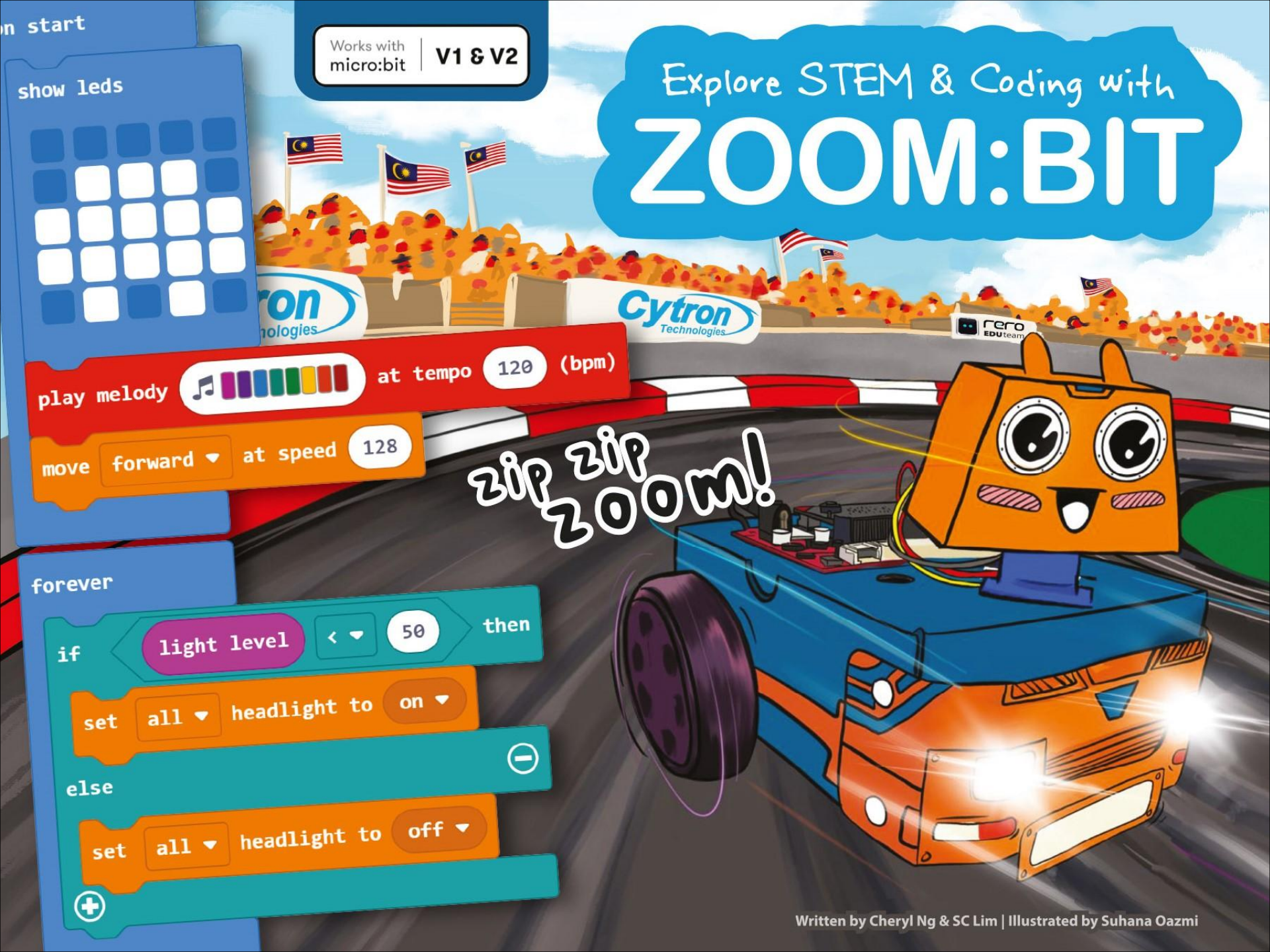

Works with micro:bit V1 & V2
Explore STEM & Coding with
ZOOM:BIT
on start
show leds
play melody at tempo 120 (bpm)
move forward at speed 128
forever
if light level < 50 then
set all headlight to on
else
set all headlight to off
Cytron Technologies
zip zip zoom!
Written by Cheryl Ng & SC Lim | Illustrated by Suhana Oazmi

Note from vero EDUteam @ Cytron

Dear Jr Maker,

Welcome aboard! We are Adam & Anna. And we're so excited to have you join our team of Makers.

In the following pages, we will walk you through the steps to build ZOOM:BIT... and very soon you will have your own robot car. You're also going to learn to code and train ZOOM:BIT to perform some tricks to WOW your friends. We're sure you will have lots of fun together.

If you encounter any problem along the way, you can reach us on Telegram t.me/zoombit_support. We'll be there to guide you. So are you ready? Let's start!

https://t.me/zoombit_support

Adam & Anna

Explore STEM & Coding with ZOOM:BIT

Written by

Cheryl Ng & SC Lim

Illustrated by

Suhana Oazmi

3rd Printing 2022

Published by

Cytron Technologies

ISBN-978-967-19475-1-7

Published by
Cytron Technologies Sdn. Bd.
No 1, Lorong Industri Impian 1,
Taman Industri Impian,
14000 Bukit Mertajam,
Pulau Pinang, Malaysia.
Tel: +604-5480668

www.cytron.io/p-zoombit

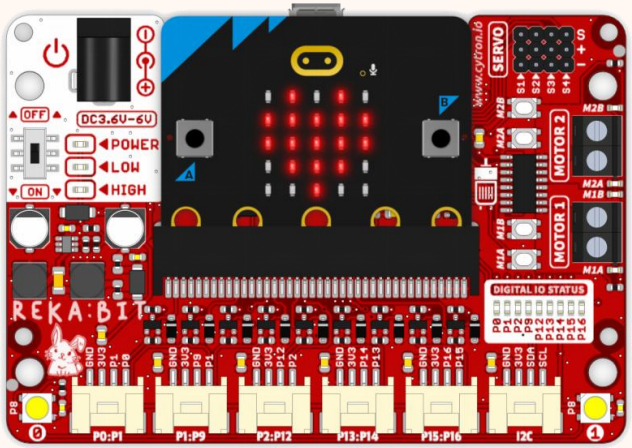
Printed in Malaysia

CONTENT

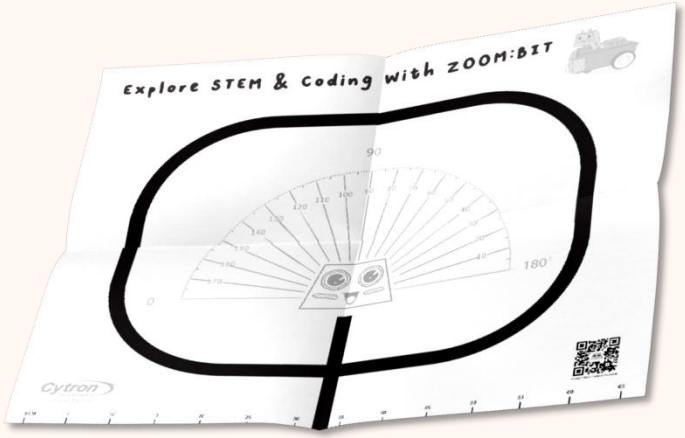
What's in the box? 1
Let's Build! 3
Chapter 1 : Hello World (LED matrix on micro:bit) 17
Chapter 2 : Sing Us A Song (Piezo buzzer/speaker on micro:bit) 32
Chapter 3 : Turn Those Lights ON... Tadaa! (LED headlights) 43
Chapter 4 : Let's Get Moving (DC motors) 50
Chapter 5 : Signal Where You're Going (RGB LEDs on REKA:BIT) 58
Chapter 6 : Look to the Left, Look to the Right (Servo motor) 63
Chapter 7 : Obstacle Avoidance (Ultrasonic sensor) 71
Chapter 8 : Stay On Track! (Maker Line Sensor) 80
Chapter 9 : All in One, Changing Modes 93
Bonus Chapter : Remote Control (Radio Communication) 103
My Learning Journal 108

What's in the Box?

Quick Start Guidebook

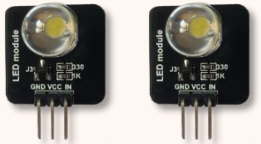


REKA:BIT (with or without micro:bit)

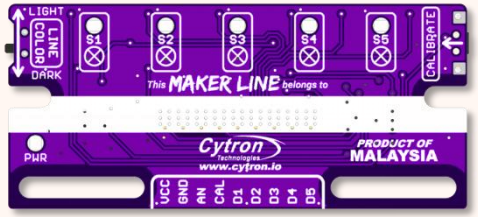


Zoom Track

Ultrasonic Sensor



LED Module x2



Maker Line Sensor

Grove Cable x4

DC Motor & Wheel x2

Screwdriver

Castor

Screw & Nut x4

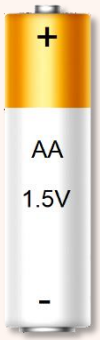
Power & Data Cable

Servo Motor

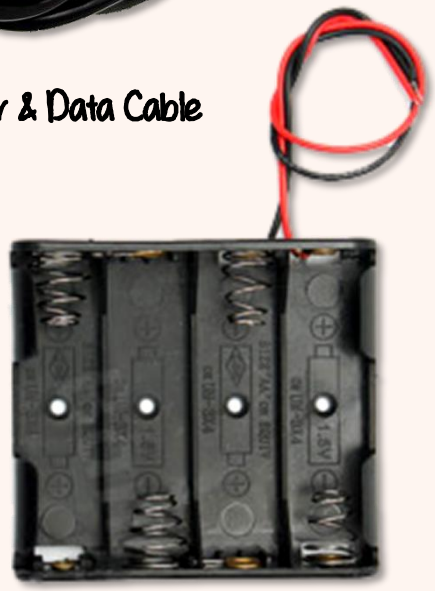
Double-sided Tape x5

White Rivet x4

Black Rivet x8

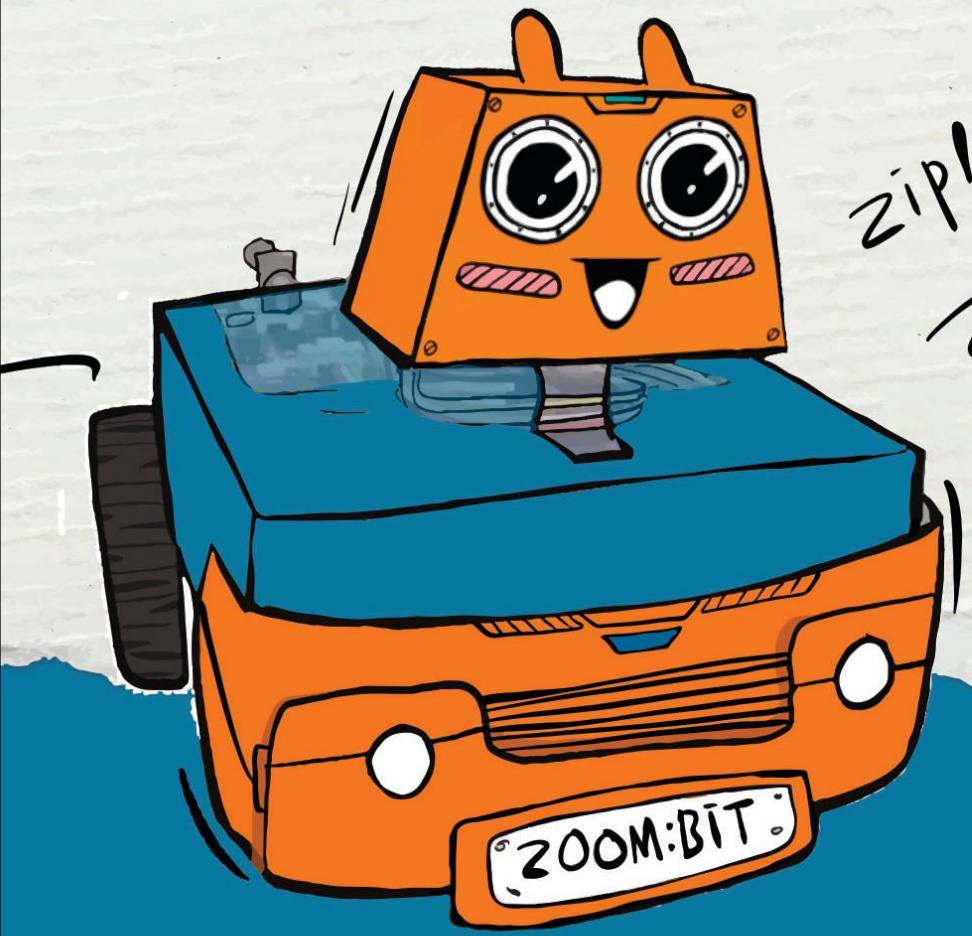


AA Battery x4

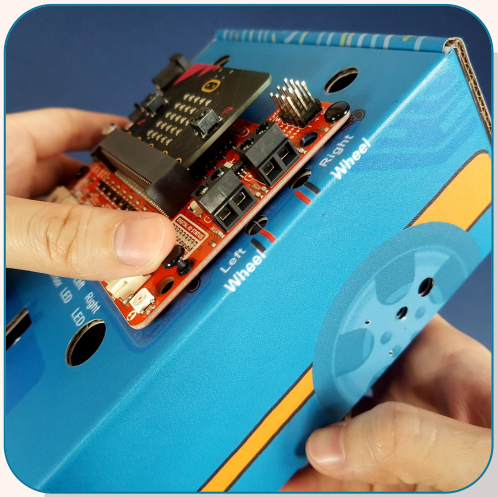


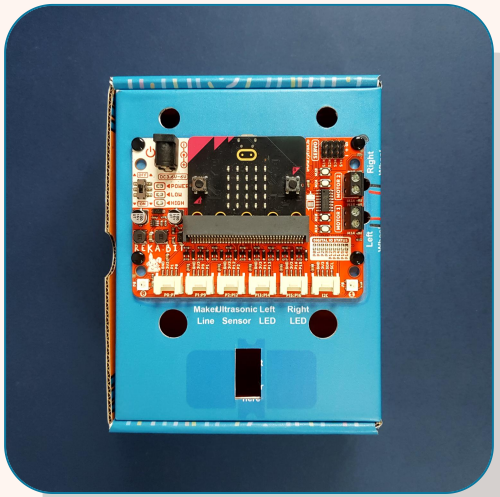
Battery Holder

Let's Build!



https://link.cytron.io
/zoombit-resource-hub

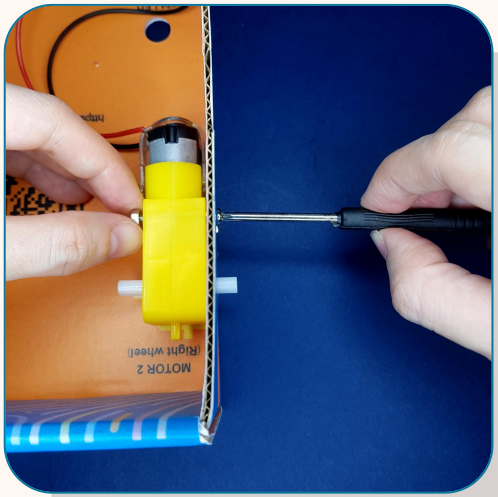


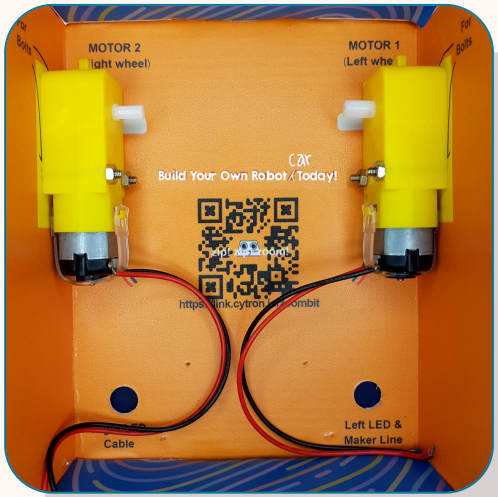


1. Empty out the box and detach all perforated parts as shown.

2. Position REKA:BIT on top of the box; you can refer to the rivet holes for alignment.

3. Insert four (4) black rivets through the holes and then press firmly to secure REKA:BIT in place.

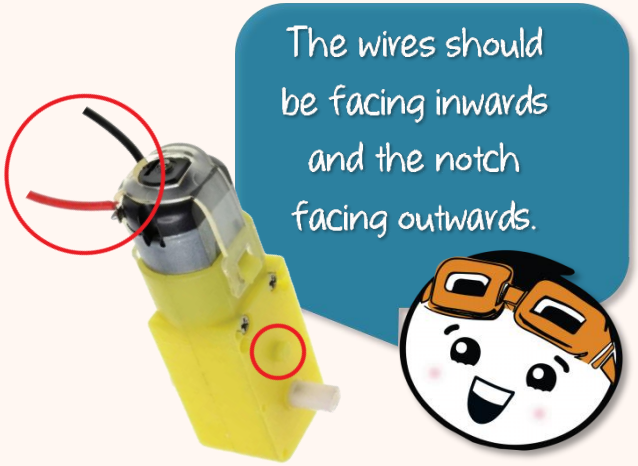


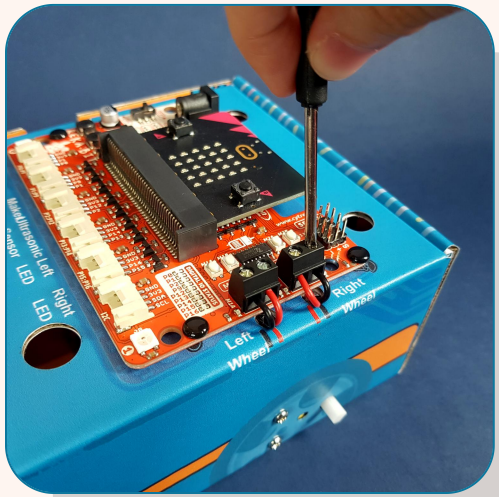


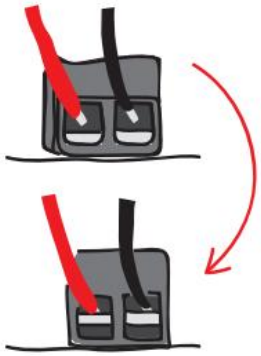
4 Use bolts and nuts to fasten a DC motor to the side of the box labelled MOTOR 2 (right wheel) as shown above.

5 Repeat to fasten the other DC motor to the side labelled MOTOR 1 (left wheel).

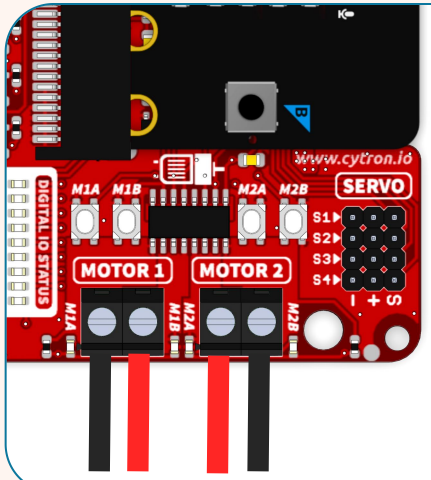
6 Feed the wires through the holes labelled MOTOR 1 wires (left wheel) and MOTOR 2 wires (right wheel).







* Clamp the exposed wire lead, and not the insulator.



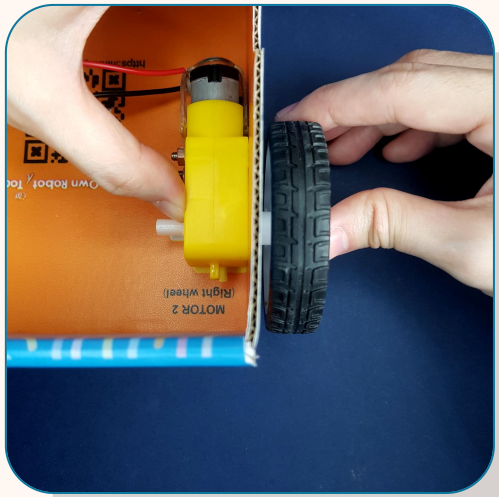
7 Connect the motor wires to MOTOR 1 and MOTOR 2 terminals on REKA:BIT :-

(i) insert the exposed wires, and then

(ii) tighten the screws using the screwdriver provided to secure the connections in place.

Wire Connections:

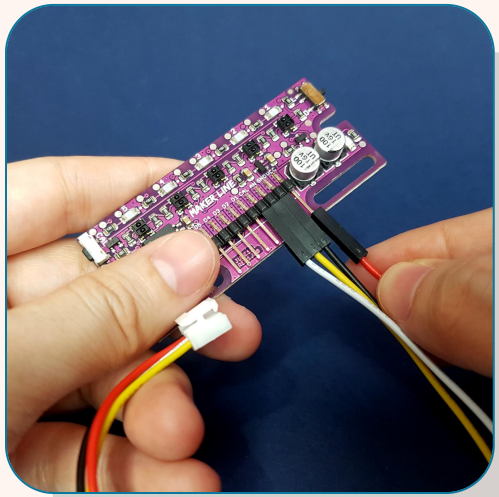
Motor		Motor Terminal
MOTOR 1	Black (-)	M1A
	Red (+)	M1B
MOTOR 2	Red (+)	M2A
	Black (-)	M2B

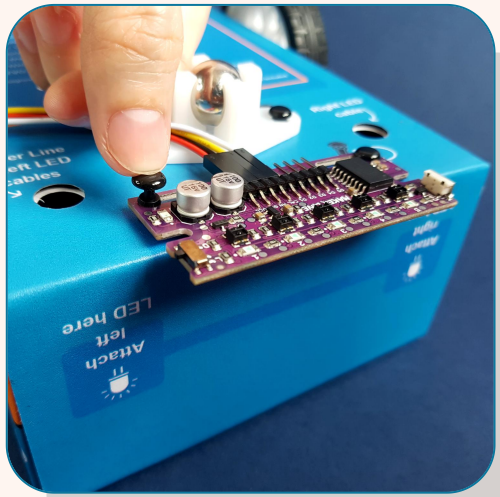


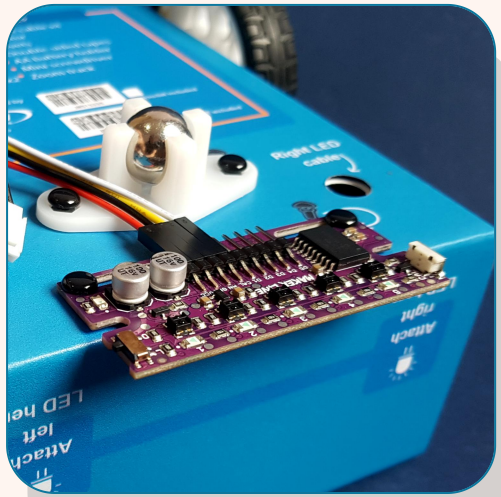
8 Attach wheels securely to the DC motor shafts.

9 Turn the box around and position the castor at the indicated position.

10 Insert two (2) black rivets through the holes and press firmly to secure the castor to the bottom of the box.







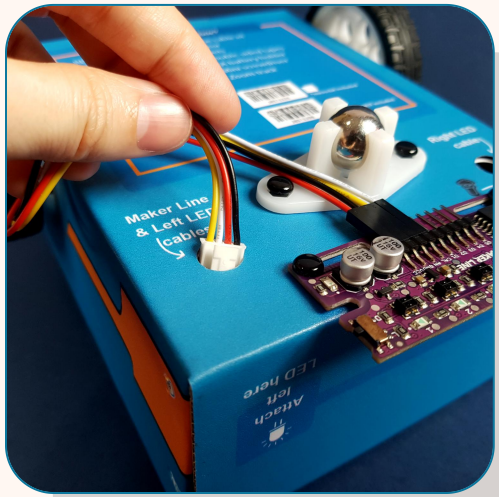
11 Attach one Grove to 4-pin female jumper cable to Maker Line sensor.

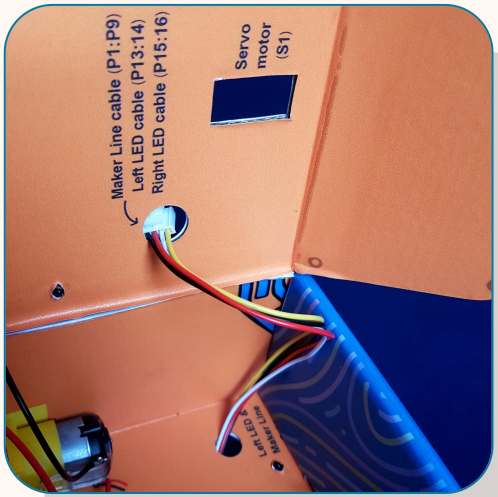
*If you're using micro:bit V1, do NOT connect the WHITE wire to Maker Line sensor. Leave the white wire unconnected.

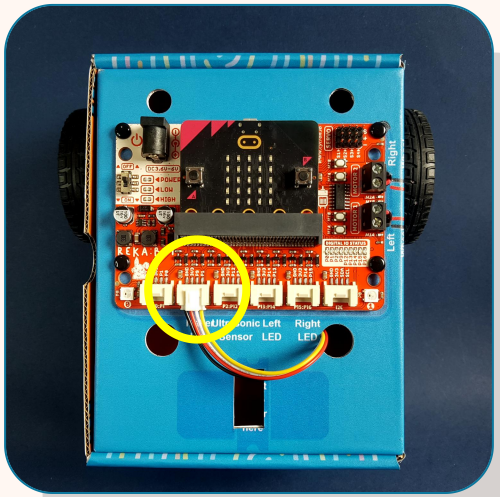
12 Position the Maker Line sensor at the indicated position and use two (2) black rivets to secure it in place.

Wire Connections:

Grove cable	Maker Line Sensor
White	CAL (Calibrate)
Yellow	AN (Analog)
Black	GND (Ground)
Red	VCC (Power input)

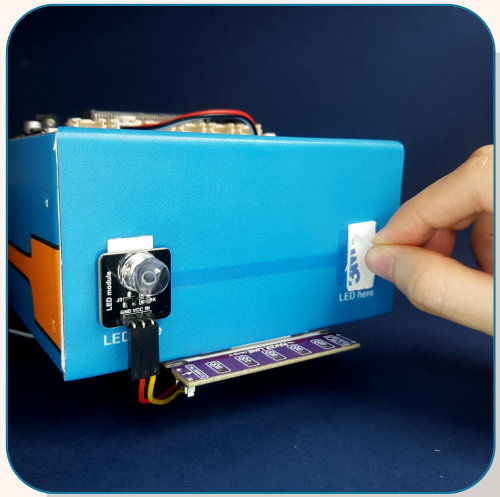


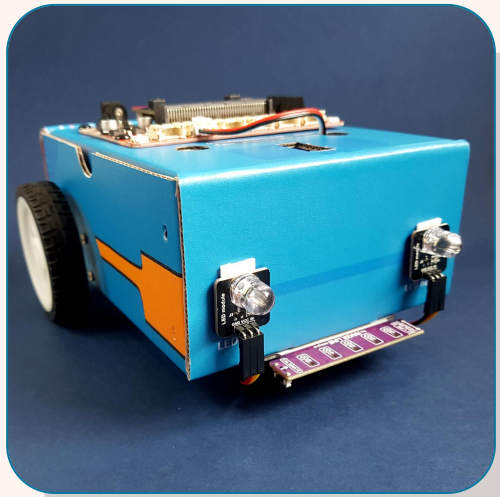




13 Feed the cable through the holes as shown.

14 Connect the Maker Line sensor cable to port **P1:P9** on REKA:BIT.





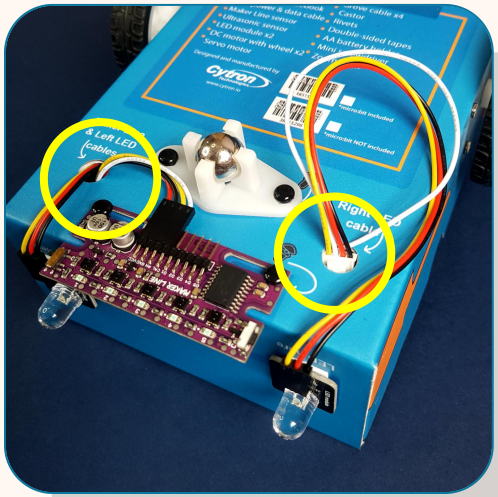
15 Attach one Grove to 4-pin female jumper cable to an LED module as shown.

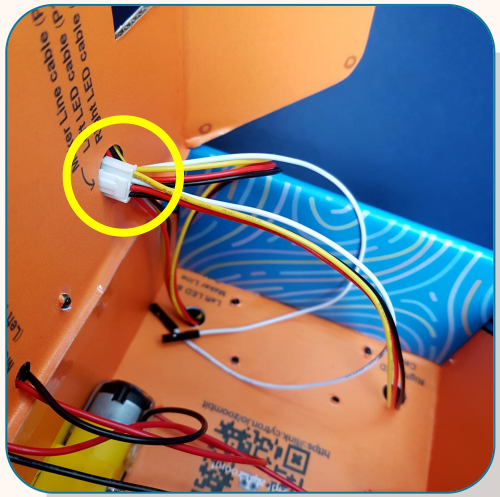
16 Repeat for the other LED module.
* Leave the white wire unconnected.

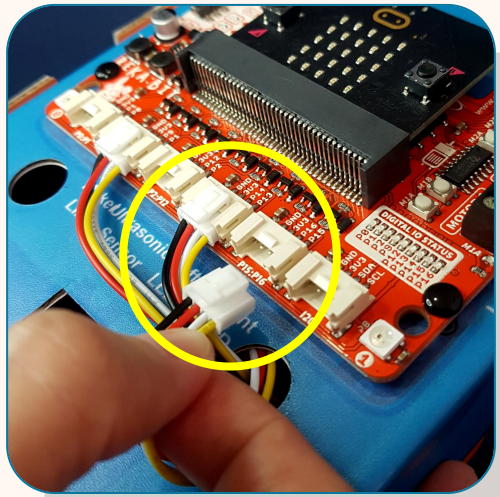
17 Use two pieces of double-sided tape to attach the LED modules to the front of the box as shown above.

Wire Connections:

Grove cable	LED module
White	Not connected
Yellow	IN (Input)
Black	GND (Ground)
Red	VCC (Power input)







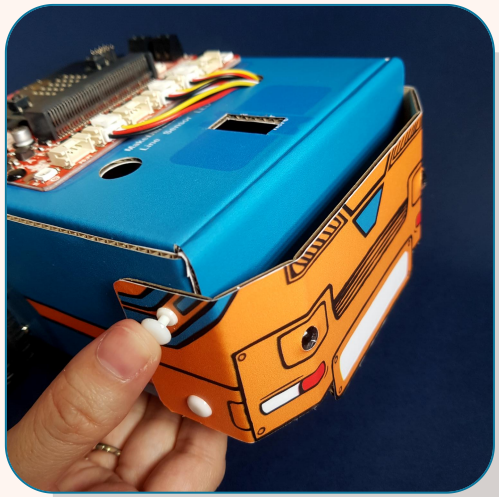
18 Feed both right and left LED cables through the holes as labelled.

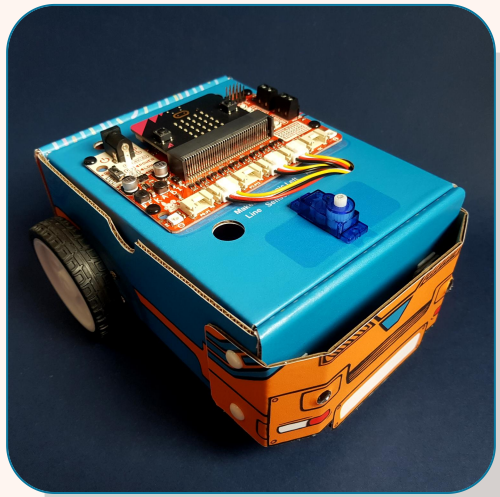
19 Connect the left LED cable to port **P13:P14** on REKA:BIT.

20 Connect the right LED cable to port **P15:P16** on REKA:BIT.

Wire Connections:

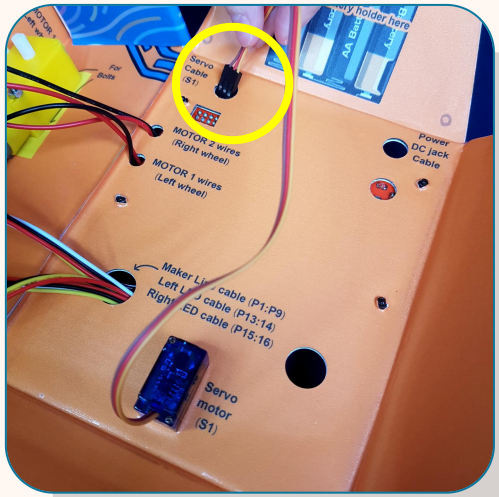
LED Module	REKA:BIT Port
Left	P13:P14
Right	P15:P16

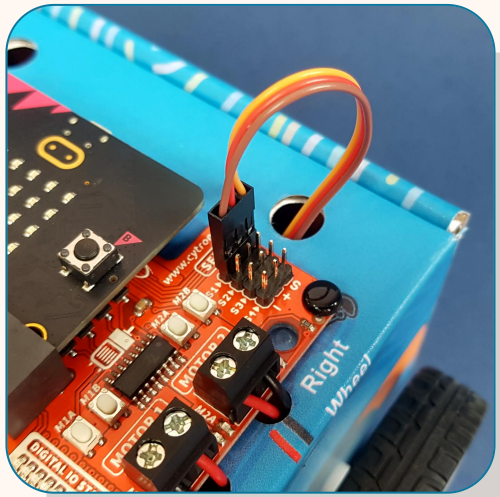


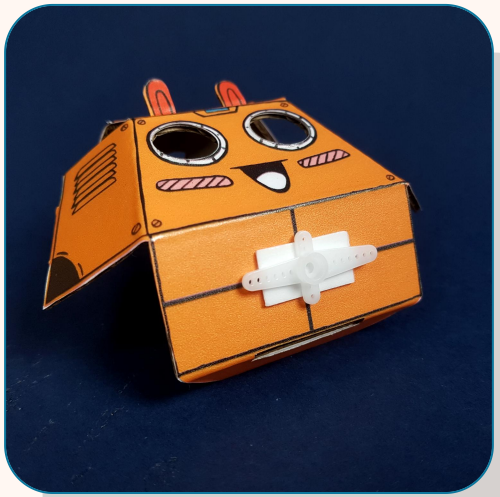


21 Attach the front bumper cardboard to the box with four (4) white rivets; use two rivets on each side.

22 Lower the servo motor into the opening until it sits firmly in place as shown above.







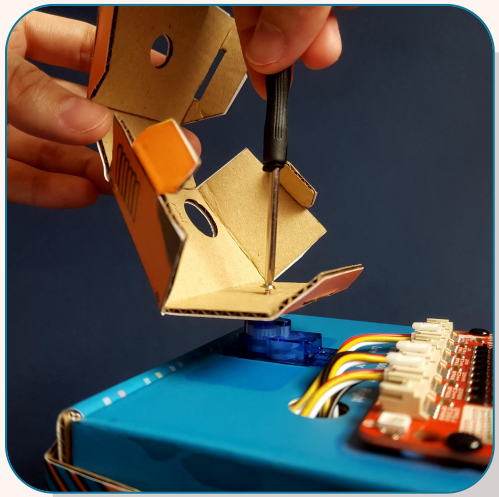
23 Feed the servo motor cable through the hole as labelled.

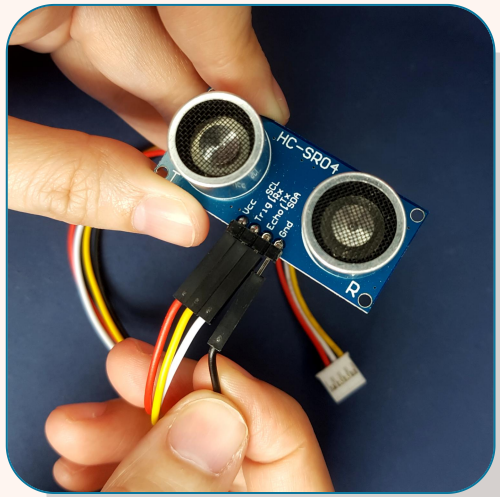
24 Connect the servo motor cable to servo port labelled S1.

25 Use a piece of double-sided tape to attach a servo motor horn to the head cardboard piece as shown.

Wire Connections:

Servo motor cable	Servo port S1
Orange	S (Signal)
Red	+ (Power)
Brown	- (Ground)





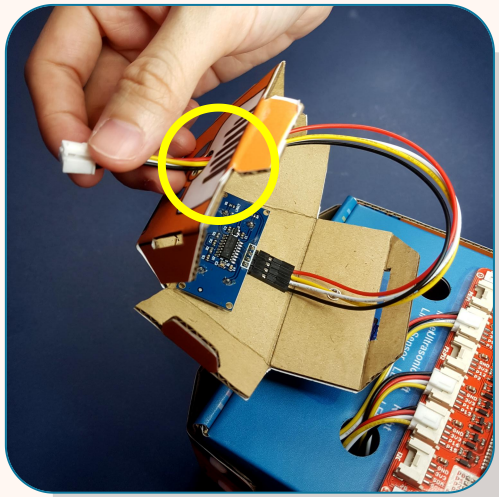
26 Attach the horn to the servo motor shaft. Use the screw and screwdriver provided to secure the head cardboard piece in place.

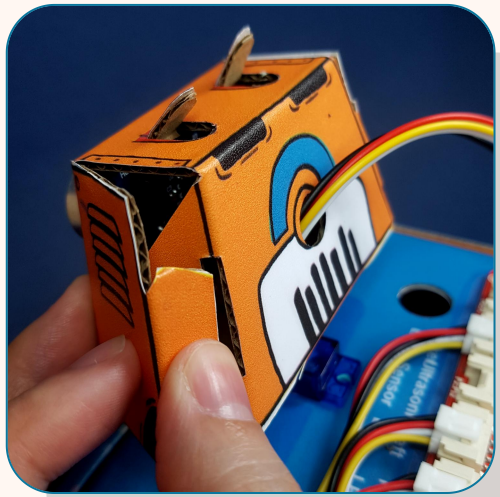
27 Attach one Grove to 4-pin female jumper cable to the ultrasonic sensor.

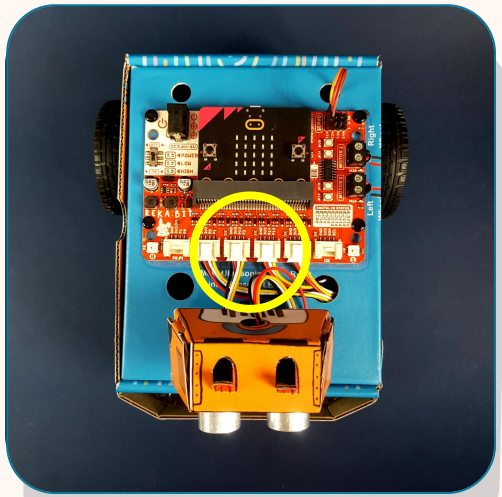
28 Attach the ultrasonic sensor to the head cardboard piece as shown above.

Wire Connections:

Grove cable	Ultrasonic Sensor
Red	VCC (Power Input)
Yellow	Trig (Trigger)
White	Echo (Echo)
Black	GND (Ground)



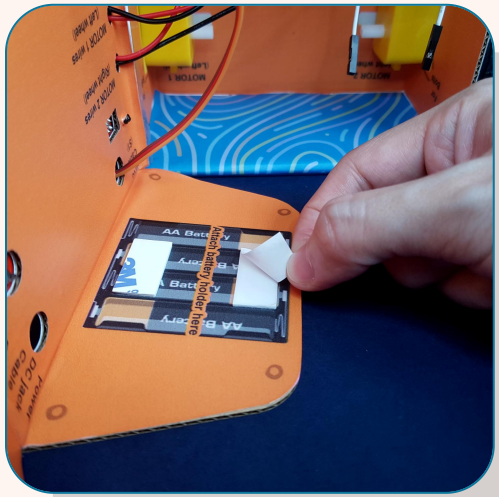


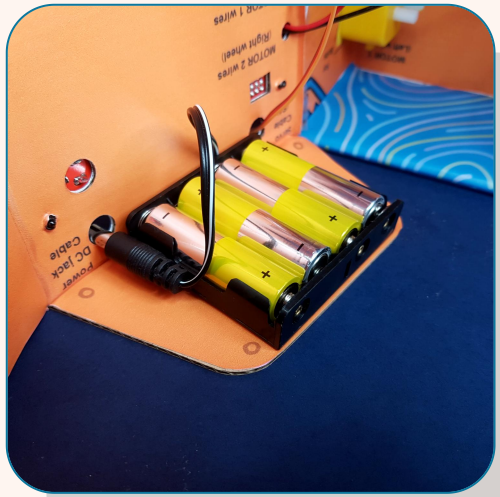


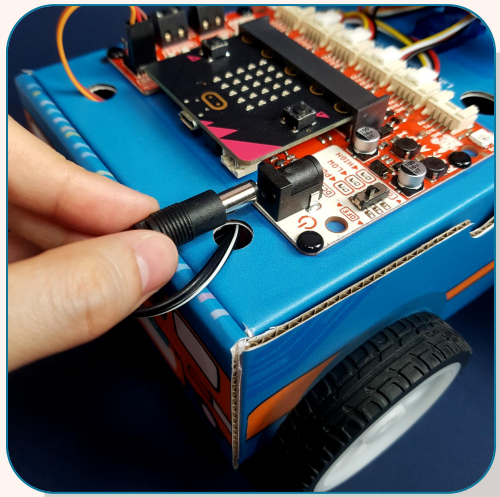
29 Feed the ultrasonic sensor cable through the hole.

30 Fold the cardboard along the crease lines and insert flaps into their slots to form the head.

31 Connect the ultrasonic sensor cable to port **P2:P12** on REKA:BIT.







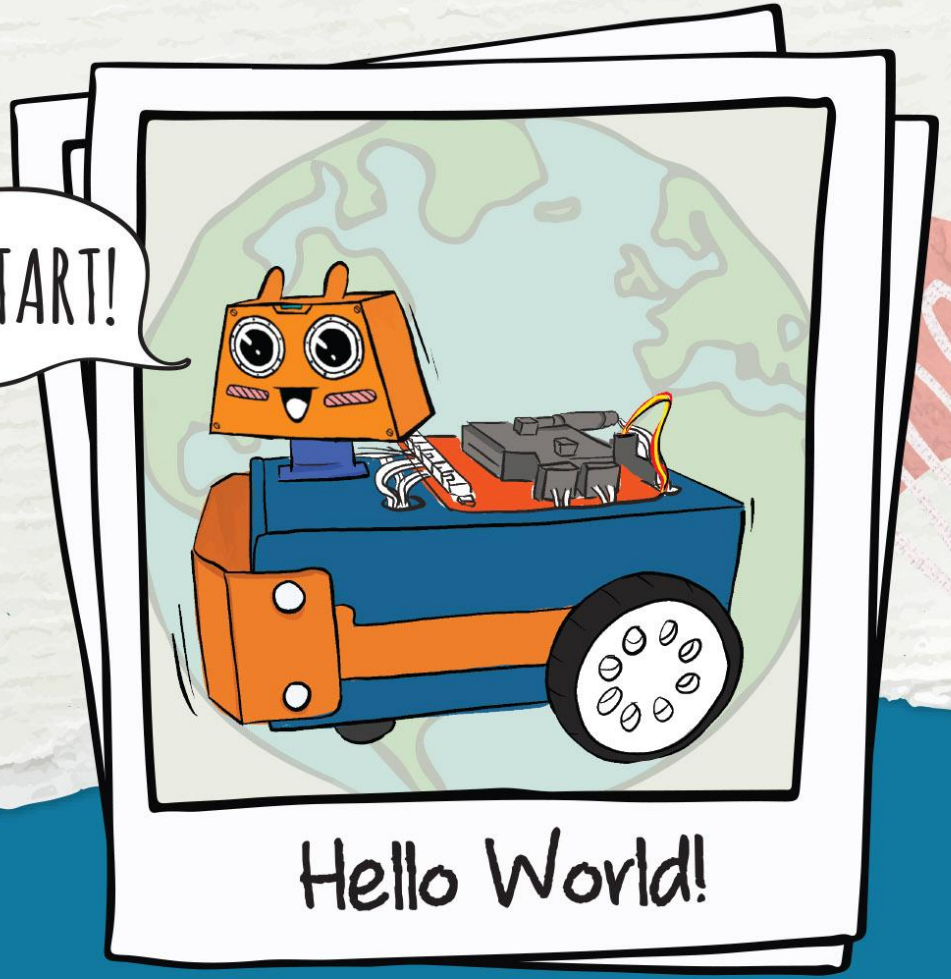
32 Use two pieces of double-sided tape to attach the battery holder to the flap of the box.

33 Insert four AA batteries into the battery holder. Feed the cable through the hole and then close the cover.

34 Connect the cable to the power jack.

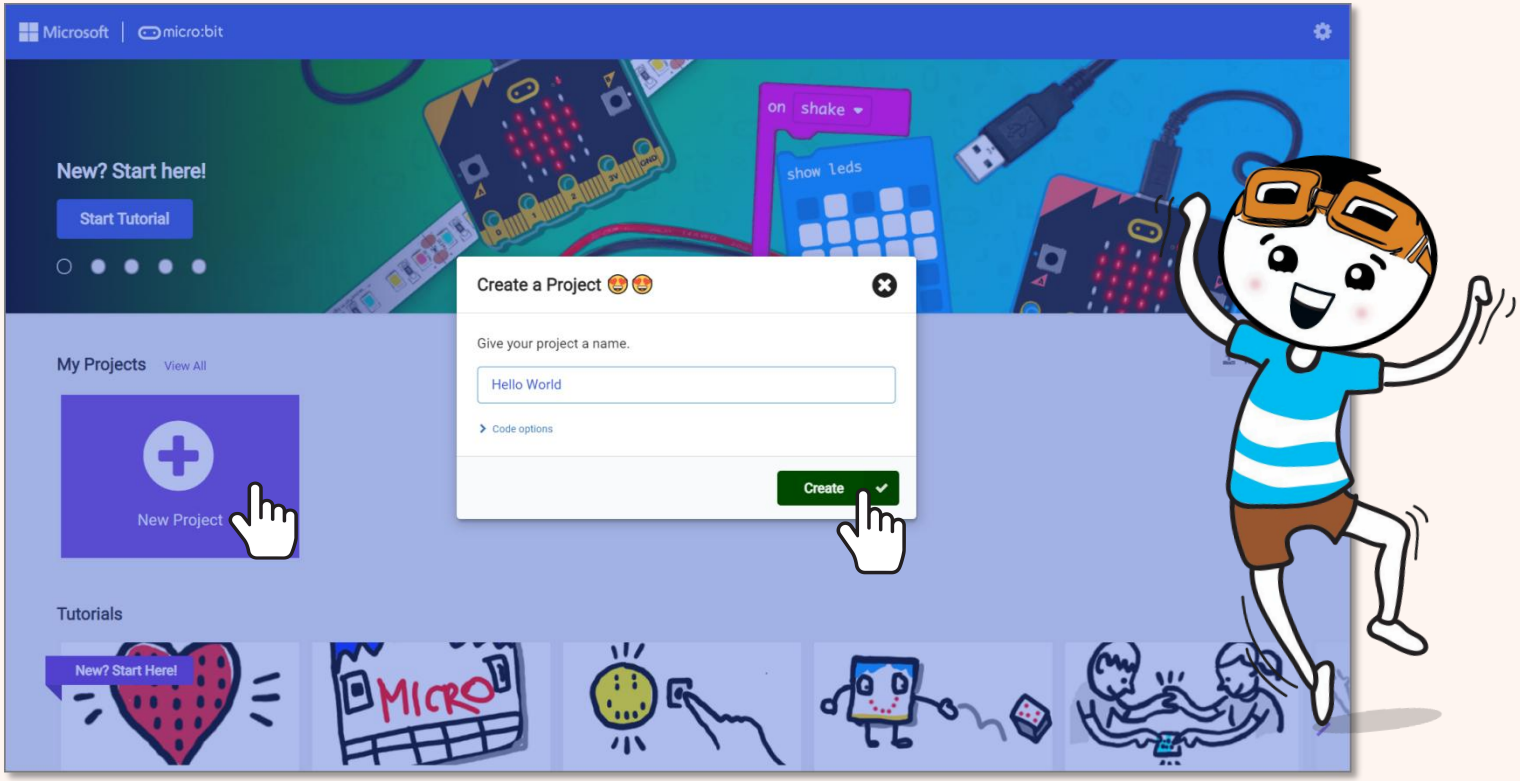
CHAPTER 1

https://link.cytron.io
/zoombit-chapter-1

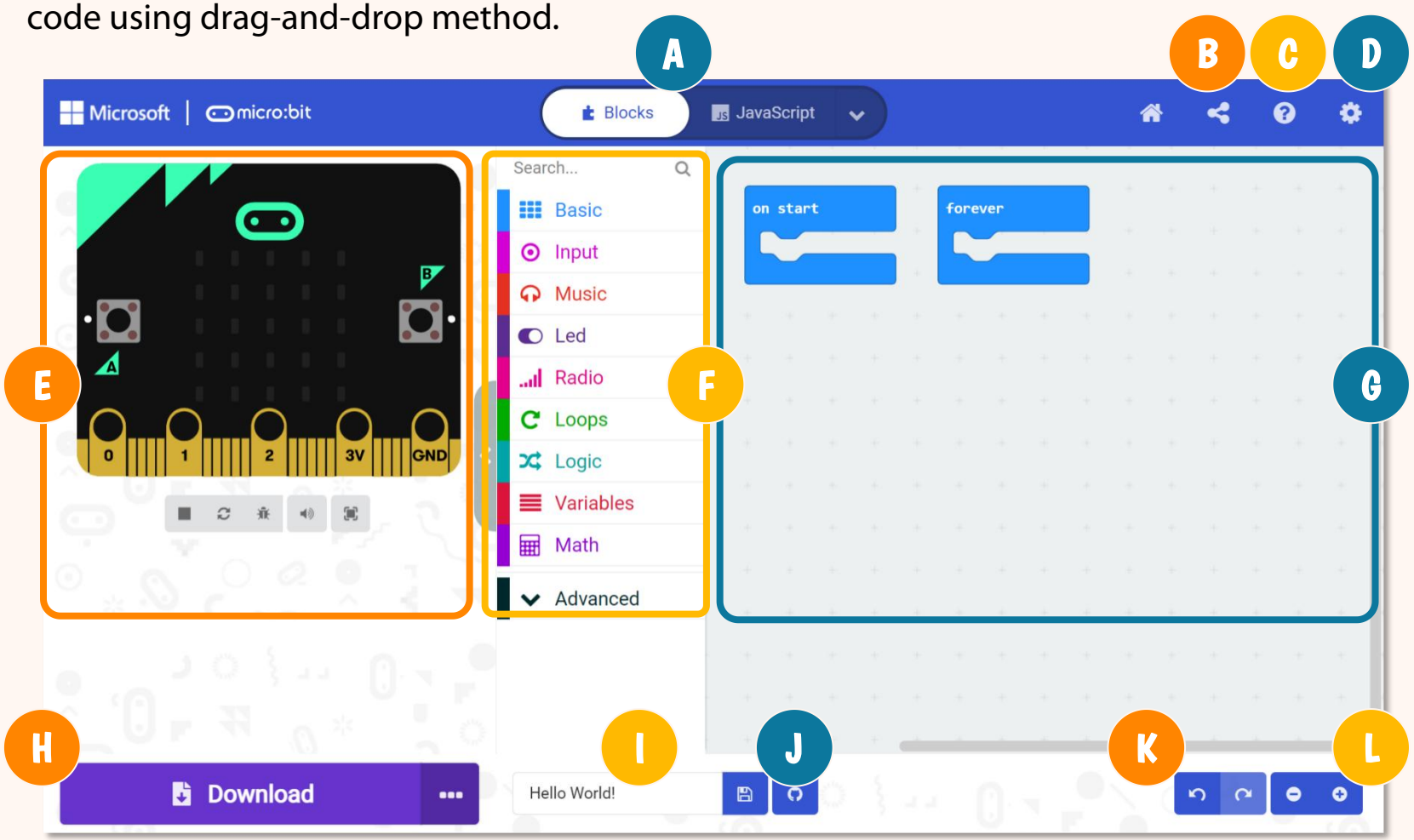


1 Open your browser and go to https://makecode.microbit.org.

2 Click **[New Project]**. Name your project and then click **[Create]**.



You will see this **Microsoft MakeCode Editor** page which allows you to easily build your code using drag-and-drop method.



A Choose to program in Blocks, JavaScript or Python.

B Publish and share your project.

C Open Help menu.

D Change settings, add extensions, connect device, etc.

E **Simulator** - Show you a simulation of your code.

F **Toolbox / Category Drawers** - Get the coding blocks that you need here. Click to see available coding blocks for each category.

G **Programming Workspace** - Build your code here by snapping blocks together.

H Click to download your code to ZOOM:BIT.

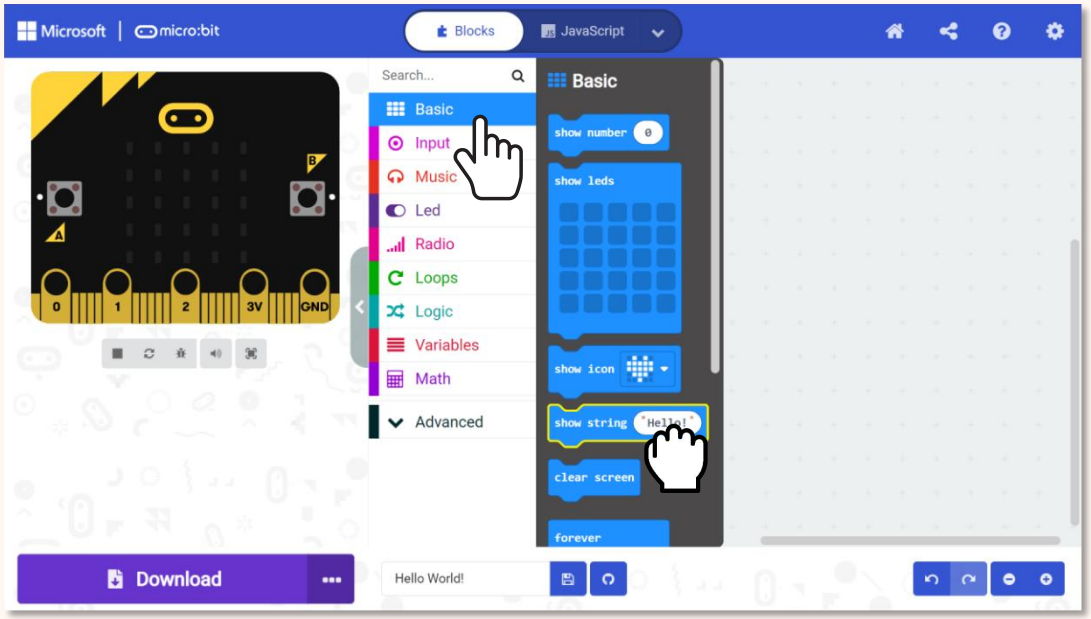
I Name and save current project to your computer.

J Create GitHub repository.

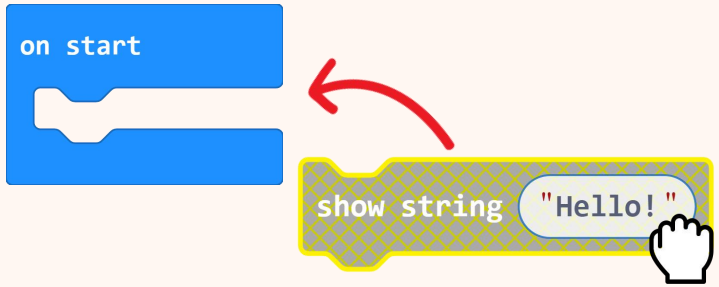
K Undo / Redo

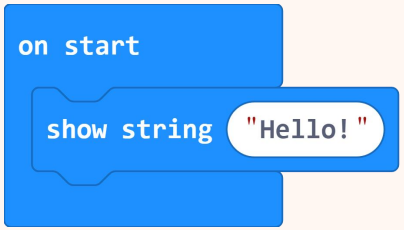
L Zoom in / out.

3 Click [Basic] category and select [show string (“Hello!”)] block.

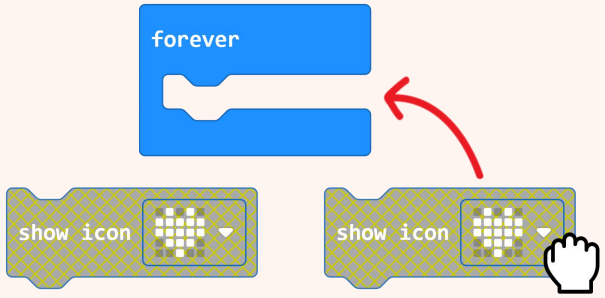


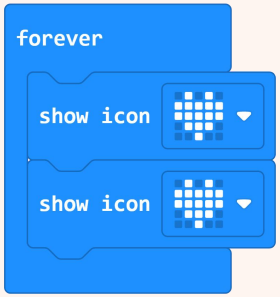
4 Click and snap the [show string (“Hello!”)] block to the [on start] block.



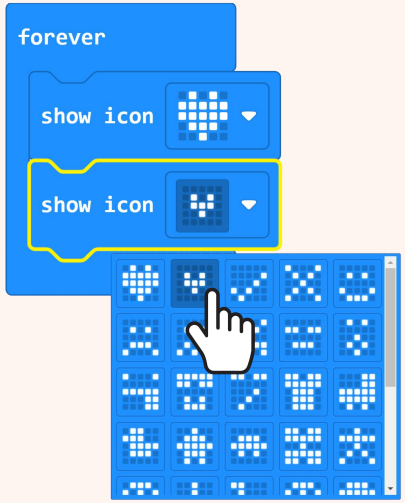


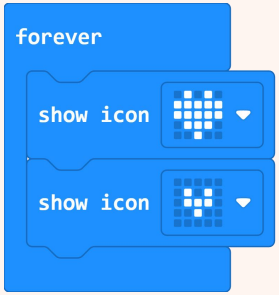
5 Click [Basic] category again and select [show icon] block. Repeat to add another [show icon] block. Click and snap both the [show icon] blocks to the [forever] block.



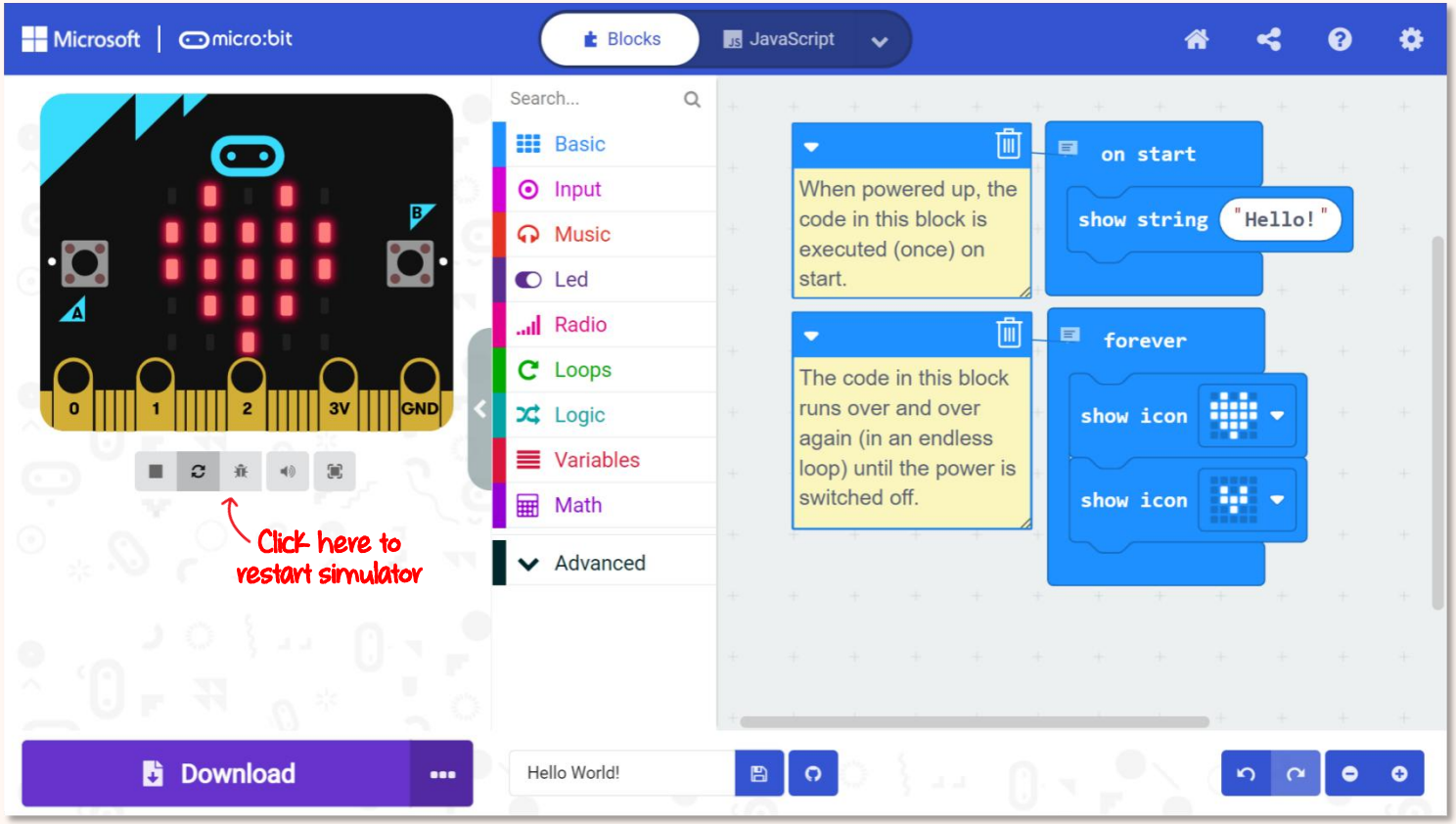


6 Click on the icon of the second [show icon] block and select the '**small heart**' image from the pop-up window.

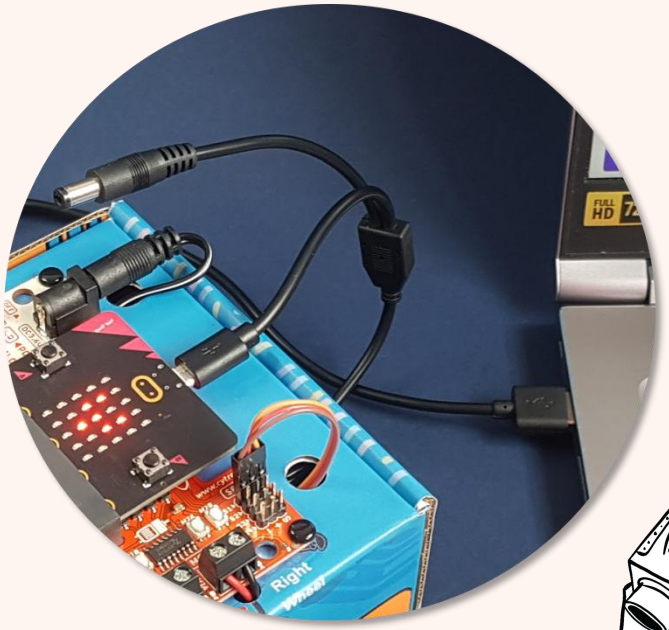


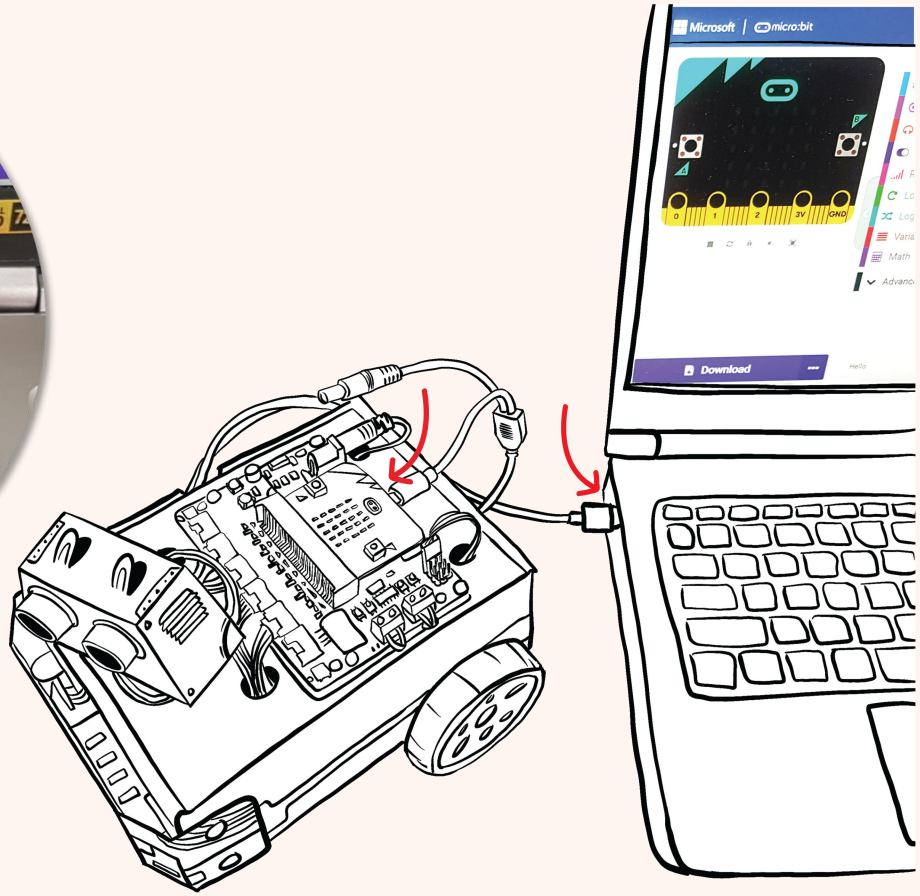


You can view a simulation of your code in MakeCode Editor. You will notice that the "Hello!" text only scrolls across the display one time but the beating heart animation keeps looping over dan over again. Do you know why?



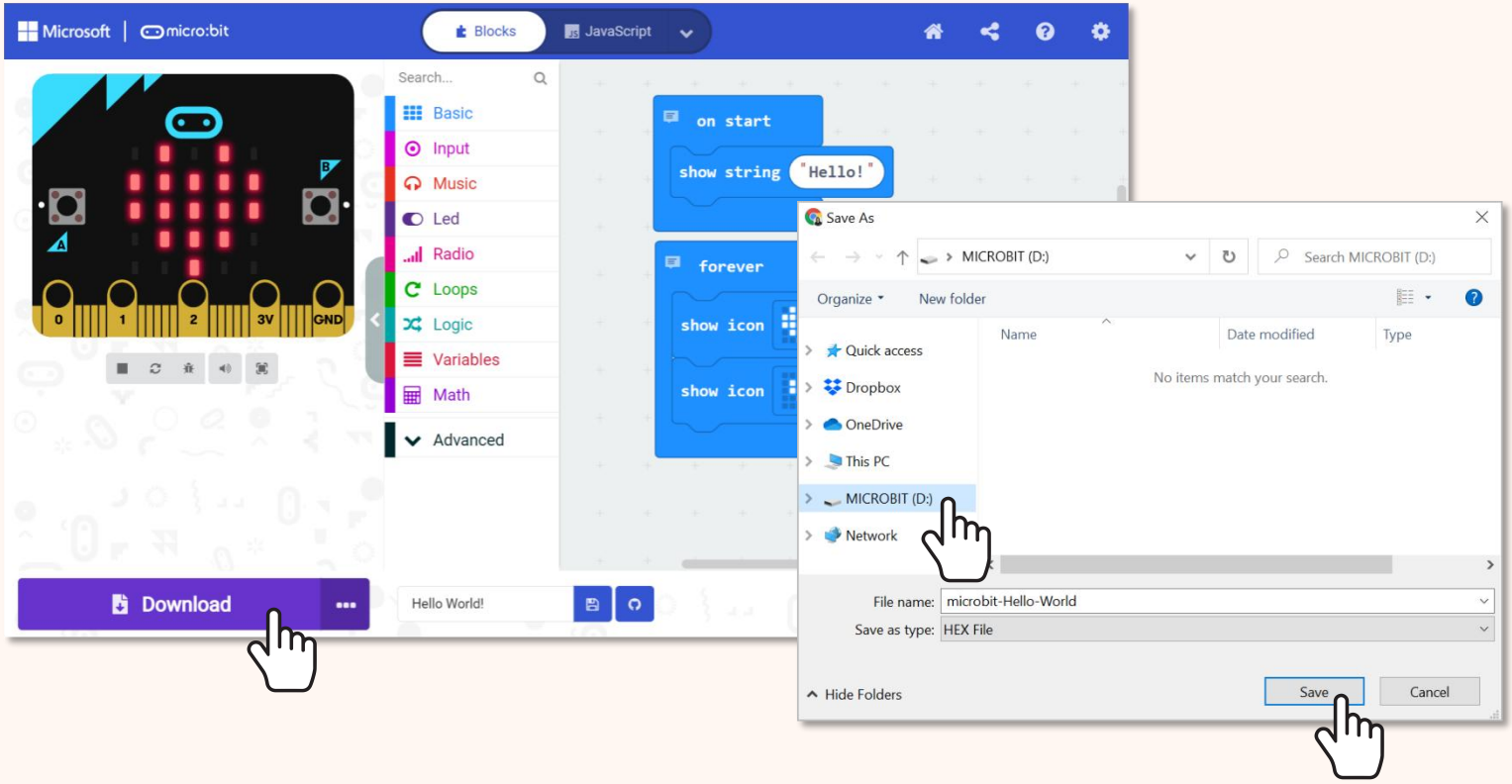
7 Connect the USB cable to your computer and robot as shown below.



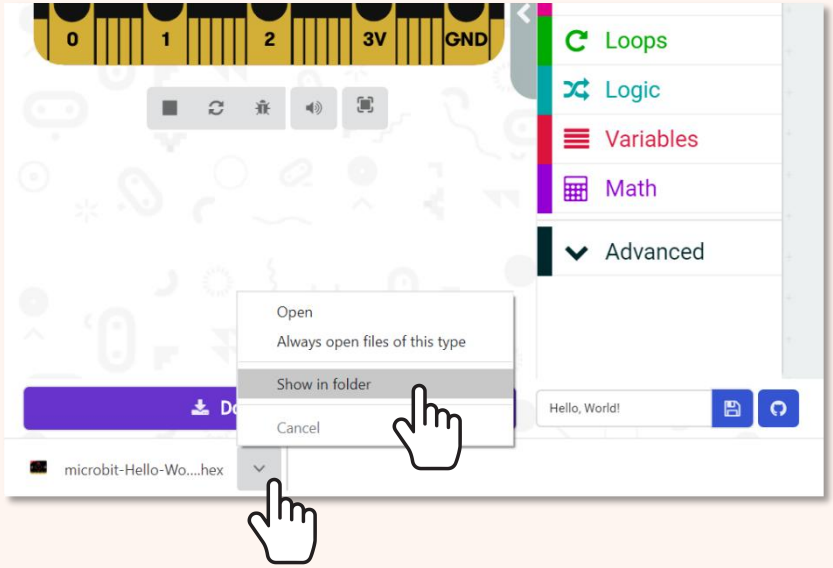


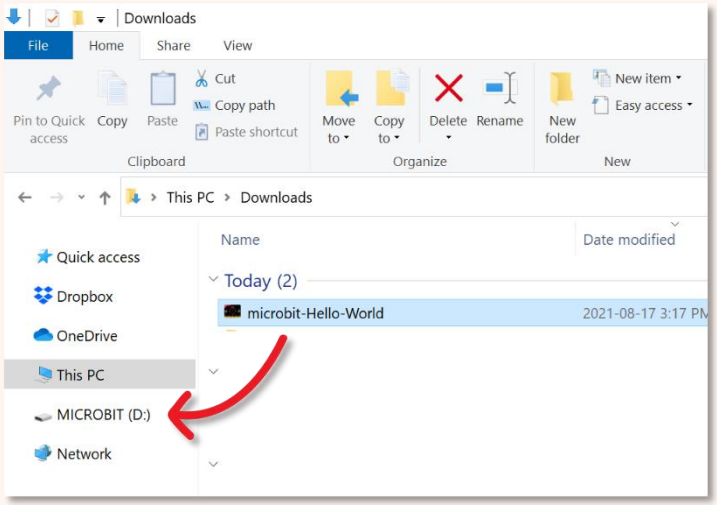
8 Click **[Download]** button. In the pop up window, choose to download your project to the MICROBIT drive and then click **[Save]**.

9 Click **[Done]** to close the pop-up window when it says "**Download completed**".

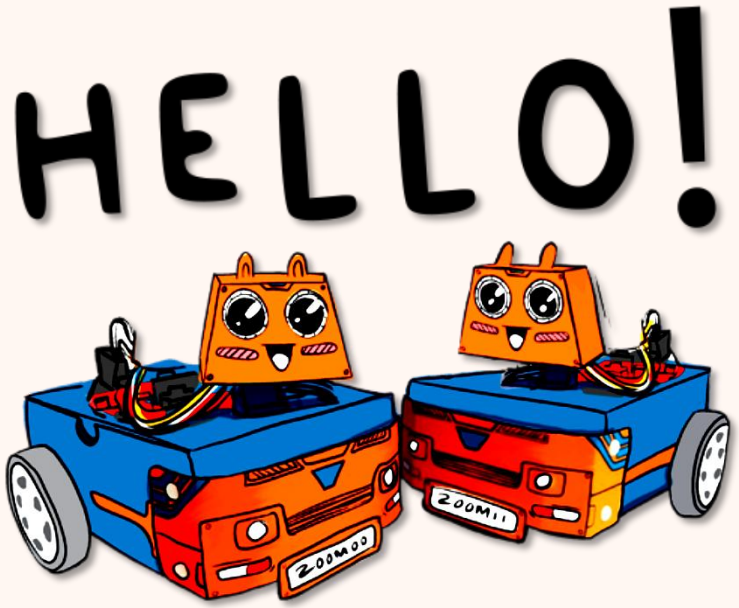


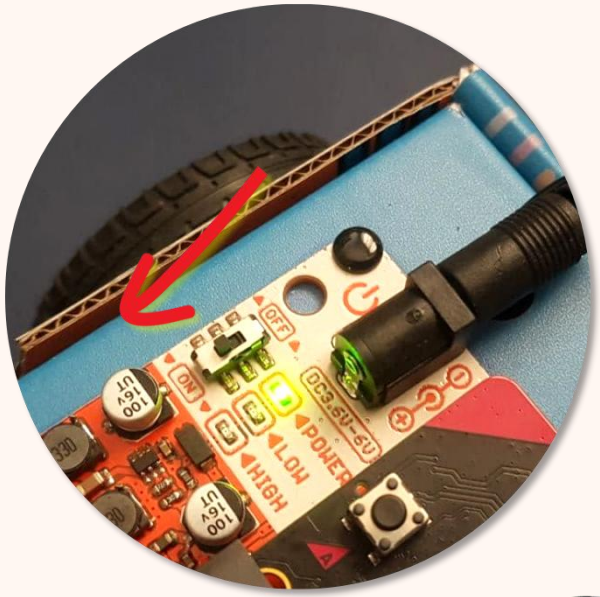
Notes: If the pop-up window does not appear, it means that the file has been automatically downloaded to the location where your browser is set to save downloads. Right-click on the downloaded .hex file which will appear at the bottom of the window and select 'Show in folder'. Click and drag the downloaded "microbit-xxxx.hex" file to the MICROBIT drive, as if you were copying a file to a flash drive.





10 Unplug the USB cable and power up ZOOM:BIT by sliding the power switch to ON.



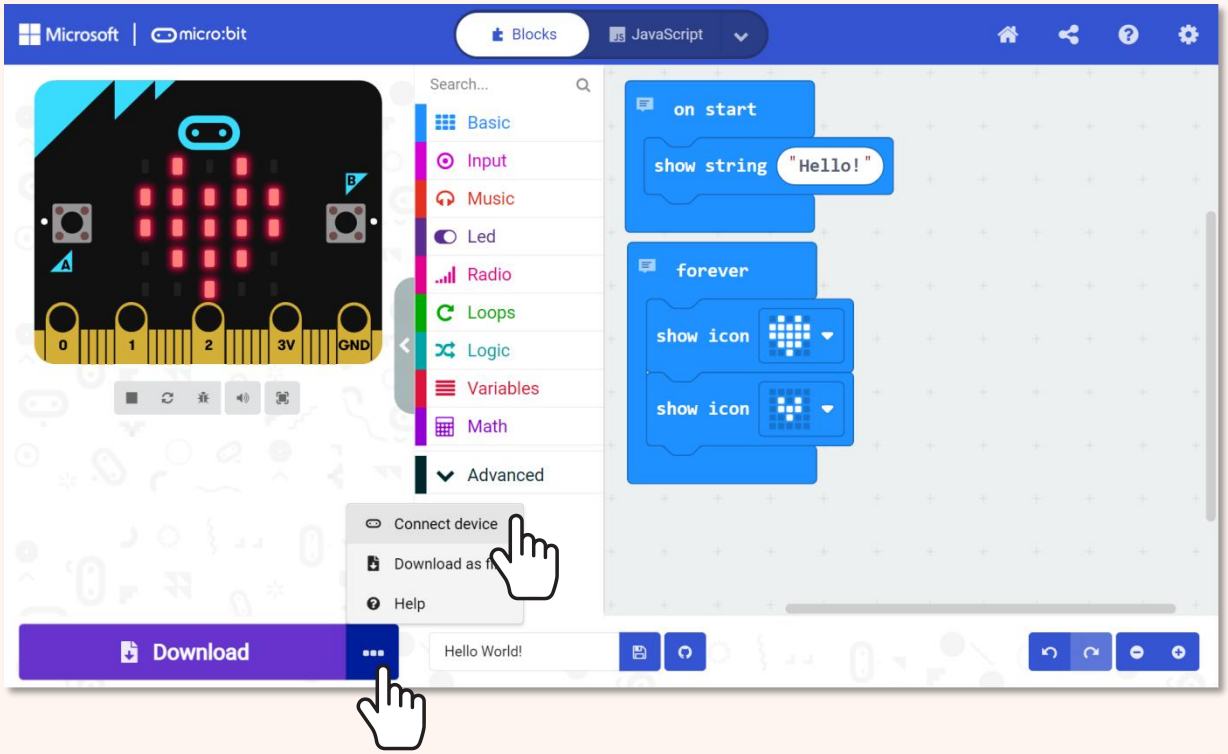


Do you see "Hello!" scroll across the LED matrix followed by a beating heart animation?
If you missed it, you can slide the power switch to OFF, and then turn it ON again to reset.

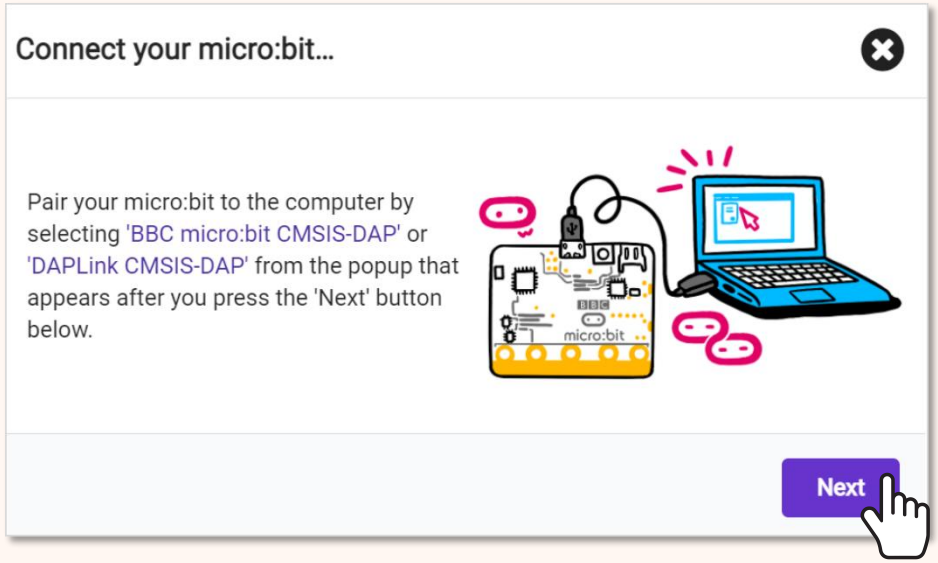
Do You Know?

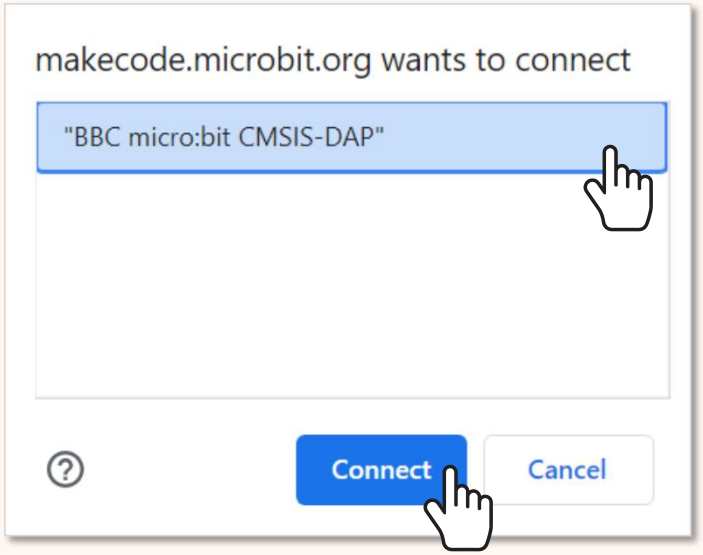
You can "connect device" to make it easier to download your code. After you've connected your device, you can directly flash code to your ZOOM:BIT with just ONE single click. Yeah! ~

11 Plug in ZOOM:BIT to your PC. Click the three dots next to the **[Download]** button, and then select **[Connect device]**.



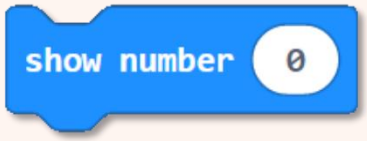
12 Follow the on-screen instructions. Select **'BBC micro:bit CMSIS-DAP'** or **'DAPLink CMSIS-DAP'** from the list and then click **[Connect]**.



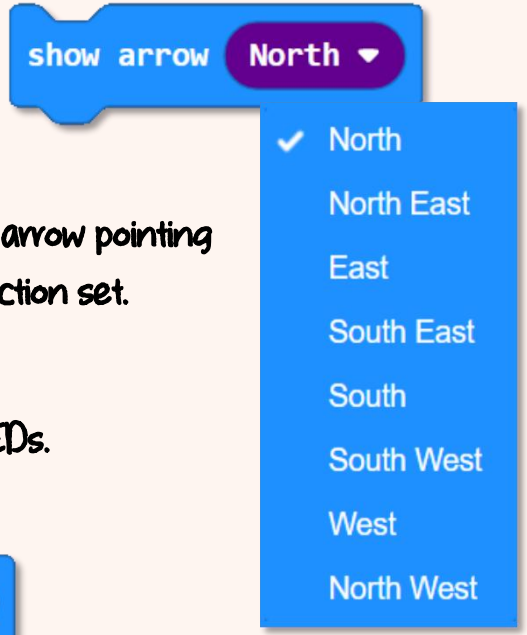


Notes: You need to use either the new Edge or Chrome browser, and have the latest firmware on your micro:bit device. If you have problems connecting your device, you can refer to https://makecode.microbit.org/device/usb/webusb/troubleshoot for more info.

Discover More Blocks

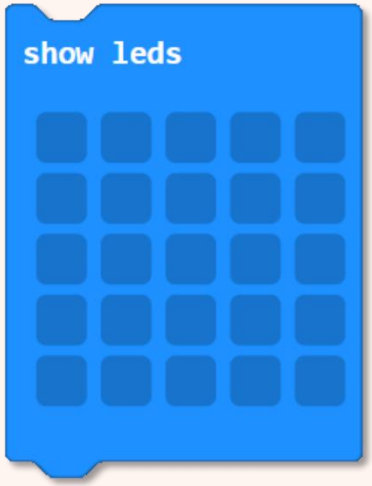


Display number.

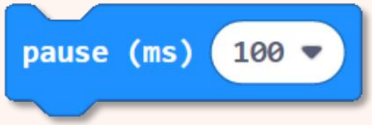


Display an arrow pointing to the direction set.

Turn off all LEDs.



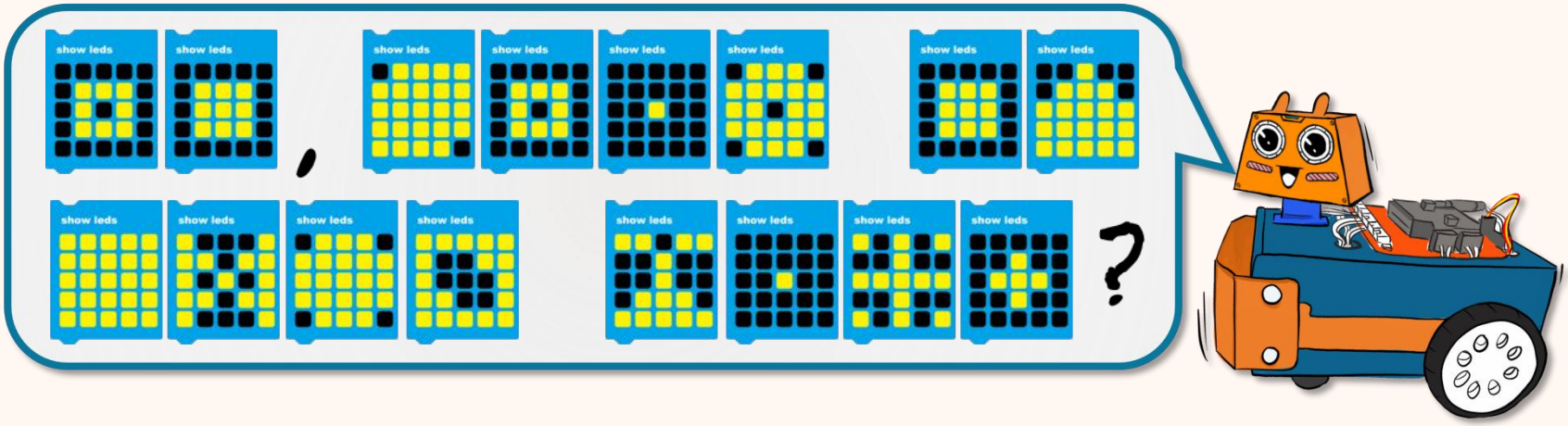
Design image to be displayed on the LED matrix. Click the rounded squares to turn on/off the corresponding LEDs.



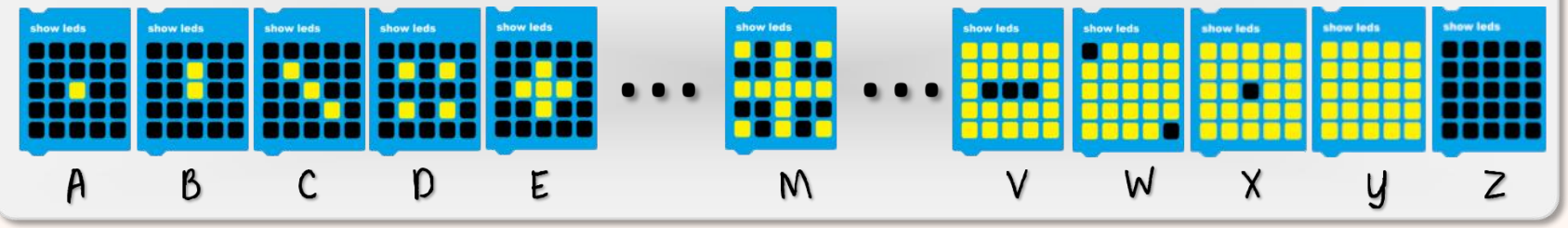
Add a delay to slow down the program, i.e. pause for the number of milliseconds (ms) that is set.
1000 millisecond = 1 second

Here's a FUN challenge for you !

Can you decode what ZOOM:BIT is asking? Program your robot to reply using the same secret code.



Here are some clues to help you out ~



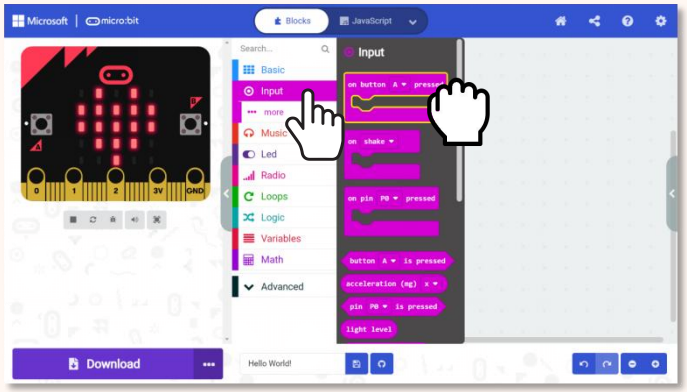
CHAPTER 2

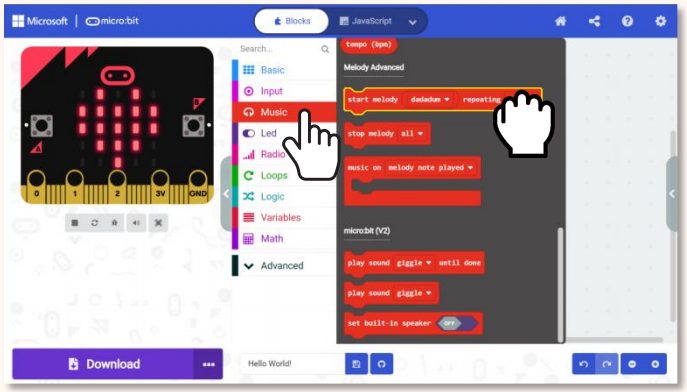
https://link.cytron.io
/zoombit-chapter-2

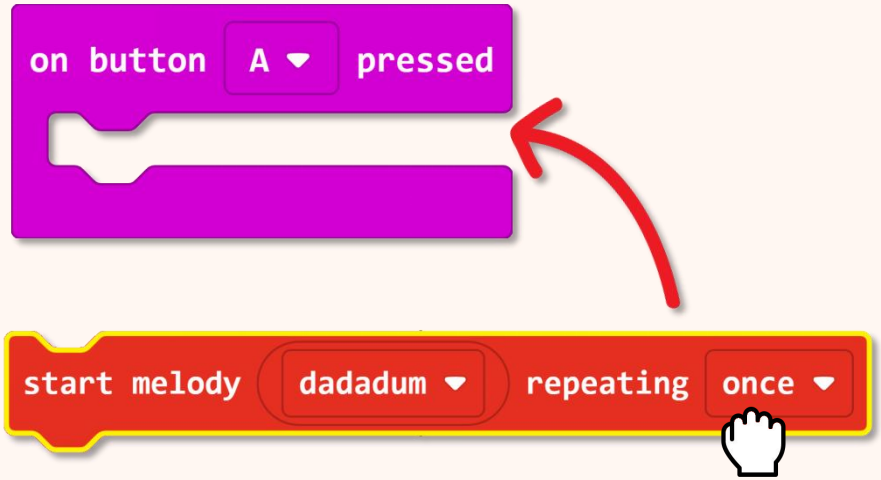
1 Click **[Input]** category and then select **[on button (A) pressed]** block.

2 Click **[Music]** category and then select **[start melody (dadadum) repeating (once)]** block.

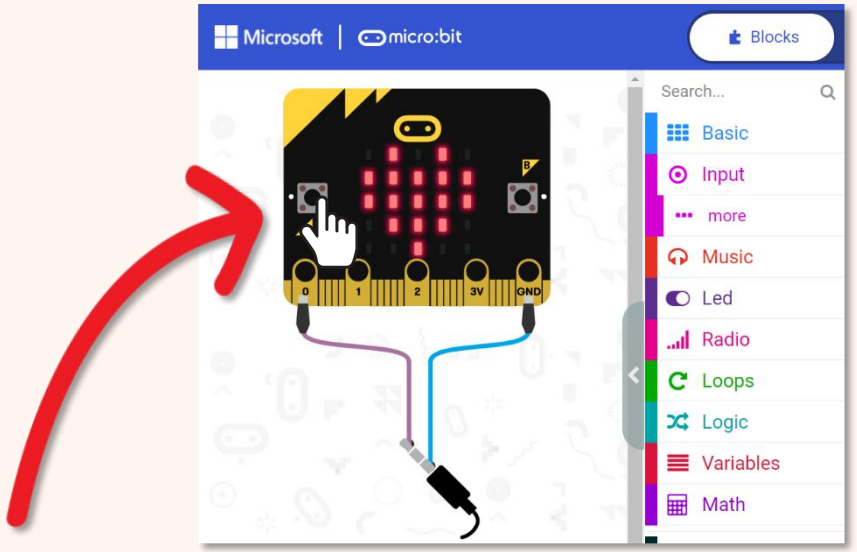
Let's teach ZOOM:BIT to sing... Do Re Mi ~
You can create a new project or continue adding blocks to your earlier code.







3 Click on [dadadum] and select **'birthday'** melody from the drop down list.

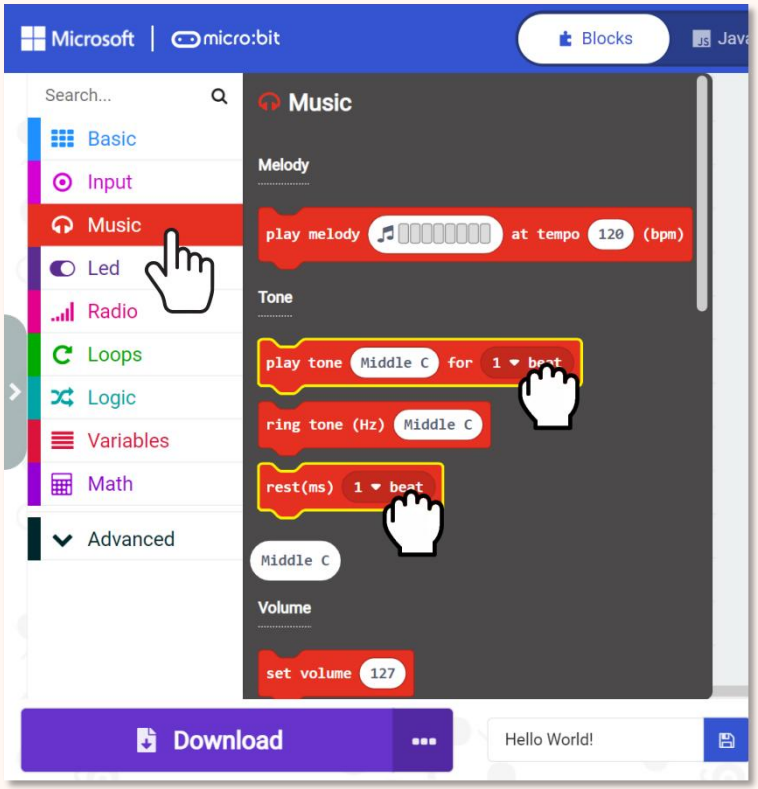


Click on Button A of your on-screen simulator. Do you hear a familiar tune? Have fun checking out the other melodies too~

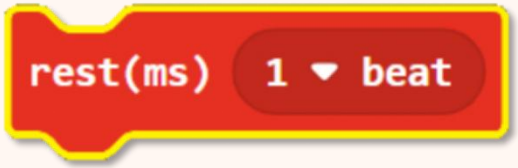
Notes: Make sure your computer speakers are turned ON.

Do You Know?

Besides the list of preset melodies, you can also program ZOOM:BIT to play any song you like. However, you will need to teach it note by note using [play tone (middle C) for (1 beat)] and [rest (ms) (1 beat)] blocks from [Music] category.

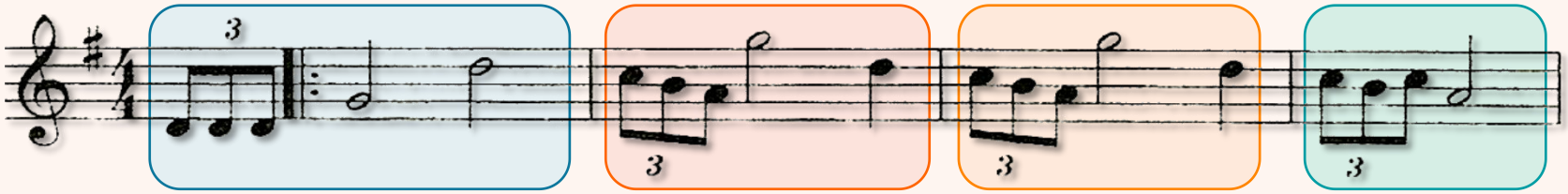


Which note to play, and for how long (duration)



Pause, i.e. do not play any note, for the duration set.

Let's try to program ZOOM:BIT to play the opening bars of the STAR WARS theme song~



Tone	Middle D	Middle D	Middle D	Middle G	High D
Beat	1/3	1/3	1/3	2	2

Tone	High C	Middle B	Middle A	High G	High D
Beat	1/3	1/3	1/3	2	1

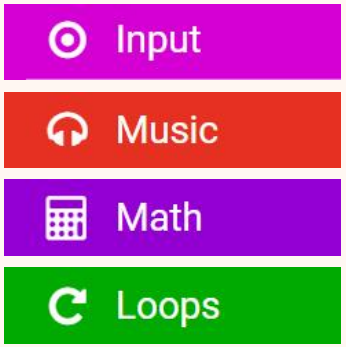
Tone	High C	Middle B	Middle A	High G	High D
Beat	1/3	1/3	1/3	2	1

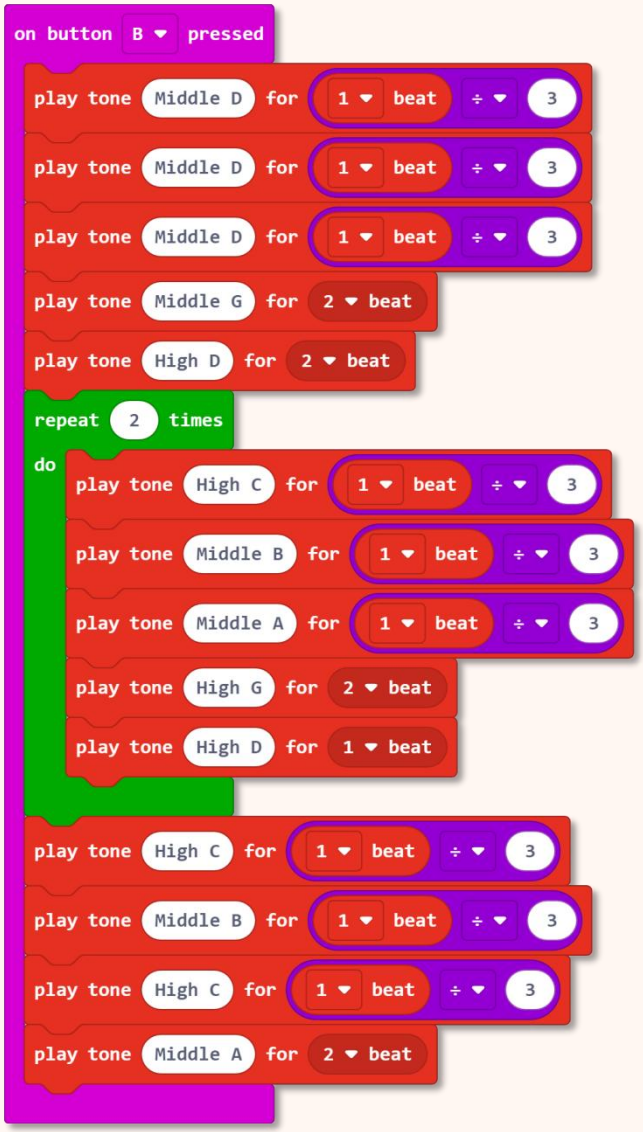
These two lines are the same. You can use a loop block to make your code more compact.

Tone	High C	Middle B	High C	Middle A
Beat	1/3	1/3	1/3	2

4

Add the following blocks to your code.
You can find the blocks you need from the category drawers with the same colour.



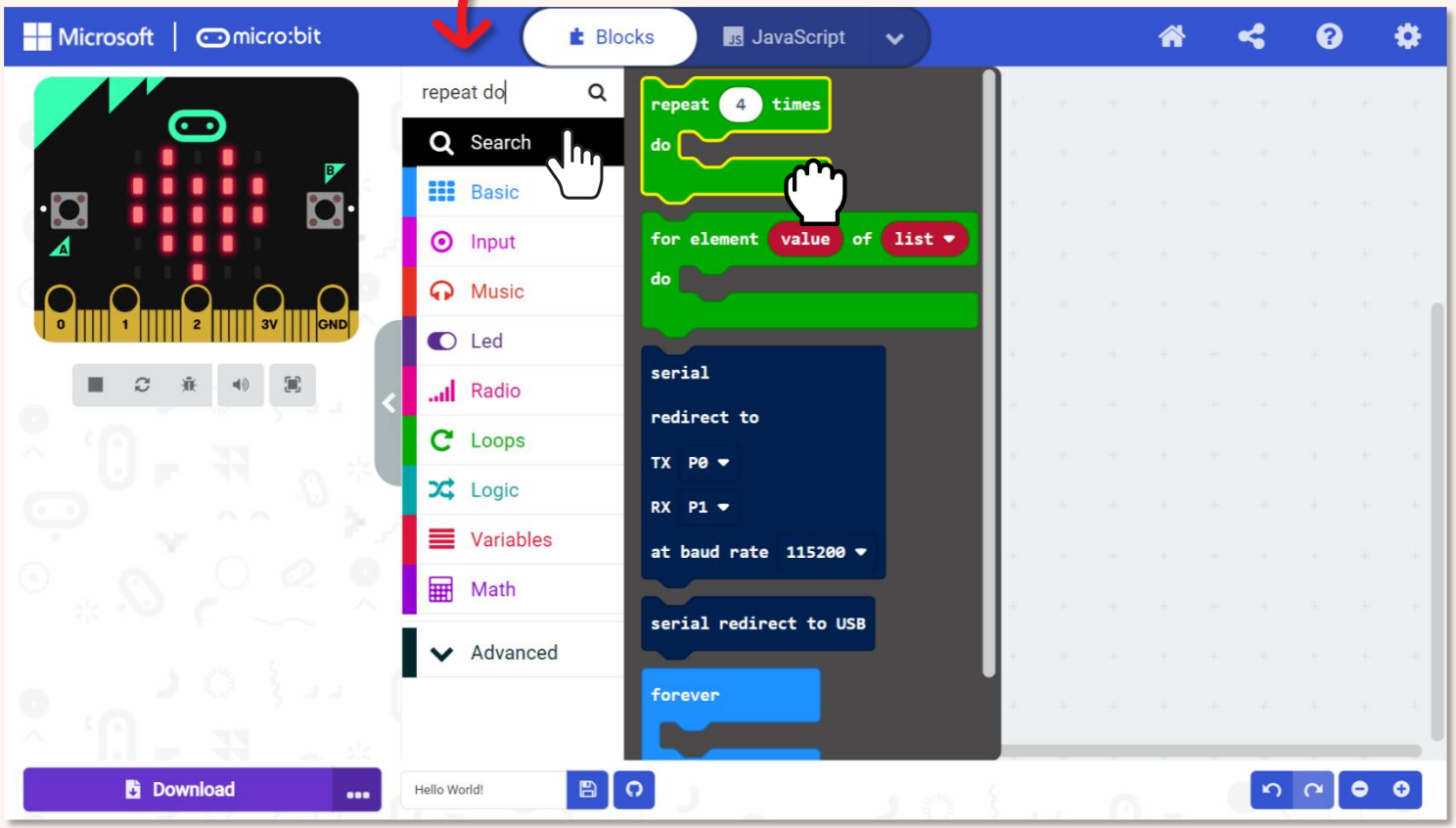


Do You Know?

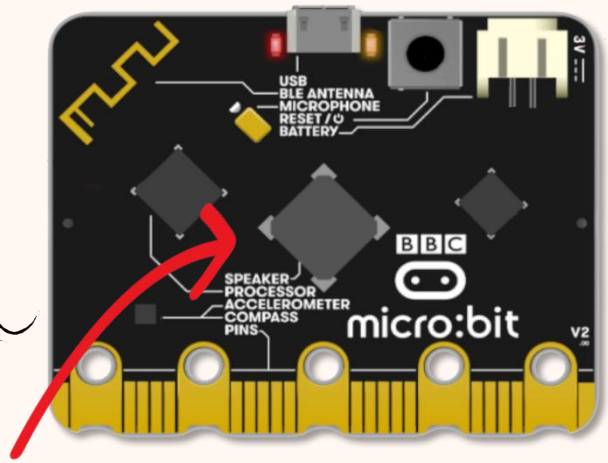
All coding blocks are colour coded. You can find blocks you need in the category drawer with the same colour.

If you need more guidance, you can go to https://link.cytron.io/zoombit-tutorial-2 for step-by-step guide to build the code.

Do You Know? You can also type keywords in the search box to find the blocks you need.



5 Flash the completed code to your ZOOM:BIT. Power it up and press Button B.



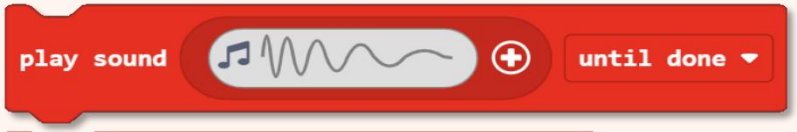
Notes: Your ZOOM:BIT (with micro:bit V2) can "sing" and make music because it has a built-in speaker which enables it to produce sounds. If you're using micro:bit v1 (without built-in speaker), you need to plug in a Grove buzzer to Port P0:P1 to play sounds. You can refer to https://link.cytron.io/zoombit-grove-buzzer for more details.

Discover More Blocks

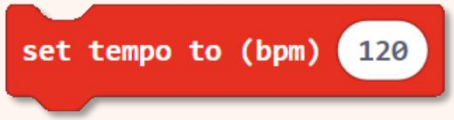
Use Melody Editor to compose and play a short melody of notes at the tempo set

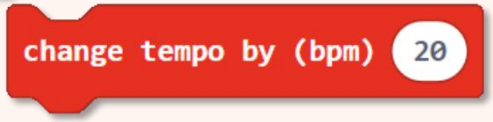
Use conditions, such as melody note played or melody ended etc, as event triggers in your code.

Stop all the sounds that are currently playing and also sounds waiting to play.

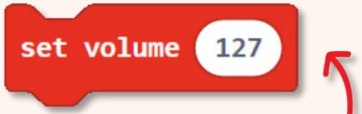


Play sound expressions (for micro:bit V2 only).

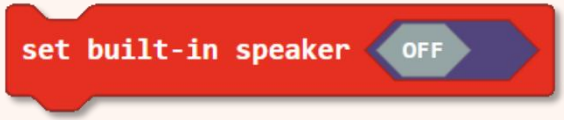




Set or change the "tempo" (i.e., the pace of your song). The higher the bpm (beats per minute), the faster or livelier your tune will be.

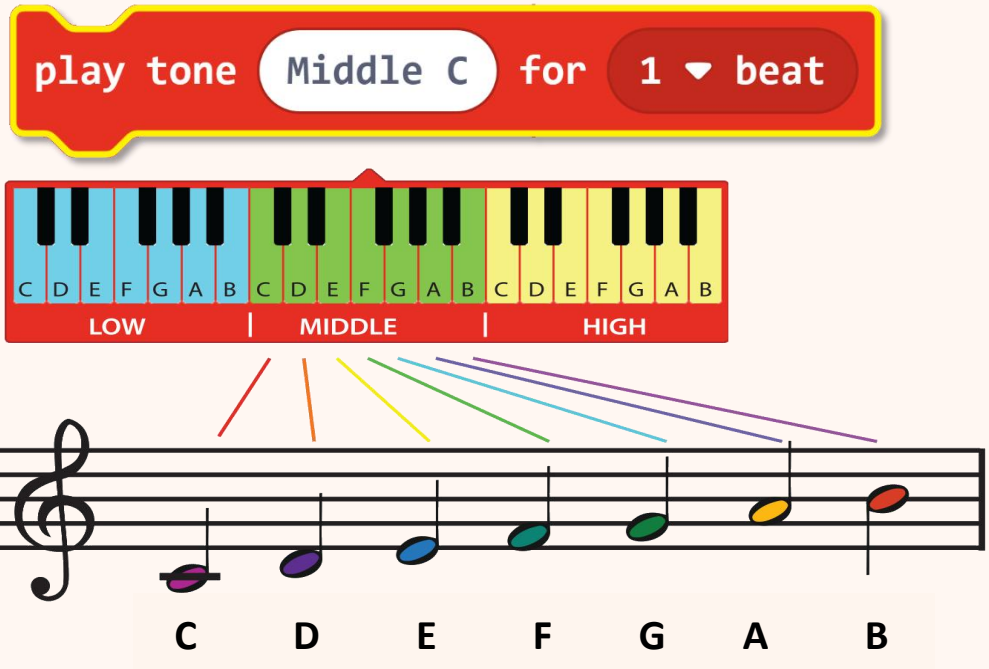


Set the volume, ranging from 0 to 255 (max loudness).



Enable, or disable, the built-in speaker to play sounds (for micro:bit V2 only).

You can program ZOOM:BIT to play other songs if you know how to read music. Here's a simple guide to help you to "decode" a music score.



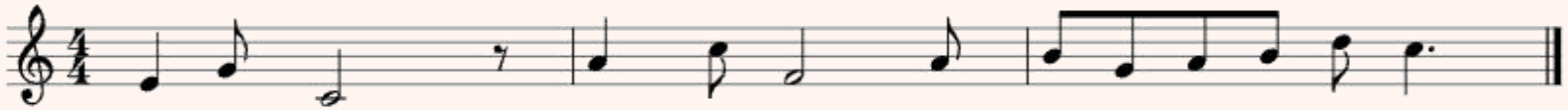
Note	Rest	Duration
𝅝	𝄻	4 beats
𝅗𝅥	𝄼	2 beats
♩	𝄽 or 𝄾	1 beat
♪	𝄿	1/2 beat
𝅘𝅥𝅯	𝅀	1/4 beat

The position of a music note on the staff (i.e. the five horizontal lines) tells us which tone to play. The higher the note sits on the staff, the higher the pitch/frequency of the sound, and vice versa..

Different musical notations are used to tell us the duration (i.e. how long) a note is to be played.

Here's a FUN challenge for you !

Teach ZOOM:BIT to "sing" your favourite song. You'll need to program it, note by note.
If you don't have a song in mind, then try the following :-



Tone	Middle E	Middle G	Middle C	Rest	Middle A	High C	Middle F	Middle A
Beat	1	1/2	2	1/2	1	1/2	2	1/2

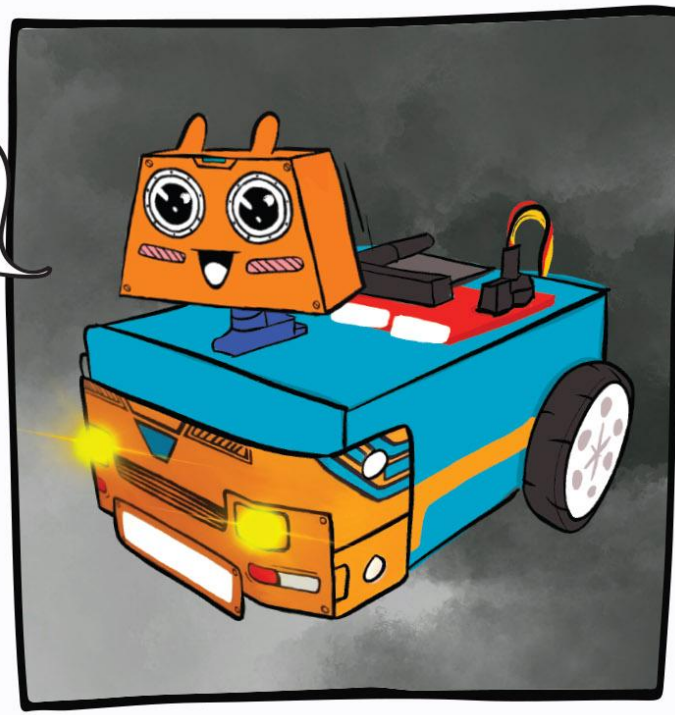
Tone	Middle B	Middle G	Middle A	Middle B	High D	High C
Beat	1/2	1/2	1/2	1/2	1/2	1 1/2

It's a very familiar tune. Can you guess what melody it is?

CHAPTER 3

https://link.cytron.io/zoombit-chapter-3

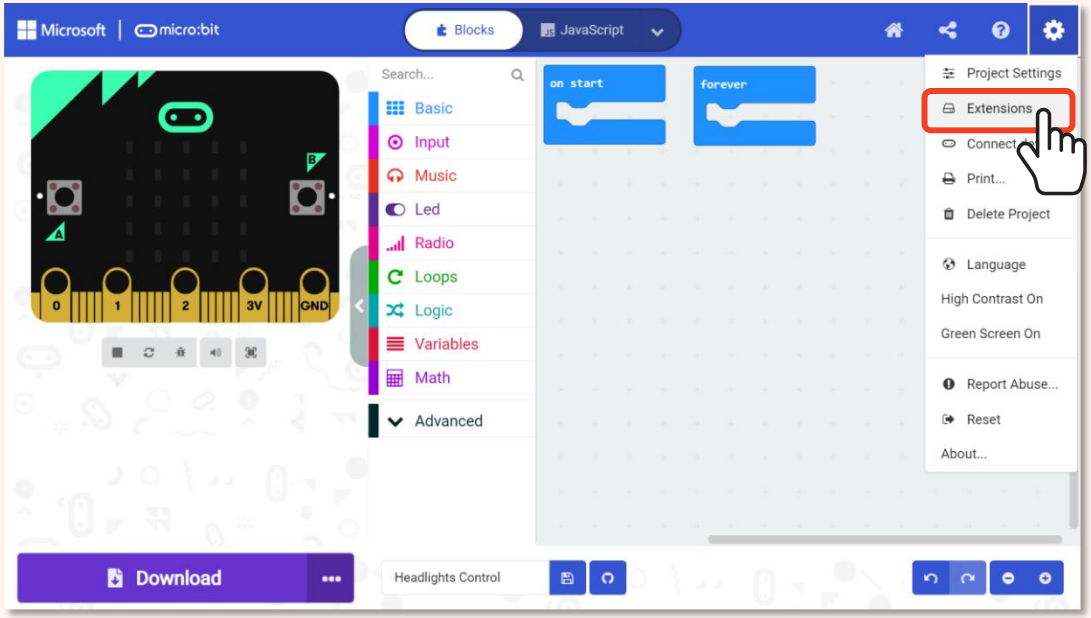
LET'S START!



Turn Those Lights ON

Do You Know? The LED matrix on micro:bit can also function as light level sensor. Let's program ZOOM:BIT to automatically turn ON its headlights when the surrounding is dark, and turn OFF when it is bright.

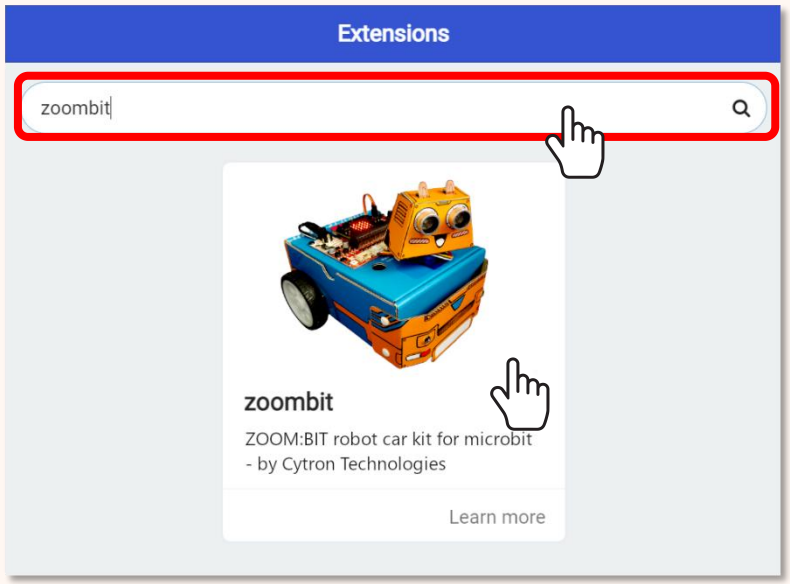
1 Create a new project in your MakeCode Editor. Click the cogwheel icon and then select **'Extensions'**. *You need Internet connection to add extensions.

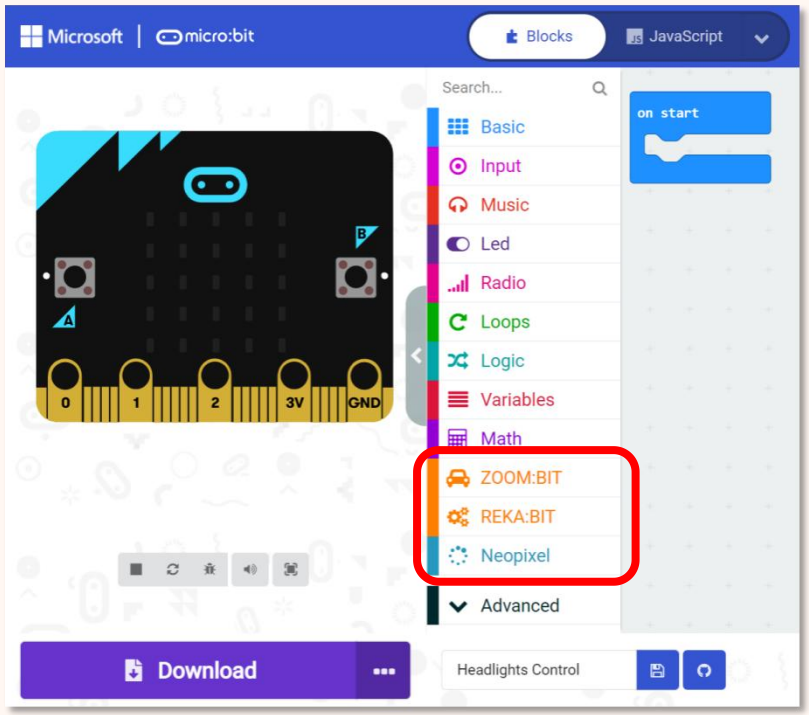


Extensions are sets of custom blocks that we add to MakeCode Editor to enable us to easily program micro:bit accessories, such as our ZOOM:BIT robot.

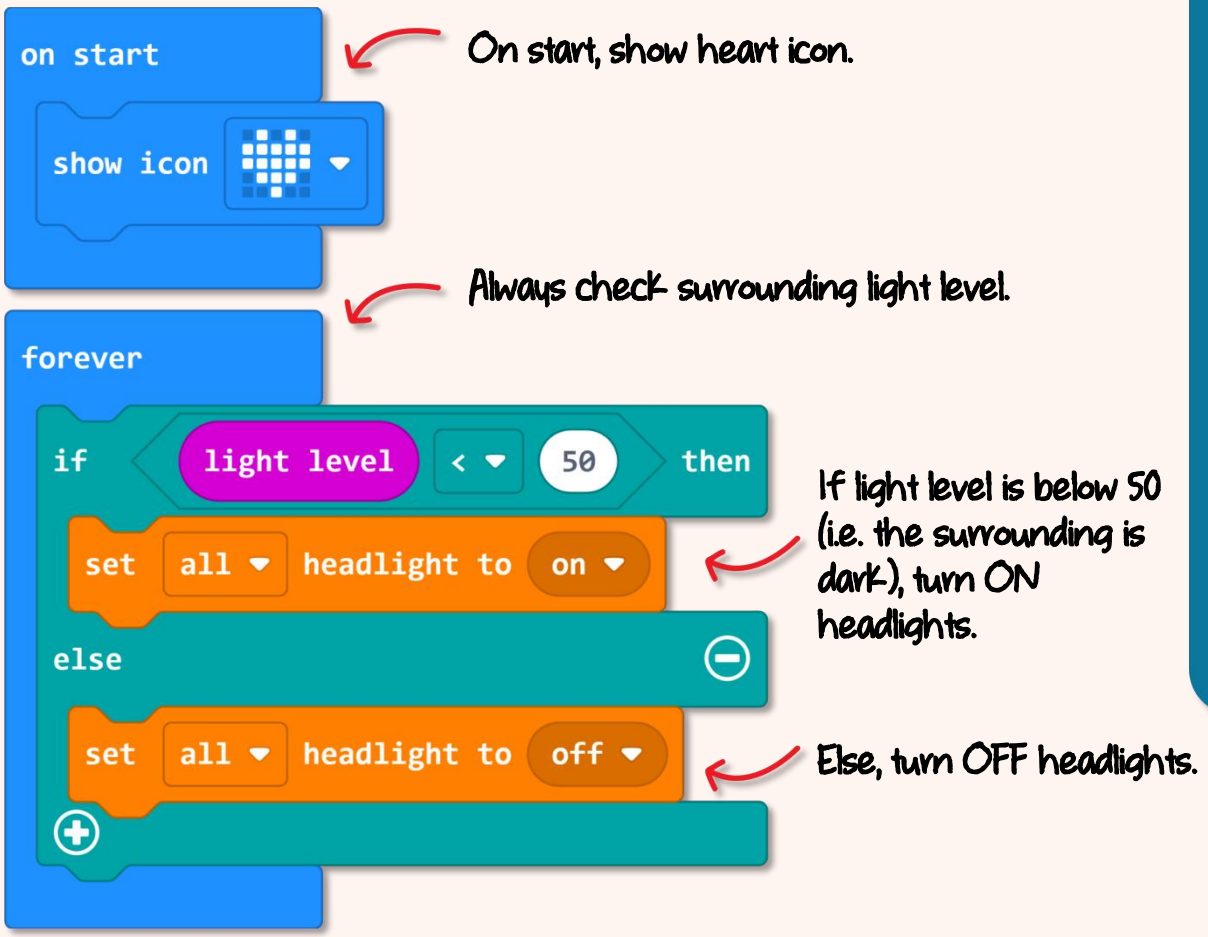
2 Type '**zoombit**' (or https://github.com/CytronTechnologies/pxt-zoombit) in the search box and click Enter.

3 Click to select **"zoombit"** extension. Wait for it to load and you'll notice the following new category drawers added to your MakeCode Editor.

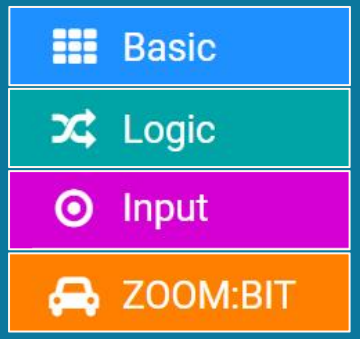




4 Build the following code.

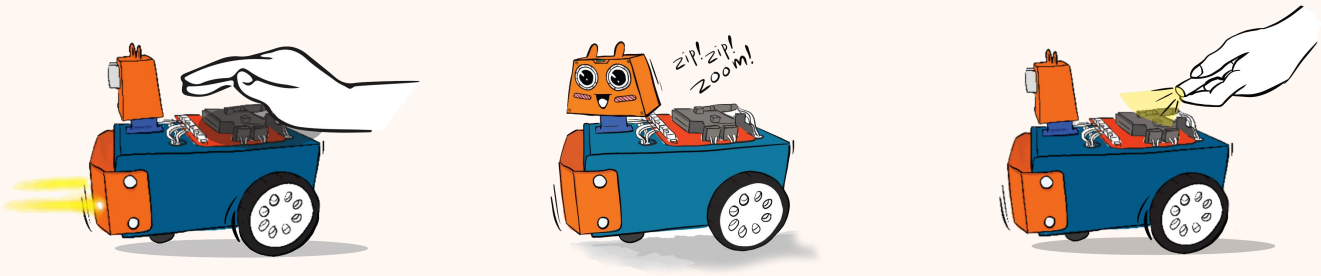


You can find the blocks you need from the following category drawers:



For more guidance, you can go to https://link.cytron.io/zoombit-tutorial-3 for step-by-step guide to build the code.

5 Download the code to your ZOOM:BIT and power it up. Observe its headlights.

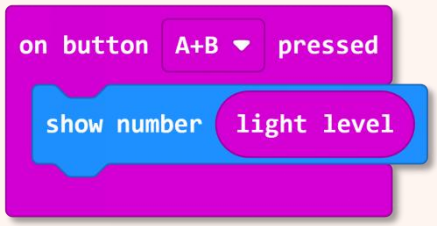


Does ZOOM:BIT turn on its headlights? If not, try to cast a shadow over the LED matrix. Next, try to shine a bright flashlight at the LED matrix; what do you observe?

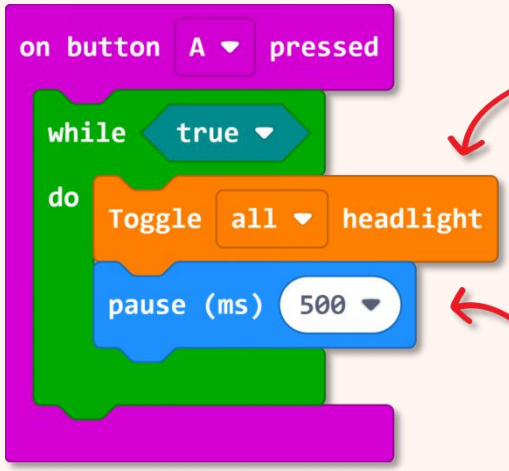
Notes: Light level reading ranges from 0 (no light detected) to 255 (maximum brightness).

Discover More Blocks

In your MakeCode Editor, create a new project, add the blocks below and then download to your ZOOM:BIT. What do you observe when you press Buttons A + B? How about Button A?



Get light level reading and display the value when Buttons A+B are pressed.



"Toggle" means to switch from one state to another. If the current state is ON, then it will switch to OFF, and vice versa.

This block slows down the program so that you can observe the headlights turning ON and OFF.

(i) What's the light level reading in your room now? What's the light level reading when you shine a bright light at the LED matrix?

* For accuracy, record at least 3-4 readings and then calculate the average value.

(ii) Do you see the headlights blinking after you press Button A? Power off to make it stop.

Here's a FUN challenge for you !

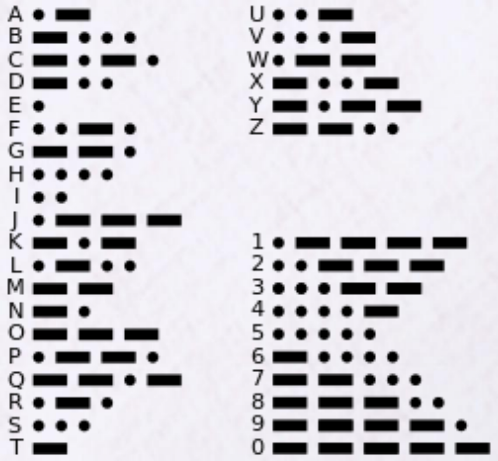
Teach ZOOM:BIT to communicate in Morse Code. Program your ZOOM:BIT to flash its headlights on button A or B pressed.

On Button A Pressed	Turn **ON** both headlights for **500ms** and then turn **OFF**	**Dot**
On Button B Pressed	Turn **ON** both headlights for **1500ms** and then turn **OFF**	**Dash**

Referring to the International Morse Code chart provided, can you get ZOOM:BIT to flash an S.O.S. message by pressing button A and button B in the correct sequence? Demo video available at https://link.cytron.io/zoombit-morse-code

International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.



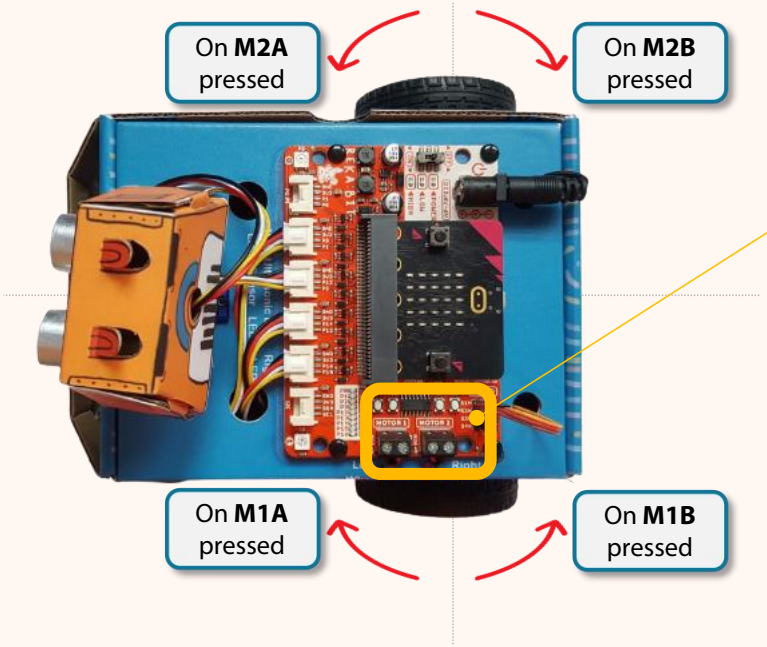
CHAPTER 4

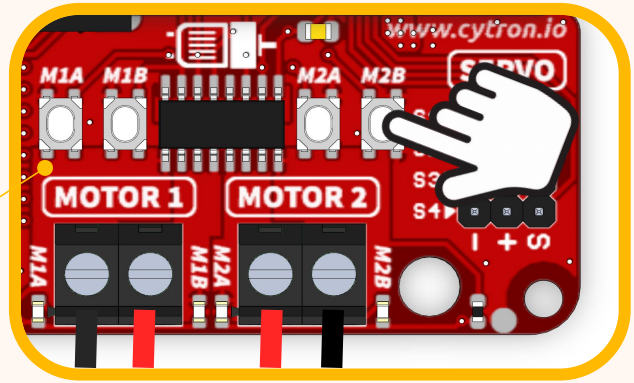
https://link.cytron.io
/zoombit-chapter-4

Before we start programming ZOOM:BIT to move, let's check to make sure that we've wired it up correctly.

1. Slide the power switch to ON.

2. Press buttons - M1A, M1B, M2A and M2B on REKA:BIT, one by one and observe the spinning direction of the wheels.





Notes:

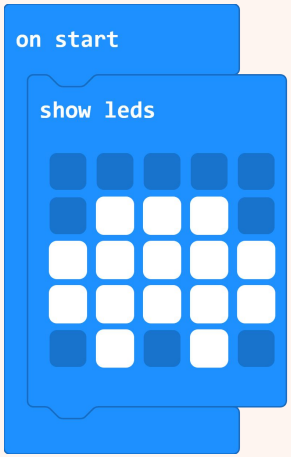
If the wheels are not spinning in the directions as shown by the red arrows, you need to check and correct the DC motor connections. You can refer to pp. 5-6.

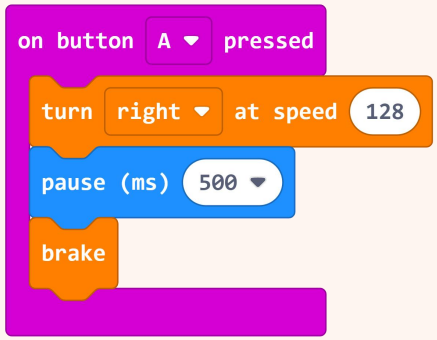
Now we're ready to program ZOOM:BIT to move around... Let's start! Zip zip Zoom ~

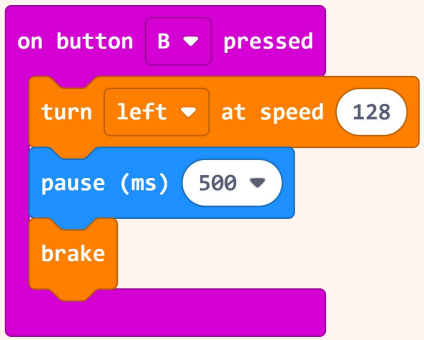
1. Create a new project in your MakeCode Editor and add ZOOM:BIT extension (you can refer to pp. 44-45).

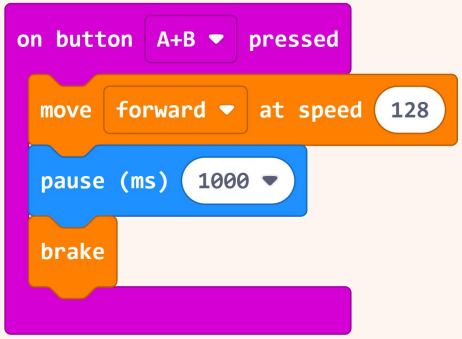
2. Build the code below. You can get the blocks you need from these category drawers:

Basic | Input | ZOOM:BIT







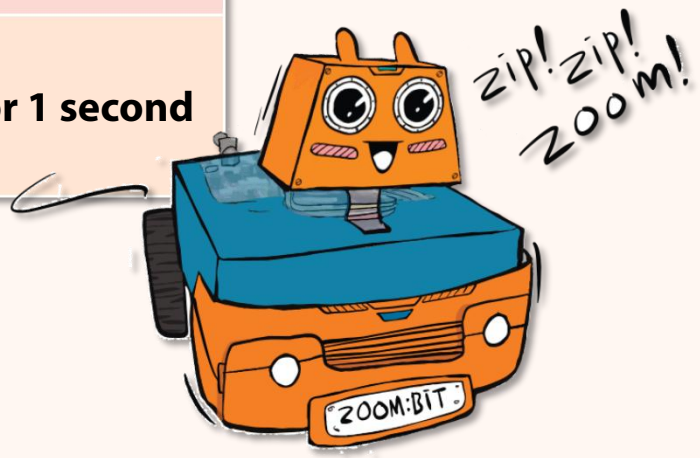


You can go to https://link.cytron.io/zoombit-tutorial-4 for step-by-step guide to build the code if you need more guidance.

3 Download the code to ZOOM:BIT and power it up.

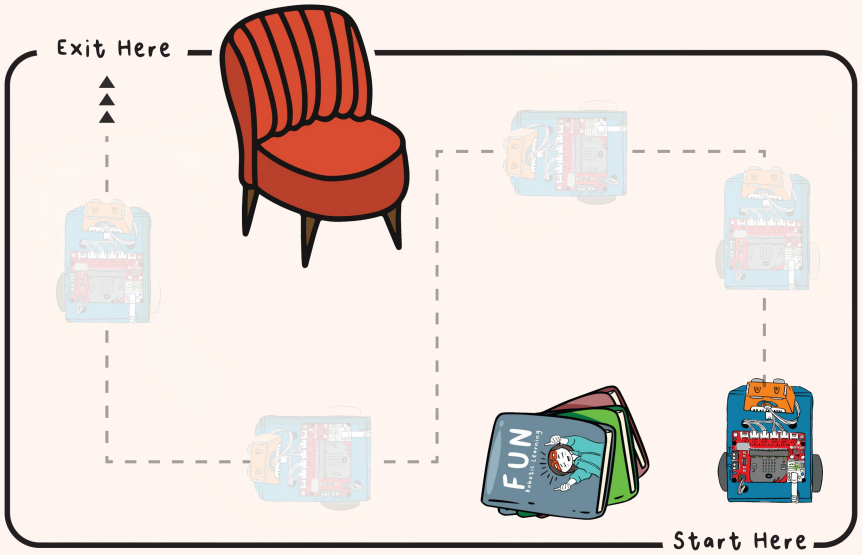
4 Press Button A, Button B, and then buttons A+B at the same time. Observe ZOOM:BIT's response.

On Button A Pressed	**Turn right for 500ms.**
On Button B Pressed	**Turn left for 500ms.**
On Buttons A+B Pressed	**Move forward for 1 second**



Here's a FUN challenge for you !

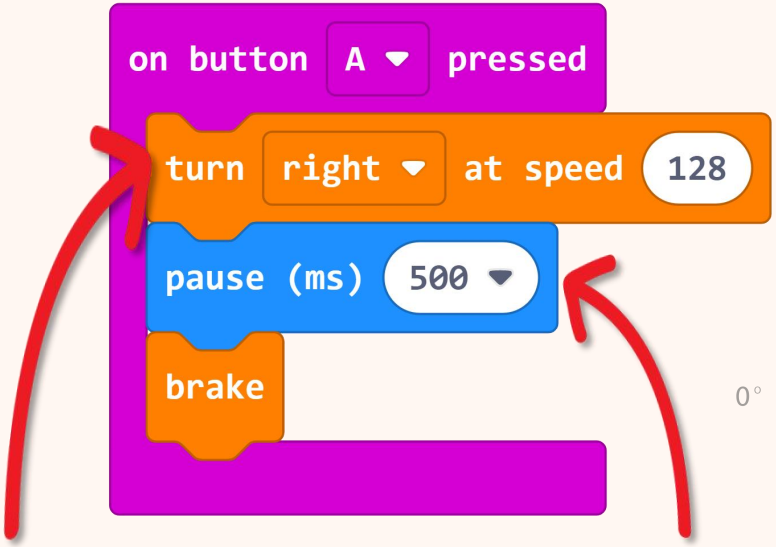
Find an open space and set up an obstacle course by randomly placing objects, such as chairs, books or cardboard boxes, along ZOOM:BIT's path. Challenge your siblings or friends to manually guide ZOOM:BIT to navigate its way out.

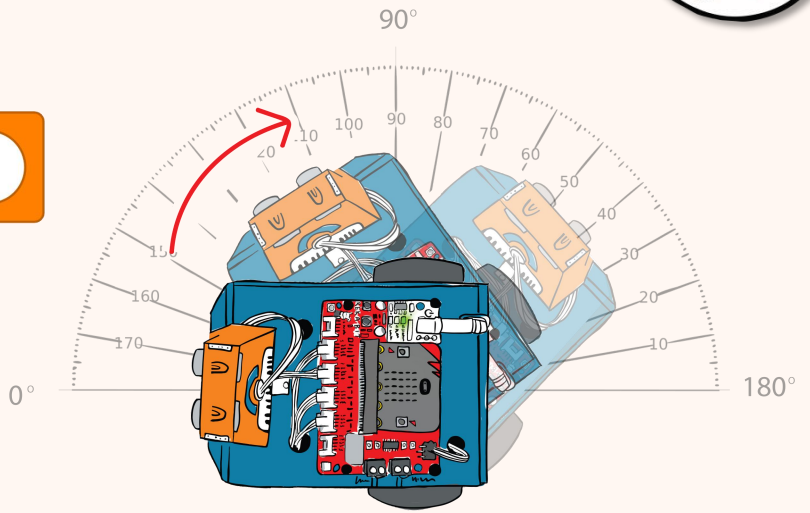


- Press A+B to move forward.
- Press Button A to turn right.
- Press Button B to turn left.

The challenger who takes the shortest time (or the least number of moves) to guide ZOOM:BIT out from the obstacle course is the WINNER!

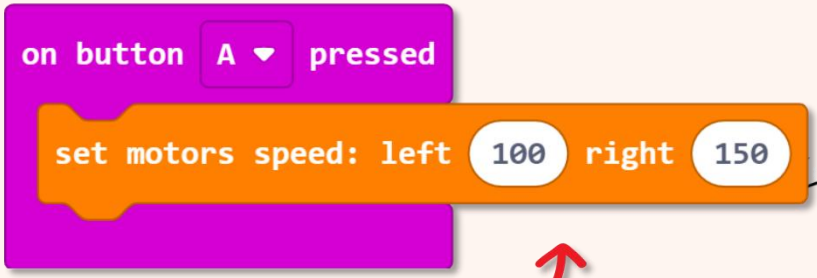
Do You Know? You can change the speed and delay settings to control the rotation angle. Try different values to complete the following table.



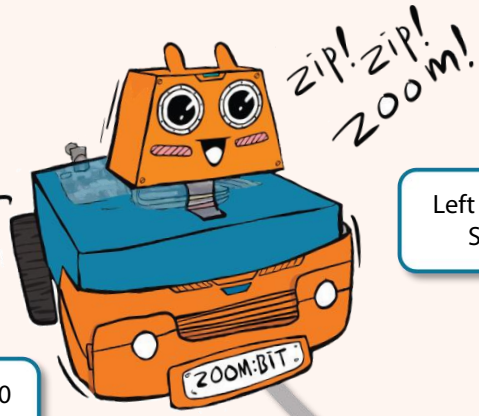


Speed	Delay	Angle
128	500	
	250	60
255	500	

Discover More Blocks



Use this block to control the spinning speed of each wheel.



Left Speed 100
SLOWER

Right Speed 150
FASTER

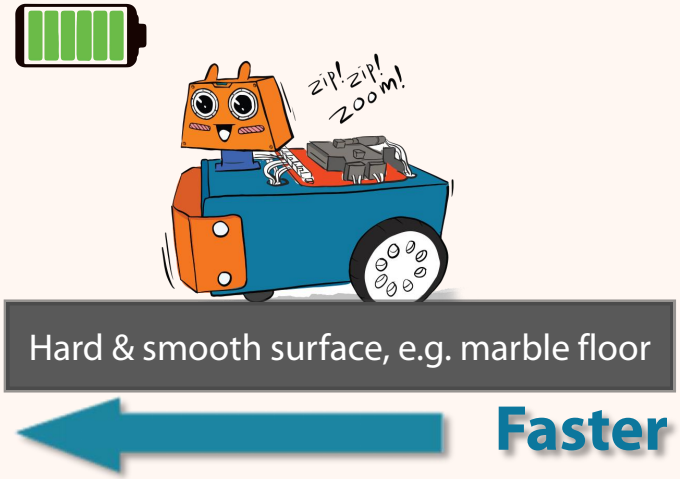
Do You Know? If both wheels spin at different speeds, ZOOM:BIT will steer towards the side of the wheel that is spinning at a lower speed. In the example above, ZOOM:BIT will move forward but steer to the left over time because the left wheel is spinning at a lower speed.

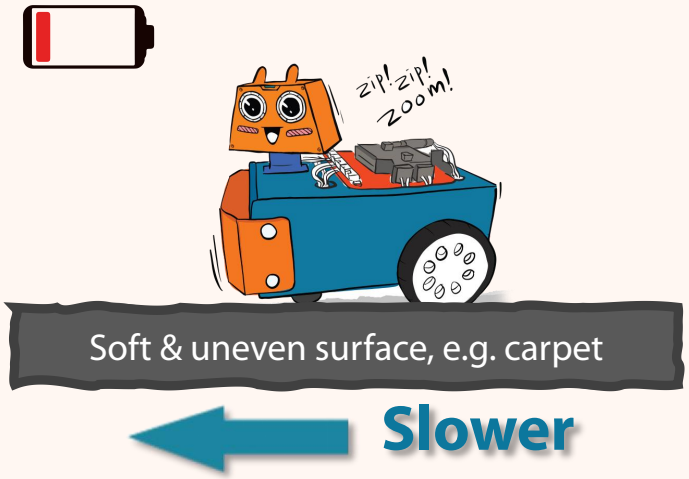
Can you predict which direction ZOOM:BIT will be moving if we set Left Speed to -150 and Right Speed to -200 ? Test it out and see if you're right.

Do You Know?

There is inevitably a slight difference between a motor's specifications and its actual performance. Seemingly identical motors are likely to rotate at slightly different speeds even though they are supplied with the same voltage. In other words, even though you program your ZOOM:BIT to move straight (i.e. same speed for both left and right wheels), it is still likely that ZOOM:BIT will veer slightly to the right, or left, after some time.

The accuracy and consistency of ZOOM:BIT's movements can also be affected by its **battery level** and also the **condition of the surface** it is on. ZOOM:BIT might move slower when its battery level is low and when the ground is too soft or uneven.

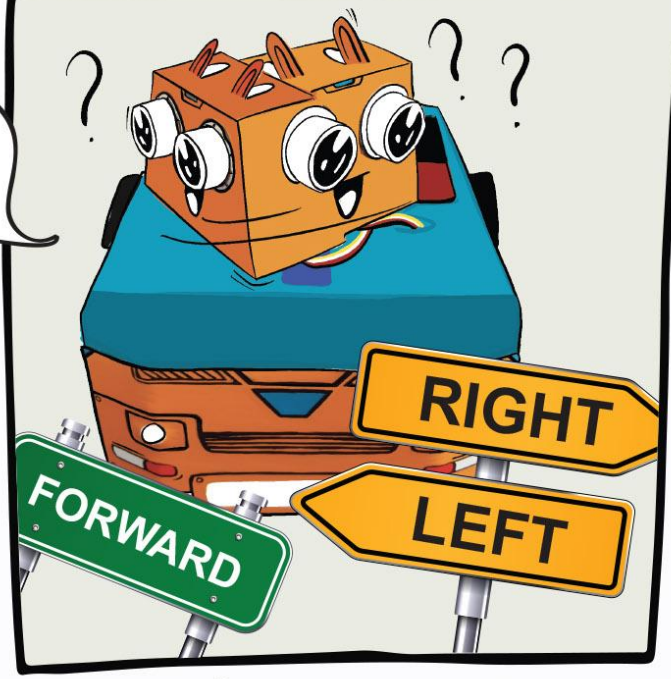




CHAPTER 5

https://link.cytron.io
/zoombit-chapter-5

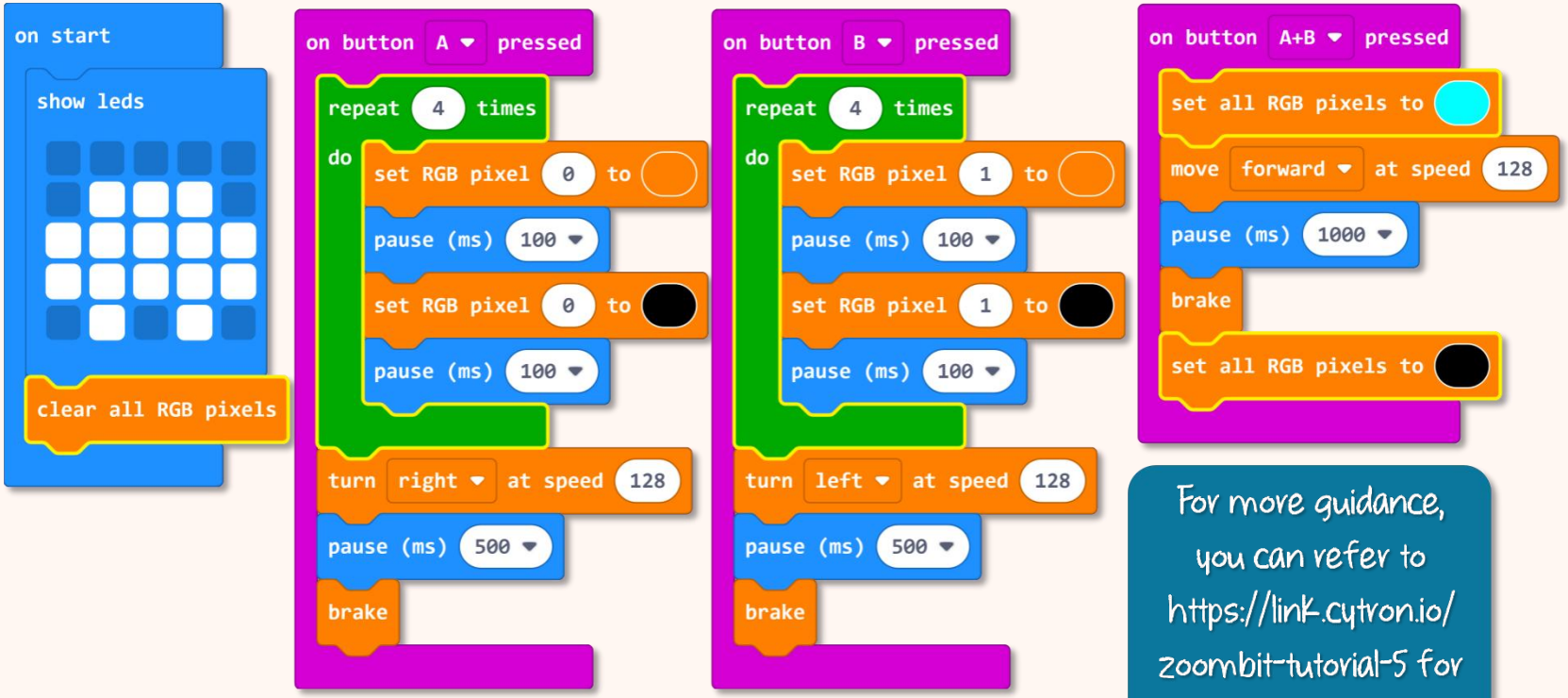
LET'S START!



Left? Right? Please Signal
Where You're Going~

There are two RGB LEDs on REKA:BIT board, labelled " 0 " and " 1 ". You can program them to light up in different colours using blocks from [REKA:BIT] category drawer.

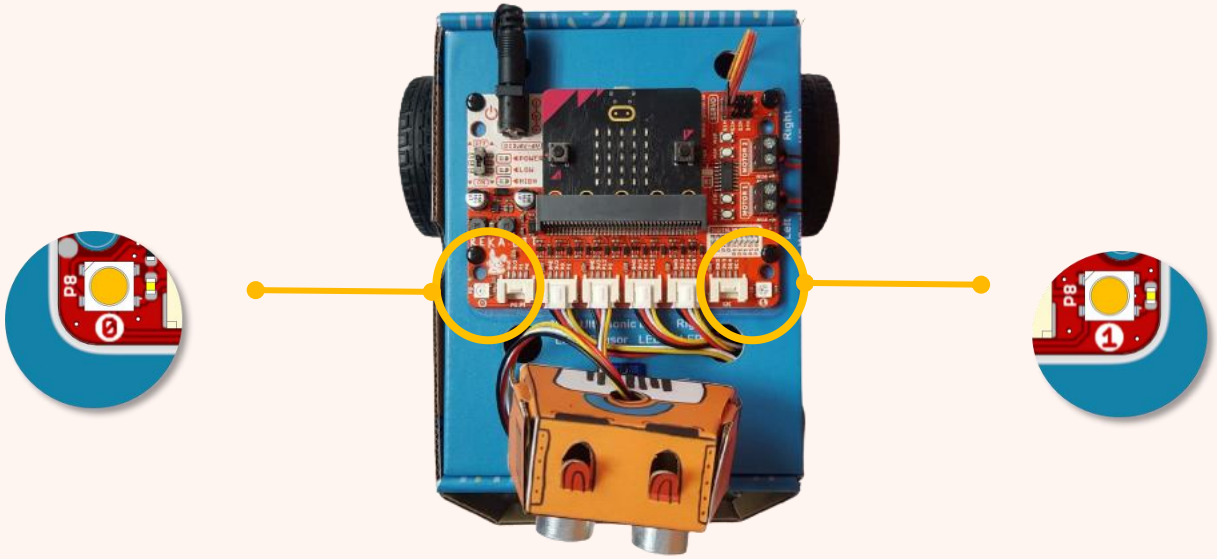
1 Add the following highlighted blocks from **[Basic]**, **[Loops]** and **[REKA:BIT]** category drawers to your code from the previous lesson.



For more guidance, you can refer to https://link.cytron.io/zoombit-tutorial-5 for step-by-step guide to build the code.

2 Download the code to ZOOM:BIT and power it up.

3 Press Button A, Button B, and then buttons A+B at the same time. Observe the RGB LEDs on REKA:BIT board.



Do you notice RGB LED " 0 " on the right blinking before ZOOM:BIT turns right ? And RGB LED " 1 " on the left blinking before ZOOM:BIT turns left ? And both RGB LEDs light up in blue when ZOOM:BIT is moving forward ?

Discover More Blocks

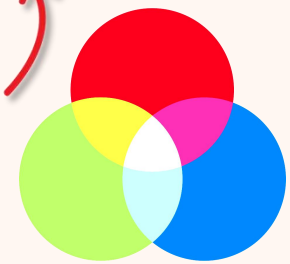
Set RGB pixel(s) to the selected colour. To change colour, click on the oval and select the colour you want from the colour palette.

Black = turn off RGB LED

Turn off all RGB pixels.

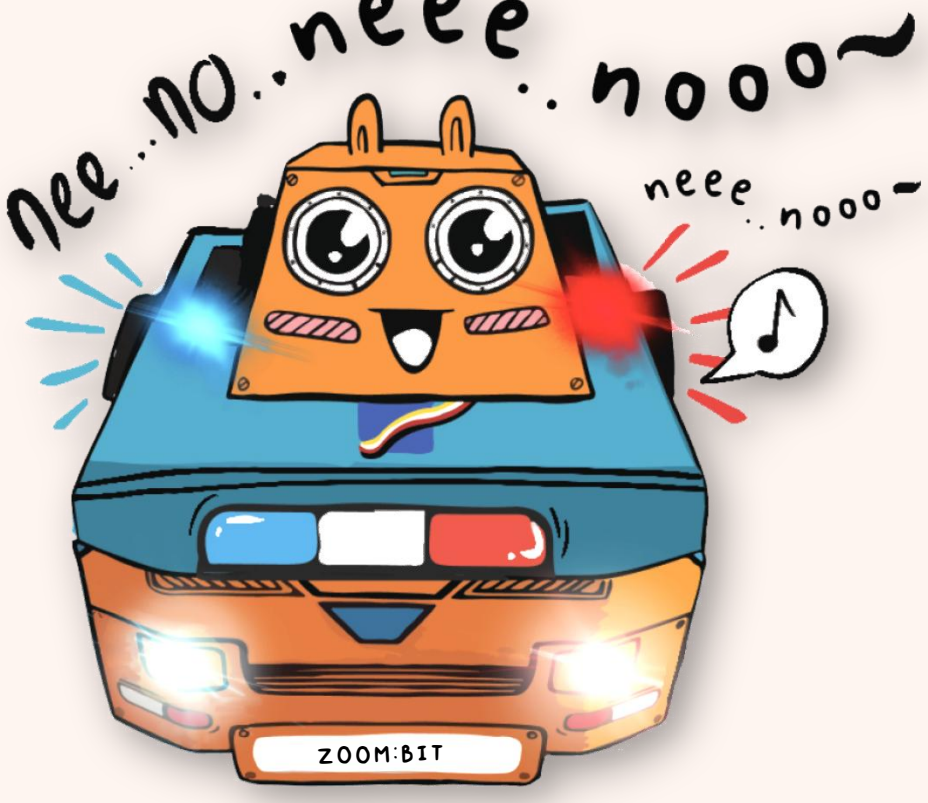
Change the brightness of the RGB pixels. The brightness value ranges from 0 to 255 (maximum brightness).

If the colour you want is not available on the colour palette, you can use this block to customise.



Here's a FUN challenge for you !

Can you program ZOOM:BIT to flash its RGB LEDs like the emergency lights of a police car? And for greater effect, make it sound the siren* as well?



* For the siren, you can alternate between middle C and middle F# notes in a loop.

* You can skip the siren part if you're using micro:bit V1 without speaker/buzzer.

CHAPTER 6

https://link.cytron.io
/zoombit-chapter-6

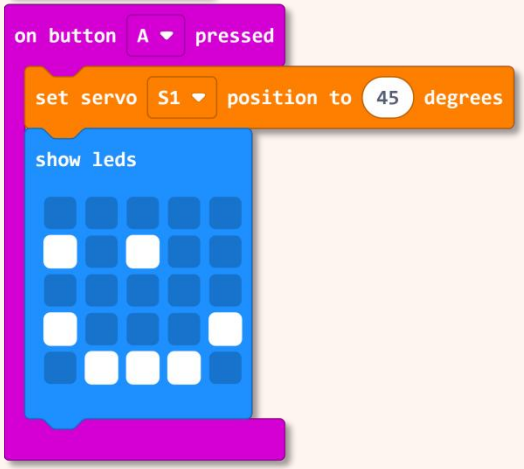
LET'S START!

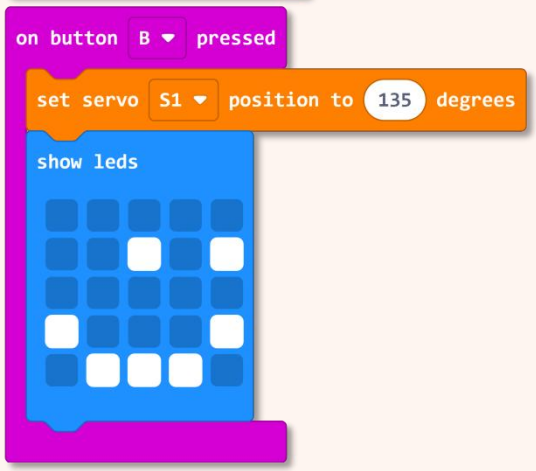
ZOOM:BIT's head is attached to a 180 degree servo motor. In other words, you can program ZOOM:BIT to look straight ahead and likewise you can make ZOOM:BIT turn its head to the left, or right, by controlling the servo to turn to your desired angle. Let's try!

1. Create a new project in your MakeCode Editor and add ZOOM:BIT extension. You can refer to pp. 44-45.

2. Build the following code. You can get the blocks you need from the following drawers:

Basic | REKA:BIT | Input

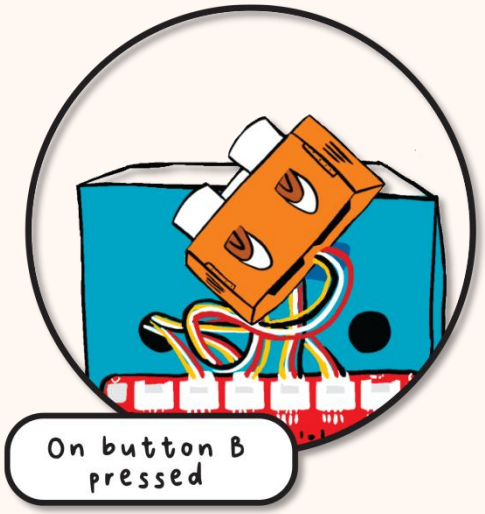


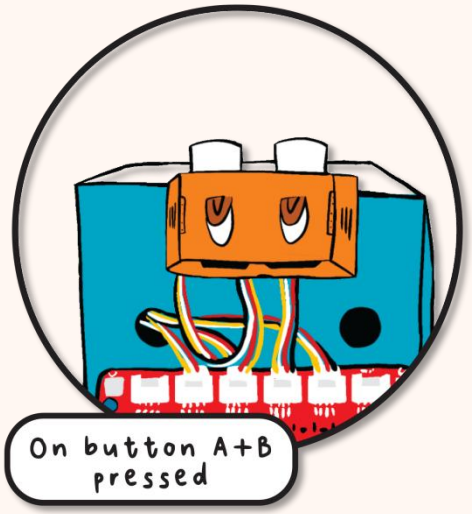


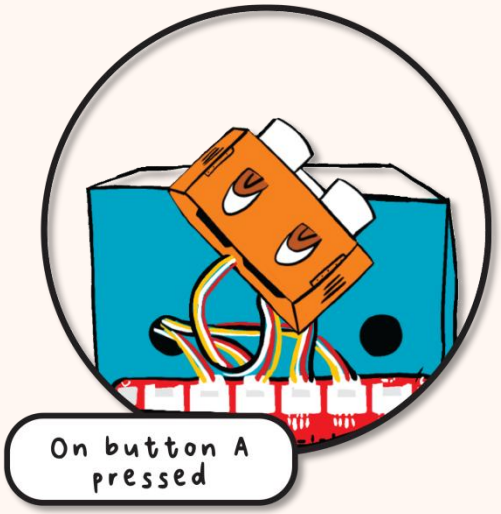
For more guidance, you can refer to https://link.cytron.io/zoombit-tutorial-6 for step-by-step guide to build the code.

3 Download the code to ZOOM:BIT and power it up.

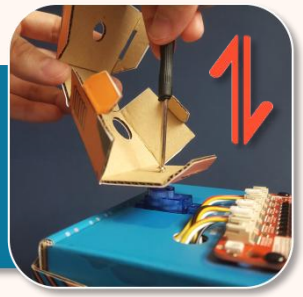
4 Press Button A, Button B, and then buttons A+B; observe the direction that the head is facing.







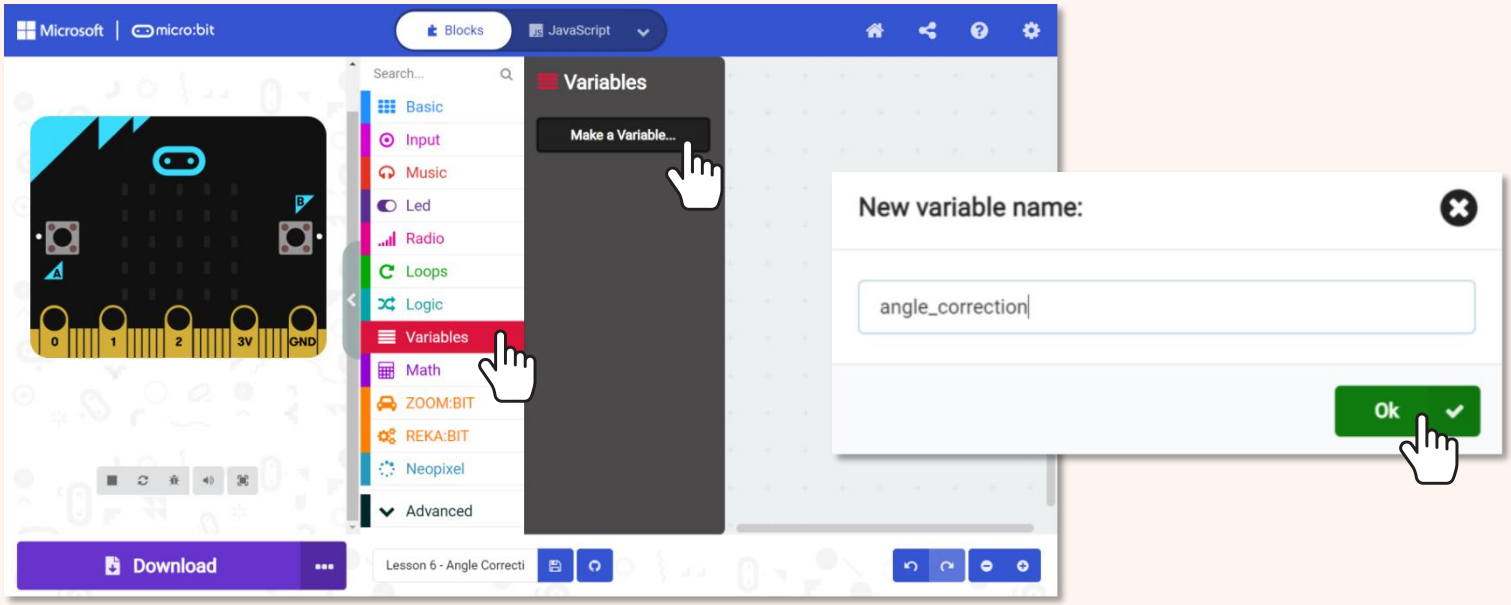
Does your ZOOM:BIT look straight ahead after you press Buttons A+B? If it is NOT properly aligned, then you need to unscrew the head, reposition it and then reattach it back to the servo motor horn.



If after manually readjusting the head, you still find that it is turned slightly to the right/left when it should be looking straight, you can correct it by making adjustment to your code. Follow the steps below to determine the "angle correction" for your ZOOM:BIT.

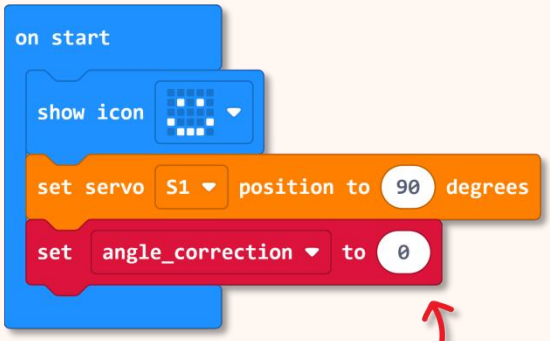
1 Create a new project in your MakeCode Editor and add ZOOM:BIT extension (you can refer to pp. 44-45).

2 Click **[Variables]** category and then select **[Make a Variable]**. Name your variable (e.g. "angle_correction") and then click **[Ok]** button.



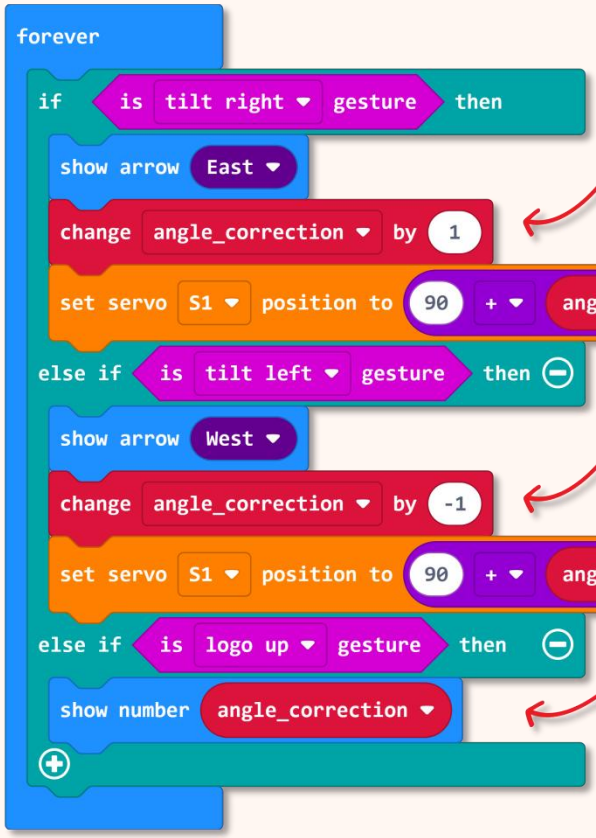
3 Build the following code. You can get the blocks you need from these drawers:

Basic | REKA:BIT | Variables | Input | Logic | Math



On start, variable [angle_correction] is set to 0.

For more guidance, you can refer to https://link.cytron.io/zoombit-tutorial-6a for step-by-step guide to build the code.



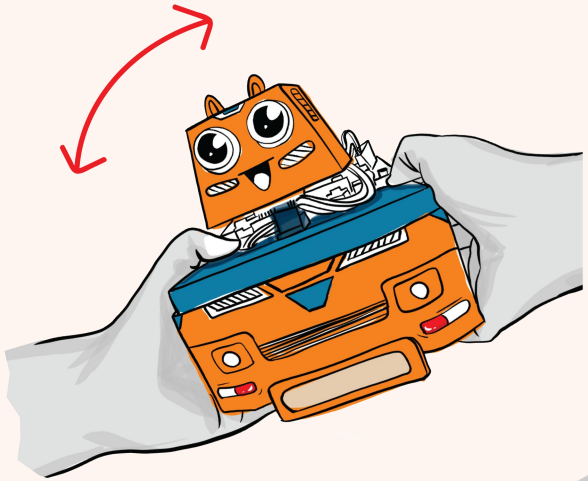
If ZOOM:BIT is tilted right, the variable [angle_correction] is changed by 1 and the head turns to the right by 1 degree.

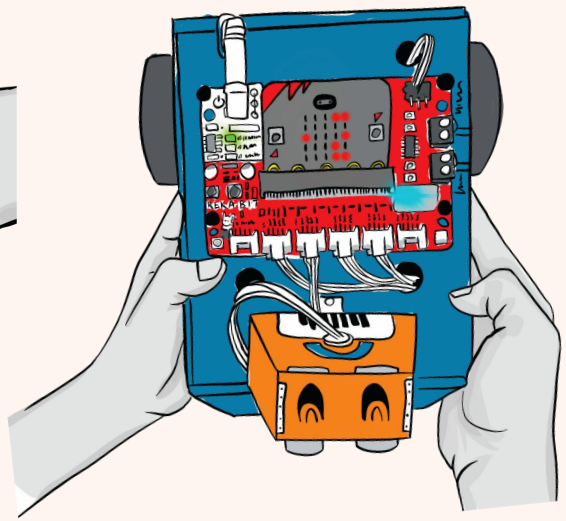
If tilted left, the variable is changed by -1 and the head turns to the left by 1 degree.

To read the value of [angle_correction], lift the robot so that the micro:bit logo is up (and robot head is facing down).

4 Download the code to ZOOM:BIT and power it up. Tilt ZOOM:BIT to the left (or to the right, whichever applies) to turn the head in that direction.

5 When you're satisfied that the head is facing straight ahead, hold ZOOM:BIT with the micro:bit logo up (and ZOOM:BIT's head facing down) to get the "angle_correction" value.





Record the angle_correction reading here.

Now that you know the [angle_correction] value for your ZOOM:BIT, you can use that in your future projects to ensure that the head is turned to the angle you want.

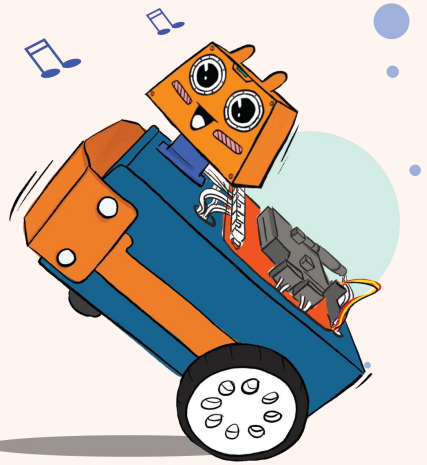
Fill in the [angle_correction] value you recorded earlier here.

Here's a sample code which includes angle correction.

When powered up, ZOOM:BIT will look to the left, then to the right and finally face straight ahead.

Here's a FUN challenge for you !

Can you program ZOOM:BIT to dance? Get creative with the moves; make ZOOM:BIT twist and turn~



With micro:bit V2, you can use [On Loud Sound] block from [Input] category as the trigger for ZOOM:BIT to start dancing; and you can add [Music] blocks too to make the performance more lively!

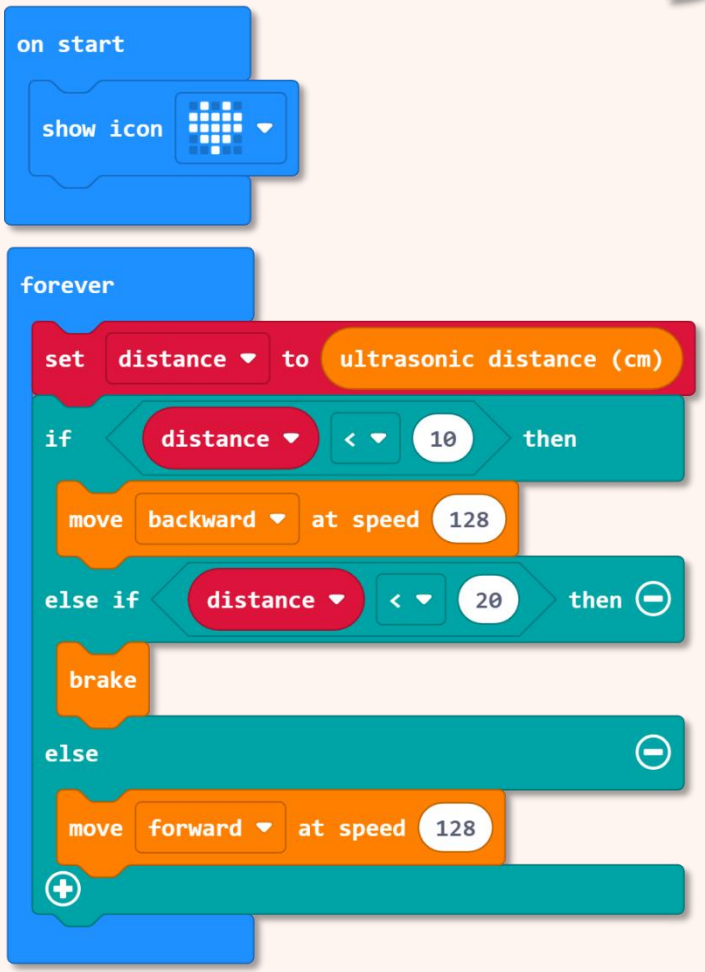
CHAPTER 7

https://link.cytron.io
/zoombit-chapter-7

Now that ZOOM:BIT is mobile, let's teach him not to bump into obstacles in its path.

1. Create a new project in your MakeCode Editor and add ZOOM:BIT extension (you can refer to pp. 44-45).

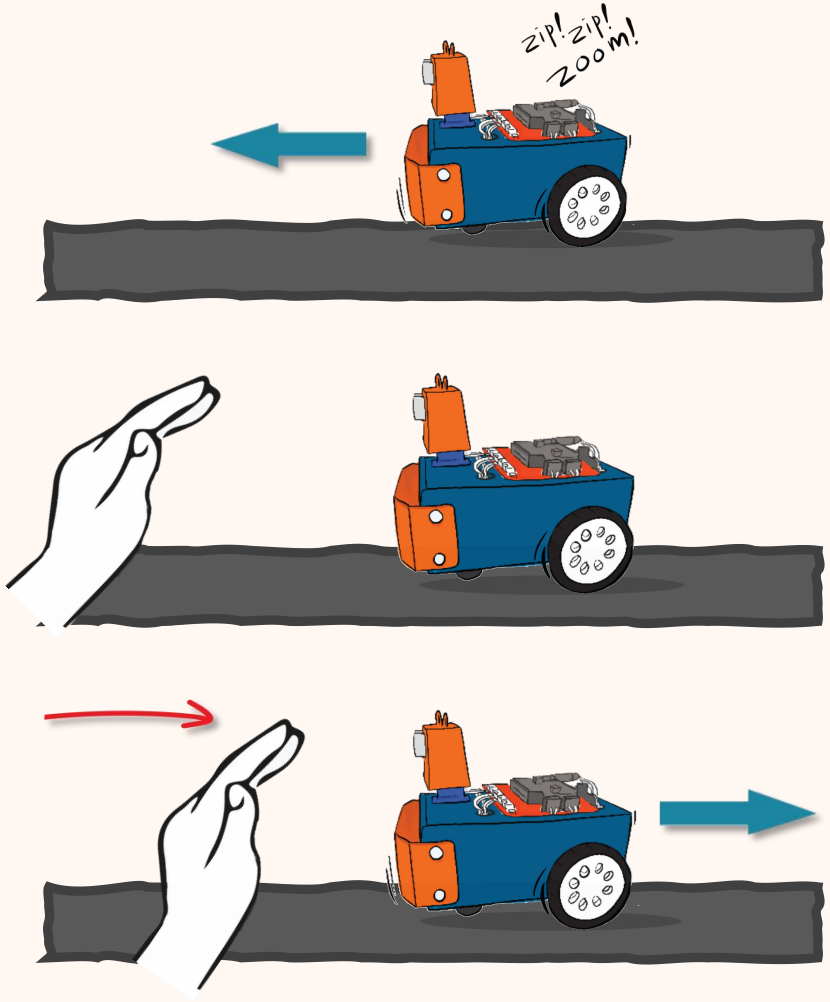
2. Build this code. You can get the blocks you need from these category drawers:



You'll need to create a new variable ("distance") and set it to always get the ultrasonic sensor value.

You can go to https://link.cytron.io/zoombit-tutorial-7 for step-by-step guide to build the code if you need more guidance.

3 Download the code to your ZOOM:BIT and power it up.



ZOOM:BIT will keep moving forward when no obstacle is detected.

Try to hold up your hand in front of ZOOM:BIT. Does your robot stop when it is about 10cm from your hand?

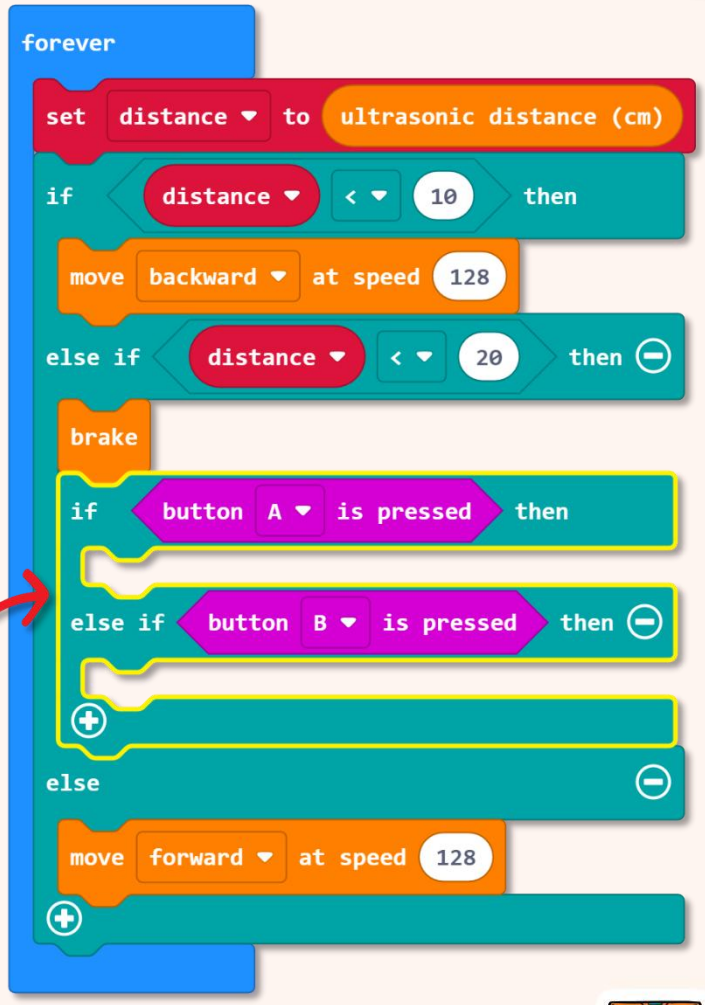
Slowly move your hand towards ZOOM:BIT. Observe its response when the distance is less than 10cm.

Let's make ZOOM:BIT turn right when Button A is pressed and turn left when Button B is pressed when it is in a stationary mode, i.e. stops 10 cm away from an obstacle.

4 Add the highlighted blocks after the **[brake]** block.

You can get the blocks you need from these category drawers:

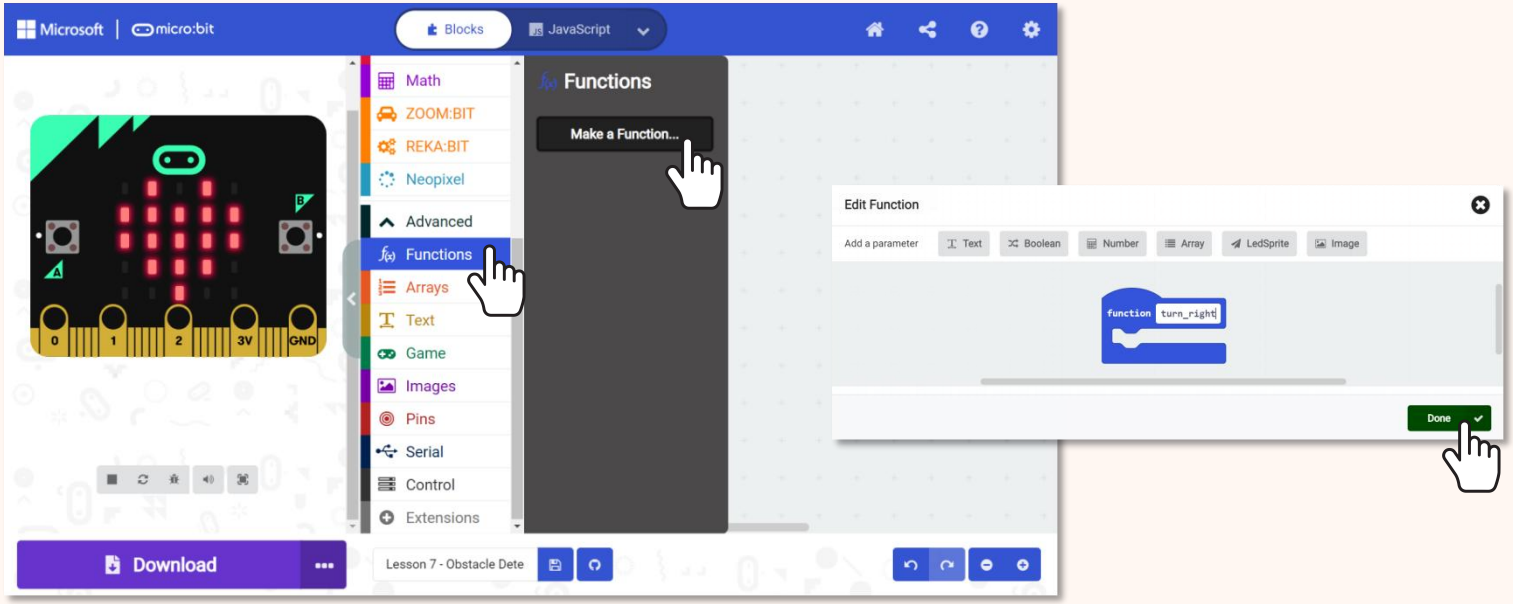
- Input
- Logic



This is called a "nested if condition".

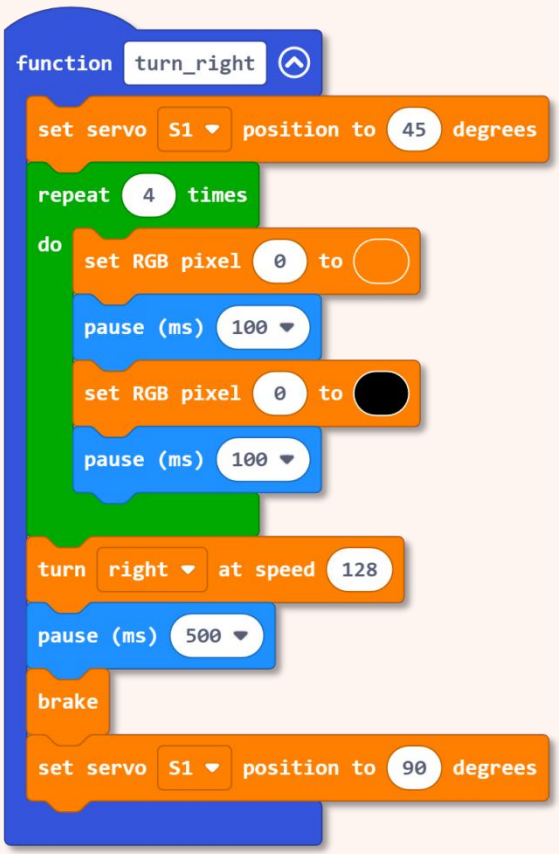
Do You Know? We can make blocks of code that perform a specific task into a function. After creating a function, you can use the function in multiple places in your program without having to build the same blocks of code over and over again. In addition, professional programmers also use functions to make their code more easily readable by others.

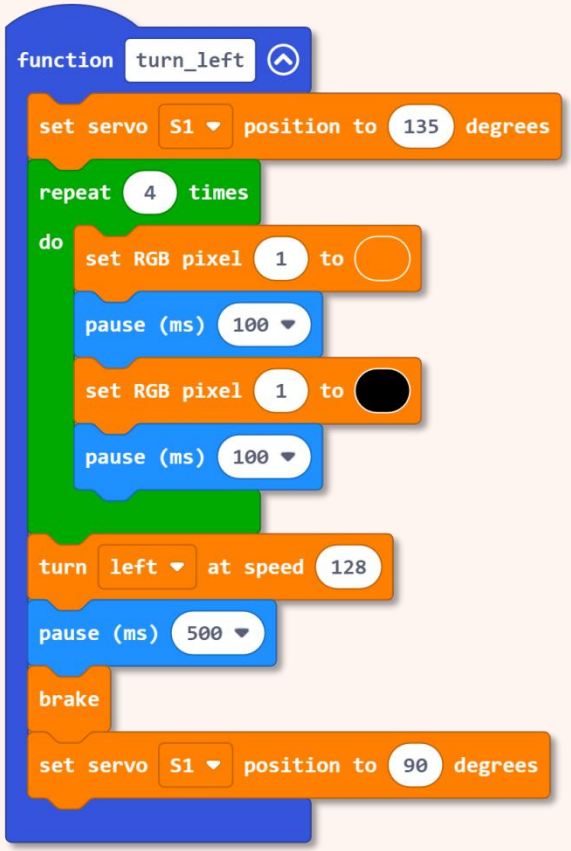
5 Click **[Advanced]** and then select **[Functions]** category. Click **[Make a Function]**, rename doSomething to **'turn_right'** and then click **[Done]** button. A **[funtion turn_right]** block will be added to your workspace.



6 Repeat to create another function and rename it **'turn_left'** .

7 Continue building your code by adding the following blocks to your **[function turn_right]** and **[function turn_left]** blocks.

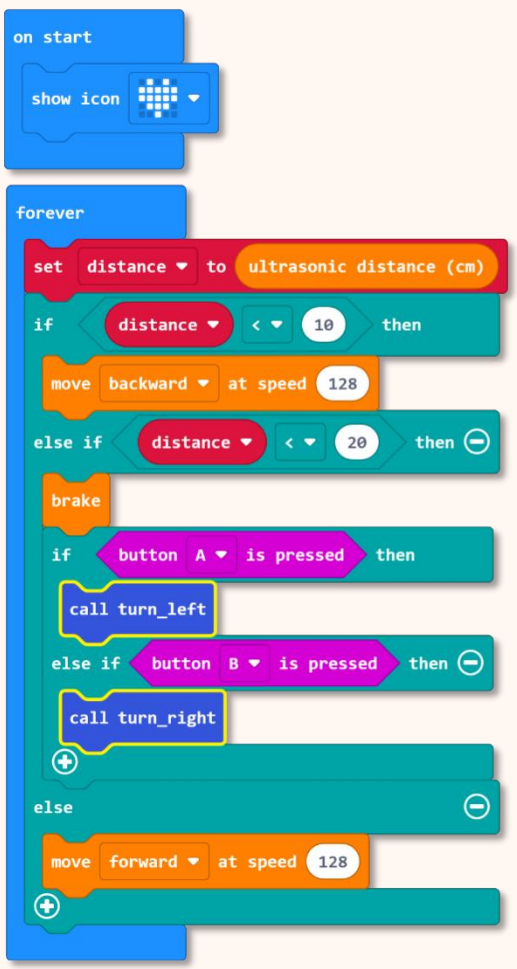


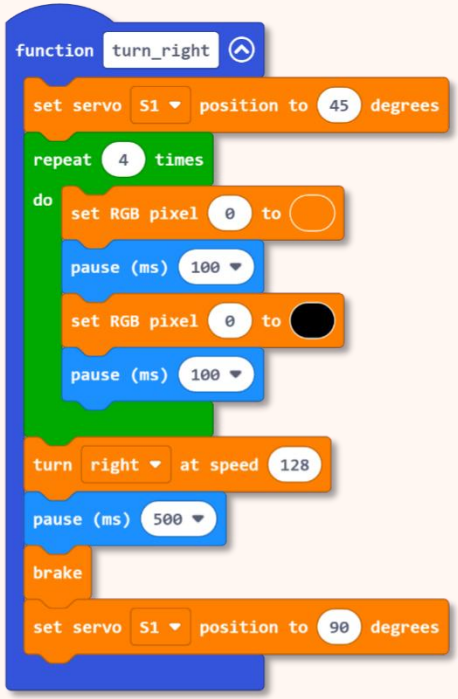


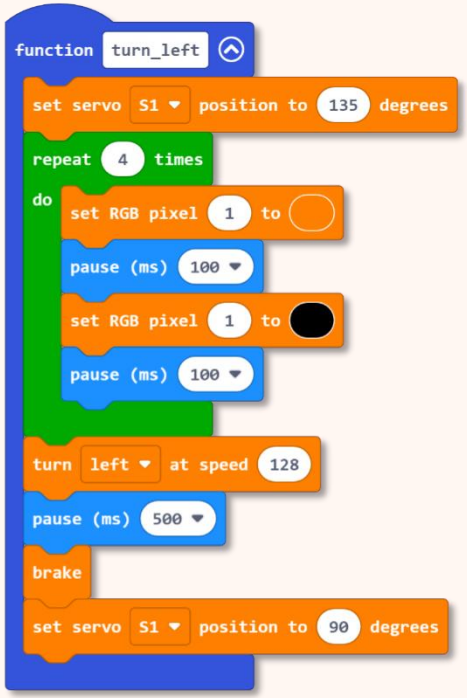
You can click icon to collapse the blocks of code after you're done building a function.

Click icon to open if you need to review or edit your code.

8 Finally, click **[Functions]** category and add **[call turn_right]** and **[call turn_left]** blocks to your code. Here's the complete code:

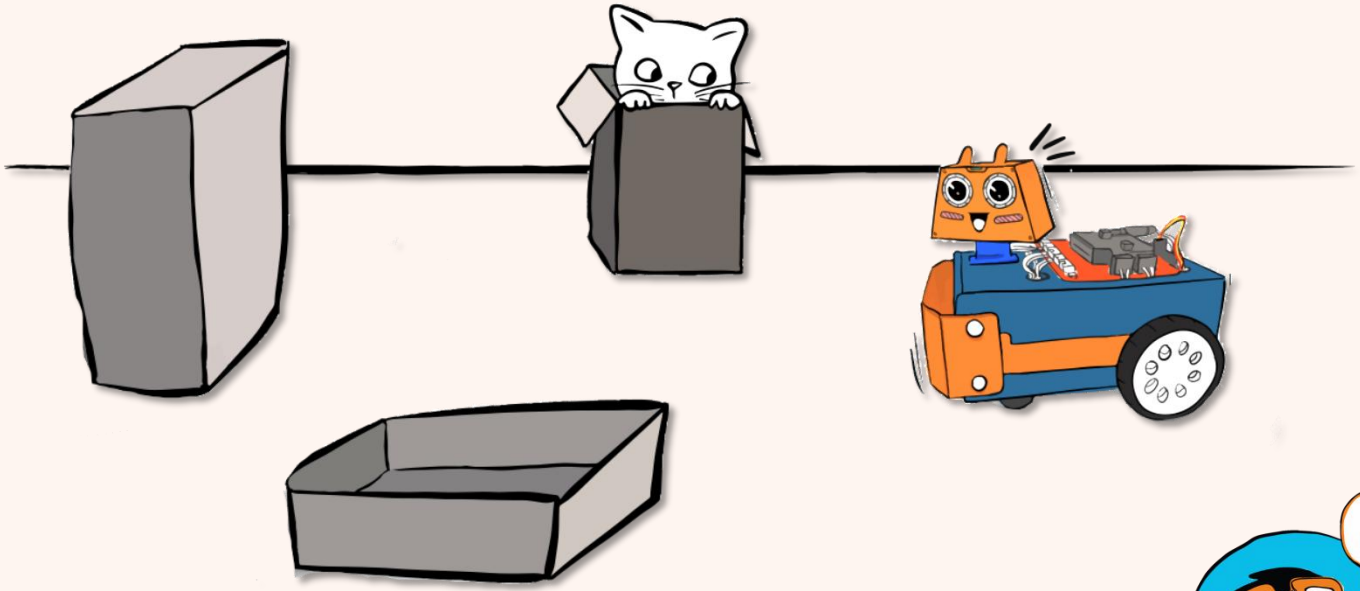






Yeah!! Now ZOOM:BIT can roam freely in your room without bumping into things. When ZOOM:BIT's path is blocked by an obstacle, you can press Button A (to turn right) or Button B (to turn left) to guide ZOOM:BIT to bypass the hindrance.

9 Download the code to your ZOOM:BIT and power it up.



Do you think you can modify the code so that ZOOM:BIT can autonomously move away from obstacles without waiting for your help? Give it a try ~

Here's a FUN challenge for you !

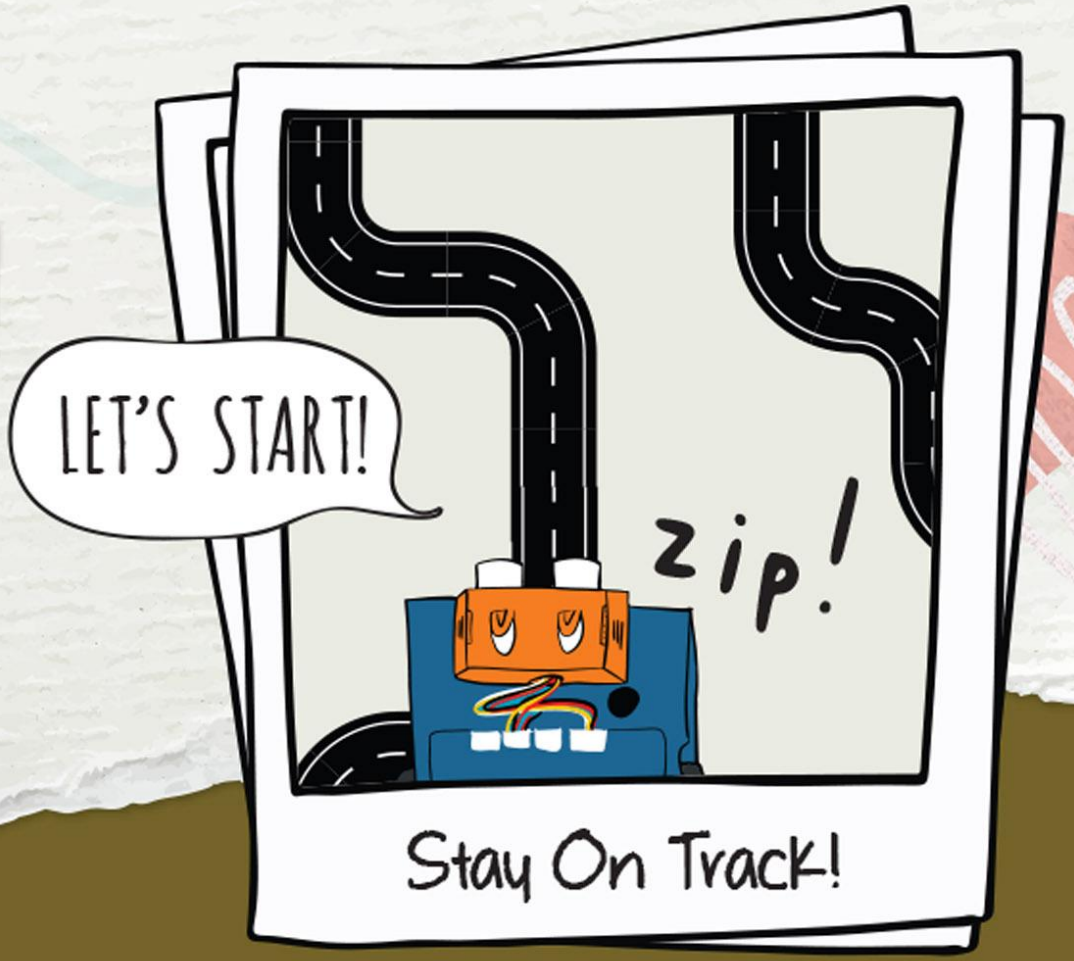
Transform ZOOM:BIT into an ultrasonic piano. Program ZOOM:BIT to play different tones in response to the reading of its ultrasonic sensor.

When an object is placed						
< 5	< 10	< 15	< 20	< 25	< 30	< 35
__ cm away, ZOOM:BIT plays tone						
C	D	E	F	G	A	B
for 1/2 beat and displays the letter note.						

Get ZOOM:BIT to sing a song. Move the palm of your hand towards, or away, from its face to get ZOOM:BIT to play the tone that you want. If you're not sure how it works, scan the QR code to watch a demo video.

CHAPTER 8

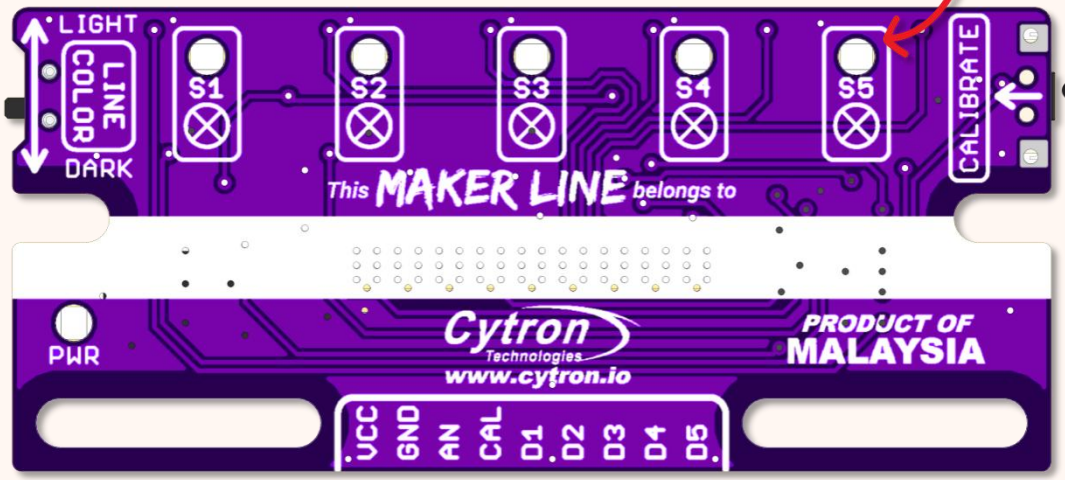
https://link.cytron.io
/zoombit-chapter-8



Do You Know? ZOOM:BIT can be programmed to follow a line? ZOOM:BIT can easily do that because it is equipped with Maker Line sensor. The sensor enables it to detect a line (either black or white) against a background with a contrasting colour.

LED indicators for each of the 5 IR sensors (S1-S5). LED will light up when line is detected by the corresponding IR sensor.

Slide the line color switch to "DARK" as the track provided is using a black line (against a white background),

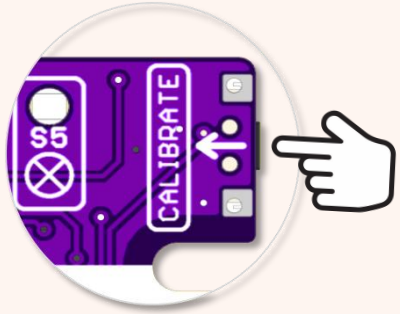


Press this button to calibrate the sensor.

Maker Line Sensor - Top View

Before you start programming ZOOM:BIT, follow the steps below to calibrate the Maker Line sensor first. Calibration only needs to be carried out once unless the sensor height, line or background color has changed.

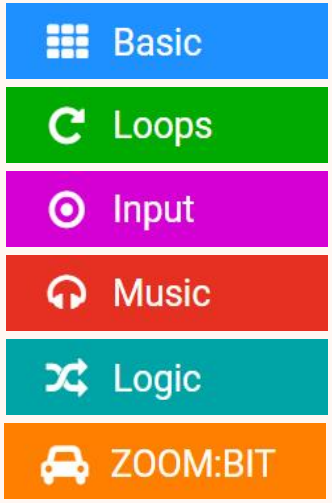
1. Spread out the track provided. Place ZOOM:BIT on the track and power it up.
2. Press and hold the CALIBRATE button until all 5 LEDs light up; only release the button when all the LEDs are blinking (i.e. Maker Line has entered calibration mode.)
3. Manually move ZOOM:BIT from side to side over the black line. Repeat several times and make sure that all sensors have been exposed to the line.
4. Press the CALIBRATE button again to exit calibration mode.

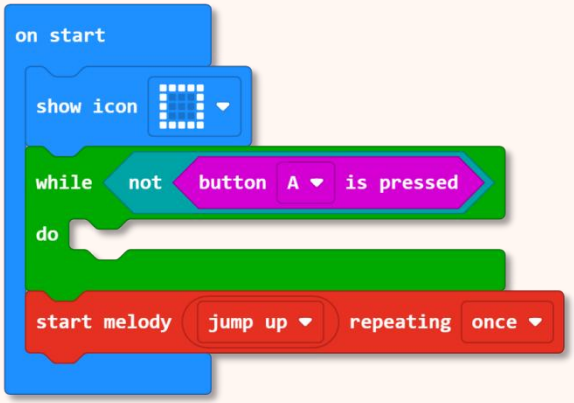


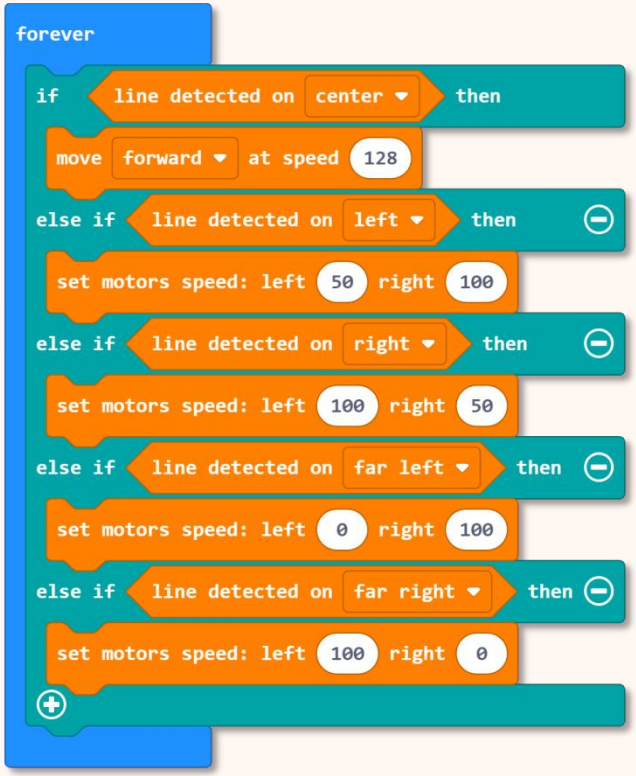
If the calibration is successful, you'll see a running light effect; your MAKER LINE is now ready to use. Scan this QR code to watch a demo video if you're not sure what to do or how it works.

1. Create a new project in your MakeCode Editor and add ZOOM:BIT extension (you can refer to pp. 44-45).

2. Build the following code to instruct ZOOM:BIT to follow the track. You can get the blocks you need from the listed category drawers:

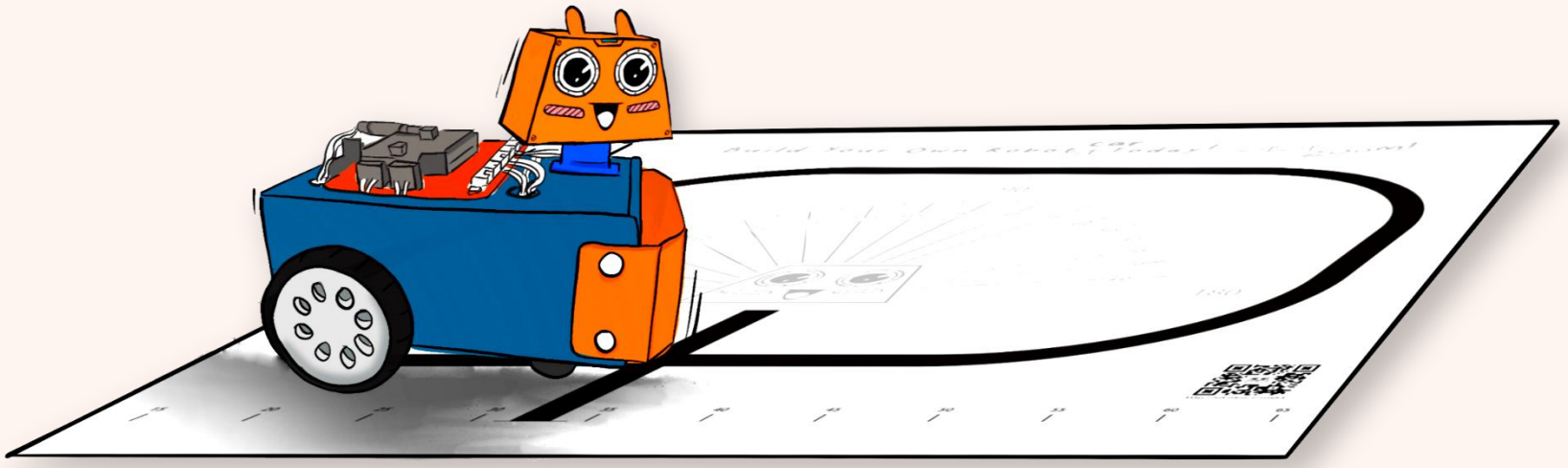






For more guidance, you can refer to https://link.cytron.io/zoombit-tutorial-8 for step-by-step guide to build the code.

3 Download the code to your ZOOM:BIT. Power it up, place it on the track and press Button A.



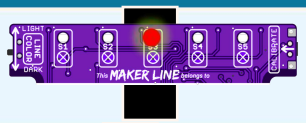
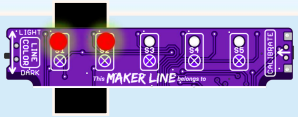
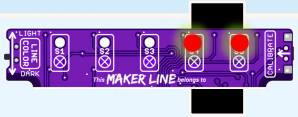
Watch in awe as your ZOOM:BIT zooms off and moves around track after you press Button A. Can you figure out how the code works?

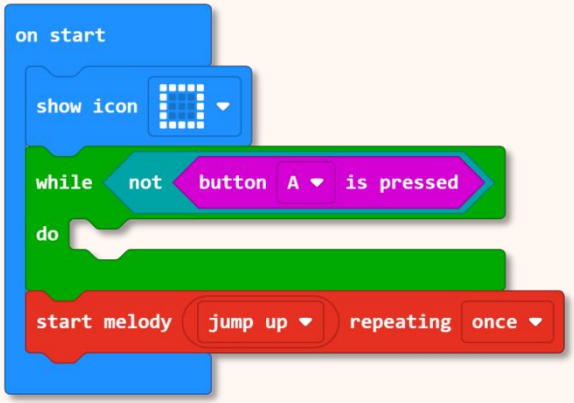
On start, show icon ☐.

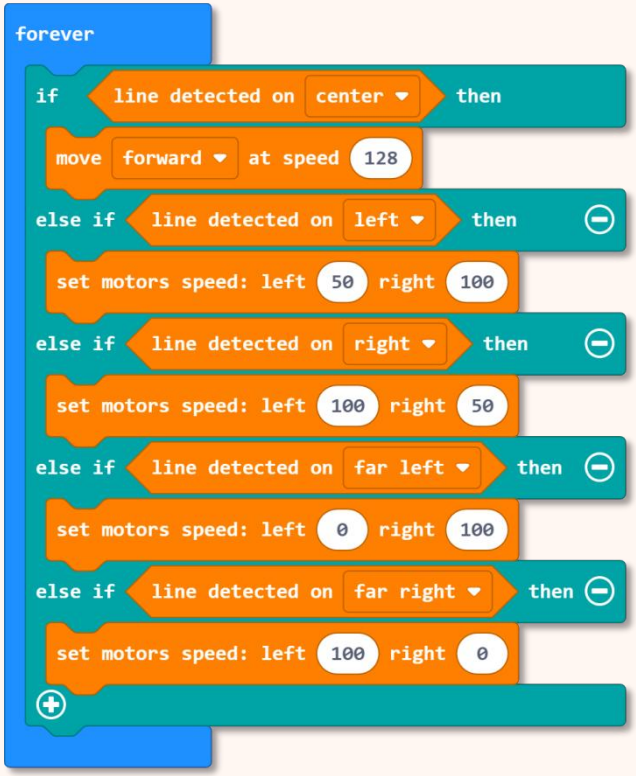
Then do nothing as long as button A is NOT pressed.

If button A is pressed, exit the while loop.
Play melody 'jump up' once.

Forever check Maker Line reading and respond accordingly.

Condition	Line detected ?	What to do to stay/ get back on track ?
	Centre	Move straight.
	Left	Turn left slightly.
	Right	Turn right slightly.
	Far left	Turn left.
	Far right	Turn right.

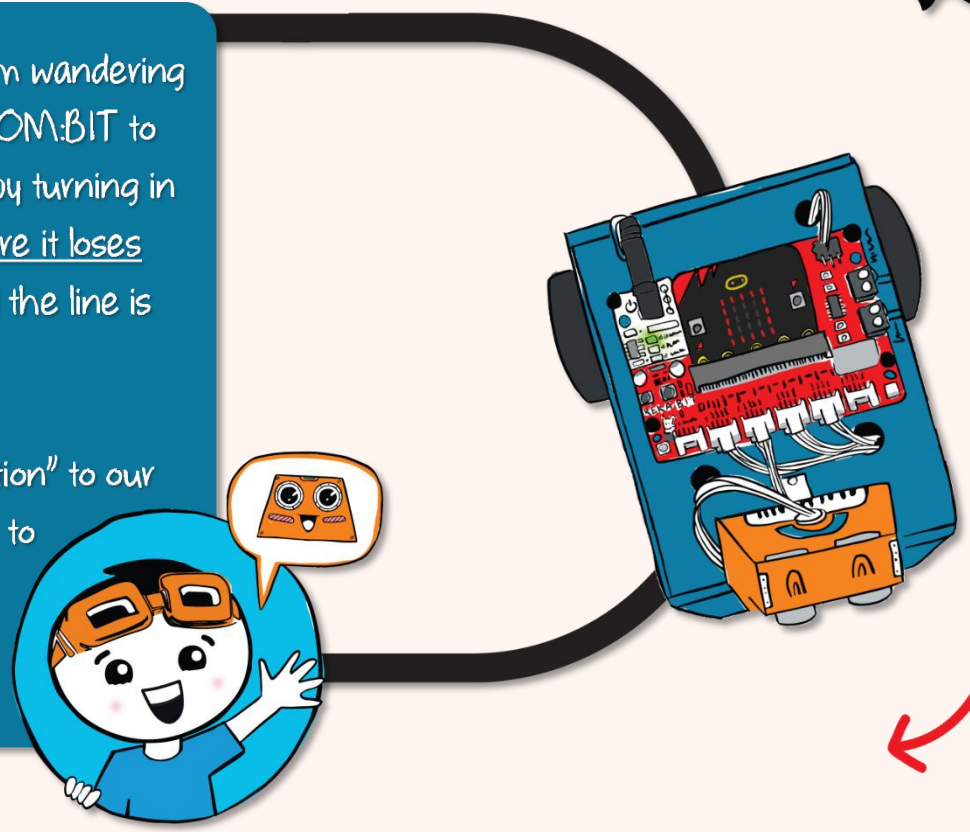




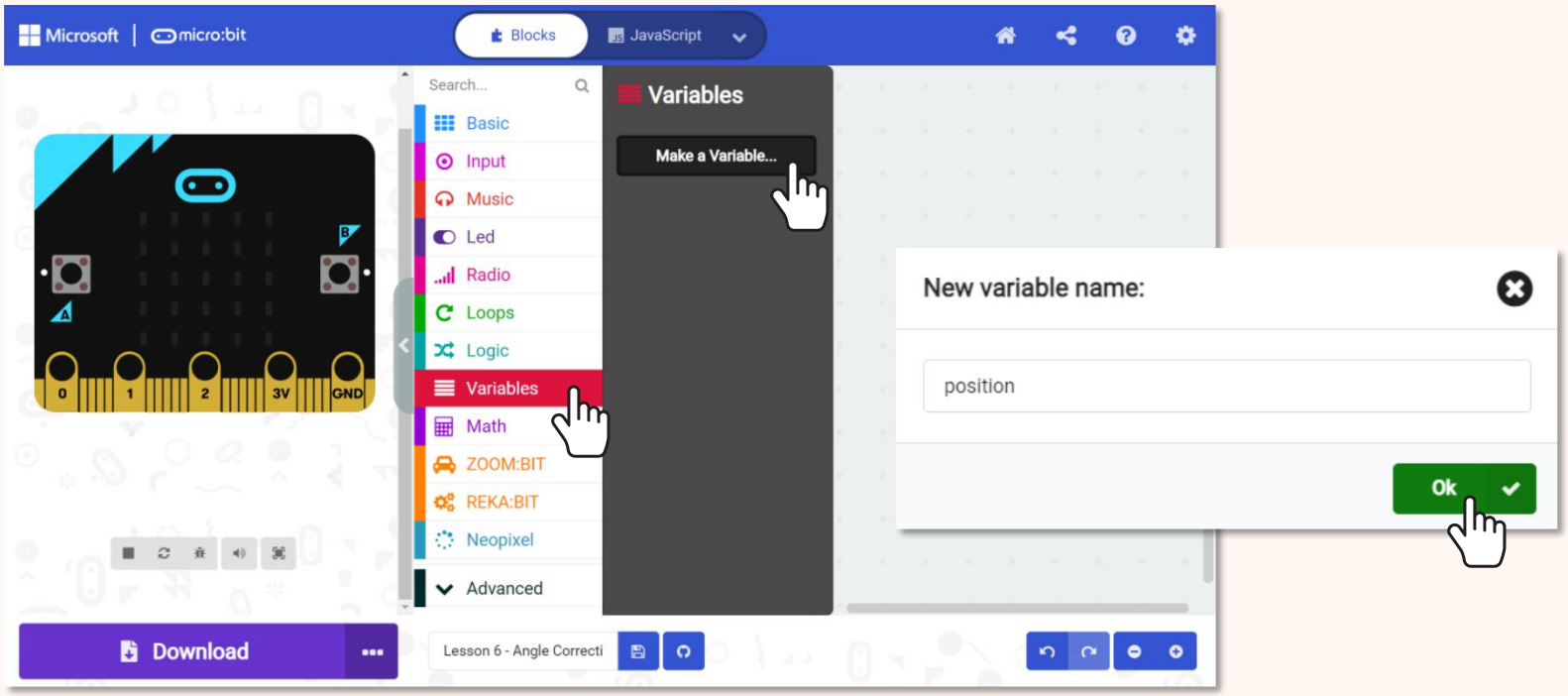
Does your ZOOM:BIT sometimes wander off track, especially when it is going around the curve? When ZOOM:BIT is turning the corner, its Maker Line sensor might be momentarily away from the line (as shown below). When this happens, ZOOM:BIT gets confused because in our code earlier we did not tell ZOOM:BIT what to do when no line is detected.

To prevent ZOOM:BIT from wandering away, we need to teach ZOOM:BIT to find its way back on track by turning in the same direction (as before it loses detection of the line) ... until the line is detected again.

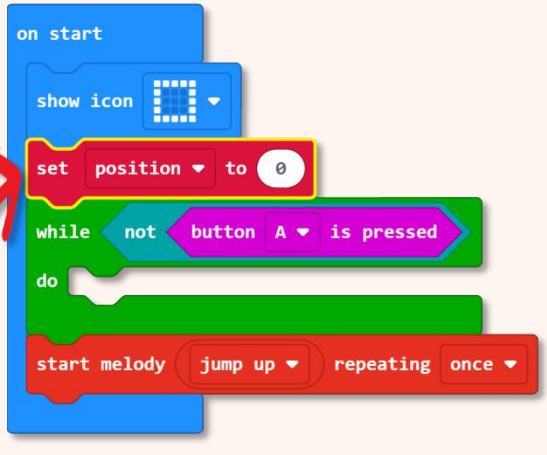
We can add a variable "position" to our code for that purpose. Turn to the next page to learn how to improve our earlier code.



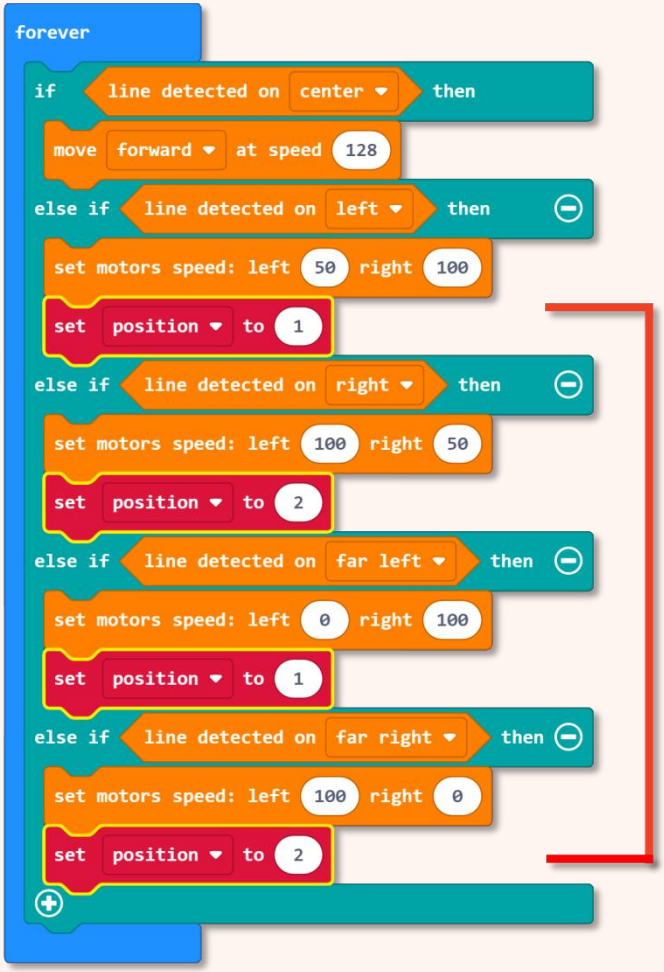
4 Click **[Variables]** category and then select **[Make a Variable]**. Name your variable (e.g. "position") and then click **[Ok]** button.



5 Add the following **[set (position) to (__)]** blocks from **[Variables]** category drawer to your code.

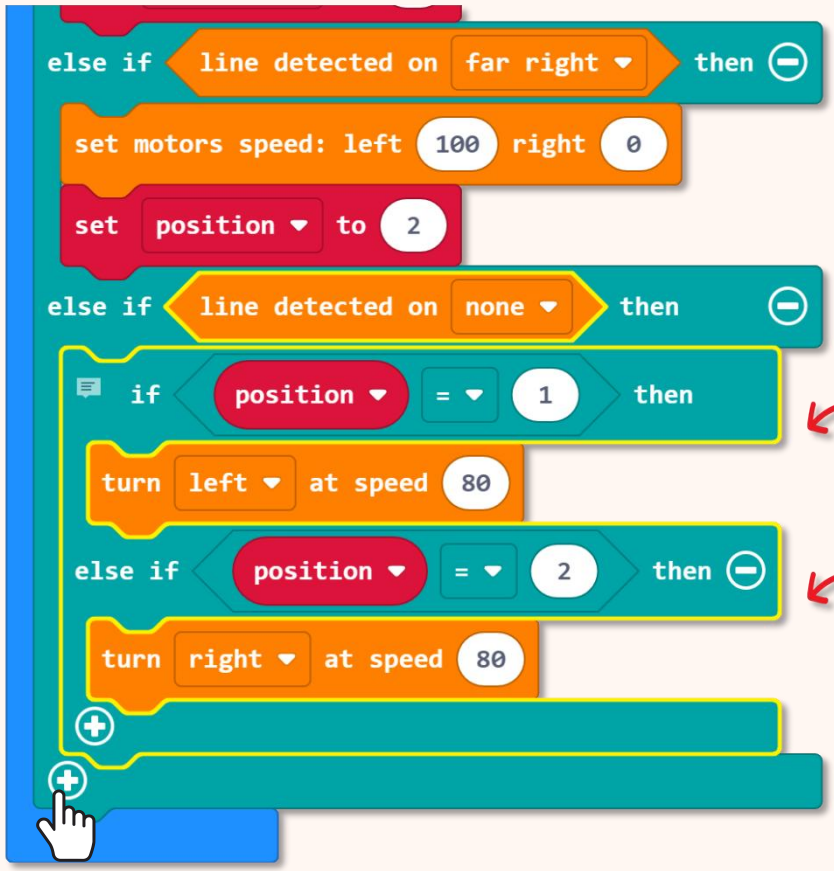


Set the variable [position] to 0 when ZOOM:BIT is powered up.



Set variable [position] to "1" when line is detected on left or far left; set to "2" when line is detected on right or far right.

6 Click the ⊕ icon to add another **"else-if"** condition. Then add the following highlighted blocks to your code.



A new condition whereby no line is detected.

This is a nested if condition. Check if variable [position] is 1, then turn left;

else if variable [position] is 2, then turn right.

7 Download the completed code to your ZOOM:BIT. Power it up, place it on the track and press Button A.

```
on start
    show icon
    set position to 0
    while not button A is pressed
    do
    start melody jump up repeating once
```

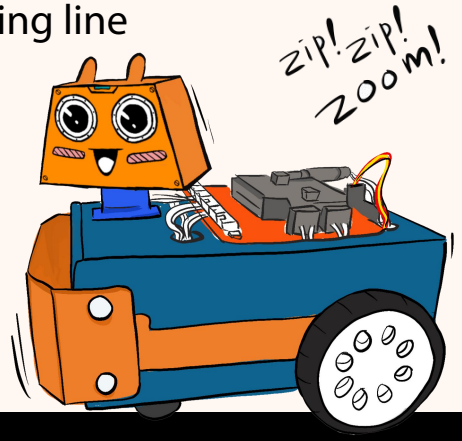
```
forever
    if line detected on center then
        move forward at speed 128
    else if line detected on left then
        set motors speed: left 50 right 100
        set position to 1
    else if line detected on right then
        set motors speed: left 100 right 50
        set position to 2
    else if line detected on far left then
        set motors speed: left 0 right 100
        set position to 1
    else if line detected on far right then
        set motors speed: left 100 right 0
        set position to 2
    else if line detected on none then
        if position = 1 then
            turn left at speed 80
        else if position = 2 then
            turn right at speed 80
```

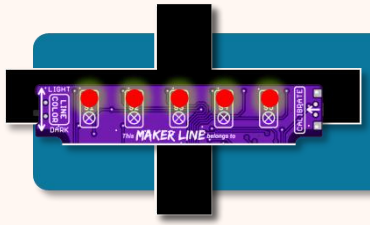
Let's test. Try to push ZOOM:BIT off track (until no line is detected by Maker Line). Do you notice ZOOM:BIT readjusting its position to get back on track, instead of wandering off?

Here's a FUN challenge for you !

Can you program ZOOM:BIT to do the following :-

- race around the track on Button A pressed
- play a tone for 1/2 beat whenever it passes the finishing line
- display the number of lap(s) it has completed, and
- stop after it has completed three (3) laps.

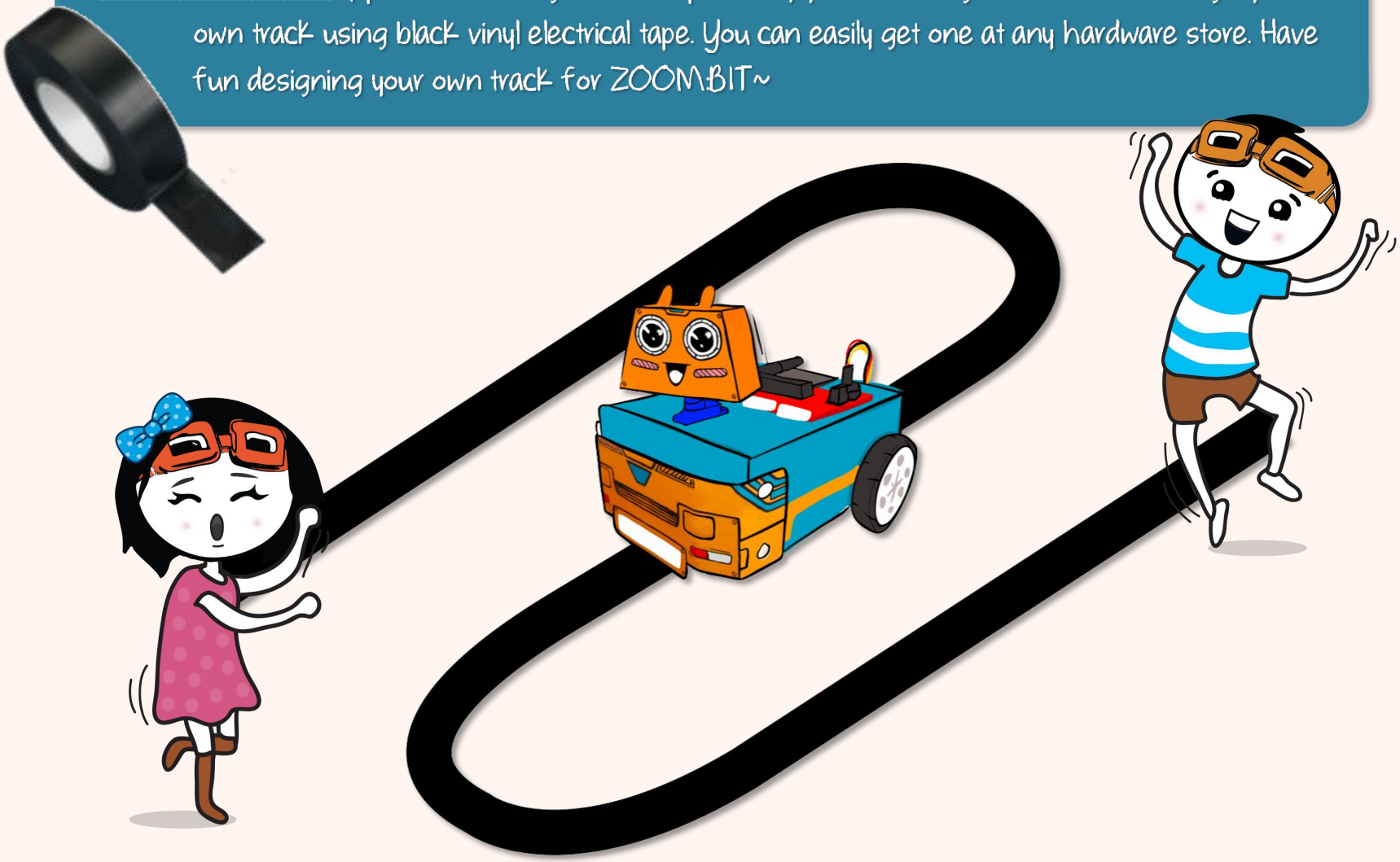




Tips: We can program ZOOM:BIT to 'Know' that it has crossed the finishing line by using the [line detected on (all)] block.

Do You Know?

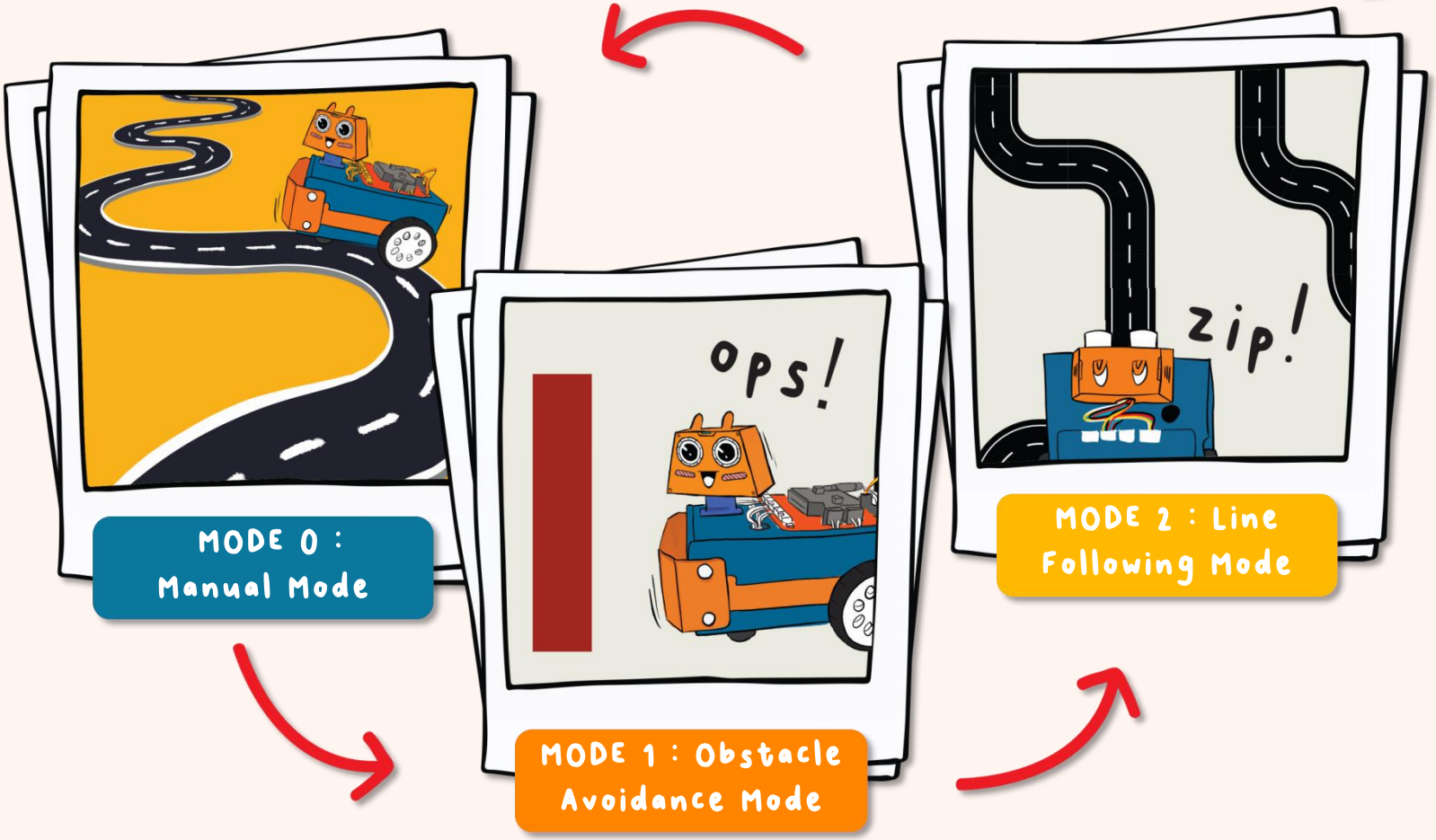
Apart from using the track provided, you can also get creative and design your own track using black vinyl electrical tape. You can easily get one at any hardware store. Have fun designing your own track for ZOOM:BIT~



CHAPTER 9

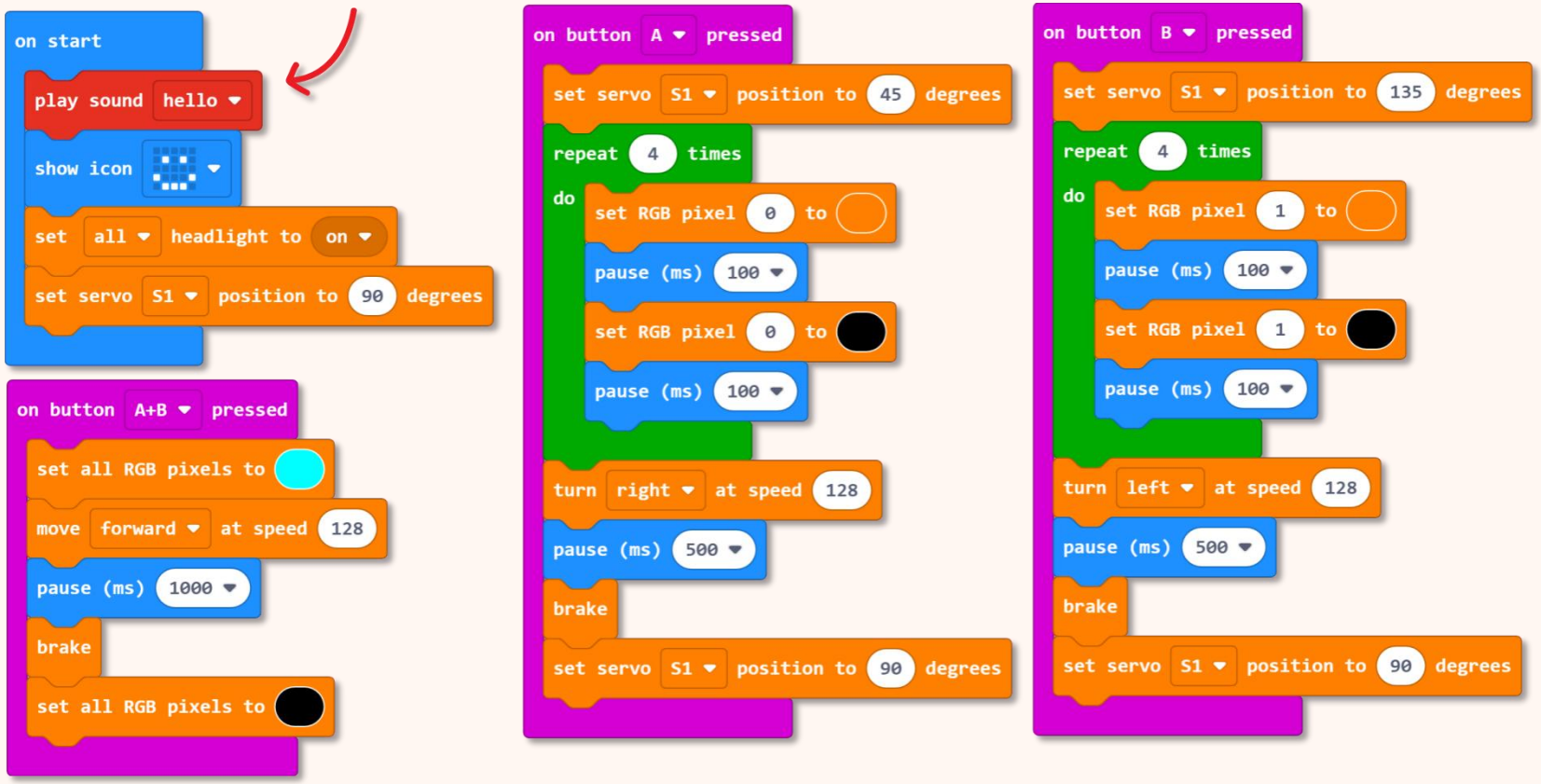
https://link.cytron.io
/zoombit-chapter-9

Together, we've taught ZOOM:BIT many tricks and he has learned them one by one. Let's now train ZOOM:BIT to juggle them all - switching from one mode to another effortlessly.



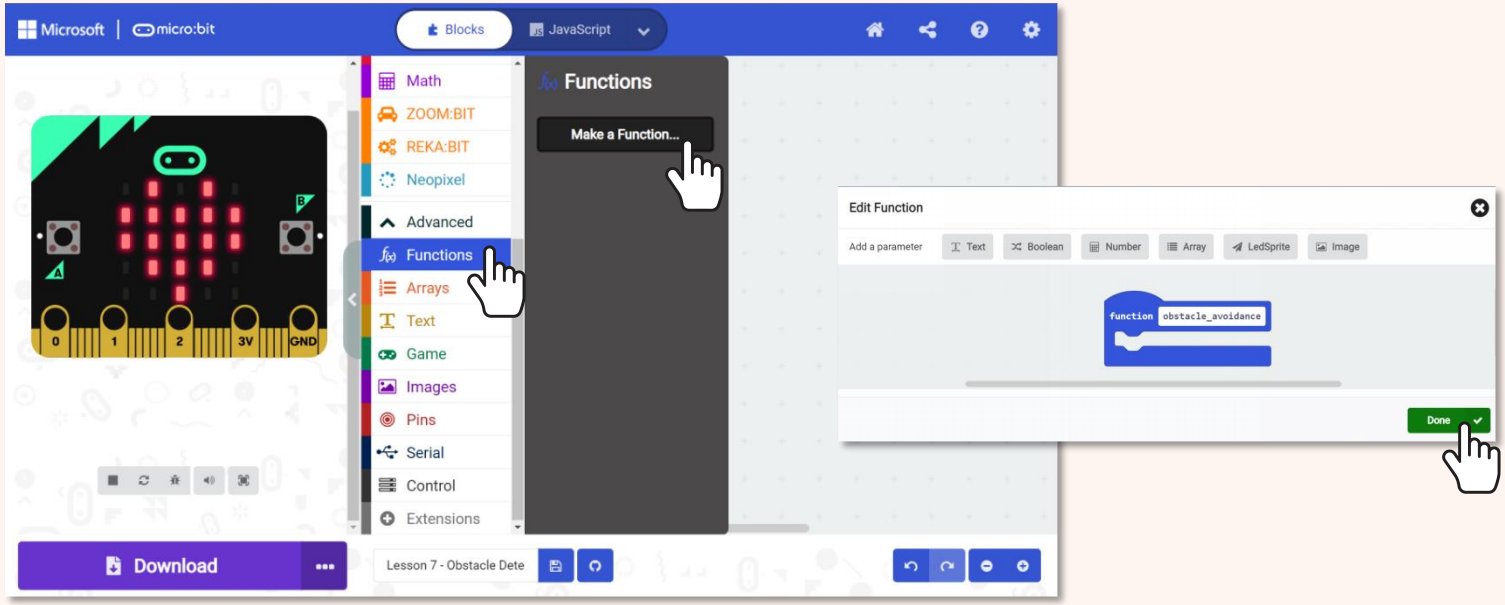
1 Create a new project in your MakeCode Editor and add ZOOM:BIT extension (you can refer to pp. 44-45). Then, build the following code for the **manual mode**.

Exclude this music block if you're using micro:bit V1.

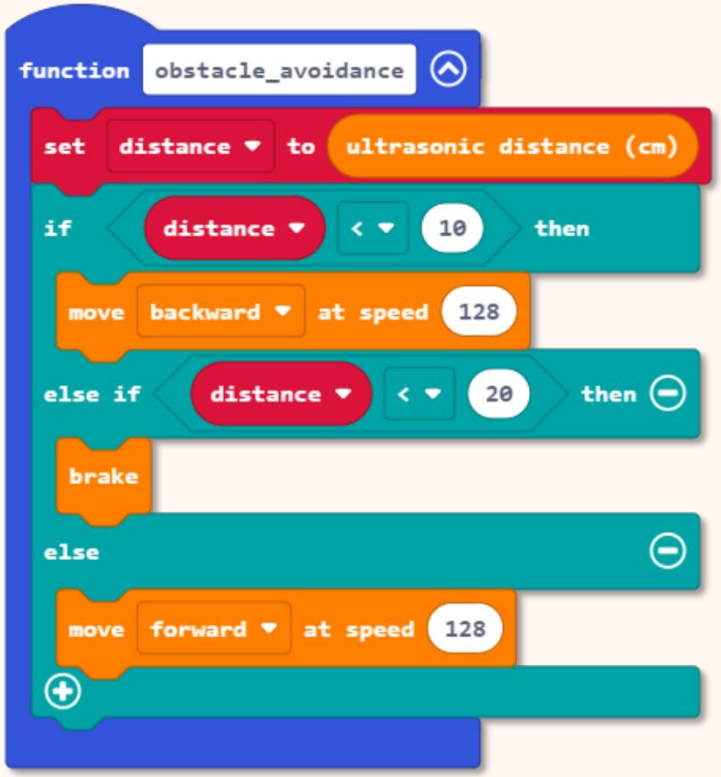


Next, let's add the other modes. To do that, we're going to use functions.

2 Click **[Advanced]** and then select **[Functions]** category. Click **[Make a Function]**, rename doSomething to **'obstacle_avoidance'** and then click **[Done]** button. A **[function obstacle_avoidance]** block will be added to your workspace.



3 Continue building your code for **obstacle avoidance mode** by adding blocks to the function block.



You can click icon to collapse the blocks of code after you've done building the function.

Click icon to open if you need to review or edit your code.

Do you notice that the code is similar to what you built in Chapter 7 ? However, here the blocks are in a [function obstacle_avoidance] block, instead of [forever] block.

4 Repeat Step 2 to create another function for **line following mode.**

5 Add blocks to the **[function line_following]** block as shown.

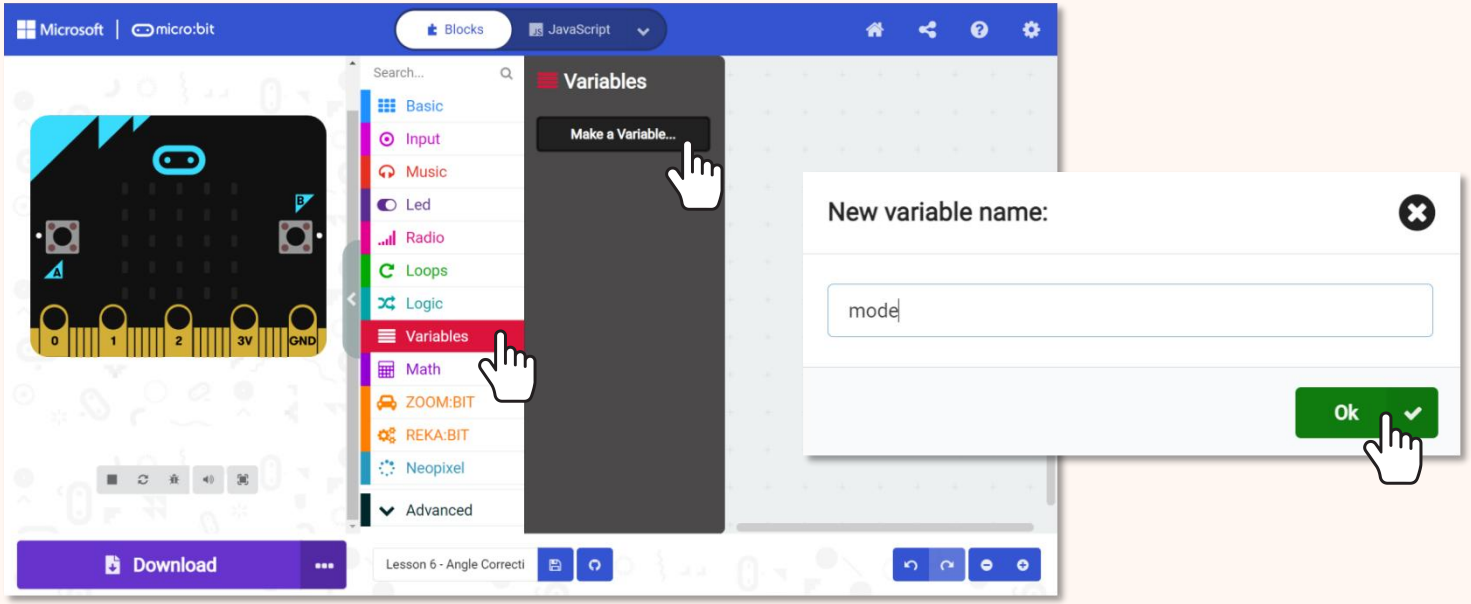
The code is similar to what you built in Chapter 8. However, here the blocks are in a [function line_following] block, instead of a [forever] block.

*Take note that you'll need to create a new variable [position].

If you're not sure how to do that, you can refer to https://link.cytron.io/zoombit-tutorial-9 for step-by-step guide to build the code.

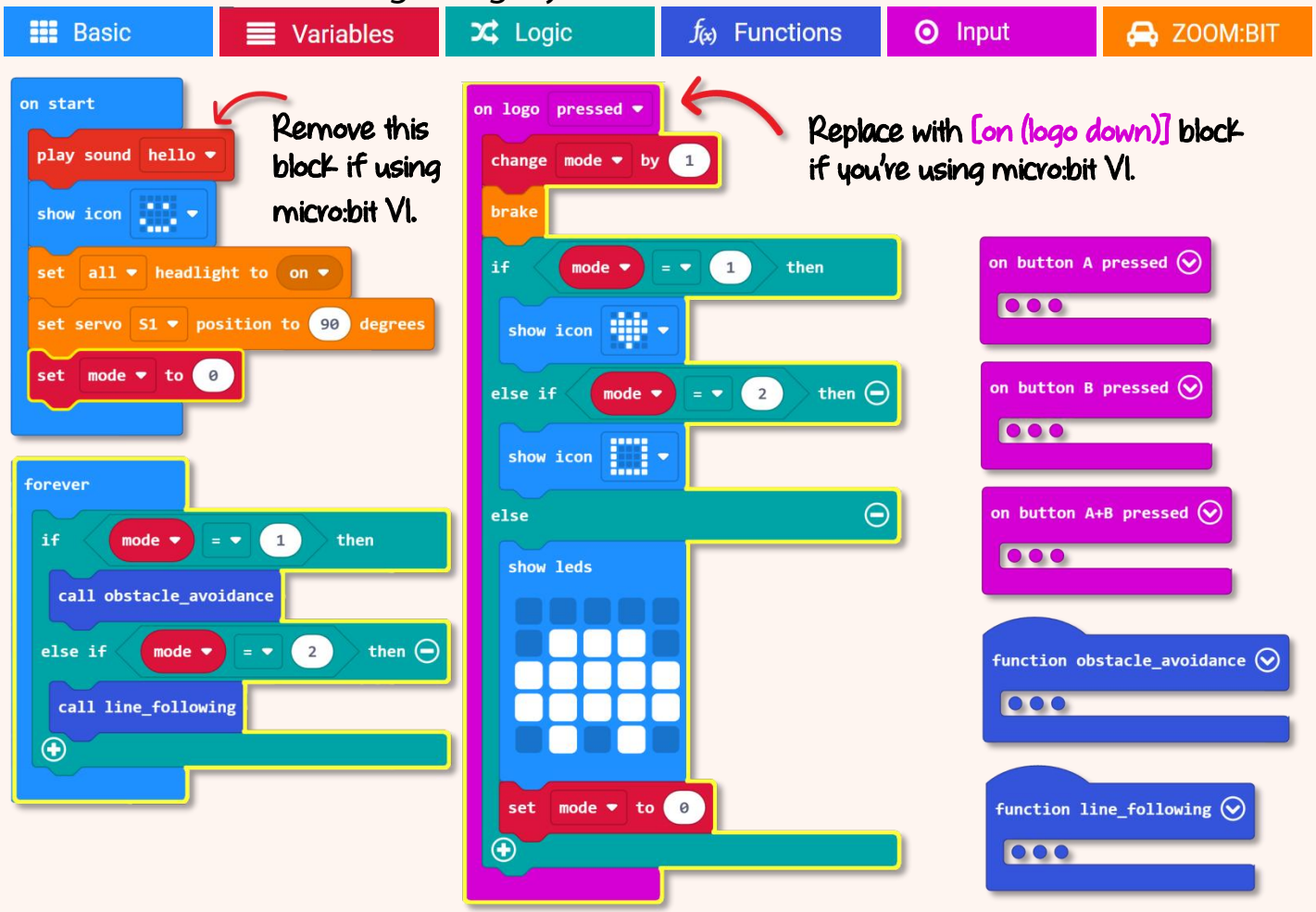
Next, we are going to add "modes" to our program so that when we change from one mode to another, ZOOM:BIT will automatically perform the corresponding task which we assign to that particular mode.

6 Click **[Variables]** category and then select **[Make a Variable]**. Name your variable (e.g. "mode") and then click **[Ok]** button.



7

Add the following highlighted blocks to your code. You can get the blocks you need from the following category drawers:

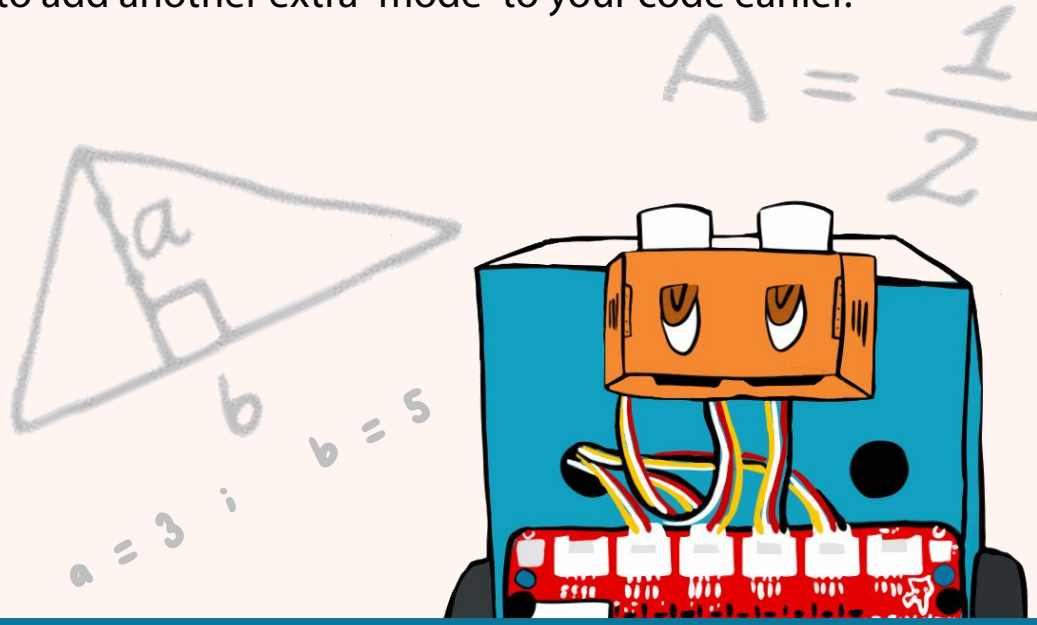


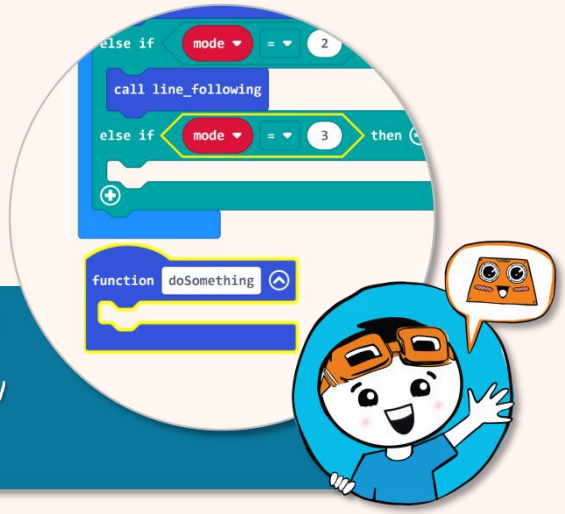
8 Download the code to your ZOOM:BIT and now you can bring your robot car wherever you go, and show off the tricks it can do to your friends~

On Start	Play sound (hello), display a smiley face, and turn on both headlights with head facing front. Set mode to 0.
Forever	Always check "mode". If Mode = 1, run Obstacle Avoidance function; else if Mode = 2, then run Line Following function.
On logo pressed (or 'on logo down' if using micro:bit V1)	Change mode by 1, stop moving and display the icon for the current mode - car (Mode 0), heart (Mode 1) and square (Mode 2). If mode is neither 1 nor 2, then set mode to 0.
On Button A Pressed	Turn right
On Button B Pressed	Turn left
On Buttons A+B Pressed	Move forward

Here's a FUN challenge for you !

Can you teach ZOOM:BIT a new trick? Teach him to solve maths equations, perhaps?
Try to add another extra "mode" to your code earlier.





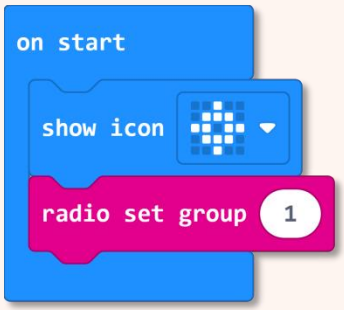
Here's a tip to help you get started - You'll need to add another else-if condition for [mode = 3] and also create a new function for your new mode. Give it a try!

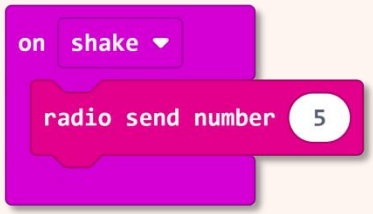
BONUS CHAPTER

https://link.cytron.io
/zoombit-bonus-chapter

Do You Know? The micro:bit on your ZOOM:BIT is equipped with radio communication function. In other words, if you have another micro:bit, you can program it to be used as a remote controller to control your ZOOM:BIT. Let's try!

1 Build the following code and download to your micro:bit board that you're going to use as the remote controller. You can get the blocks you need from these category drawers: Basic | Input | Radio





2 Create a new project in your MakeCode Editor and add ZOOM:BIT extension (you can refer to pp. 44-45).

3 Build the following code to enable your ZOOM:BIT to receive instructions from the remote controller.

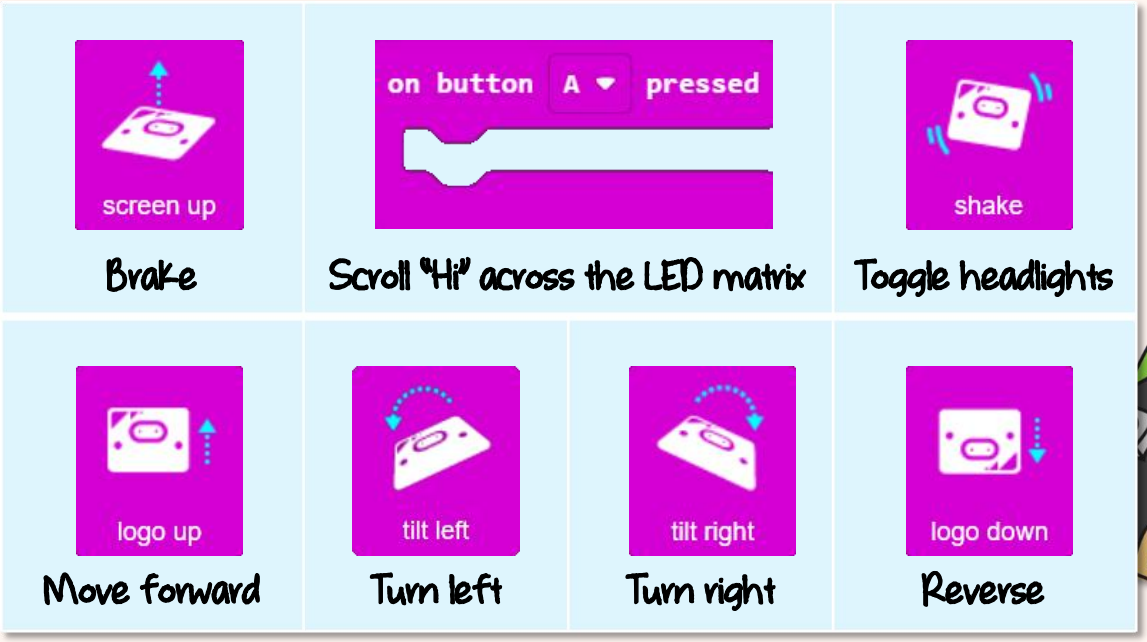
We need to set both the micro:bit (remote controller) and ZOOM:BIT to the same radio group in order for them to transmit and receive radio signals from each other. In this example, we set both to radio group 1.

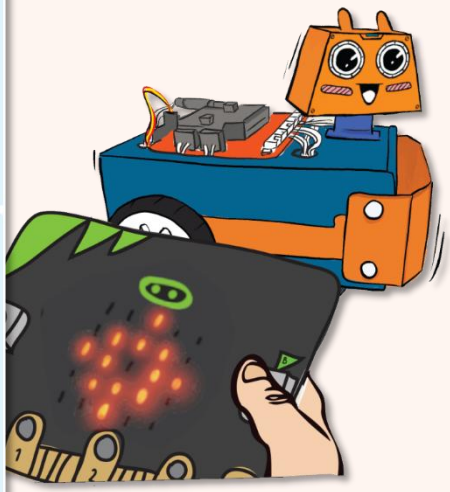
```
on start
    show icon
    radio set group 1
```

```
on radio received receivedString
    show string receivedString
    show icon
```

```
on radio received receivedNumber
    if receivedNumber = 0 then
        brake
    else if receivedNumber = 1 then
        move forward at speed 128
    else if receivedNumber = 2 then
        turn left at speed 75
    else if receivedNumber = 3 then
        turn right at speed 75
    else if receivedNumber = 4 then
        move backward at speed 128
    else if receivedNumber = 5 then
        Toggle all headlight
```


4 Download the code to your ZOOM:BIT. Power up both the micro:bit (remote controller) and ZOOM:BIT.

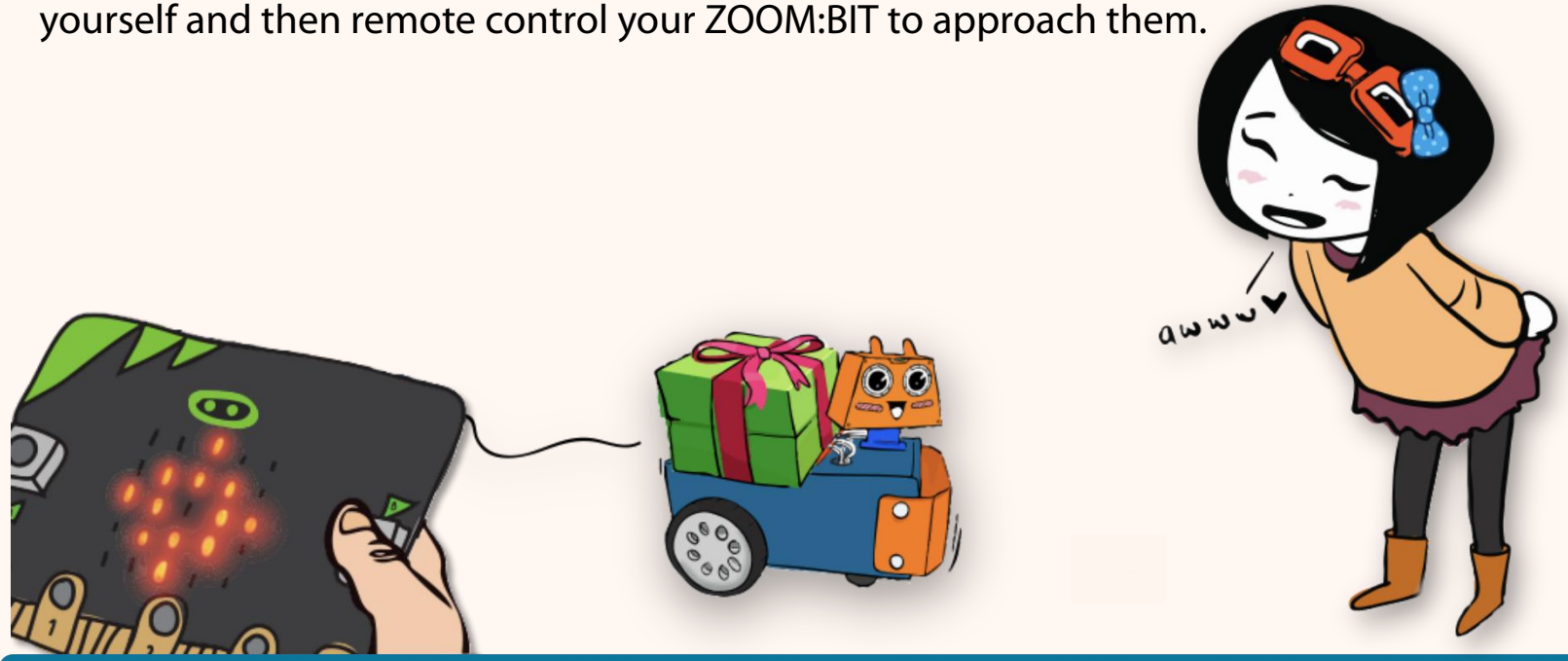




Now you can remote control your ZOOM:BIT to roam the territory. Have fun !

Here's a FUN challenge for you !

Modify the code to add more "instructions" for ZOOM:BIT to carry out - perhaps, to convey a secret message or to deliver a gift. Give your family members or friends a surprise - hide yourself and then remote control your ZOOM:BIT to approach them.



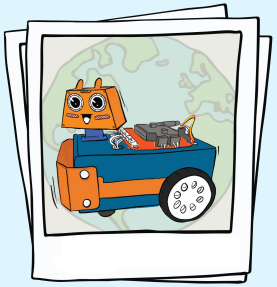
Here's a tip for you. You can add [on button B pressed] and [on button A+B pressed] to your remote controller code (micro:bit) ; and you'll need to add new else-if conditions to your ZOOM:BIT's code.

My Learning Journal with ZOOM:BIT

I finished building my ZOOM:BIT on _____ ; and together, we explored the lessons in this book and attempted all the challenges.

CHAPTER 1

Display text and animation.



Lesson & challenge completed on

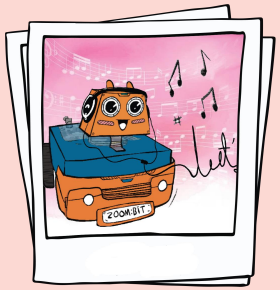
_____

Verified by :

_____

CHAPTER 2

Play simple melodies.



Lesson & challenge completed on

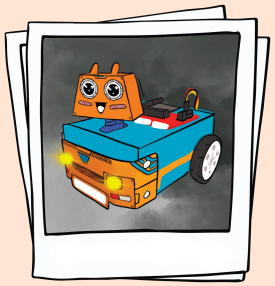
_____

Verified by :

_____

CHAPTER 3

When dark, turn on headlights.



Lesson & challenge completed on

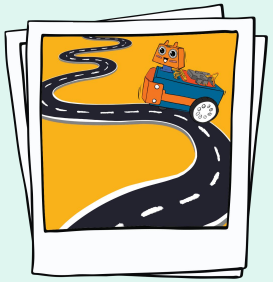
_____

Verified by :

_____

CHAPTER 4

Move & turn on command.



Lesson & challenge completed on

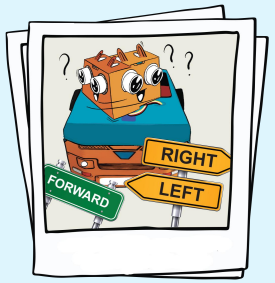
_____

Verified by :

_____

CHAPTER 5

Signal using RGB LEDs.



Lesson & challenge completed on

_____

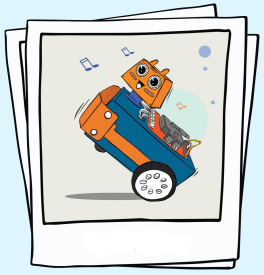
Verified by :

_____

My Learning Journal with ZOOM:BIT

CHAPTER 6

Control head movement/angle



Lesson & challenge completed on

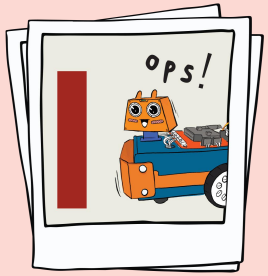
_____

Verified by :

_____

CHAPTER 7

Detect and avoid obstacles



Lesson & challenge completed on

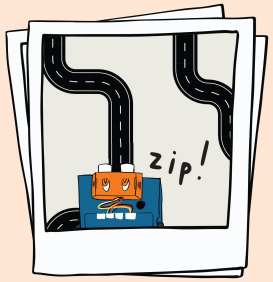
_____

Verified by :

_____

CHAPTER 8

Follow the line & stay on track



Lesson & challenge completed on

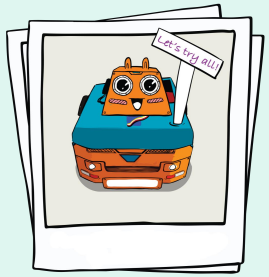
_____

Verified by :

_____

CHAPTER 9

Change from one mode to another



Lesson & challenge completed on

_____

Verified by :

_____

BONUS CHAPTER

Remote control

Lesson & challenge completed on

_____

Verified by :

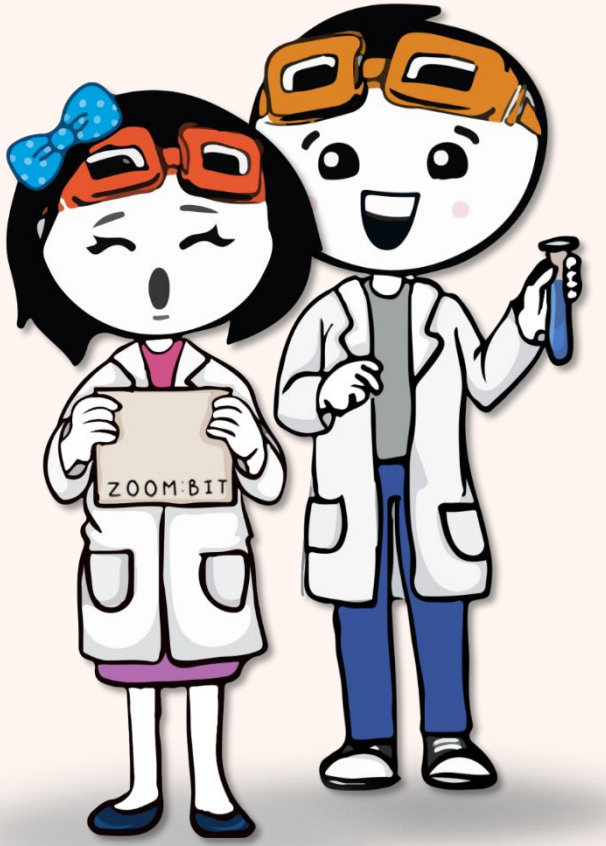
_____

P/S Get a teacher/parent to check & verify for you.

Note from vero EDUteam @ Cytron

Woohoo... CONGRATULATIONS!! You've successfully built your own robot car; and together with ZOOM:BIT, you've learned to code and completed challenges as a team. Great job! We hope you've also had fun along the way.

So what's next? You can visit www.cytron.io to explore and get add-on sensors or parts to customize your robot car. How about adding a Grove OLED Display to the I2C port? Or perhaps, adding more servo motors to form a robot arm? The possibilities are endless. Have fun exploring~



https://t.me/zoombit_support

Do share your ZOOM:BIT adventures with us on Telegram t.me./zoombit_support.
We'd love to hear from you. Cheers~

Adam & Anna

ISBN 978-967-19475-1-7