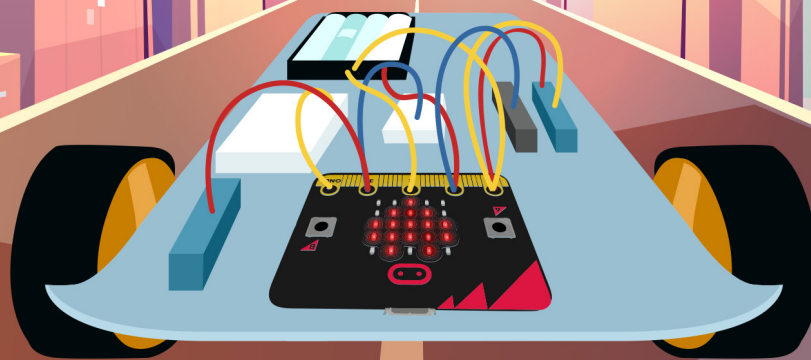


micro
:mag

Issue 6
November 2019

BUILD A ROBOT

Build your own low-cost micro:bit
powered robot



Contents

News

- P4 micro:bit Live 2020 - Call for hosts!
- P5 4tronix's new motor driver board
- P6 Python Editor V2
- P8 Team Update

Features

- P10 3D Printed battery holder + stand
- P12 The creation of Z-Bot
- P14 Meet Avey Couloute - Girls into coding
- P16 Stepping stones towards student activism
- P18 What Pimoroni Learned at Deer Shed Festival

Cover Feature

- P20 Low Cost DIY Robot
- P22 Budget Robotics
- P23 Build the robot
- P25 Code it with MakeCode
- P27 Code it with MicroPython
- P30 Robot Showdown

Makes

- P32 3D Printed Servo Magic 8 Ball
- P34 Learn how to use MakeCode functions
- P36 Create a virtual pet with Tynker

Reviews

- P38 :VIEW text32 LCD
- P39 :ZIP Halo
- P40 Tobbie II
- P41 New Accessories Showcase

BUILD A DIY ROBOT

Cover Feature | P20



Pimoroni + micro:bit At Deershed | P18

function SWIntro

play tone Middle G for 1/4 ▼ beat

rest(ms) 1/8 ▼ beat

play tone Middle G for 1/4 ▼ beat

rest(ms) 1/8 ▼ beat

play tone Middle G for 1/4 ▼ beat

play tone Middle C for 2 ▼ beat

play tone Middle G for 2 ▼ beat

How to use Functions in MakeCode | P34

Hello World



Kerry Kidd
Editor In Chief

Welcome to issue 6 of micro:mag! It has been a busy few months for us here at micro:mag. As we were getting this issue ready for release, the team took a trip to micro:bit LIVE. to show off micro:mag and meet a lot of you lovely people from the micro:bit community in person rather than over email or Twitter! It was great meeting you all and hearing your love for micro:mag. This has made us want to commit more time to the magazine and maybe even make bigger and better issues. Anyway enough about the future, let's take a look at what we have in store for you in Issue 6!

Within this issue, you can make a cheap micro:bit robot using our cover feature produced by Les Pounder. Thanks Les! As well as this you can read about Pimoroni's do's and don'ts for running workshops within a tent in the middle of a field, make a magic 8 ball using a servo and a 3d printed bracket, plus hear about Avey's mission to inspire more girls into STEM.

As well as all that we also have some news for you about the new python editor & the new motor controller from 4tronix.

On behalf of the micro:mag team enjoy issue 6!

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

- » Les Pounder
- » David Whale
- » Vlastimil Hovan
- » Avey Couloute
- » Helene Virolan
- » Michelle Valtas
- » Dave Del Gobbo
- » Tanya Fish
- » Robert Wiltshire
- » Meridith Ebbs
- » Nicole Parrot
- » Tynker
- » Kitronik
- » Micro:bit Foundation
- » Elenco
- » 4tronix

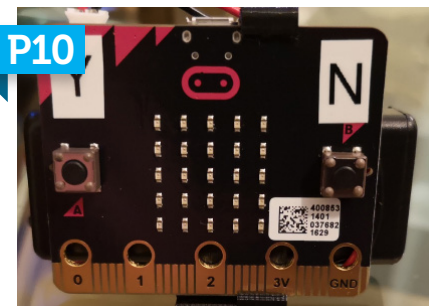
P04



CALL FOR HOSTS! M:B LIVE 2020

Now that micro:bit Live 2019 is over, the foundation are looking for hosts for 2020!

P10



3D PRINTED BATTERY HOLDER

We talk to David & Vlastimil about the 3D printed battery holder & stand.

P40



TOBBIE II REVIEWED

We review a super cool micro:bit powered humanoid robot kit.



After a successful micro:bit Live 2019, the foundation are looking for hosts of micro:bit Live 2020 regional events!

Call for hosts!

ABOVE:
micro:bit Live.
Credit: Les
Pounder

Last month, over 200 micro:bit community members gathered together at the BBC in Media City, Salford for two days of micro:bit themed talks, workshops and even parties.

The micro:mag team got the chance to attend the event and enjoyed every part of it, so did all the attendees. The Micro:bit Educational Foundation are excited to announce their plans for micro:bit Live 2020 which will be a different format to this years event. The foundation want people from around the globe to have the chance to attend micro:bit Live, however, they realise that everyone travelling to Salford isn't possible. The

idea for micro:bit Live 2020 is to have regional events around the world under the "micro:bit Live" banner and they want the community to offer to host an event. If you think this is something you'd be interested in knowing more about, you can fill in the form linked on the right to get added to the mailing list. We can't wait to see where the event will go next year and hopefully attend some of them! If you're interested in reading about micro:bit Live 2019, we've got a micro:bit Live special coming up next issue so make sure you're subscribed to the magazine to be the first to know when Issue 7 is released.

Fill in the form below to hear back from the foundation about hosting a micro:bit

Live 2020 event:
bit.ly/microbit2020



MINIBIT



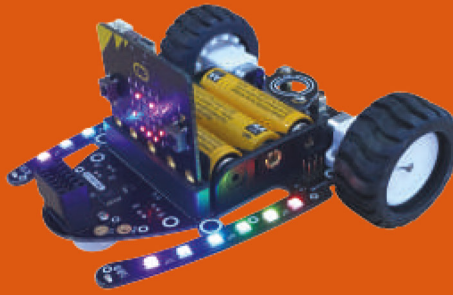
£20.80

PLAYGROUND KITS



FROM
£24.00

BITBOT XL



£35.00

ROBOBIT MK3



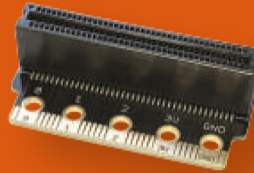
£32.00

BIT: COMMANDER



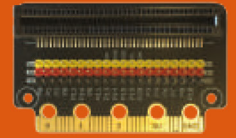
£16.50

ANGLE:BIT



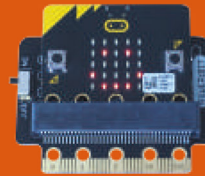
£4.15

BIT:ZERO



£4.50

POWER:BIT



£5.00

BIT:170



£5.50

4tronix drive:bit

By Les Pounder

Some hot news that has just made the deadline for this issue! 4tronix have just released a new robot board for micro:bit.

Drive:bit is a low cost, small profile motor controller based around the popular DRV8833 chip. It features a slot to connect and power the micro:bit and two outputs for motors. There are also connections for other micro:bit GPIO pins and a single neopixel LED. Both the motors and micro:bit can be powered from a single 7-10V power source which makes for a tidy and hassle free robot build. To code Drive:bit there is

a MakeCode extension which will enable any coder to quickly build their first robot. For the Python programmers amongst you, it can also be controlled using MicroPython.

In this issue we covered building a robot as cheaply as possible, and with Drive:bit we can build a cost effective and simple robot for only a little more money. Drive:bit would be an ideal platform for schools with limited budgets and space wishing to build motorised micro:bit based projects to ignite the spark of imagination.

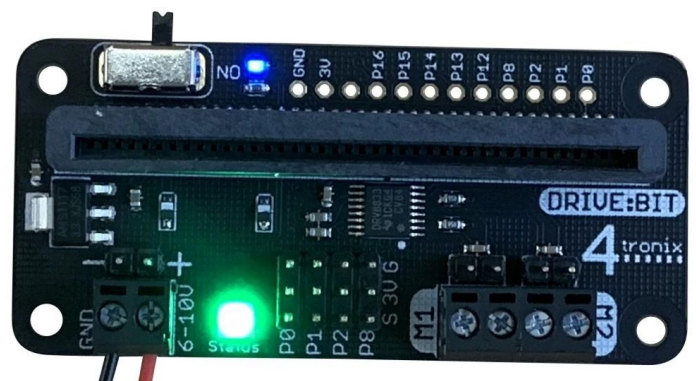
Drive:bit is smaller and lower in cost than other motor control boards, and this means

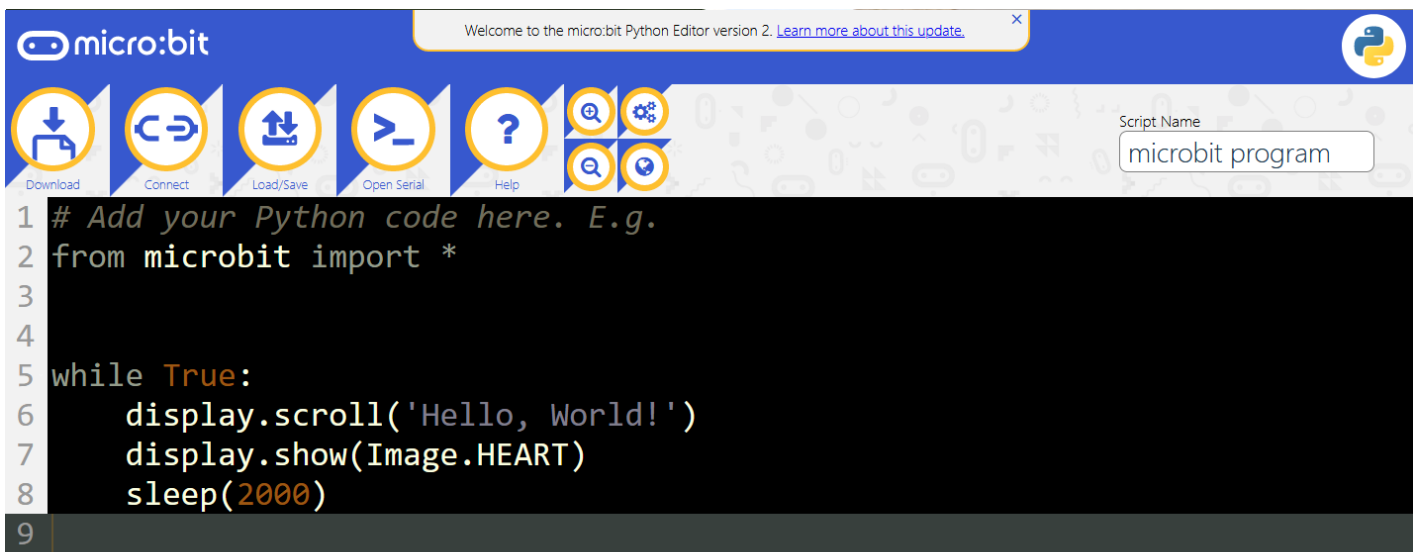
that our robots can be smaller and more cost effective.

Drive:bit is available as a black and gold PCB for £9 from the

link below and there will be a full review in the next issue of micro:mag, so don't miss it!

go.micromag.cc/drivebit

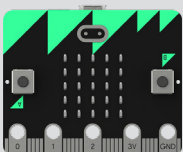




PYTHON EDITOR V2 RELEASED!

The Official Python Editor for the micro:bit has just got an update.
By **Micro:bit Educational Foundation.**

About the Author



The Micro:bit Educational Foundation is a not for profit organisation. Our vision is to inspire every child to create their best digital future.

The Foundation was legally established with the support of our founding members in September 2016.

microbit.org

The October 2019 release of the micro:bit Python Editor (python.microbit.org) brings with it a number of new features and UI changes. This release is version 2 of the editor. We are currently testing this version out in the beta editor python.microbit.org/v/beta before moving this version of the editor live at

python.microbit.org.

Previous programs still work

You are still be able to import .Hex files and Python scripts created in the previous version of the editor and use them in the latest version. Load them into the editor by drag-and-drop, or using the file picker and simply download them again to ensure they are up to date. The underlying version of MicroPython

in the v2 editor has not changed from the current live editor, so all programs will continue to work in the same way.

Access the old editor

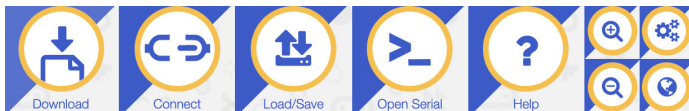
Should you need to, you can always access the previous version of the editor, which will be available for the next two years at

python.microbit.org/v1

The Menu

The updated menu consolidates the Load/Save buttons into one place that also provides access to the new File System. You will also see options to connect and interact with your micro:bit via the serial interface.

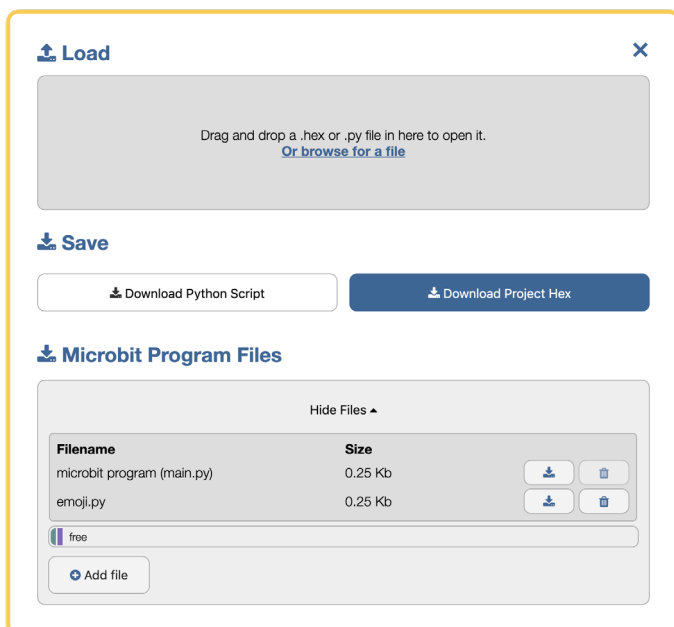
These are enabled if you're using a browser that supports WebUSB, like Google Chrome and you have a recent version of the firmware on your micro:bit. If you are using a browser that doesn't support WebUSB, a message will tell you that these features are unavailable.



Above:

The new Python Editor menu with updated icons

The File system



When the micro:bit is flashed with MicroPython, it enables a simple file system to store files on device. Selecting the load button in the menu will bring up the files modal window and you are able to inspect the files on your micro:bit. Initially this will just be your program script main.py, but you can also add modules here too that can be called from your program. This makes using python libraries for micro:bit add on boards much easier to use with the online Python editor.

WebUSB

When using a browser that supports WebUSB, like Google Chrome or the latest Edge Beta, the options for connecting to and interacting with your micro:bit in the browser will be enabled.

The Connect button allows you to connect to your micro:bit via webUSB. When you choose to connect, a window will pop up in your

browser asking you to choose the micro:bit, you want to connect to.

If you see the message 'no compatible devices found', you may need to update the micro:bit firmware. When you are connected you will see this menu item change to 'Disconnect'.

The Open/Close Serial button allows you to dynamically interact with MicroPython on the micro:bit once you have entered the REPL. You will need to flash the micro:bit with a MicroPython hex in order to use the REPL.

The Read, Evaluate, Print Loop(REPL) is a way of dynamically interacting with the micro:bit using MicroPython. To interact with the REPL on the micro:bit, when you Open Serial you will be asked to 'Click here or press CTRL-C to enter the REPL'. Once you have done so, try typing something in MicroPython. For example:

```
>>> help()

>>> from microbit import *

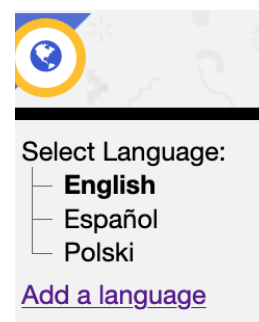
>>> display.scroll("Hello from the REPL")

>>> import this

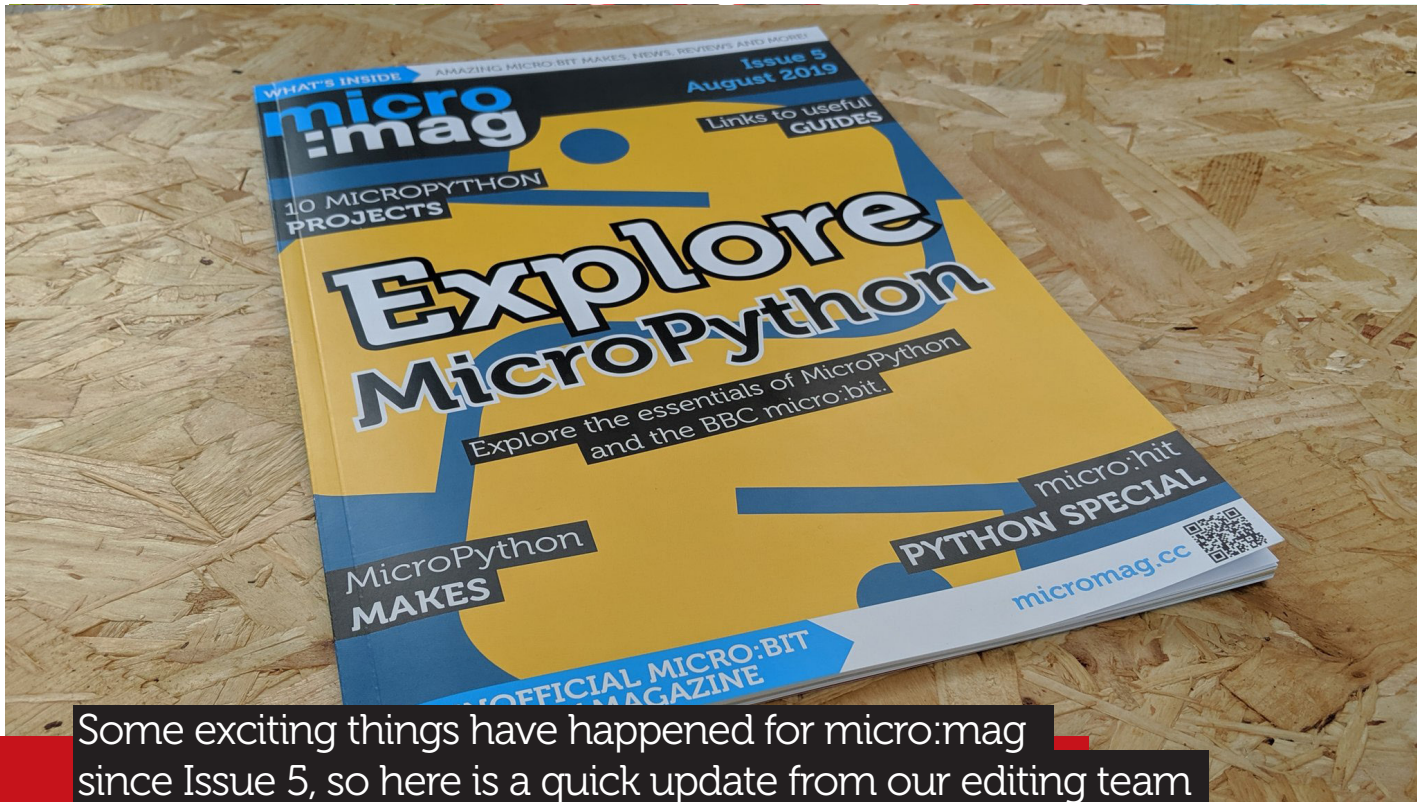
>>> import love

>>> import antigravity
```

Translations



From this menu you have the option to select a language for the editor. The functionality here is demonstrating the capability to load translations and as yet there are only basic examples in place. Learn more on how to do this at go.micromag.cc/pyeditortranslate



Some exciting things have happened for micro:mag since Issue 5, so here is a quick update from our editing team

Team Update

ABOVE:
A printed copy of Issue 5. Buy one today!

Since the launch of Issue 5, some really exciting things have happened in the world of micro:mag which we'd like to make our readers aware of. We hope you are as excited as we are to see the magazine grow from a small volunteer project to a whole community effort. As always, we couldn't do this without your support. Here's a few updates from the team about what's happened since Issue 5!

micro:mag CIC

When we started the magazine, we had no real idea of how it would take off. It was hard to imagine us getting past Issue 1. As we started to grow and do new things (like bring on advertisers), we soon realized that we needed to register a business here in the UK. As of the 1st October 2019, we are now a "Community Interest

Company" which basically means that what we do is for the benefit of the community and we are committed to investing in it. We're really excited by registering micro:mag CIC and see this as a big step forward for the magazine.

Printed Copies

You may or may not have seen on Twitter that we started selling the micro:mag as a physical magazine on our website. The first run went really well and we learnt a lot. We're proud to announce that pre orders for physical copies will be available for every future issue for a 4 week period from release date. We did have some printing issue on the first run, but we managed to sort them out as fast as possible to make sure that everyone got a quality product. You'll be able to buy your own printed copy of micro:mag as well as the occasional

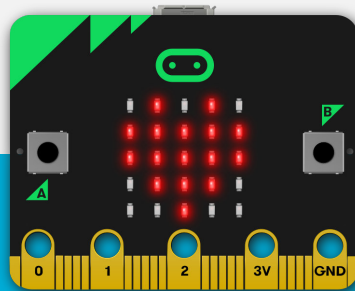
special item on our brand new store website over at micromagstore.com.

Next Issue & The Future

Next Issue's special cover feature will be from micro:bit manufacturers Farnell where they'll be sharing how the micro:bit is made and what is involved in the process. We're really looking forward to sharing this exclusive article with you in December when Issue 7 is released. Our team have some exciting plans for other things we'd like to do in 2020 to help the magazine grow even more. 2019 has been an amazing year for micro:mag and we're glad you've been with us on the journey. We can't wait to see where 2020 takes us and the exciting micro:bit content we'll publish in the magazine. Once again, thank you for reading micro:mag.

do your :bit

A micro:bit digital challenge for the Global Goals



A micro:bit challenge for the Global Goals for young people across the world.

Visit doyourbit.microbit.org for inspiration, teaching resources and information on how to take part.

Co-creators



In collaboration with





3D PRINTED MICRO:BIT BATTERY HOLDER AND STAND

We interviewed **David Whale** and **Vlastimil Hovan** on why the battery holder and stand was developed

About the Author



David Whale

David is an embedded software engineer, book author, book and magazine editor, general micro:bit Wizard.

@whaleygeek

micro:mag: We understand you've designed a 3D printed battery holder for the micro:bit – can you tell us a bit about it?

David Whale: Well, Vlastimil did all the hard work, I was just the first user of the battery holder. It's great because it slides securely onto the micro:bit holds a standard battery pack on the back and even has a slot for the battery wire to stop it getting tangled.

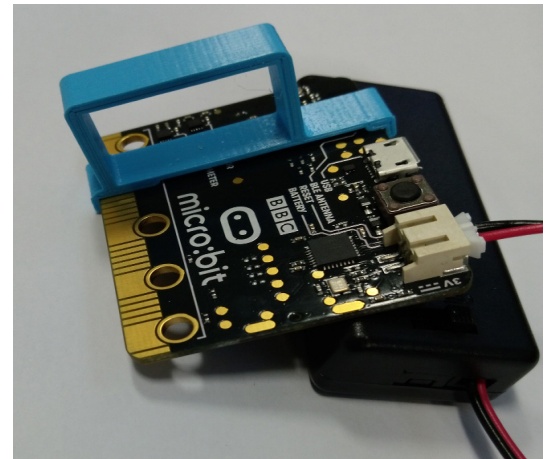
Vlastimil Hovan: I first got into 3D design after I bought a 3D printer. I used my new skills to design the first micro:bit battery pack holder, using a free package called "Design Spark Mechanical" from RS Components.

m:m: Where did the idea for the design come from?

DW: I had been invited by BBCClick to build a 300 micro:bit orchestra for their live stage show in 2017, and I needed a fast and simple way to fix 300 battery packs! I tweeted the challenge and had many good (and strange!) replies over a few days. I used elastic bands in the end, but Vlastimil also had this innovative idea for a 3D printed clip.

VH: I saw the challenge posted by David and got my gears in motion. It was a good way to practice my newly developed 3D design skills, and as micro:bit user I shared the same issue as to what to do with the battery pack. I enjoy a challenge and after a little while, the first battery pack holder was born. The first design took 15 minutes to print but the latest model takes about 60 minutes.

DW: ... and I didn't have my own 3D printer so Vlastimil printed some and posted them to me!



ABOVE:

The original blue holder

m:m: Fantastic! How has the clip been developed since?

VH: After the first design, I kept on thinking about different ways that the micro:bit is used, and came up with the idea of an electronic badge. That's when I created a different version of the holder with a hole at the top for a lanyard clip.

DW: ... don't forget the stand feature!

VH: oh yes! One night I was balancing the holder on my table and thought if I make the bottom of the holder the correct angle it will stand up on its own, and version 3 was born.

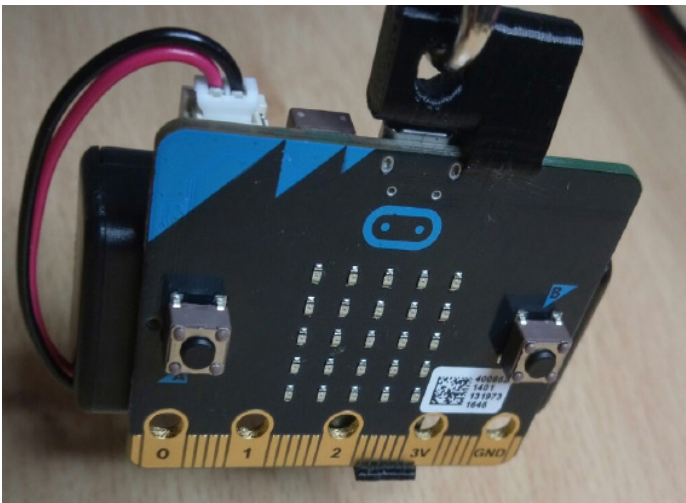
About the Author



Vlastimil Hovan

Vlastimil is a full-time college lecturer in electronics, loves designing electronic circuits, tinkering with software, and 3D printing anything and everything.

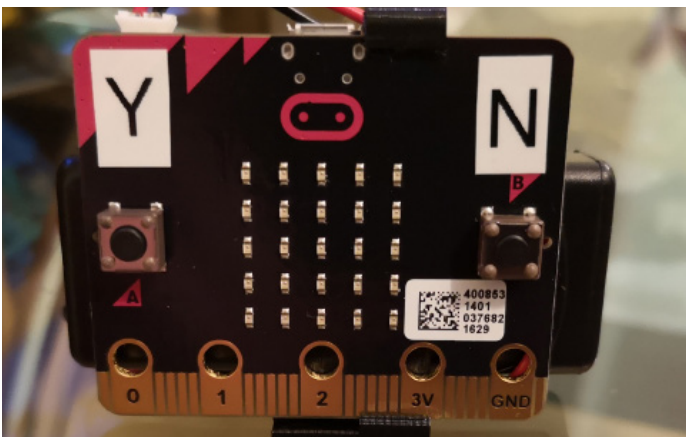
@Vlastimi_Hovan



Above:
Added lanyard-clip feature

m:m: How many different generations of the battery holder are there?

VH: There are around five different designs but even I am losing track. The great thing about 3D printing is that you can re-iterate a design many times quickly to perfect it.



Above:
Fig 3: Now self-balancing

DW: They make great prizes or handouts at events, the school teachers I work with love them, and the students find them really useful too!

VH: After great feedback from various users I looked at all the different designs and thought hard on how to combine them together into a single combined design, which became the final holder version. It can be clipped on a lanyard, stand on its own on a flat surface, and hold the battery pack and micro:bit securely in place secured by a single grub screw. Also note how the USB port is not obstructed, so that you can upload software without moving the holder, and a refinement that allows for crocodile clips to still be attached. Last but

not least, there is a slot for a strap, so it can be turned into a wearable/watch.



Above:
Line-up of all the versions

DW: Yes, Vlastimil keeps refining the design, it's great that all the ideas have made it into a single integrated product now.

m:m: That's great! How do you know each other?

VH: Via Twitter!

DW: Vlastimil and I are 'virtual friends' – we've never actually met in real life. But it shows the power of the community and social media, we've become great friends over the last year!

VH: Yes, we chat on twitter several times a week, even though we are at opposite ends of the country.

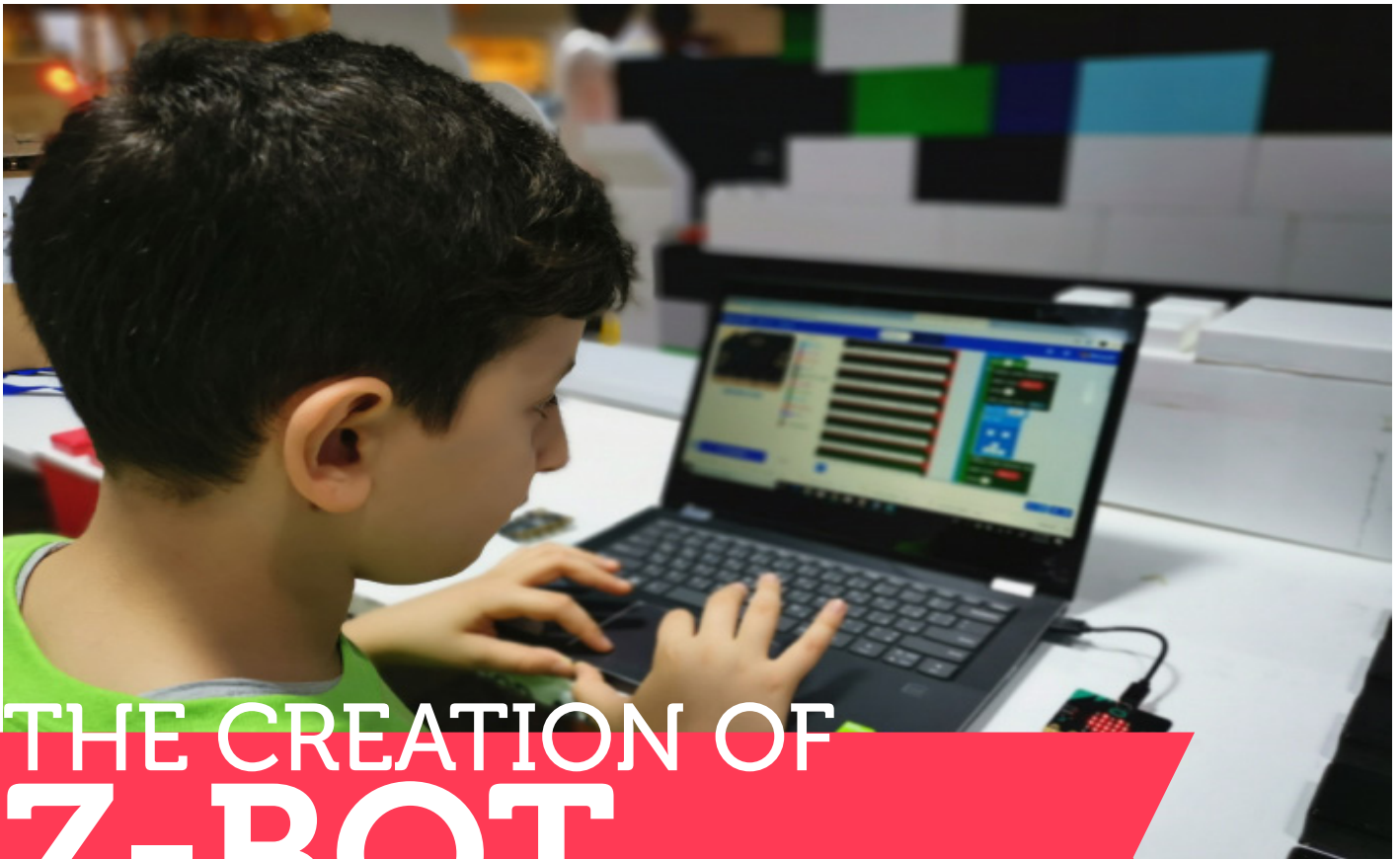
m:m: What's next, do you think?

VH: I've been designing a 3D printed tray that holds a class-set of micro:bits with battery holders fitted. I'm hoping that school teachers will find it useful to keep everything organised.

DW: I think customised engraving on the back would be great for making 'show specials' for handing out at events; who knows, there might be a new craze with people collecting them all. Oh, and one day, I hope to meet Vlastimil in real life and shake his hand in thanks for all his great work!

m:m: Where can our readers get hold of the battery holders?

VH: If your school or college has a 3D printer, you can get the design of the older version of the battery holder from Thingiverse <https://www.thingiverse.com/thing:2666671> I am currently looking at how to develop the latest model into a product that can be widely available; but please get in contact (e.g. via Twitter) as I do sell them directly at the moment.



THE CREATION OF Z-BOT

The Humanoid robot that can move and talk. By Zayd Nashed.

About the Author



I is 11 years old, from Syria living in Saudi Arabia

He started in electronics and coding when he was 9 years' old

I won the micro: bit Global Challenge for the Middle East region in 2018.

This summer, I volunteered in a tech camp in Saudi Arabia, which aims to teach kids electronics and coding to achieve the targets of global goals.

During that time, I was working on my own project, which was called Z-Bot; it was a humanoid robot that can move and talk by using the micro:bit

Therefore, I decided to make my robot to guide and introduce the global goals for kids.

What I Used

- 3 x Card board for the head, body, and the front box
- 2 x MAX7219 LED Matrix
- 3 x servo motor, one for the mouth, and two for the neck
- 1x Kitten bot extension Keypad
- 4 x Microbits
- 1x Speaker

What Z-bot Does

The main idea of this project is that Z-bot has a keypad in front of it, so when anyone presses any global goal number, the robot will move it's head and mouth, in addition, to say the global goal name and scroll it on a long LED Matrix.

Link for Z-bot's video go.micromag.cc/zbotvid

How Z-bot Works

I used four micro:bits three of them connected together by radio, while the last one was an individual micro:bit and each micro:bit has different tasks.

The first one is the main micro:bit, which is connected to the keypad and knows what global goal number has been pressed; it is also connected to three servos to move the mouth and the head, I used MakeCode to code it. The second micro:bit is a speech one and coded by MicroPython, its task is to speak the global goal name.

The third micro:bit is connected to an LED Matrix to scroll the goals name on the robot's body. I used



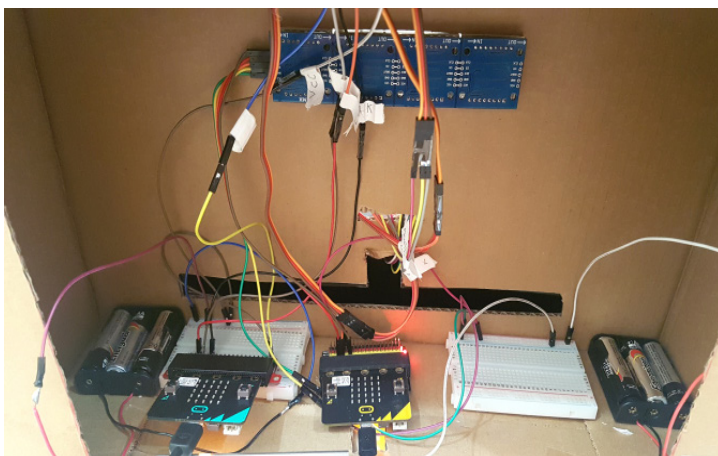
Left:

Z-bot on display

MakeCode to code it too.

While the last one was a separate micro:bit, which is connected to the LED Matrix for eyes. I also used MakeCode to code it.

When any number was pressed on the keypad, the first micro:bit will move the servos (mouth and head) and send two radio notifications, one for the second micro:bit to say the global goal name and the other notification to the third micro:bit to scroll the global goals name on an LED Matrix.



Above:

micro:bits all wired up within Z-bot

What problems I faced

I faced many problems during the work on this robot, one of them was when I tried to send a message from blocks to python, the python wasn't able to understand any character from blocks instead it showed me a bunch of question marks, and the solution was a piece of code needed to be written before the "on start".

Another problem I faced was when he tried to add some additional words to explain the global goals, the micro:bit showed the number 844 and then a sad face, which mean that is an error in the memory; the solution for this problem was to make the code shorter by cancelling some words.

Conclusion

To conclude I thought it would be difficult to make when I only had the thought, but it turned out to be super easy after hard work, searching and never giving up, and that's how I ended up with my amazing project: Z-Bot.

```

1 from microbit import *
2 import radio
3
4 radio.config(group=132)
5 radio.on()
6
7 def get_message():
8     while True:
9         try:
10            msg = radio.receive_bytes()
11            if msg is not None:
12                if len(msg) >= 13 and msg == 2:
13                    lstr = msg[12] #length byte
14                    text = str(msg[13:13+str], 'ascii')
15                    return text

```



Avey Couloute

INTERVIEW

This issue, we got the chance to interview girls in tech pioneer Avey Couloute whose mission is to inspire more girls to choose a career in STEM subjects. Recently, Avey has started her own business, Girls into Coding, where she organises regular events in London for girls to attend and learn how to code through a number of talks and workshops. In this interview, we learn more about these events and how the micro:bit plays a big part in them.

micro:mag: Thanks for letting us interview you for micro:mag, please tell us a bit about yourself!

Avye: I'm Avye, I've just turned 12 years old. I love coding & making stuff and I'm on a mission to inspire more girls to get into coding and technology. In my spare time, I lead regular coding & physical computing workshops for young people and I enjoy playing football & swimming with my local swim squad.

MM: What made you get into coding?

AV: When I was 7 years old, I started going to coding clubs and workshops which introduced me to scratch & then the micro:bit. I loved how I could use code to move things on the screen or to control electronic components.

MM: You inspire lots of young people in the UK through your "Girls into Coding" events. Please tell us about them.

AV: Well, in the past girls and women have been under-represented in STEM and this generation has a chance to change that. For me, "Girls Into Coding" is my way of contributing to that change. The events are free to attend and provide an opportunity for girls aged 10 -14 to explore coding and physical computing in a supportive environment. As well as participating in hands-on workshops, the events give the girls an opportunity to listen to lightning talks throughout the day, delivered by inspiring female role models who are doing cool stuff in the tech world. With support from members of the tech & maker community, the girls receive a really positive experience and get a micro:bit & a

physical computing starter kit, so they can continue their tech journey at home and beyond.

MM: What do you like the most about the micro:bit?

AV: I love the size of the micro:bit and the number of things it's got packed into it. It's also compatible with so many different add-ons and extensions.

MM: What's your favourite project that you've made with the micro:bit?

AV: A set of robot kits which I jointly developed for my last girls into coding event. The kits used proximity sensors, LEDs, motors, buzzers & radio controllers. The idea for the project was a new challenge for myself because the kits were designed so that all of the main components could be slotted on to the robot chassis without the use of tools.

MM: Have you got any future plans for micro:bit related projects?

AV: I am working on ideas for my next Girls into Coding workshop; I'm quite interested in developing a micro:bit robotics activity using the new kitronik servo & motor driver board. I think it will be a good way to bring servos & motors together in one project. I have a couple of other things in the pipeline which I'll soon be able to say a bit more about.

Learn more about Avye's work:

Blog: 10tonolimit.com

Twitter: [@helenevirolan](https://twitter.com/helenevirolan)

Microsoft Video: go.micromag.cc/avyemsft



Above:

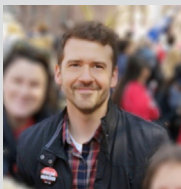
Avye's July Girls into coding event at Microsoft



STEPPING STONES TOWARDS STUDENT ACTIVISM

Dave Del Gobbo tells us about Social issues in micro:bit Board Games

About the Author



Dave is a Canadian science and special education teacher at Peel District School Board. He enjoys 3D printing, gaming in education and giant espresso machines.

@DaveDelGobbo
@FCLedu

Right:

The game featured the micro:bit displaying a custom victory message.

Critical thinking, collaboration and creativity have been identified as important skills for modern students to attain. In this project, students leverage these skills, while teaching fellow classmates about complex social issues via self-created multimodal board games.

This article has been submitted by Fair Chance Learning on behalf of Dave. We've been a fan of Dave's work for some time. We value our relationship and the leadership he offers the education space.

General Learning Strategies (GLE) is a special course offered to Ontario secondary students with Individualized Education Plans. In these classrooms, you will find students identified with a wide range of exceptionalities working to enhance their literacy and numeracy skills. My challenge has been to design projects that meet my students' personal needs, while also providing opportunities for them to find their own voice.

As I explored options for what to include in my GLE course, coding was selected as a vehicle to explore critical thinking and numeracy goals. Initially, students worked on an introductory set of lessons based on the great work of Douglas & Mary Kiang. Coding on the micro:bit allows students to iterate quickly and 'fail for-

ward', so that they strive to be conscientious. After only a week of learning of computer science concepts, a pair of students developed a tribute to Nintendo's 1984 classic "Duck Hunt"; view the video here:

Our final class project was inspired by the work of Professor Larry Bencze, a University of Toronto professor who encourages students to take an activist stance on social issues. Students were asked to create micro:bit based board games that revolved around an issue important to them. Once again, I was impressed by the student's ingenuity in creating games that showed both technical sophistication and a nuanced understanding of their issue.

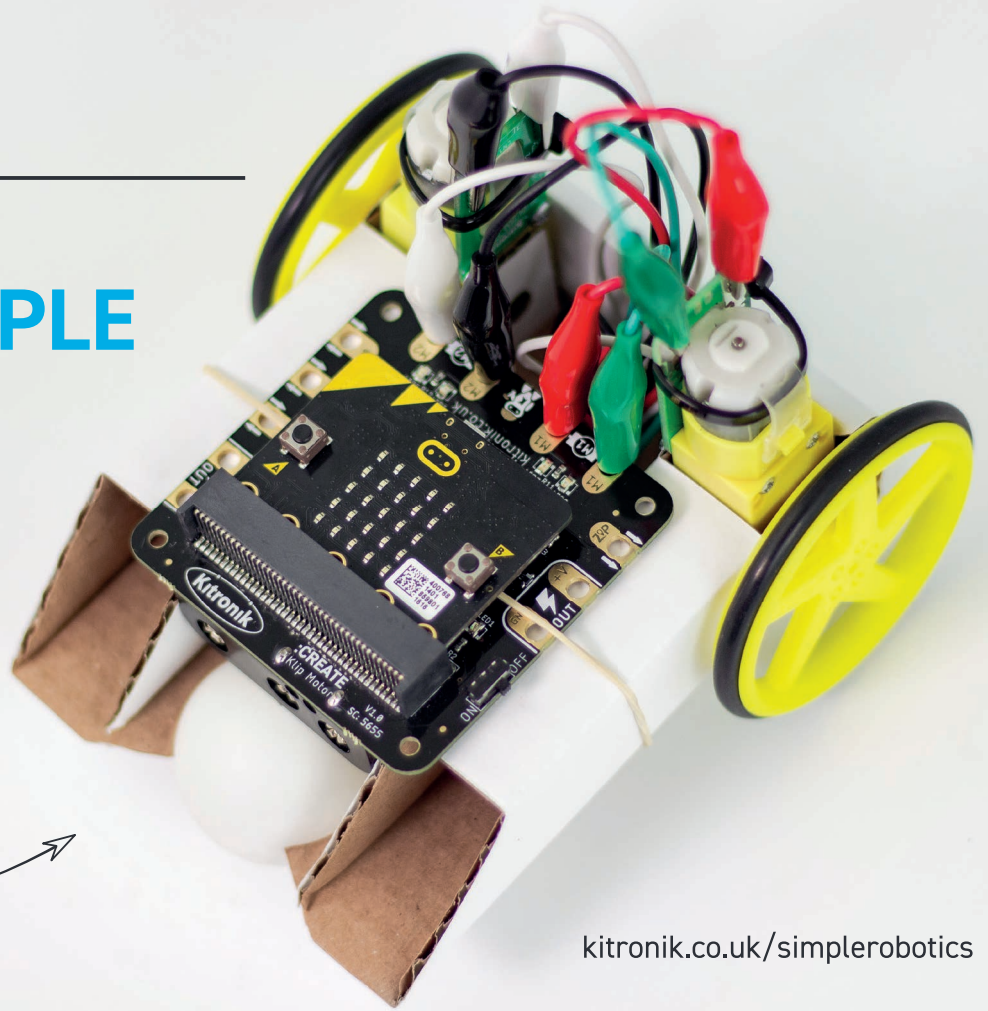
The game creator also 3D scanned (using a Kinect Sensor) and 3D printed a small version of himself to act as the playing piece, essentially putting 'himself' in the game. These projects highlight the micro:bit's key advantage: it allows abstract computer science concepts to come alive in a physical device a student can interact with, untethered to the confines of a screen.





ROBOTICS MADE SIMPLE

for less than £25 (Ex. VAT)



*the simple
robotics kit*

kitronik.co.uk/simplerobotics

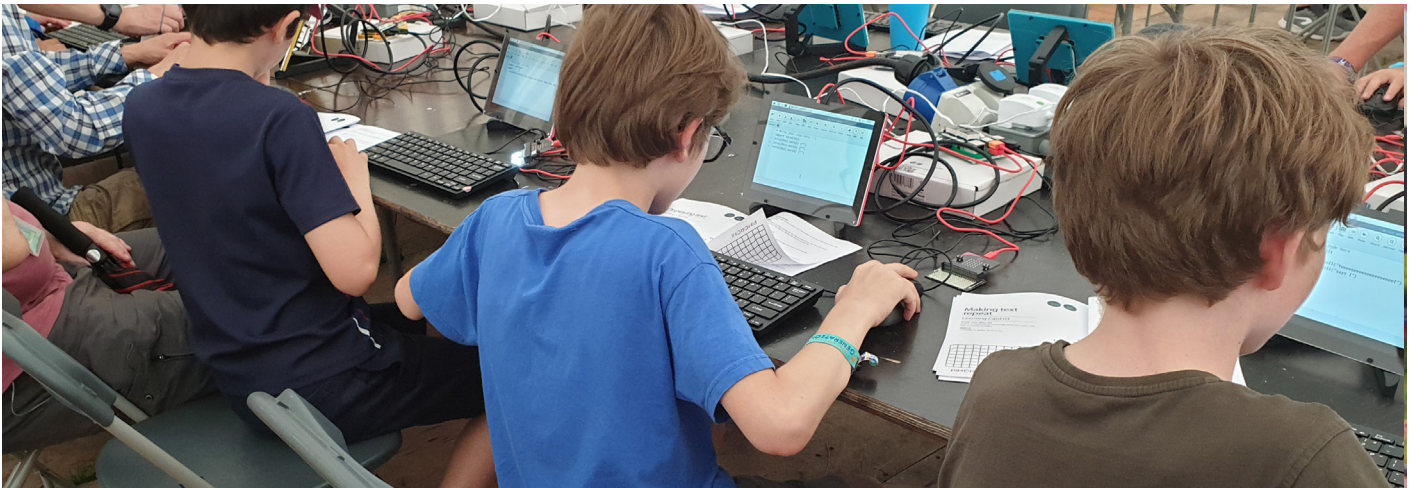
SUBSCRIBE TO EMAIL UPDATES

FRESH COPIES IN YOUR INBOX



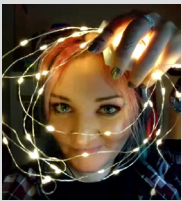
SIGN UP FOR FREE AT:
MICROMAG.CC/EMAIL

We'll only contact you a few times a year about new issues and micro:mag news



WHAT PIMORONI LEARNED FROM THEIR FESTIVAL WORKSHOP

About the Author



Tanya has been a Pimoroni pirate since 2016 - making learning materials for schools, running workshops, & doing talks. Often to be found in possession of cardboard and glitter.

Twitter: @tanurai

Web: tanyafish.com

Above:

The queue of people waiting to get in and sign up stretched back as far as the eye could see

Deer shed Festival (deershedfestival.com) is a family festival that attracts over 10,000 people for music, arts, science and spoken word based fun. We were invited to run a coding workshop in the science tent last year, and then this year we were invited back (so we must have done something right)!

In this article, we'll share lessons learned, top tips, and how to prepare for the onslaught of sign-ups. We decided that we'd teach MicroPython by using a micro:bit and an add-on board (the Pimoroni scroll:bit display).



The first thing to note about working at a festival is that you're very limited in space, resources, and climate

control. We learned from the previous year that we would have to carry all of the resources across a bumpy grass field. To add to the fun, it was raining when we unloaded. We used Really Useful boxes with clip lids and a trolley. The sealable clip lids kept everything dry and it meant that we could stack the boxes. Taking no chances, inside each box were more boxes like little technical matryoshkas. If you want to stop them sliding around inside the box, a layer of packing foam with slots cut in it is a good way to store things.



Above:

Safely packed micro:bits

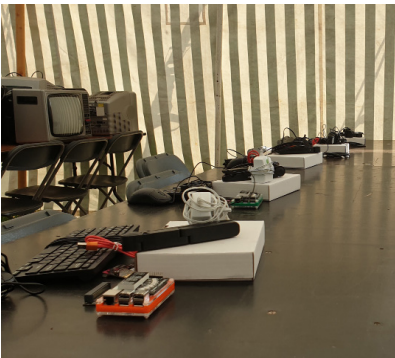
When you're staying in a tent yourself, there is not really an option to rework things if you have problems on site. If you are running Raspberry Pi with SD cards, it is a good idea to take some spare ones that are already prepared with software in case of corruption. We made sure ours had the Mu editor pre-loaded, along with the scroll:bit libraries (because there's no internet in the middle of nowhere - more about that later).



Left:

Prepared SD card

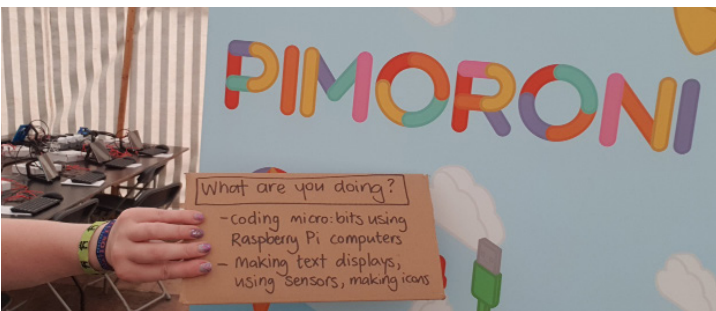
Again, due to the way that you have to store and carry all of your materials, more streamlined it is the better. We designed some tiny portable screens for the occasion and you can see that these pack into small white boxes. Monitors are just not an option due to the bulkiness and not being stackable. For the same reason, we chose Raspberry Pi, as it was easy to take a pair of backups without taking up too much space. It's also a good idea to take spare mice - we took two full sets of equipment spare. Stuff *will* fail over the course of a long weekend of muddy, rainy, suncream-covered children.



Above:

Equipment laid out ready

In a marquee, you have to be aware of leaks in the event of rain or other inclement weather. Never set up right next to the tent wall. We found this out the hard way when a deluge started and a steady drip drip drip of water started plopping down onto one of our unfortunate participants. Also, if you have ever been camping, you'll know that things have a tendency to get damp overnight, and this had an adverse effect on our printed materials. Store paper and other moisture-sensitive materials in an airtight box overnight, if possible, instead of leaving them out with the equipment.

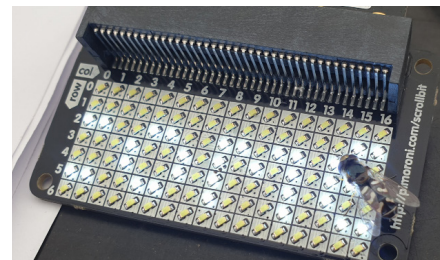


Although the venue provides signs at a low level, it can be difficult

to see when the venue is full of people so make sure you take something above eye level for people to spot easily (we used the banner). Another thing to consider is that your audience is very mixed. People bring tiny babies right up to grandparents, and it helps to think about your audience. Think about what skill level they are at, and what other skills will be needed such as reading level or motor skills like using a mouse or typing. We decided to set a limit on ours, based on a reading age of eight. We also had examples of the materials at signup, so that parents could judge for themselves whether their child was able to work independently. We write our materials specifically with this event in mind. When the tent is full it is excessively noisy, constantly. This means that you are unlikely to be able to teach like a classroom environment, and instead, it has to be self-led and have troubleshooting built-in.

We chose to do a booklet with stand-alone tasks that did not necessarily build on the previous task. This meant that if children wanted, they could just do the tasks that interested them and skip the others, without any loss of progress. We tried to anticipate problems, and added a troubleshooting box to the bottom of each activity. In addition to this, we added extension tasks so that each task could be expanded if the child wanted to or they could skip straight onto the next task. We aimed at having half an hour of activities so that we could have 120 children through during each day.

Another consideration we had was that in the middle of a field, there is no Wi-Fi. No Wi-Fi means everything had to be off-line. As mentioned before, this means you need to do all the updates and installations *before* you head off to the festival. Out of the off-line editors we could use, we chose an editor called Mu. The reasoning was that we were able to change font sizes and theme for accessibility reasons, and the simplicity of having recognisable buttons such as save, flash, check, and quit, saved on the explanation of the interface to people.



Above:

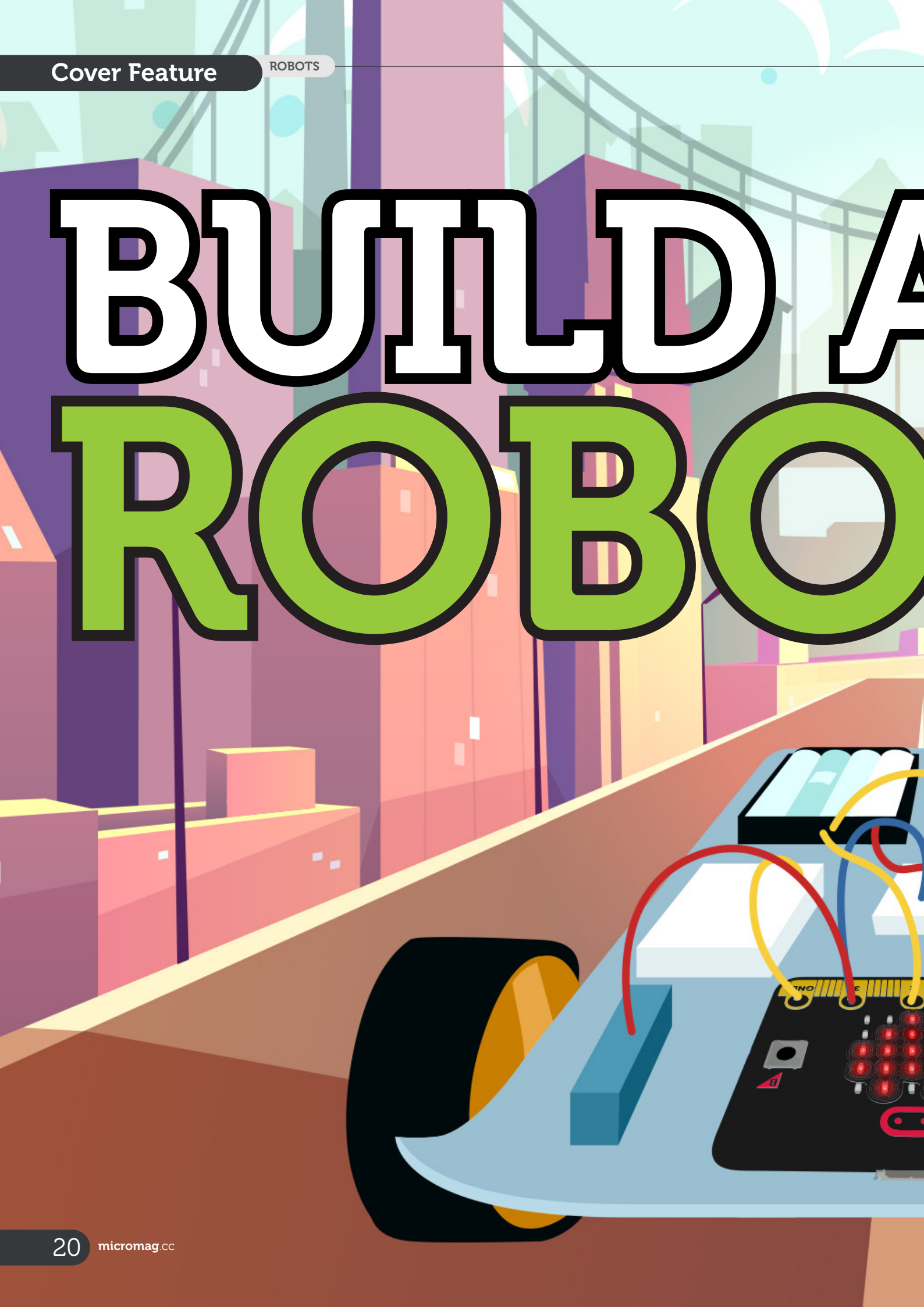
A bug during coding

We wanted to use MicroPython, to show that the micro:bit can be coded in ways that are not just block coding. It also meant that printed materials did not have to be in colour, which was a consideration when releasing them to teachers as not everybody has access to a colour printer.

All in all, the workshops went smoothly. We did have a few bugs, but they were of the hoverfly variety, attracted to the lights on the scroll:bit. Having a clear explanation of what the workshops were about meant that our audience was largely self-selected and able to access the materials.

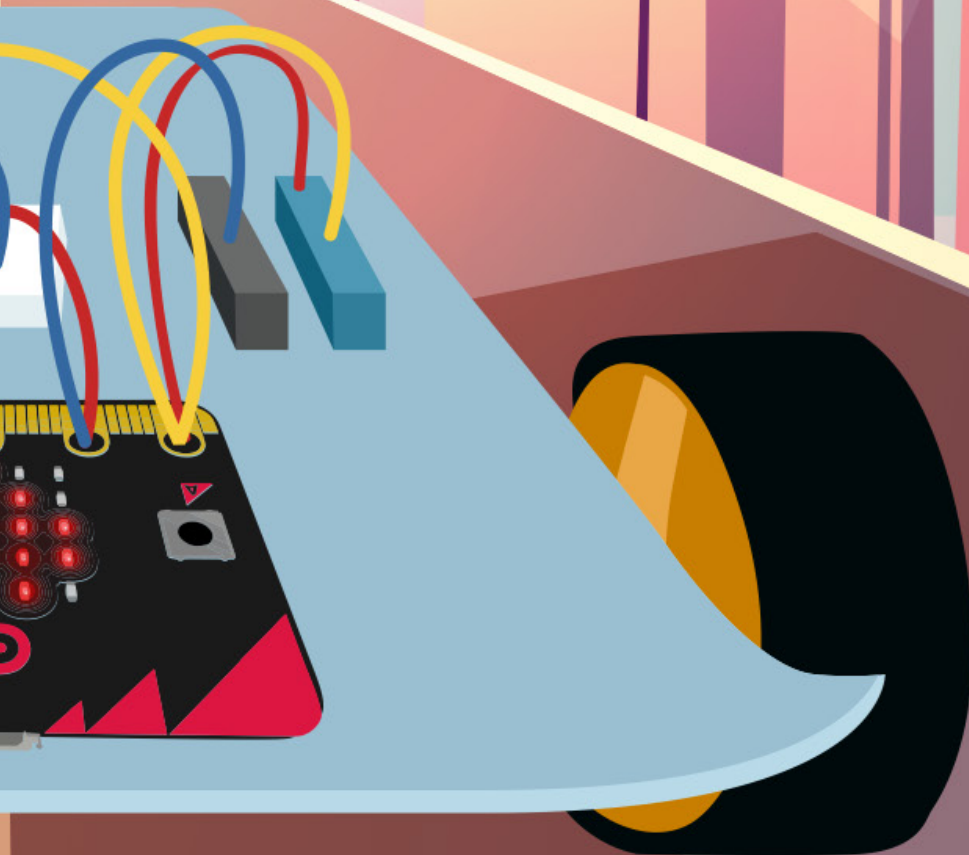
The materials are now available at edu.pimoroni.com, for download as a PDF in black and white. We can truly say that these were destruct tested in the harshest of environments and next year we hope to go back with more materials, more workstations, and even more fun stuff to tinker with.

BUILD A ROBO



Les Pounder shows us how the micro:bit can be built into a cost effective robot for beginners

One of the most common projects people build with the micro:bit is a robot. The micro:bit makes this super simple both on the hardware and software side of things. The majority go out and purchase a prebuilt robot that makes it super easy to get started with robotics and the micro:bit. However, most of the time, this can set you back £40+. For this special issue, we set Creative Technologist and regular micro:mag contributor **Les Pounder** the challenge of building the ultimate low-cost micro:bit powered robot that anyone can build, with just a few parts.

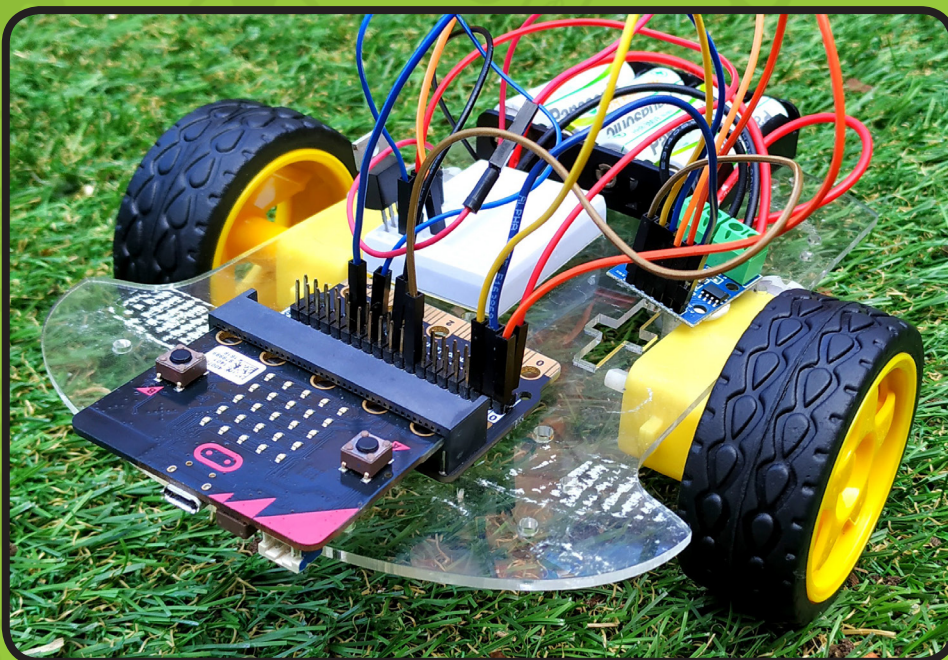


Budget Robotics

Robots are cool and they are even more cool when you make your own. In this feature we take a look at building a really simple budget robot that will provide you with a foundation to learn more about robot building. We shall code the robot in two ways. Firstly using the Makecode editor, and secondly via MicroPython. But before a line of code is written, we need to make sure that we understand what a robot is and what components makes a robot.

A robot will follow instructions blindly, if we tell it to go forward, it will and it will never stop until we tell it to do so. So what does this mean? Well it means that we must be careful when writing and issuing instructions. If we get something wrong then the robot will quickly show us, typically by driving off in any direction.

The components that make a robot are two (or more) motors with wheels. These are typically bright yellow DC motors that can be found cheaply online. But these motors need a controller in order for them to safely work and these controllers come in all manner of configurations. From large industrial precisely calibrated controllers to simple hobbyist level models, every motor needs a controller and there is one for every budget. But what is the brains behind the robot? Well in our case it is the humble micro:bit on to which we will write code that will send signals from the micro:bit GPIO to the controller board, turning the motors on and off to drive in any direction we choose.



Building the robot

Building the robot is where the fun begins!

Parts List:

- » A micro:bit
- » Pimoroni Pinbetween
- » L9110S motor controller
- » LD1117AV33 3.3V Regulator
- » 4 x AA battery box
- » 4 x AA batteries
- » Breadboard
- » 3 x Female to male jumper wires
- » 5 x Female to female jumper wires
- » 1 x Male to male jumper wire
- » A robot chassis (from eBay)

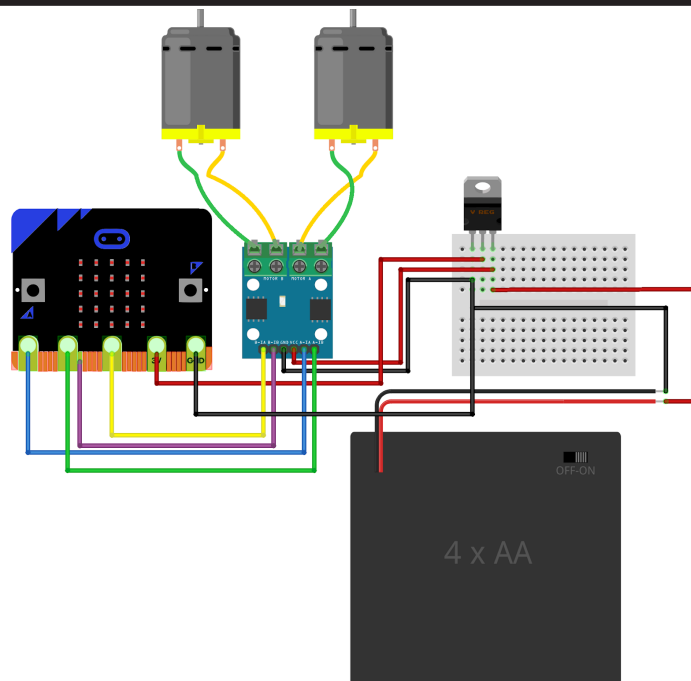


Top Tip:

All of the code and high resolution images / diagrams can be downloaded via our Github repository:



go.micromag.cc/robotgh

1. The build starts by inserting the micro:bit into the pinbetween board. This will open up all of the usable GPIO pins on the micro:bit.



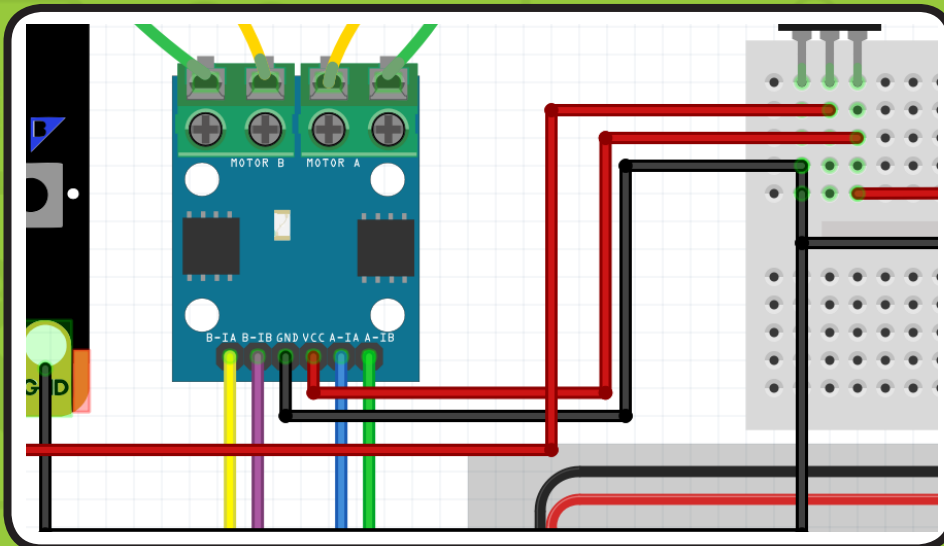
2. The complete circuit sees the micro:bit connect to the L9110S motor controller using female to female jumper wires. This connection will enable the micro:bit to control the L9110S which in turn will control the motors connected to the two screw terminals of the L9110S.

The connections from the micro:bit to L9110S motor controller are:

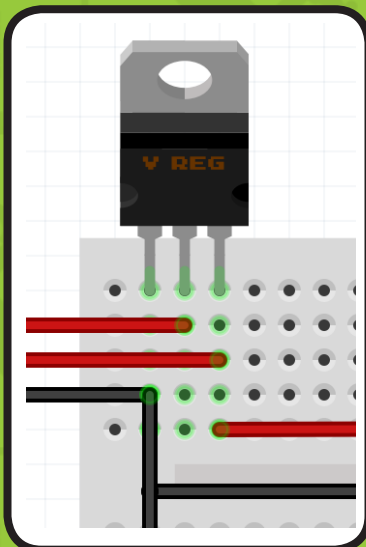
 micro:bit	 L9110S
0	A-1A
1	A-1B
2	B-1A
8	B-1B

Building the robot

Building the robot is where the fun begins!



From the pin between to the L9110S motor controller we need to make a connection between the GND of the micro:bit and the L9110S. This ensures that the two components share a common GND.



To create a single power source to power both the micro:bit and motors we need to use a voltage regulator to drop the 5V from the AA batteries to 3.3V for the micro:bit. Into the breadboard insert the voltage regulator so that the black side of the chip is facing you. Using a female to male jumper wire, connect the first leg (from the left) of the regulator to GND on the micro:bit then connect the GND (black) wire from the battery pack to the same leg. For the second leg connect this to the 3V pin of the micro:bit using another female to male jumper wire. Lastly connect the third leg of the regulator to the VCC pin of the L9110S motor controller and then connect the VCC (red) wire to the same leg. This completes the power supply for the micro:bit and motor controller.

The last step is to connect the motors to the screw terminals of the L9110S motor controller. Don't worry about the polarity, motors do not have a polarity.



Important!

When programming the micro:bit, or powering it from the micro USB lead, remove the VCC wire from the battery to the voltage regulator from the breadboard. But reconnect when the robot is ready to run.

Coding the Robot

With Microsoft MakeCode

1. For this part of the project we shall use the Makecode editor found on the micro:bit website microbit.org.

Create a new project and for now ignore the "on start" and "forever" blocks.

To control the robot we shall create a series of functions that store a series of commands. To use them we simply call the function and the function runs. To make a function we need to go to Advanced >> Functions and click on "Make a Function..."

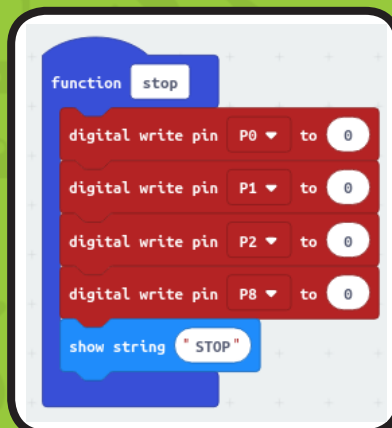
In the Edit Function window, change the name of the function from "doSomething" to "stop" and click Done.

Edit Function

Add a parameter



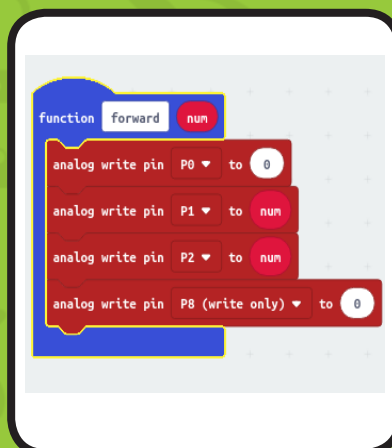
2. The code for the function is made using blocks from the Pins section. Here we see the blocks to turn off pins 0,1,2,8, which will turn off any running motors. Then the "show string" block from Basic will print a STOP message to the LED matrix.



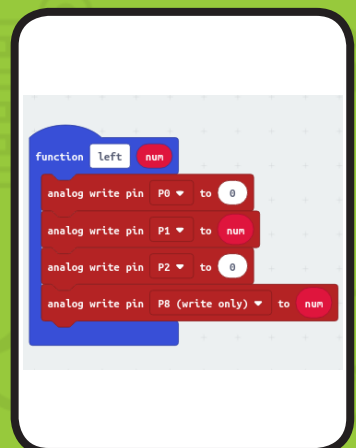
3. Let's make another function, again go to the Functions blocks and make a new function. But this time our "forward" function requires a parameter (an extra instruction) which is going to be a number between 0 and 1023. Click on Number and then Done.



4. To write the code for the forward function we use "analog write pin" from Pins. We number the pins to match those used to control the motor (0,1,2,8) and then set pins 0 and 8 to zero. But pins 1 and 2 will use the parameter number. To use this drag the red "num" block next to "forward" and drop it over the correct space.



5. The process is repeated to create a function to turn the robot left. Note that the pins that will receive the parameter are different as to turn left, we need the right wheel to go forward, and the left wheel to move backwards.



Coding the Robot

With Microsoft MakeCode

```
function right num
  analog write pin P0 to num
  analog write pin P1 to 0
  analog write pin P2 to num
  analog write pin P8 (write only) to 0
```

6. The next function will make the robot turn right

```
function backward num
  analog write pin P0 to num
  analog write pin P1 to 0
  analog write pin P2 to 0
  analog write pin P8 (write only) to num
```

7. The next function will trigger the robot to reverse

```
on start
  call stop
  call forward 512
  pause (ms) 2000
  call backward 512
  pause (ms) 2000
  call left 512
  pause (ms) 2000
  call right 512
  pause (ms) 2000
  call stop
```

8. With the functions created, let's create a short test script that will call each of the functions in turn, so that we can check that our robot is working. Inside "on start" we shall use the "call" blocks from Functions and call each of the functions. We start with "stop", then forward at half speed (512) then use a pause from Basic to force the robot to travel for 2 seconds. And the process carries on with each function call, and pause until we reach the end and add another stop.

Download the code and copy it to your micro:bit. When it has finished, remove the micro USB lead and connect the micro:bit to the 3.3V output of the voltage regulator. Your robot will come to life and start driving around. Check that it follows the correct sequence, forward, backward, left, right. If a motor moves in the wrong direction, unscrew the wires from the motor controller and swap them around. When you are happy, change the sequence of code so that your robot draws a square, navigates around the room, delivers you food! This is your robot, and it will do what you tell it to do!

Coding the Robot

With MicroPython

1. To take this project further we can code the robot using MicroPython using Mu. Mu can be downloaded by visiting <https://codewith.mu/> and installing for your operating system.

We start writing the code for the robot by importing the micro:bit library and then create four variables that refer to the connections between the micro:bit and the L9110S motor controller.

```
from microbit import *
motorA1 = pin0
motorA2 = pin1
motorB1 = pin2
motorB2 = pin8
```

2. The next step is to write a series of functions that will group the instructions needed to perform a task and by calling the name of the function, the task is completed. The first function is called "stop" and it will turn off all of the pins connected to the motor controller, ensuring that the motors stop moving. Lastly we scroll "STOP" across the LED matrix of the micro:bit.

```
def stop():
    motorA1.write_digital(0)
    motorA2.write_digital(0)
    motorB1.write_digital(0)
    motorB2.write_digital(0)
    display.scroll("STOP")
```

3. The next function will drive the motors backwards (reverse) and the way in which our motors are wired we turn on A1 and B2 and turn off the other connections. This will flip the motors to turn in the direction that we require. This function also takes an argument (an extra instruction) which in this case is the speed, as a value between 0 and 1023.

```
def backward(speed):
    motorA1.write_analog(speed)
    motorA2.write_analog(0)
    motorB1.write_analog(0)
    motorB2.write_analog(speed)
```

Now we'll create more functions for forward, left & right.

Forward

```
def forward(speed):
    motorA1.write_analog(0)
    motorA2.write_analog(speed)
    motorB1.write_analog(speed)
    motorB2.write_analog(0)
```

Left

```
def left(speed):
    motorA1.write_analog(0)
    motorA2.write_analog(speed)
    motorB1.write_analog(0)
    motorB2.write_analog(speed)
```

Coding the Robot

With MicroPython

Right

```
def right(speed):
    motorA1.write_analog(speed)
    motorA2.write_analog(0)
    motorB1.write_analog(speed)
    motorB2.write_analog(0)
```

That's all the functions complete. If you were to download and run the code you've just written, it won't actually do anything. This is because we haven't "called" the functions. The next part of the code will do this. Making functions is a great way to reuse code multiple times in a program without having to copy and paste it every time, therefore making the code shorter and more efficient.

4. In the last section of code we create a sequence that will test each of the functions to ensure that our robot has full movement. For functions that involvement movement, for example forward, we pass a value to control the speed of the motor. We chose to use 512 as this provides half speed for the motors. Fast enough to move around freely, but the robot will not travel too far. Between each function we add a sleep so that the robot performs the movement/task for a set amount of time.

```
stop()
forward(512)
sleep(3000)
backward(512)
sleep(3000)
left(512)
sleep(3000)
right(512)
sleep(3000)
stop()
```

5. Flash the code to your micro:bit and check that the robot performs the sequence of code. If the motors move in the wrong direction, simple unscrew their wires from the motor controller and swap them over. Re-test and when happy design a new sequence to perform a cool task.

So there we have it, one simple, cost effective robot which can be programmed in Makecode or MicroPython. This is your robot, decorate and add to the project, have competitions with your friends and their robots and above all else. Have fun building new robots.

Share your robots!



We want to see what you've built using our Low Cost DIY micro:bit powered robot guide. Tweet us a picture using:

[@micro_mag](#) [@biglesp](#)

About Les Pounder



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

[@biglesp](#) [bigl.es](#)



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
MICROMAGSTORE.COM**

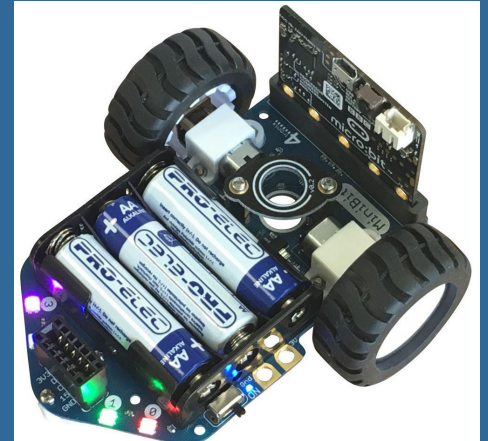
You can still download the free digital edition from our website. View the catalogue at micromag.cc/issues

Robot Showdown

4Tronix MiniBit

This brand new robot from 4tronix is certainly impressive at only £25. It's the cheapest robot on the list and is by far the best for those on a budget.

The 4tronix minibit packs a bunch of features like 4 NeoPixel LEDs, ultrasonic sensor connectors, grippy wheels and a pen holder that you can find on the more expensive robots we've featured. However, with choosing a cheaper robot you can expect to lose some features like line following sensors, these do not appear on the minibit. Fear not though, the robot has some "croc clip" pads which allow you to attach add-on modules like line following sensors (bought separately from 4tronix). There is also a super cool connector on the front for the ultrasonic sensor you can buy which is also "Breakout Garden" compatible. This robot is super easy to work with thanks to its handy MakeCode library which is an adaptation of the already easy to use BitBot library. If you want to go a bit further than MakeCode, there's Python support too.

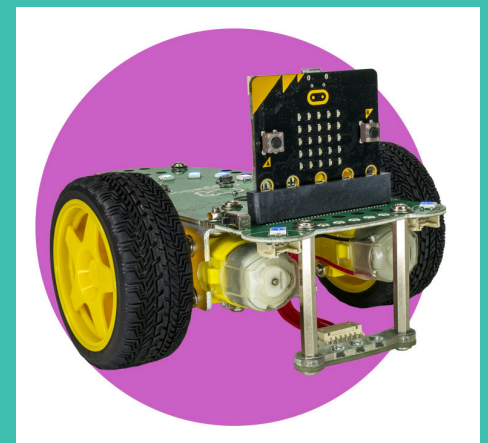


Buy a minibit
go.micromag.cc/minibit

Dexter GiggleBot

So far out of the robots we've done a full review of, the Dexter Industries GiggleBot is the only one to get a 10/10. It just gets so much right across the board. Here's why.

The Dexter Gigglebot has all the features you could expect from a robot: NeoPixels, Grippy Wheels, Line Following, Light Sensing, Pen Holder, Lidar sensor support, Servo Connections & Pin Breakouts. If you're looking for a robot that does it all, the GiggleBot is the one to get. The software support is also incredible. The GiggleBot has extensions for all the major micro:bit editors (MakeCode, Python and EduBlocks). One welcome feature of the GiggleBot is the fact it comes prebuilt so if you're buying in bulk for a school, you don't need to worry about assembling them all. Dexter also run a Free Teacher Trial for schools in the USA and Canada for 45 days where they'll loan you a GiggleBot to try before you buy.



Buy a GiggleBot
go.micromag.cc/buygiggle

Final Verdict:

Best for Budget



MiniBit

We've reviewed a fair few robots over the past six issues of micro:mag, so we thought it would be good to get some of the kids at the Harris Museum Code Club to rate them.

Inksmith K8

We reviewed the Inksmith K8 only last issue and gave it a 7/10. This robot is more on the expensive side, however, it lacks a lot of features for the money.

Don't get us wrong, the Inksmith K8 is really well designed and thought out and the curriculum that goes alongside it is a welcome addition. Though, it's really hard for us to recommend when you can get the Dexter Giggiebot for the same price. The robot requires quite a bit of assembly and is quite fiddly in places. However, there are some upsides. The grippy wheels provide a good grip to the floor, something we find useful at Code Club due to the uneven wooden floor. There are also line sensors and an ultrasonic sensor included however the line sensors are not very secure in the casing. Two big things are missing with the K8 and that's neopixels and a pen holder. This might not be important for some, but it gives a robot some life and is a feature that we definitely expect on a robot of this price range. The Inksmith K8 isn't bad at all, we'd just not pick it up over the others.



Buy a K8

go.micromag.cc/buyk8

4Tronix BitBot XL

A familiar robot in a brand new package. Introducing the 4tronix BitBot XL. An improved version of the already popular 4Tronix BitBot.

If you've looked into micro:bit robots before, you'll have seen the 4tronix bitbot. This is an amazing mid-range micro:bit robot that packs a number of features: 12 NeoPixels, Ultrasonic sensor support, Grippy Wheels, Pen holder, line following and buzzer. The robot is super easy to control and zips around a robot course with no problems. The software support for BitBot XL is good too with extensions for MakeCode and examples on the 4tronix website for MicroPython. One thing that sets the BitBot line of robots apart from the others is its range of community written resources, this is due to the popularity of the original BitBot. You won't be short of examples, tutorials and lesson plans for the BitBot whether you're a beginner or pro programmer.



Buy a BitBot XL

go.micromag.cc/bitbot

Ease of use



GiggieBot

Software

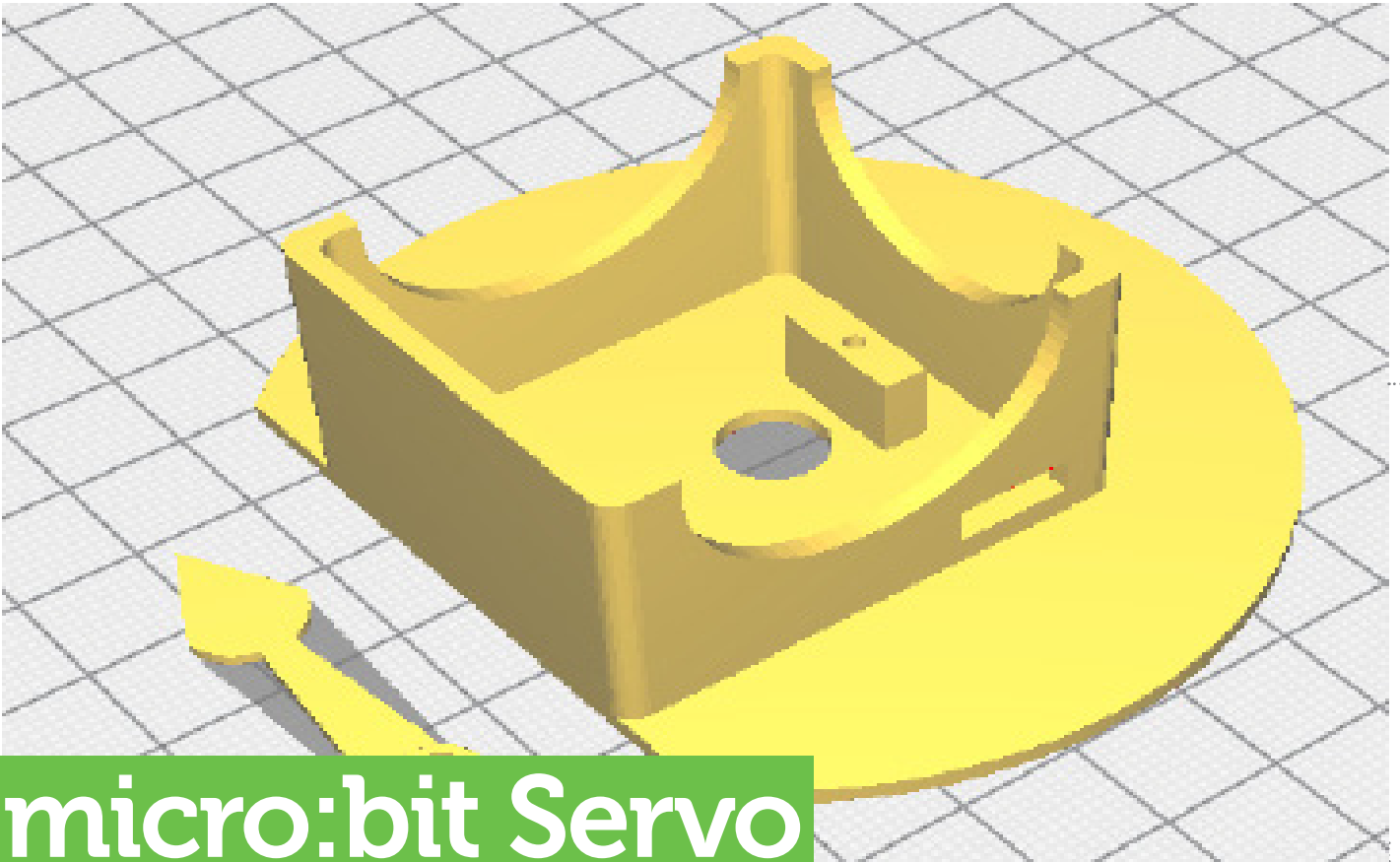


GiggieBot

Resources



BitBot XL



micro:bit Servo MAGIC 8 BALL

Above:
Image of STL
files

How to 3D print and build your very own BBC micro:bit powered Magic 8 Ball.

BY Robert Wiltshire

About the Author



Robert works for Software Cornwall in Education Outreach for the software industry. Software Cornwall connects, promotes, represents and supports Cornwall's digital tech community.

softwarecornwall.org

The Magic 8 Ball is a device that answers questions you put to it with random prearranged answers.

This exercise produces much the same thing except that the answer is indicated by a pointer.

Get Creative

The first stage is to create artwork that will adorn your creation. This will consist of the base with your answers and the pointer. Consider what answers you want your device to give. They may be helpful or not! It is entirely up to you.

Draw around the rounded base and the arrow onto some paper. Get creative with how the base looks. Also, design your pointer too. Cut the designs out, glue gently onto the servo base and the arrow. Don't

put the arrow onto the servo just yet.

First Code

These instructions are a simplified version. The full unabridged text is available online.

Startup the Mu editor with a new tab and enter the following code. With the servos we have the value for the middle is 75.

```
from microbit import *
pin0.set_analog_period(20)
pin0.write_analog(75)
```

Of these three lines of code, the last is the one that controls where the servo rotates to. Connect the micro:bit to the computer and the servo to the

micro:bit. Pin0 is just because the servo is connected to zero. Run the code by flashing to the micro:bit using the Flash button. The servo should rotate and stop. Now place the arrow onto the servo spindle pointing up and screw into place.

Finding the Limits

Using the code below keep adjusting the values upwards towards 180 and down towards 1 until the arm swings smoothly. Keep changing the clockwise/anticlockwise values and reflashing the code until you are happy.

```
from microbit import *
pin0.set_analog_period(20)
clockwise = 30
anticlockwise = 130

while True:
    pin0.write_analog(clockwise)
    sleep(2000)
    pin0.write_analog(anticlockwise)
    sleep(2000)
```

Point to the Answers

Once you have found the limits then find the positions of your artistically placed and drawn answers. Add more variables and adjust until the arrow can point at the answers.

Use the code below to set your positions up. Label them to match your answers. Your labels must be a single word such as not_likely or notLikely.

```
from microbit import *
pin0.set_analog_period(20)
yes = 25
never = 58
likely = 93
no = 145

while True:
    pin0.write_analog(yes)
    sleep(2000)
    pin0.write_analog(never)
    sleep(2000)
    pin0.write_analog(likely)
    sleep(2000)
    pin0.write_analog(no)
    sleep(2000)
```

Obtaining the Random Choice

Our code has four answers and four corresponding servo values. The task now is to randomly select one answer.

To demonstrate this random ability switch to the Read, Evaluate, Print, Loop mode on Mu, by clicking the REPL button.

Type in this at the chevrons.

```
>>> outcomes = {"yes":25, "never":58, "likely":93, "no":145}
```

This is a dictionary with names and corresponding values.

To access the values, code needs to ask for a key and then the value of that key.

```
dictionaryName = {"keyName1" : value1, "keyName2" : value2}
```

A specific value for a known key can be found by asking for it. But this presumes the name is already known and therefore is not a random choice.

```
>>> outcomes["yes"]
>>> 25
```

This extracts just the values as a list.

```
>>>list(outcomes.values())
>>>[25, 58, 93, 145]
```

The following lines import the random ability into Python and then makes a random choice from the list of outcomes values.

```
>>>import random
>>>random.choice(list(outcomes.values()))
>>>25
```

The Full Code

The final part is to get an answer when a button is pressed. So if button A is pressed only then look up a random value and point the servo using that value.

```
from microbit import *
import random

pin0.set_analog_period(20)
outcomes = {"yes":23, "never":58, "likely":93, "no":140}

while True:
    if button_a.is_pressed():
        answer = random.choice(list(outcomes.values()))
        pin0.write_analog(answer)
        sleep(200)
```



Create a VIRTUAL PET

Above:

Virtual pet mobile games have been popular for many years.

Use the Tynker iPad & android app to create your own virtual pet with the BBC micro:bit

BY The Tynker Team

About the Author



Tynker is an online coding platform used by over 60 million kids across 150 countries. Tynker's self-guided courses enable students to learn at their own pace

tynker.com

In this tutorial, we'll create a Virtual Pet using Tynker and your micro:bit. This simple, crowd-pleasing project will have kids create an on-screen animal that responds to the micro:bit's buttons. Even better, it's easy for kids to expand on the project and explore their own creativity.

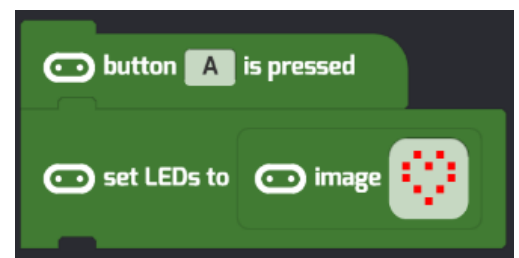
Tynker Workshop (tynker.com) has recently added micro:bit support to their popular online block-based coding platform. It's a great simplified coding environment for micro:bit, especially as it allows for wireless Bluetooth deployment.

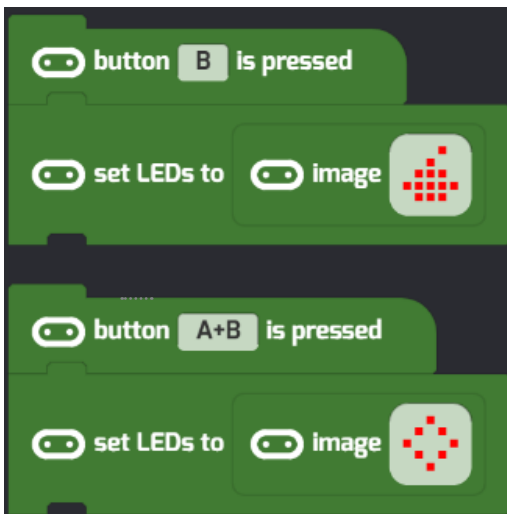
You can follow along at gotyn.kr/virtualpet or open a blank micro:bit Tynker project at

gotyn.kr/microbitprojects

Programs for the micro:bit

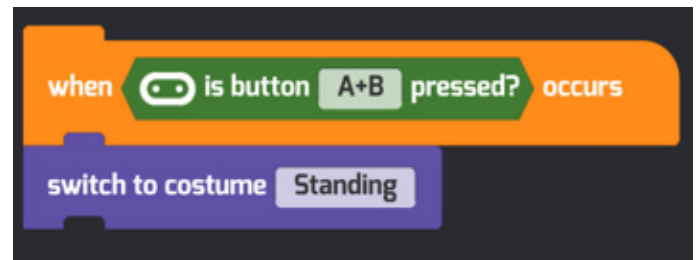
First, let's write some code for the micro:bit itself. With these, the micro:bit will change its LED display whenever the buttons are pressed. You'll find the micro:bit blocks in the Hardware section of Tynker's coding palette.





Above:

Virtual pet mobile games have been popular for many years.



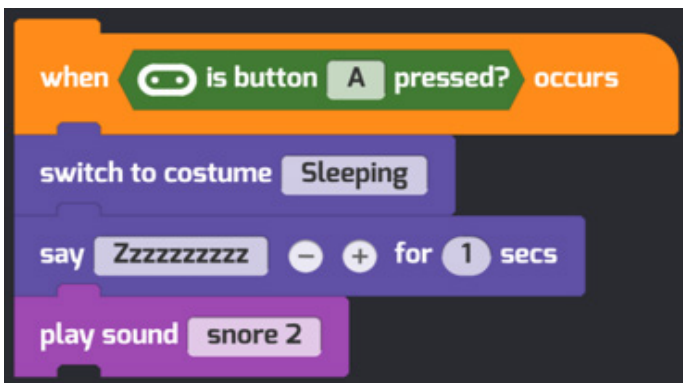
Now let's get the code onto your micro:bit. To set up your micro:bit for wireless interaction with Tynker, click the SET UP MY MICROBIT button above the Stage. Follow the on-screen instructions. Once that's complete, click the big orange run button to deploy this project to the micro:bit.

Then try interacting with your new pet. Both the micro:bit display and the on-screen pet should respond to the micro:bit's buttons. What other commands could you add to your pet? What about using variables to keep track of how hungry your pet is?

This virtual pet project is a fun and simple way to get kids to see the possibilities of physical computing!

Take a nap

Then let's program our Virtual Pet to respond to button A. We'll have our Virtual Pet change costumes and play a sound.



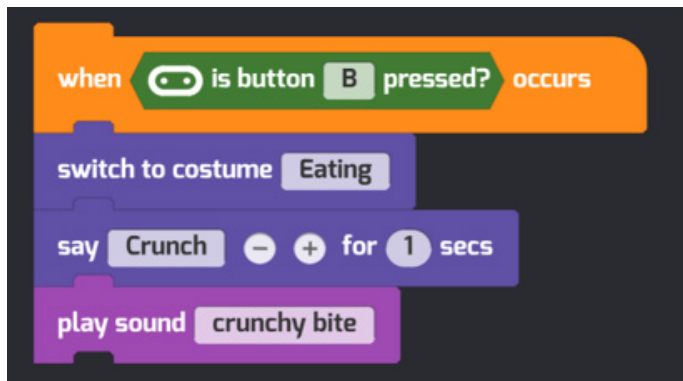
Get the Tynker app!

You can download the Tynker app for free on your iPad or Android device using the links below!



Eat Food

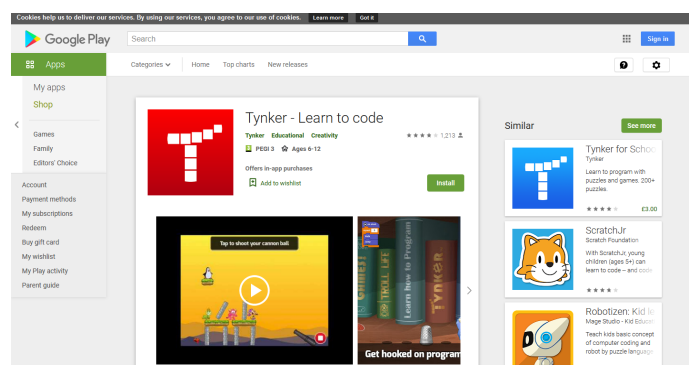
We'll program button B to make our pet chow down!



Stand Up

We'll program button A+B to make our pet stand back up.

go.micromag.cc/tynkerapple



go.micromag.cc/tynkerandroid

```

function SWIntro
  play tone Middle G for 1/4 ▾ beat
  rest(ms) 1/8 ▾ beat
  play tone Middle G for 1/4 ▾ beat
  rest(ms) 1/8 ▾ beat
  play tone Middle G for 1/4 ▾ beat
  play tone Middle C for 2 ▾ beat
  play tone Middle G for 2 ▾ beat

function SW2ndPhrase
  play tone Middle F for 1/4 ▾ beat
  rest(ms) 1/8 ▾ beat
  play tone Middle E for 1/4 ▾ beat
  rest(ms) 1/8 ▾ beat
  play tone Middle D for 1/4 ▾ beat
  rest(ms) 1/8 ▾ beat
  play tone High C for 2 ▾ beat
  play tone Middle G for 1 ▾ beat

function SW3rdPhrase
  play tone Middle F for 1/4 ▾ beat
  rest(ms) 1/8 ▾ beat
  play tone Middle E for 1/4 ▾ beat
  rest(ms) 1/8 ▾ beat
  play tone Middle F for 1/4 ▾ beat
  play tone Middle D for 2 ▾ beat

on button A ▾ pressed
  repeat 3 times
  do
    call SWIntro
    call SW2ndPhrase
    call SW2ndPhrase
    call SW3rdPhrase
    call SWIntro
    call SW2ndPhrase

on event from MICROBIT_ID_BUTTON_A ▾ with value MICROBIT_BUTTON_EVT_UP ▾
  show string "Star Wars - a long long time ago in a galaxy far far away..."
  
```

LEARNING FUNCTIONS WITH STAR WARS

BY Meredith Ebbs

Learn how to cut your code down by using Functions in MakeCode

About the Author



Educational Engineer |
Apple Teacher | Makey
Makey Ambassador |
Digital Technologies
| NSW CSER Project
Officer

@imerinet

The Microbit can be used to create music. I recently watched the MicroMonsters YouTube video. The video is easy to follow and the purpose of the video is to learn to use a Piezo speaker. So of course I composed the code but I found the code very long and complex. Then I wondered how can you simplify the code to make it shorter and easier to manage.

Definition

Functions are a way to store information that can be called at anytime during the program. Functions remove the need to type the same code over and over and allow you to reuse repetitive pieces of code. This reduces the number of lines of code

and errors.

Example

For example, the tune of Star Wars has several parts that are repeated throughout the song. By creating a Function for each repeating part it is possible to use the function multiple times. This creates simpler code.

The first phrase of Star Wars can be rewritten as SWIntro. It contains the opening phrase of Star Wars Da Da Da Dum Da "G, G, G, C, G". Instead of writing these 5 notes every time they can be replaced with the Function SWIntro.

```

function SWIntro
  play tone Middle G for 1/4 beat
  rest(ms) 1/8 beat
  play tone Middle G for 1/4 beat
  rest(ms) 1/8 beat
  play tone Middle G for 1/4 beat
  play tone Middle C for 2 beat
  play tone Middle G for 2 beat
  
```

The next phrase of music in the picture to the right "F, E, D, C, G" is also repeated. This was made into a Function SW2ndPhrase. The phrase is used each time you want to call that series of notes.

```

function SW2ndPhrase
  play tone Middle F for 1/4 beat
  rest(ms) 1/8 beat
  play tone Middle E for 1/4 beat
  rest(ms) 1/8 beat
  play tone Middle D for 1/4 beat
  rest(ms) 1/8 beat
  play tone High C for 2 beat
  play tone Middle G for 1 beat
  
```

The 3rd series of notes in the Star Wars these is "F, E, F, D" This series of notes is replaced with SW3rd-Phrase.

```

function SW3rdPhrase
  play tone Middle F for 1/4 beat
  rest(ms) 1/8 beat
  play tone Middle E for 1/4 beat
  rest(ms) 1/8 beat
  play tone Middle F for 1/4 beat
  play tone Middle D for 2 beat
  
```

Once you have created each function you can then use them to replace the series of notes in the program. Rather than have a big long block of code we now have a series of single blocks that call the sequence of notes as required. To make the code play longer it is put in-

side a repeat block and repeated 3 times.

```

on button A pressed
  repeat 3 times
  do
    call SWIntro
    call SW2ndPhrase
    call SW2ndPhrase
    call SW3rdPhrase
    call SWIntro
    call SW2ndPhrase
    call SW2ndPhrase
    call SW3rdPhrase
  
```

Because the music goes for a long time it is important to be able to stop the music. To do this you can add another event called On Button B pressed. When you press Button B we want to reset the code to start. This will turn off the music and replace it with a cleared screen.

In the original video they used a control block to trigger the Star Wars introduction text to play in the background while the music is playing. The way the code was written in the video the text starts playing immediately. I wanted the text to scroll when Button A was pressed and not before.

In the control menu there is a block "on the event Microbit_ID_Button_A" with an option to choose a value - for this code "Microbit_Button_Evt_Up" was selected, this means when the A button is lifted up after being pressed execute the series of blocks inside this event. For this program when Button A is pressed and lifted up it will trigger the show string block.

```

on event from MICROBIT_ID_BUTTON_A with value MICROBIT_BUTTON_EVT_UP
  show string Star Wars - a long long time ago in a galaxy far far away...
  
```

Put all the code together download the code to your micro:bit and it will play the Star Wars theme. Button A will trigger the theme to play 3 times. When Button A lifts up after being depressed it will also trigger the Star Wars text to scroll. Button B will reset the program or stop the music.

:VIEW text32

Add a super cool LCD display to your micro:bit with the :view text32 LCD display from Kitronik. By Kerry Kidd

AVAILABLE FROM
Kitronik
kitronik.co.uk/5650

What's Included?

- :VIEW text32 LCD

Price

\$24.60 USD
€22.27 EUR
£19.20 GBP

Approx

The :VIEW is a cool text display add on for the micro:bit! It allows you to display 2 lines of text at 16 characters per line.

:VIEW would be great to make a name badge for conferences where you would want your name, company name and maybe even a Twitter handle. This is a lot of information to try and display on the 5 x 5 LED matrix of the micro:bit but perfect to use the :VIEW text32 addon for.

It is very easy to get up and running with the :VIEW all you need it 3 AAA batteries which powers the :VIEW and the micro:bit. There is a MakeCode extension for the :VIEW which helps you to get the :VIEW up

and running quickly. We were able to get the text displaying on the :VIEW within a couple of minutes playing with it.

The :VIEW has another cool feature where it can be attached to another micro:bit add-on to add even more functionality to your micro:bit!

The board is reasonably priced at around £20 for what it can do and there are not really any other options on the market for an all in one LCD add-on board for the micro:bit.

The board features a super handy backlight for illuminating the back of the LCD display so that you can view it in low light or dark conditions. This can be toggled on

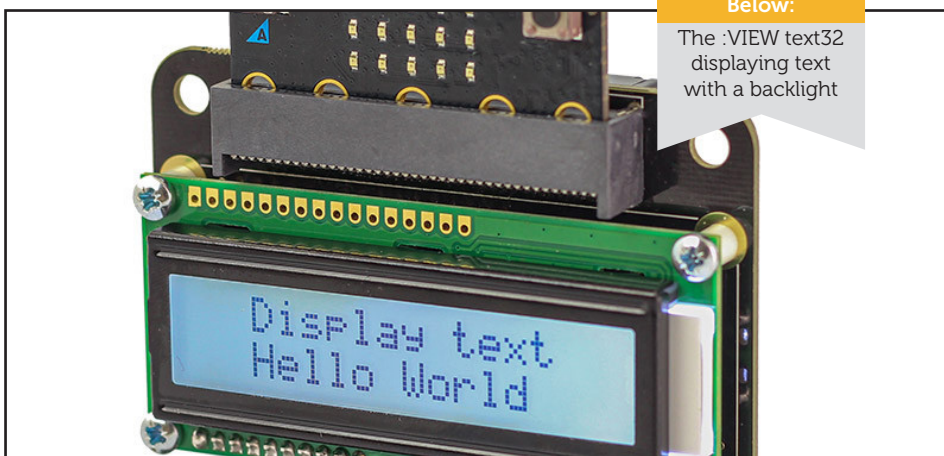
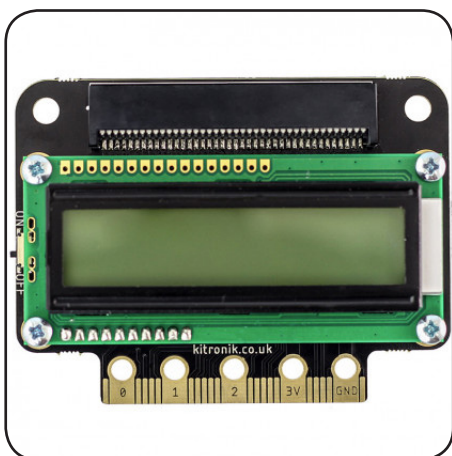
via the MakeCode library.

If you'd like to go a bit further than MakeCode, Kitronik also provide a really simple and easy to use Python library that is even easier to use now thanks to the recent MicroPython editor update! (Read about it on pages 6 & 7)

Overall this is a great product and a great way to display text messages using your micro:bit, Whether you use it for a name badge or use it in conjunction with another micro:bit add-on board to display an output.

OUR RATING

10/10



Below:
The :VIEW text32 displaying text with a backlight

Similar Products

Two other products to consider...



01 Elecrow LCD1602

This product is much cheaper at only \$5.50 but has slow delivery from China and no MakeCode library.



02 Waveshare Screen

This 1.8 inch colour LCD is an interesting add-on board which allows you to display images in colour!

:ZIP Halo

The "Halo" shaped 24 neopixel add-on board for the micro:bit from Kitronik. By Kerry Kidd

AVAILABLE FROM
Kitronik
kitronik.co.uk/5625

What's Included?

- :ZIP Halo board

Price

\$17.69 USD
€16.01 EUR
£13.80 GBP

Approx

ZIP:Halo is a ring of 24 individually addressable neopixels that connect with screws to the bottom of the edge connector of the micro:bit.

To get up and running with the :ZIP Halo you require a screwdriver to connect it to the micro:bit using the 5 screws included. :ZIP Halo can be programmed through the MakeCode or Python Editors.

The :ZIP Halo is a simple way to add a ring of Neopixels to your micro:bit projects never the less we had great fun reviewing the :ZIP Halo by adding some colour to a compass and spirit level. Using the lesson plans from Kitronik's website.

The lesson plans are very well written and great for a beginner to pick up and learn how to program the micro:bit using the :ZIP Halo.

The :ZIP Halo is part of a wider range of neopixel products from Kitronik and these products include a very nicely laid out getting started guide which shows you what you need to do to get up and running with the :ZIP Halo.

We did feel however that this product is slightly overpriced for what it is at £13.80 as many other micro:bit neopixel products are slightly cheaper.

There's also some connections on the bottom of the board which allows you to connect more :ZIP

LEDs which are sold by Kitronik.

The :ZIP halo needs to be powered externally as the micro:bit can only handle up to 8 LEDs on its own. This is slightly inconvenient but we can't complain as the board simply wouldn't work without it.

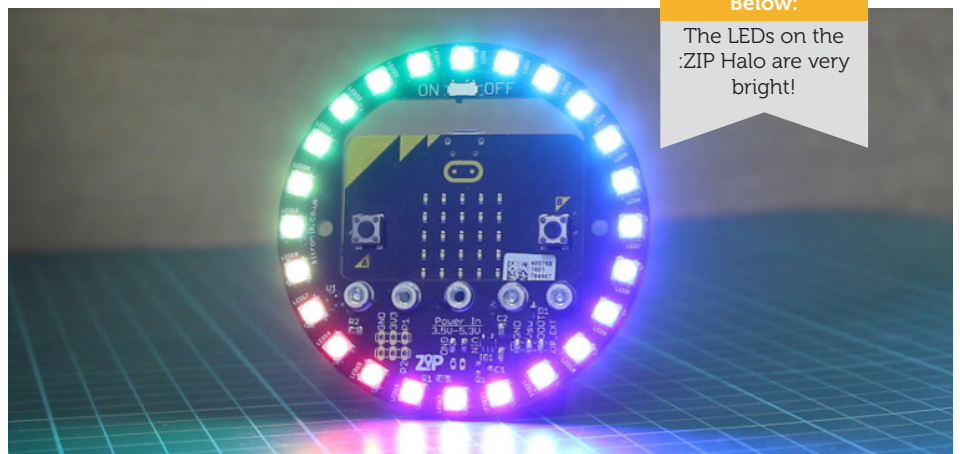
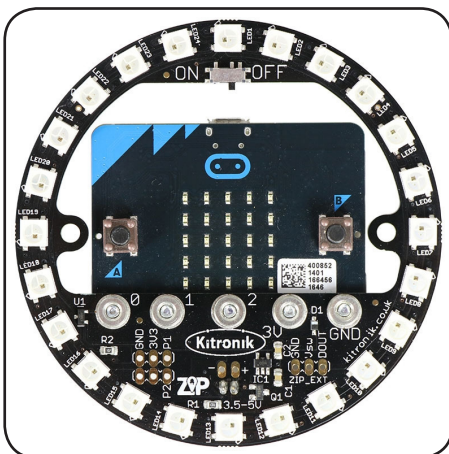
Overall this is a well-designed piece of kit and easy to put together. We had a lot of fun programming it.

OUR RATING

8/10

Below:

The LEDs on the :ZIP Halo are very bright!



Similar Products

Two other products to consider...



01 micro:pixel

Quite similar to the unicorn pHat for the Raspberry Pi. The proto:pic micro:pixel is another neopixel add on.



02 KEYES LED Matrix

Similar to the micro:pixel. This is another option for a neopixel add-on board.

Tobbie II

A unique micro:bit powered humanoid robot from the makers of Snap Circuits, Elenco. By Joshua Lowe

AVAILABLE FROM
Elenco

[go.micromag.cc/
tobbie](http://go.micromag.cc/tobbie)

What's Included?

- Tobbie II Kit

Price

\$84.95 USD

€76.88 EUR

£66.27 GBP

Approx

Here at micro:mag, we review a LOT of micro:bit robots. It's not hard to go wrong when buying a micro:bit robot (sure, some are better than others) but most of the time you'll get two wheels, a distance sensor and some LEDs. However, that trend ends with the Tobbie II, a micro:bit powered, humanoid robot that caught our attention when it did the rounds on Twitter the other month. We just knew we had to get one to review for this issue to see what it was all about. Did it live up to our expectation? Well, you'll have to read on.

The Tobbie II for the micro:bit is certainly unique. It's not got two wheels but instead, it has six legs Tobbie II uses to walk. Pretty cool right? The robot comes as a kit which took us around two hours to build. The parts are plastic and they all come in

a grid of which you can snap off the individual parts. The kit takes around two hours to build and isn't easy in places so we'd recommend getting some help from an adult if you're stuck.

Tobbie II's head houses the micro:bit. There's a PCB with an edge connector that the micro:bit slots into and a plastic flap that covers the micro:bit up. There are two sensors on the front that can detect objects and two buttons that connect to the buttons on the micro:bit so you can access them with the flap closed. We like the overall design of the Tobbie II and it's certainly something different. It's eye-catching and caught the attention of many attendees visiting the micro:mag stand at micro:bit Live last month. The Tobbie II is easily programmed via a Makecode library but unfortunately, there is no official Python support. However, Tobbie II isn't all that great. It's got poor build quality,

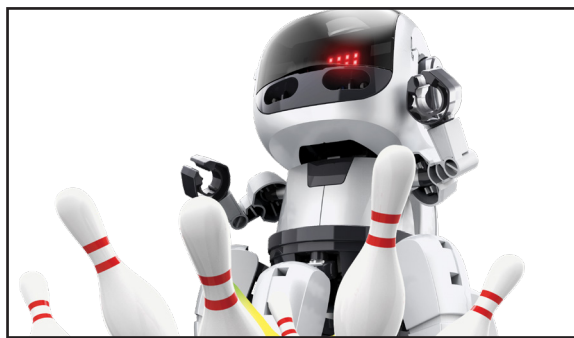
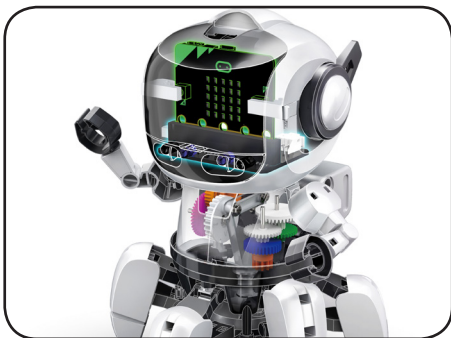
it's overpriced and just doesn't feel that premium. Within five minutes of testing both the robots we got sent with the kids at Code Club, the legs of the Tobbie II broke apart and had to be reconnected to the slots at the bottom of the casing. Whilst we appreciate this product, we wish it had better build quality and was significantly cheaper. These issues make it hard to recommend, but it's certainly worth buying if you're looking for something different and have the money to spare.

OUR RATING

6/10

Below:

Tobbie's Legs are omnidirectional!



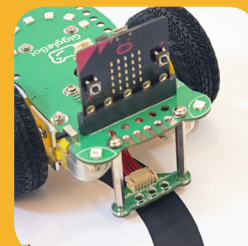
Similar Products

It's unique, but here are two other robots...



01 4tronix mini:bit

This super cheap budget robot has most of the features you need to get started with robotics & the micro:bit.

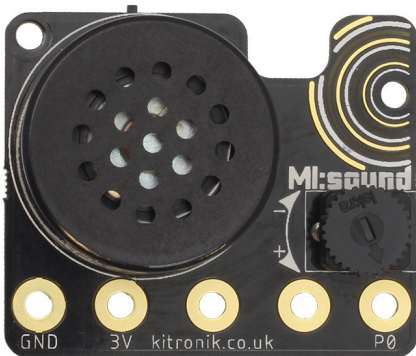


02 Dexter GiggieBot

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

New Accessories SHOWCASE

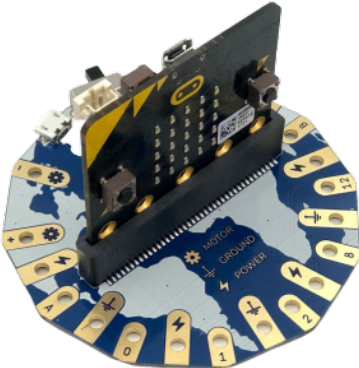
We take a look at some of the brand new micro:bit accessories that have just hit the shelves!



Kitronik MI:power board

The MI:sound board is a speaker board for the BBC microbit. It features on-board; speaker, CR2032 battery holder, thumb wheel volume control and an easy access on/off switch. The MI:sound board can either be powered by the CR2032 battery (provided) or via the micro:bit USB or JST connections.

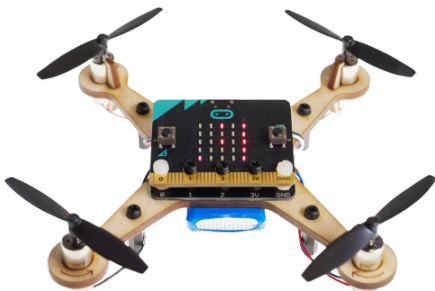
Price: £6 **Buy:** kitronik.co.uk/5649



Inksmith Climate Action Kit

With the Climate Action Kit, you can monitor the climate around you with the different sensors and peripherals in the kit. In the kit you will find the following: a Breakout Board, Alligator Clips, DC Water Pump, DC Motor & Fan, Moisture Sensor, 360 Servo Motor, Infrared Sensor, Tube & Connectors and some pH strips.

Price: £35 **Buy:** go.micromag.cc/cak



Air:bit micro:bit Drone

Air:Bit combines the micro:bit's ease of use with the excitement of a drone. It is the world's first ever developed micro:bit dronet is a STEAM learning kit, which will engage students with a more practical and creative way of learning. It is fully repairable and withstands a lot of crashes. The best of all - it is affordable compared to other drones.

Price: £137 **Buy:** makekit.no/airbit

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