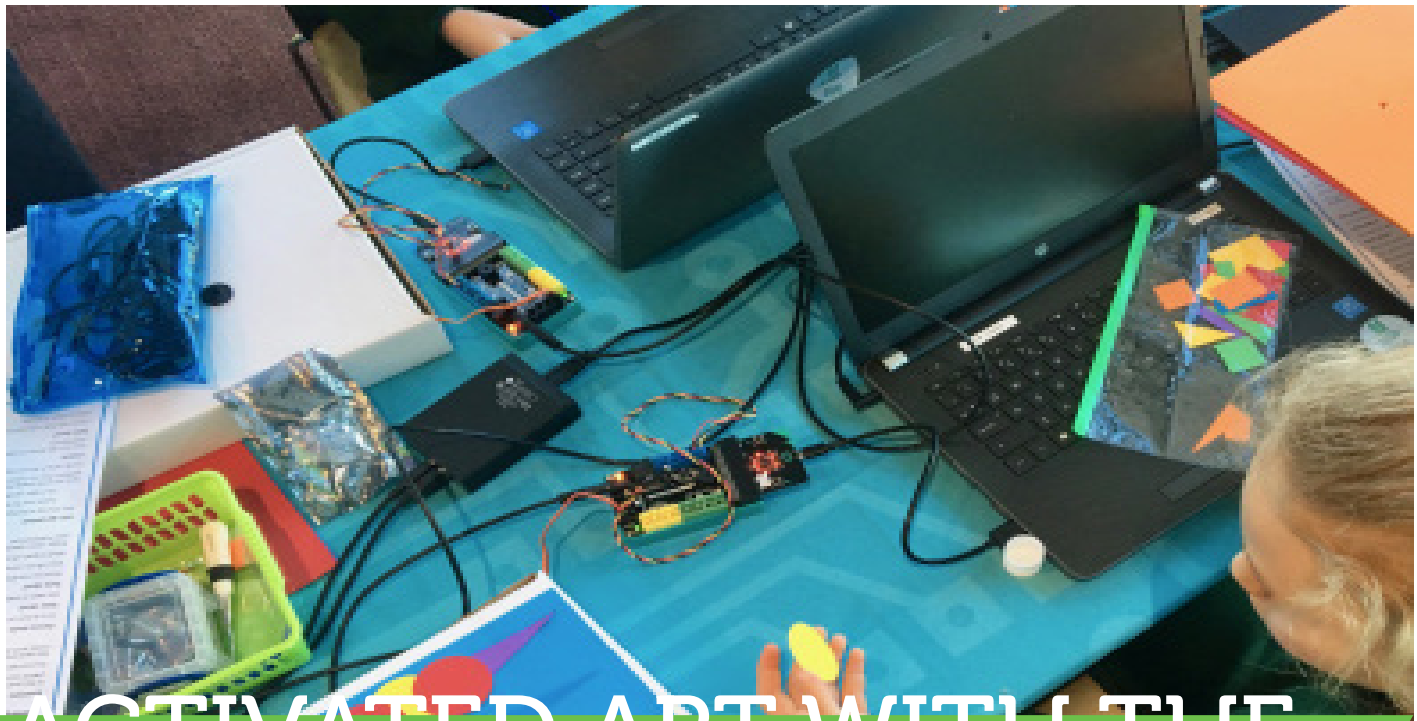
```
# This is the main loop that makes the programme work
while True:
  # resets the variable for each pass through the loop - it will only be 1 if a button has been pressed.
  doneButtons = 0
  # Pressing both buttons will send the message and play a tune
  if button_a.is_pressed() and button_b.is_pressed():
      # adds message start and end characters
      display.scroll(">" + message + "<")
      # we crossed our fingers but it works
      radio.send(message)
      # play some music, message coming
      music.play(music.FUNK)
      # send it again just in case
      radio.send(message)
      # resets the message, ready to go again.
      message = ""
      # resets the starting character
      charNum = 0
      # resets the character
      character = 0
      # makes sure that pressing a single button won't fire additional code
      doneButtons = 1
  elif button_a.is_pressed() and doneButtons == 0:
      charNum = charNum + 1
      # using modulo operation to make this work.charNum becomes the remainder of charNum/numberOfChars this makes sure you can't go beyond the available characters
      charNum = charNum % numberOfChars
      # just plays middle C
      music.play("C3:1")
      sleep(100)  # we added this while testing as it occasionally crashed
      # select one character from charList
      character = charList[charNum]
      # show the current character
      display.show(character)
      # this should then prevent the elif for a single button press of A or B
      doneButtons = 1
  # This is where we confirm the selected character and
  # add it to the end of the message and play a note
  elif button_b.is_pressed() and doneButtons == 0:
      # just plays a high C
      music.play("C5:1")
      # adds the selected character to the message
      message = message + character
      # displays the message so far.
      display.scroll(">" + message)
      # resets the position at 0 for the character string
      charNum = 0
      # makes sure that pressing a single button won't fire additional code
      # I probably don't need it here as there are no subsequent
      # button lines
      #   between here and the start of the main loop
      doneButtons = 1
  # Message receiving section.
  # This is the bit I'm most doubtful about
  # incomingMessage is the variable to store any incoming message
  incomingMessage = radio.receive()
  # this section should ONLY run if there's a message from another micro:bit
  if incomingMessage:
      # plays ringtone on receipt
      music.play(music.RINGTONE)
      # scroll that message
      display.scroll("" + incomingMessage)
      # wait one second
      sleep(1000)
      # display the message again
      display.scroll(incomingMessage)
      # reset the incoming message.
      incomingMessage = ""
```

How it works

Click on button A to display the next available character in the charList variable.

Click on button B to select that character

Click on both buttons together to send the message.



ACTIVATED ART WITH THE HUMMINGBIRD

Combine shape and colour with robotic elements of motion and light detection to create an interactive work of art. By **Su Adams & Katy Henry**

About the Authors

Su Adams is a Computing Curriculum Specialist and business owner of U Can Too

Katie Henry is the Director of Learning for BirdBrain Technologies. Creators of the HummingBird Kit

Art is a form of communication, a way of sharing feelings and ideas with others. Think about the idea or feeling you would like to share with people who will experience your masterpiece.

In this short activity, you'll combine the movement of a servo motor with shapes and colour to create your own artistic masterpiece inspired by abstract artist Wassily Kandinsky. Kandinsky explored whether shape and colour alone could be used to represent feelings and music. For example, Kandinsky felt that the colour yellow expressed a sound like a trumpet and that if certain colours were used together they could create harmonies like a chord.

You will need...

- **Cardboard Box (Any Size will do)**
- **Coloured Paper (Large sheets + Shapes)**
- **Hot Glue, Blue Tac or any Adhesive**
- **HummingBird Kit with micro:bit.**

How might Kandinsky's artwork differ if he were able to include motion as well as colour and shape?

Plan

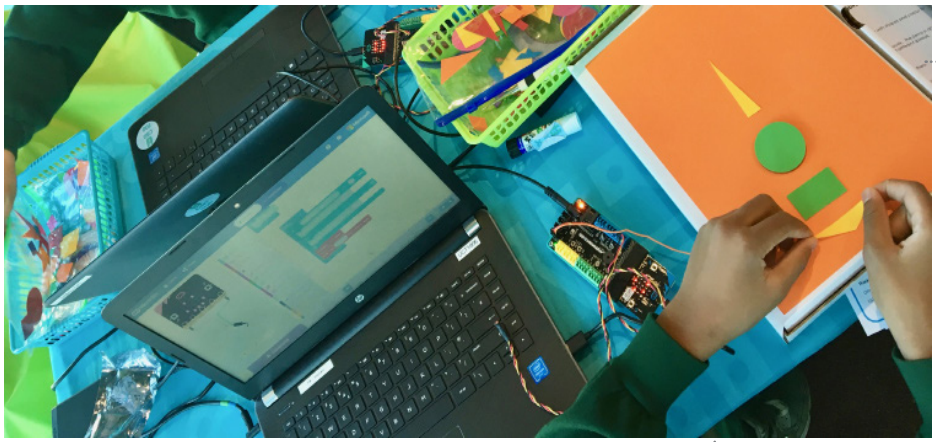
» What would you like your masterpiece to look like?

» How will you communicate the feeling or idea that you are trying to share?

» What shapes will you use and which colours?

» Which element will you make move and how will you use motion to enhance and improve your masterpiece?

Then, choose a background colour paper and lay it over top of your box. Poke a hole in the paper where the motor will push through. Next, pick the coloured shapes that you would like to use and stick them onto the coloured background in the position you would like. Attach

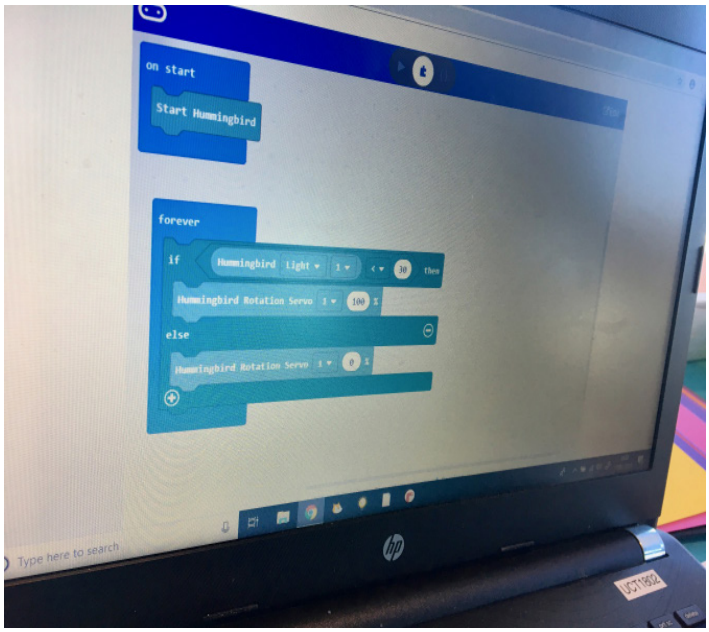


Left:
A student creates a Kandinsky-inspired work of art using the Hummingbird Robotics Kit and craft supplies

your artwork to the box, making sure that the hole in the paper lines up over the motor. Finally, affix your moving element to the top of the servo horn.

Code it

Now you are ready to explore how movement can be used to change the effect of your masterpiece. Consider adding a light sensor from the Hummingbird Robotics Kit (or using one of the sensors on your micro:bit) to control your motor. Use the free programming tutorials at birdbraintechnologies.com to help you.



Extend

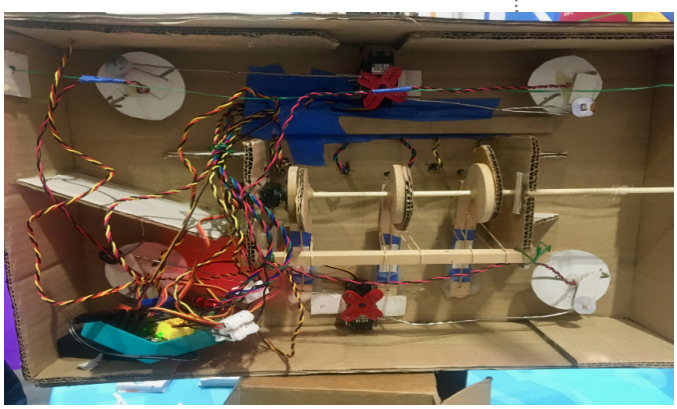
Inspired by Paul Klee's The Goldfish, STEAM lead at Devonshire Hill Primary School Pablo Joyce affixed more than just paper to his box. He used assorted craft supplies and reusable material such as discarded styrofoam and balsa wood. Each section of this fish moves because of the cam and follower mechanism Pablo built on the back. This project uses 5 LEDs, one motor (mounted in the back), one light sensor (in the eye of the fish), one ultrasonic distance sensor (top, middle of the robot), and one potentiometer (top right corner of the box).

Share with others

This is a great project for beginners and advanced makers alike. Consider hosting an Activated Art meet up in a library maker space or CoderDojo. If you want a printable copy of this project description, contact Su at Su@ucantoo.org.uk

Above:
This robot was created by Pablo, (@pablisch on twitter) using the Hummingbird Robotics Kit

Below:
This is the back of Pablo's robot. With flexible robotics kits like the Hummingbird Robotics Kit, you can easily make amazing projects



wifi:bit

Add WiFi cabability to your micro:bit with this neat ESP8266 based board from e-radionica . By **Joshua Lowe**

AVAILABLE FROM
e-radionica
go.micromag.cc/wifi

What's Included?
- wifi:bit

Price
$15.07 USD
€14.66 EUR
£12.41 GBP
Approx

THE micro:bit is packed full of features like an accelorometer, bluetooth and radio communication to name a few, however, one feature that it lacks it the ability to connect to the internet to unlock the Internet of Things.

Croatian based company E-Radionica saw a gap in the market for a WiFi add-on for the micro:bit to add this functionality. wifi:bit is based on the ESP8266, a popular microcontroller that has a wifi chip onboard. The board is super easy to connect to the micro:bit with its edge connector and one thing I liked about the board is the fact that the pins are broken out at the bottom of the board so you don't loose access to unused pins, something that many add-on boards lack. The board is super simple to use with a provided MakeCode package and a few examples on the website.

The library has lots of functionality to get you started. Only 2 blocks are required to connect to the wifi:bit and your wifi at home. Digging a bit deeper into the library, you can use the HTTP request block to interact with the external API's, for example you could control Philips Hue lights or interact with a a service on IFTTT, the possibilites are endless!

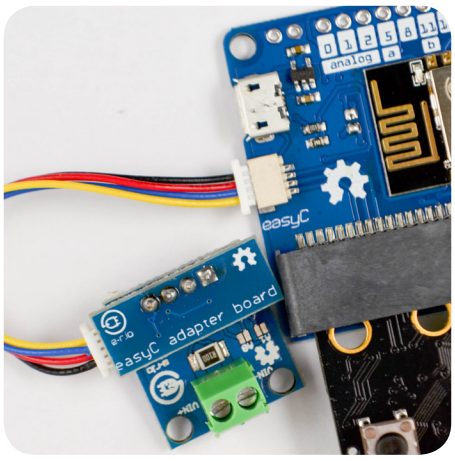
Another cool feature in the MakeCode library is the Blynk block which allows you to easily interact with the Blynk service which allows you to control pins on the Raspberry Pi and extend the functionality with Internet of Things.

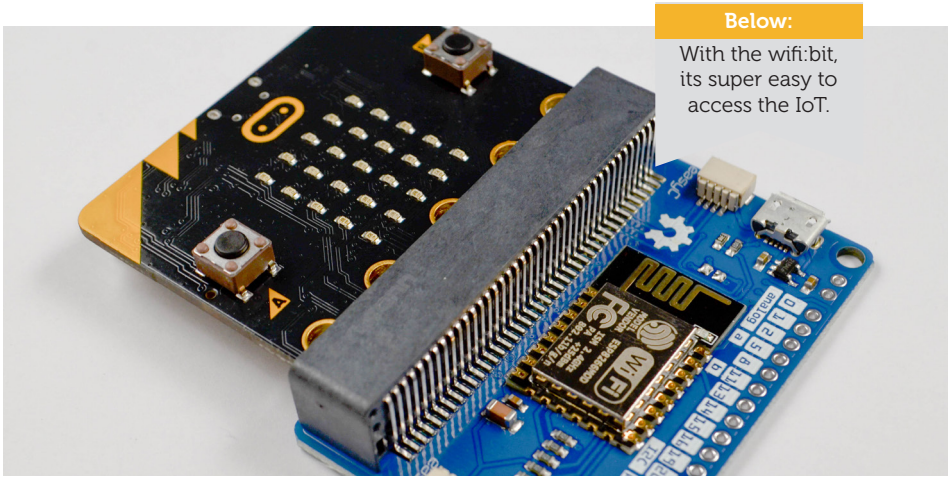
One thing that I see as missing from this development board is support for MicroPython, this seems to be a common trend for many new add-on boards. It'd be great to see MicroPython support for this board in the future.

Another cool feature of this board is the easy-c connector which allows you to connect different modules via a grove type connector to allow easy expansion where you can add sensors and other cool components. We'd pick one up if you're looking for a different type of add-on board.

OUR RATING

9/10

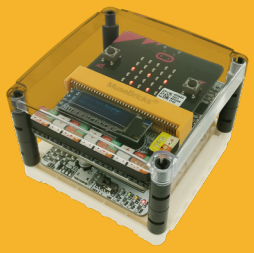




Below: With the wifi:bit, its super easy to access the IoT.

Similar Products

Two other products to consider...



01 Muselab IOT Shield

This product is much more expesnive but focuses on more than just wifi capability and even has its own screen!

02 IOT Environment

Again, more expensive. The Elecfreaks IOT Environment Kit packs a range of sensors with wifi capability too.

Inksmith K8

The modular kit for learning coding and robotics with the micro:bit. By **Joshua Lowe**

AVAILABLE FROM
Inksmith Store

go.micromag.cc/buyk8

What's Included?

- Front Ultrasonic Sensor.
- 3 line following sensors
- 2 wheels
- Battery Pack
- K8 Robot Driver board
- 2 9 Gram Servo Motors

Price

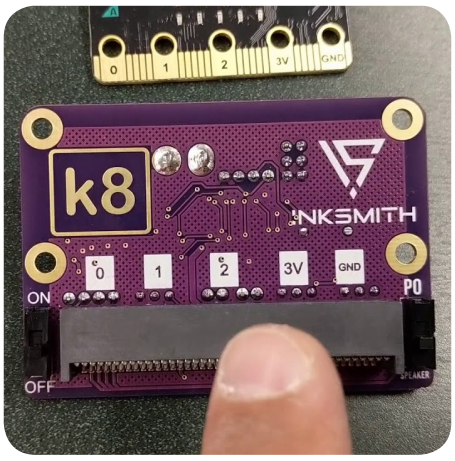
$60.43 USD
€53.99 EUR
£49.77 GBP
Approx

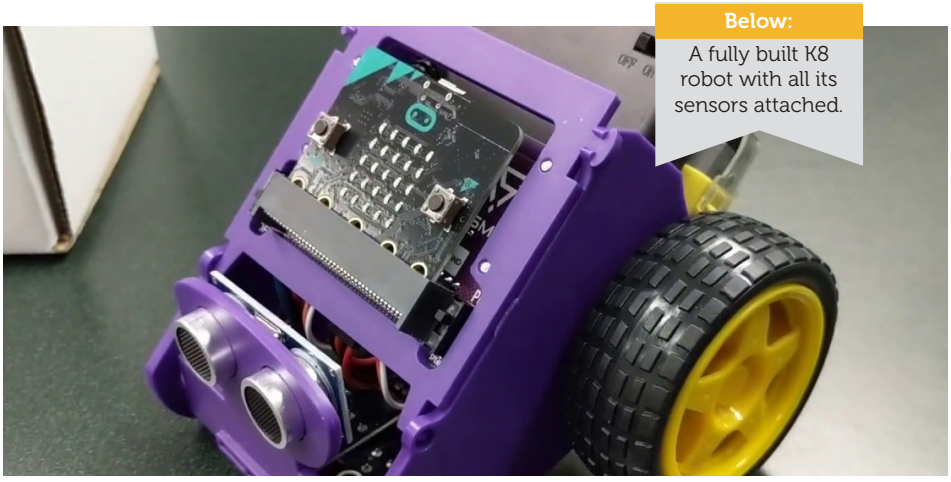
We see a lot of micro:bit robots with testing products and doing reviews for the magazine. Some have amazing features but lack support, some of them are more advanced than others. However, the Inksmith K8 robot is different, its feature packed and even has its own curriculum! Let's take a look.

Unlike many micro:bit robots, the K8 is made from a purple injection moulded plastic chassis, most of the other robots we've used or reviewed have just used the PCB as a chassis so having a nice case really makes this robot feel premium. The heart of this robot is the K8 driver PCB and it's certainly a good looking purple PCB. We really liked the design and layout of this PCB, it's a nice touch. The robot uses two classic yellow robot motors you can find in most robot kits. These motors have a good speed and work well. I found that construction was slightly fiddly compared to other robots that I have assembled in the past but once I got the hang of it, the K8 was quite fun to build. Going back to that lovely purple PCB, it has an easy to use edge connector for the micro:bit as well as connectors on the back for 3 line following sensors, a sonar sensor, two motors, 2 servos (so you can build robot arms etc.) and a barrel jack connector for the battery back power. Once assembled, the K8 robot is a feature packed robot for the price of $60 USD. One of the great things about the K8 robot is its selection of tutorials and resources. Inksmith have really excelled with the selection they offer. The courses on the Teachable website are enough to have a whole curriculum built around the robot which is super handy for schools. Unfortunately, I can't help but notice the lack of MicroPython support for this robot, I feel that this is one of its main let downs. I'd really like to see this in the future. All in all though, the K8 robot is really one of the best micro:bit robots out there, with its premium packaging, design and resource selection it really is the perfect robot for schools and code clubs alike and you should consider picking one up!

OUR RATING

8/10

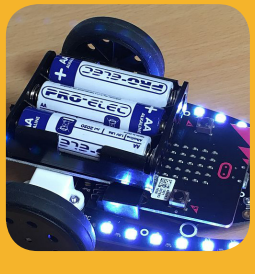




Below: A fully built K8 robot with all its sensors attached.

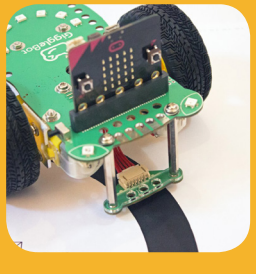
Similar Products

Two other robots to consider...



01 4Tronix Bit:Bot

Probably the most popular robot for the micro:bit out there. Great if you're looking for low cost robotics.



02 Dexter GiggleBot

We gave this robot a 10/10, we love the fact that it supports basically every micro:bit editor out there.

YOUR PRODUCT/AD HERE

ADVERTISE IN THE NEXT ISSUE OF MICRO:MAG

Grab the attention of thousands of micro:bit community members by Advertising.

GET IN TOUCH WITH OUR TEAM FOR MORE INFO AT:
hello@micromag.cc

We've got reasonable rates available for full and half page advertisements.

DONATE TO MICRO:MAG

HELP US COVER THE COSTS OF MICRO:MAG

We are run by a team of passionate community volunteers, but we still need help covering costs.

DONATE TO MICRO:MAG VIA PAYPAL:

micromag.cc/donate

Previous donations have helped us get a designer to design the new layout, cover domain costs and much more!

micromag.cc

£5.99