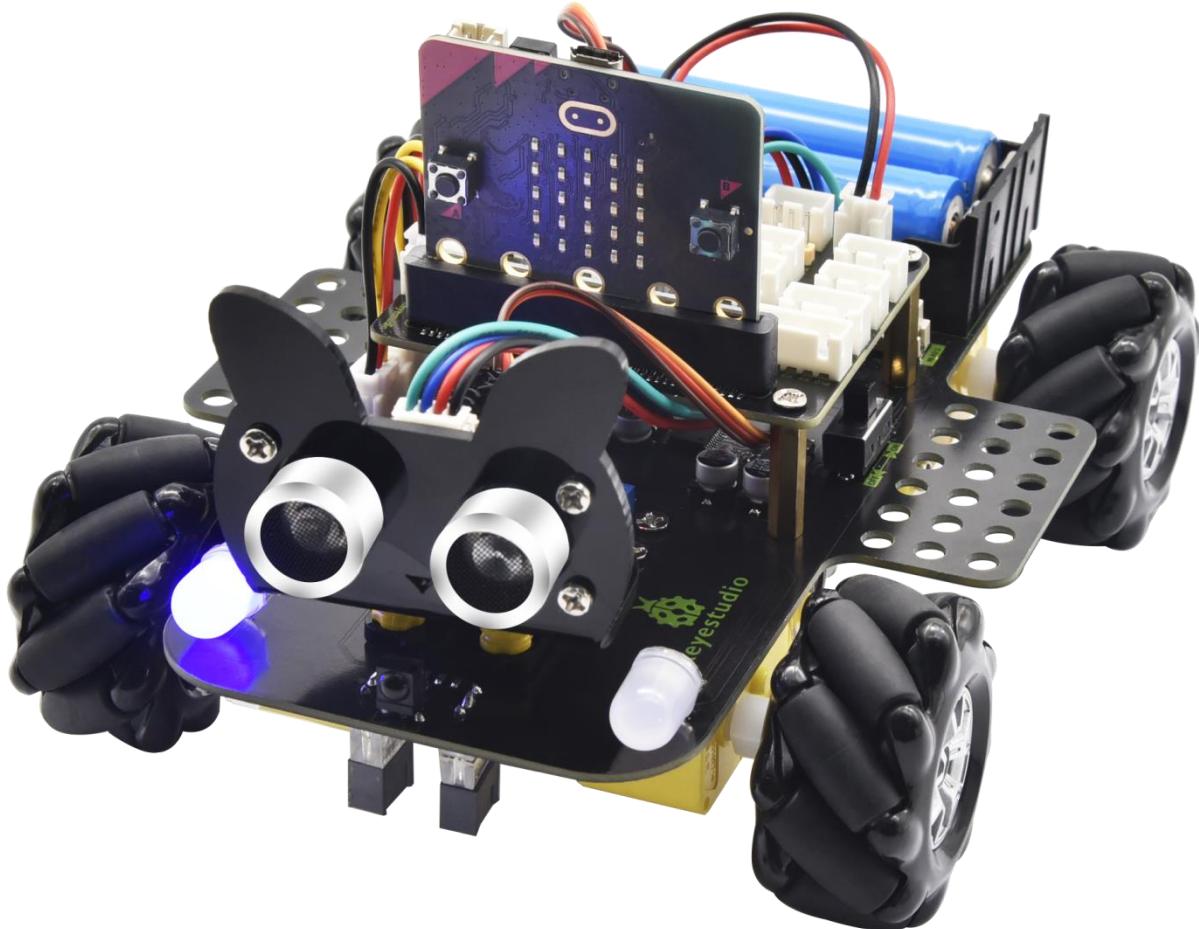




# Keyestudio 4WD Mecanum Robot Car

(Makecode)



## Contents

1. Introduction .....	4
2. Description .....	6
3. Parameters.....	6
4. Kit List.....	7
5. Preparations: .....	12



---

5.1 Background Information about Micro:bit.....	12
( 1 )What is Micro:bit? .....	12
( 2 )Layout .....	13
( 3 ) Pinout .....	14
( 4 )Notes for the application of Micro:bit main board .....	16
5.2. Install Micro:bit driver .....	17
6. Keyestudio 4WD Mecanum Robot Car.....	18
6.1. Basic Information about Keyestudio 4WD Mecanum Robot Car ..	18
6.2. the Installation of keyestudio 4WD Mecanum Robot Car .....	20
7. Get Started with Micro:bit .....	42
7.1 Write code and program: .....	42
7.2. Makecode: .....	53
7.3 Quick Download.....	54
7.4. Makecode extension library:.....	62
7.4.Resources and test code.....	71
7.5.Input test code.....	72
7.6. Install CoolTerm: .....	77
8. Projects .....	79
Project 1: Heartbeat.....	80
Project 2: Light A Single LED.....	83
Project 3: LED Dot Matrix.....	90
Project 4: Programmable Buttons.....	102



---

Project 5: Temperature Measurement .....	113
Project 6: Geomagnetic Sensor.....	127
Project 7: Accelerometer.....	142
Project 8: Light Detection .....	157
Project 9: Speaker.....	163
Project 10: Touch-sensitive Logo.....	169
Project 11: Microphone .....	175
Project 12: Bluetooth Wireless Communication.....	187
Project 13: Colorful Lights .....	208
Project 14: WS2812 RGB LEDs.....	215
Project 15: Servo .....	238
Project 16: Motor.....	247
Project 17: Line Tracking Sensor .....	260
17.1: Detect Line Tracking Sensor .....	260
17.2: Line Tracking Smart Car .....	278
Project 18: Ultrasonic Follow Smart Car.....	288
18.1: Ultrasonic Ranging .....	288
18.2: Ultrasonic Avoidance Car .....	297
18.3: Ultrasonic Follow Smart Car.....	313
Project 19: IR Remote Control .....	322
19.1: Decode IR Remote Control .....	322
19.2: IR Remote Control .....	332



---

Project 20: Bluetooth Multi-purpose Smart Car.....	348
20.1: Read Bluetooth Data.....	348
20.2: Multi-purpose Smart Car .....	363
9. Resources:.....	369

## 1. Introduction

Have you wondered to learn programming or have your own programming robot? Nowadays, programming has developed to a lower age group, and it will be a trend for everyone to be able to program thanks to the spread of simple graphical programming platforms, from micro:bit to Arduino and Raspberry Pi. Maybe you haven't heard of them before. It doesn't matter because with the help of this product and tutorial, you can easily install a multi-functional programming car and experience the fun of being a maker.

Micro:bit is a highly integrated microcontroller of powerful functions and small size. It is very suitable to be applied in STEAM education for its functions to make robots, wearable devices and electronic interactive games via the combination of code programming and graphical programming.



---

This Keyestudio 4WD Mecanum Robot Car is a smart DIY car specially designed for micro:bit. The smart car kit consists of a car body with extended functions, a PCB base plate with integrated motor drive sensors, 4 decelerating DC motors, Mecanum wheels, various modules and sensors and acrylic boards. Therefore, you can easily assemble a cool Mecanum wheel 4WD smart car by yourself, and then use Microsoft's online graphical programming platform Make Code to program the micro:bit control board to control the car. In the process, you can not only experience the fun of creation but enhance hands-on ability and learn programming skills as well.

MakeCode for micro:bit is the most widely used graphical programming environment on the micro:bit official website. It is based on the graphical programming environment developed by Microsoft's open source project MakeCode. This graphical programming can also be converted to code languages, python and javascript language. This combination makes learn programming easy. At the same time, MakeCode programming can be simulated or programmed for actual electronic components.

For your convenience, source code has been provided in every project, as well as code programming steps and code explanation in details. Hope you can better understand them.



---

## 2. Description

This product is a smart car based on Micro:bit. It boasts multiply functions including ultrasonic sound following, line tracking, infrared control and Bluetooth control. It comes with a passive buzzer which is able to play music, 4 WS2812RGB LEDs to display different colors, 2 colorful lights to make direction lights for the car. This product uses two 18650 lithium batteries for power supply.

When installing and disassembling the battery, please pay attention to the positive and negative poles of the battery, and be sure not to reverse them. By the way, the motor speed of this product is adjustable.

In order to provide you with better experience, corresponding documents about installation and test code are also provided.

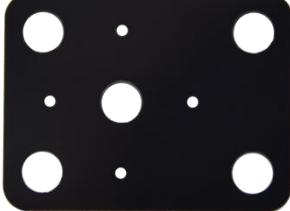
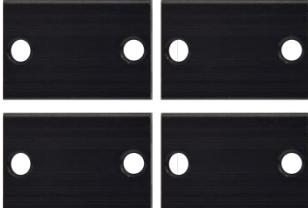
## 3.Parameters

- ◆ Connector port input: DC 6V---9V
- ◆ Operating voltage of drive board system: 5V
- ◆ Standard operating power consumption: about 2.2W
- ◆ Maximum power: Maximum output power is 12W
- ◆ Motor speed: 200RPM/1min
- ◆ Working temperature range: 0-50°C
- ◆ Size: 120\*120\*120mm
- ◆ Environmental protection attributes: ROHS

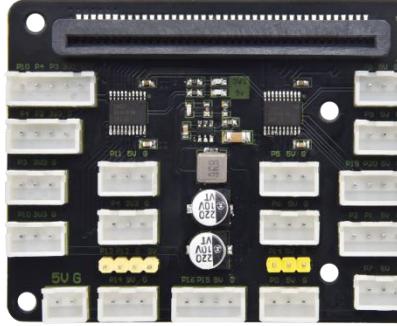


Note: working voltage of micro:bit is 3.3V, driver shield integrates 3.3V/5V communication conversion circuit.

#### 4. Kit List

#	Picture	Components	Quantity
1		KS0511 Acrylic Board T=3mm	1
2		Acrylic Board with Lego Holes T=3mm	1
3		4.5V Motor	4
4		23*15*5MM Fixing Board	4



5		Servo	1
6		Mecanum Wheels	4
7		Keyestudio Micro:bit IO Port Expansion Sensor Shield With Level Conversion	1
8		Micro:bit Main Board V2.0 with Package for KS4031	1
		Micro:bit Main Board V2.0 for KS4032	0



9	A black printed circuit board (PCB) with a central microcontroller, various components, and two battery slots at the top right.	Keyestudio Driver Board	1
10	Four gold-colored metal pillars arranged in a 2x2 grid.	M3*20MM Dual-pass Copper Pillar	4
11	Four yellow plastic LEGO Technic connector pieces, each with a central hole and four side holes.	4265c Lego Part	4
12	Four blue plastic LEGO Technic connector pieces, each with a central slot and four side holes.	43093 Lego Part	4
13	Six clear acrylic gaskets arranged in a 2x3 grid.	Acrylic Gasket Six in One Pack	1



14		M3*6MM Round Head Screw	18
15		Keyestudio Ultrasonic Module	1
16		M3 Nickle-plated Nut	14
17		M3*30MM Round Head Screw	9
18		M2 Nickle-plated Nut	3
19		M2*8MM Round Head Screw	3



20		M3*8MM Round Head Screw	5
21		Remote Control (without batteries)	1
22		Plastic String 3*100mm	5
23		USB Cable	1
24		HX-2.54 2P DuPont Wire 100mm	1
25		HX-2.54 4P DuPont Wire 50mm	2
26		XH2.54 4P DuPont Wire 160mm	1
27		XH2.54 3P DuPont Wire 50mm	2



28		3*40mm Screwdriver	1
29		M1.2*5mm Round Head Self-tapping Screw	6

## 5.Preparations:

### 5.1Background Information about Micro:bit

#### ( 1 )What is Micro:bit?

Micro:bit is an open source hardware platform based on the ARM architecture launched by British Broadcasting Corporation (BBC) together with ARM, Barclays, element14, Microsoft and other institutions. The core device is a 32-bit Arm Cortex-M4 with FPU micro-processing.

Though it is just the size of a credit card, the Micro:bit main board is equipped with loads of components, including a 5\*5 LED dot matrix, 2 programmable buttons, an accelerometer, a compass, a thermometer, a touch-sensitive logo and a MEMS microphone, a Bluetooth module of low energy, and a buzzer and others. Thus, it also boasts multiple functions.

The buzzer built in the other side of the board makes playing all kinds of sound possible without any external equipment. The golden fingers and

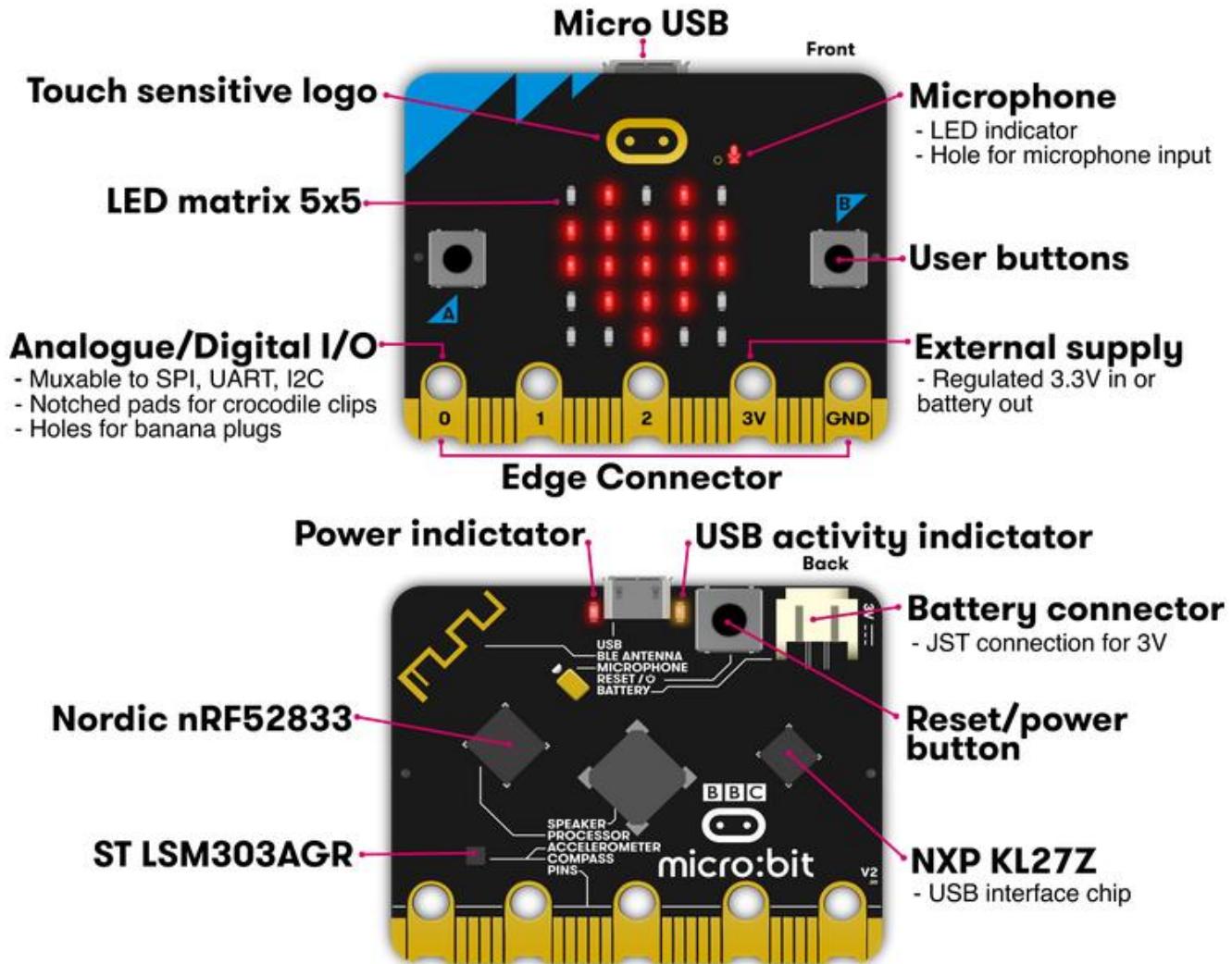


---

gears added provide a better fixing of crocodile clips. Moreover, this board has a sleeping mode to lower power consumption of batteries and it can be entered if users long press the Reset & Power button on the back of it. It is capable of reading the data of sensors, controlling servos and RGB lights and attaching with a shield so as to connect with various sensors. It also supports a variety of codes and graphical programming platforms, and is compatible with almost all PCs and mobile devices. It has no need to install drivers. It is of high integration of electronic modules, and has a serial port monitoring function for easy debugging.

The board has found wide applications. It can be applied in programming video games, making interactions between light and sound, controlling a robot, conducting scientific experiments, developing wearable devices and make some cool inventions like robots and musical instruments, basically everything imaginable.

## ( 2 )Layout



For more information, please resort to following links :

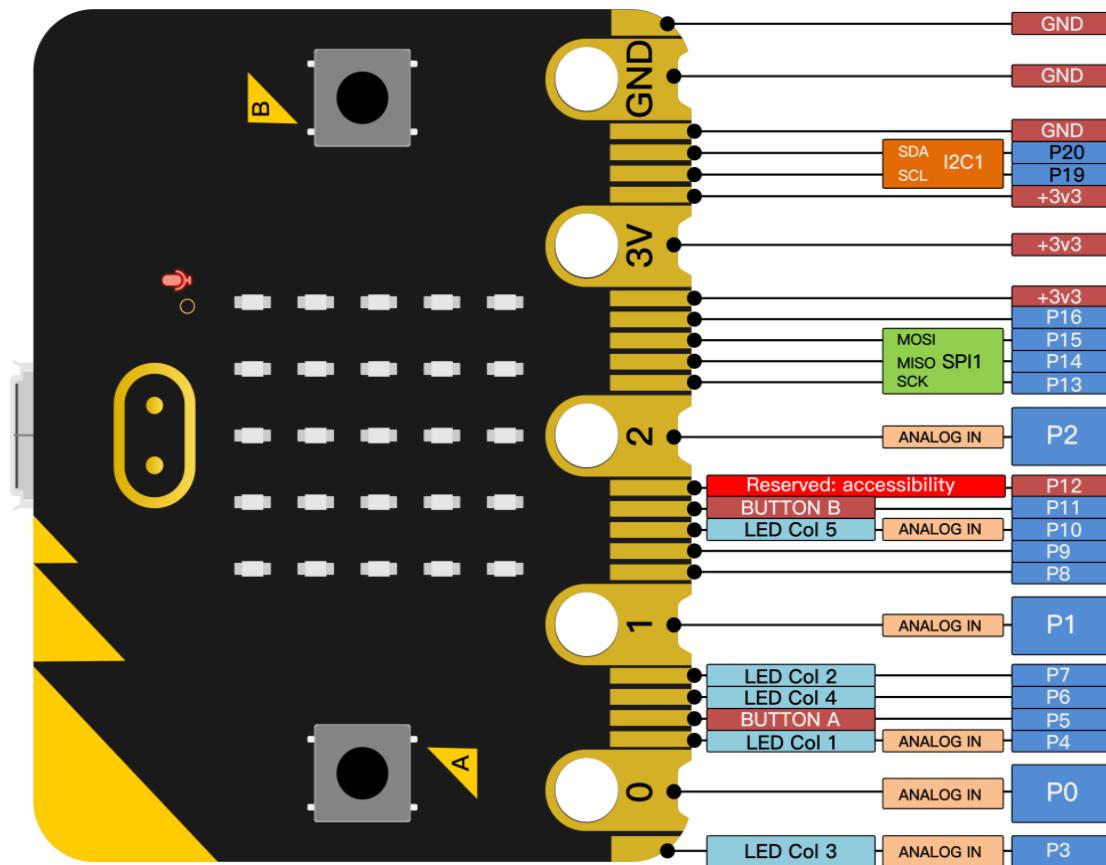
<https://tech.microbit.org/hardware/>

<https://microbit.org/new-microbit/>

<https://www.microbit.org/get-started/user-guide/overview/>

<https://microbit.org/get-started/user-guide/features-in-depth/>

### ( 3 ) Pinout



### The functions of pins:

GPIO	P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P19, P20
ADC/DAC	P0, P1, P2, P3, P4, P10
IIC	P19 (SCL) , P20 (SDA)
SPI	P13 (SCK) , P14 (MISO) , P15 (MOSI)
PWM (used frequently)	P0, P1, P2, P3, P4, P10
PWM (not frequently used)	P5, P6, P7, P8, P9, P11, P12, P13, P14, P15, P16, P19, P20



Occupied	P3(LED Col3), P4(LED Col1), P5(Button A), P6(LED Col4), P7(LED Col2), P10(LED Col5), P11(Button B)
----------	---

Browse the official website for more details:

<https://tech.microbit.org/hardware/edgeconnector/>

<https://microbit.org/guide/hardware/pins/>

#### ( 4 )Notes for the application of Micro:bit main board

- a. It is recommended to cover it with a silicone protector to prevent short circuit for it has a lot of sophisticated electronic components.
- b. Its IO port is very weak in driving since it can merely handle current less than 300mA. Therefore, do not connect it with devices operating in large current, such as servo MG995 and DC motor or it will get burnt. Furthermore, you must figure out the current requirements of the devices before you use them and it is generally recommended to use the board together with a Micro:bit shield.
- c. It is recommended to power the main board via the USB interface or via the battery of 3V. The IO port of this board is 3V, so it does not support sensors of 5V. If you need to connect sensors of 5 V, a Micro: Bit expansion board is required.



- d. When using pins(P3, P4, P6, P7 and P10)shared with the LED dot matrix, blocking them from the matrix or the LEDs may display randomly and the data about sensors connected maybe wrong.
- e. Pin 19 and 20 can not be used as IO ports though the Makecode shows they can. They can only be used as I2C communication.
- f. The battery port of 3V cannot be connected with battery more than 3.3V or the main board will be damaged.
- g. Forbid to operate it on metal products to avoid short circuit.

To put it simple, Micro:bit V2 main board is like a microcomputer which has made programming at our fingertips and enhanced digital innovation. And as for programming environment, BBC provides a website:

<https://microbit.org/code/>, which has a graphical MakeCode program easy for use.

## 5.2. Install Micro:bit driver

Micro:bit is free of driver installation. However, in case your computer fail to recognize the main board, you can install the diver too.



Just enter the link <https://fs.keyestudio.com/KS4031-4032>

to download the driver file   mbed\_usb\_2020\_x64\_1212.exe of micro:bit in file folder   5. Microbit Driver Installation

## 6.Keyestudio 4WD Mecanum Robot Car

This chapter will introduce the function and structure of keyestudio 4WD Mecanum Robot Car. It is a programmable car based on BBC micro:bit. Driven by motors, it boasts a line tracking sensor and an infrared receiver integrated into the bottom plate, an ultrasonic sensor, servos ,2 colorful lights, 4 WS2812 RGB lights. The wiring is not complicated and it has Lego jacks to facilitate connection with other peripheral devices. Abundant hardware resources will enable you to master more knowledge and skills, so that you can use your imagination to create more technological inventions.

### 6.1.Basic Information about Keyestudio 4WD Mecanum Robot Car

This car can help you to better learn to use Micro:bit and obtain electronic knowledge.

**Components:** an ultrasonic sensor, servos ,2 colorful lights, 4 WS2812 RGB



lights 4 decelerating DC motors, Mecanum wheels,

Sensor	Colorful light	Decelerating DC motor	Servo	Ultrasonic sensor	Line Tracking Sensor	Infrared Receiver	WS2812 RGB light	Power switch
#	2	4	1	1	1	1	4	1

Note: the line tracking sensor, WS2812 RGB lights and infrared receiver servo are integrated in the base.

#### Pins:

Pin on Micro:bit	Sensors of the keyestudio 4WD Mecanum Robot Car
P1 P2	Line Tracking Sensor
P14	Servo
P8	4 个 WS2812RGB Lights
P9	Infrared Receiver
P15P16	Ultrasonic Sensor

#### Power supply and Battery

The keyestudio 4WD Mecanum Robot Car is powered by two 18650 batteries. The battery holder of the car is compatible with any type of 18650 lithium battery (rechargeable). You can use a universal battery



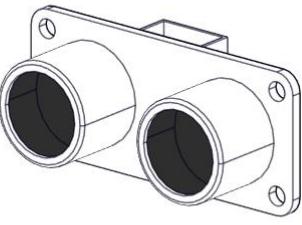
charger to charge the 18650 lithium battery.

Please note: This product does not contain batteries.

## 6.2. the Installation of keyestudio 4WD Mecanum Robot Car

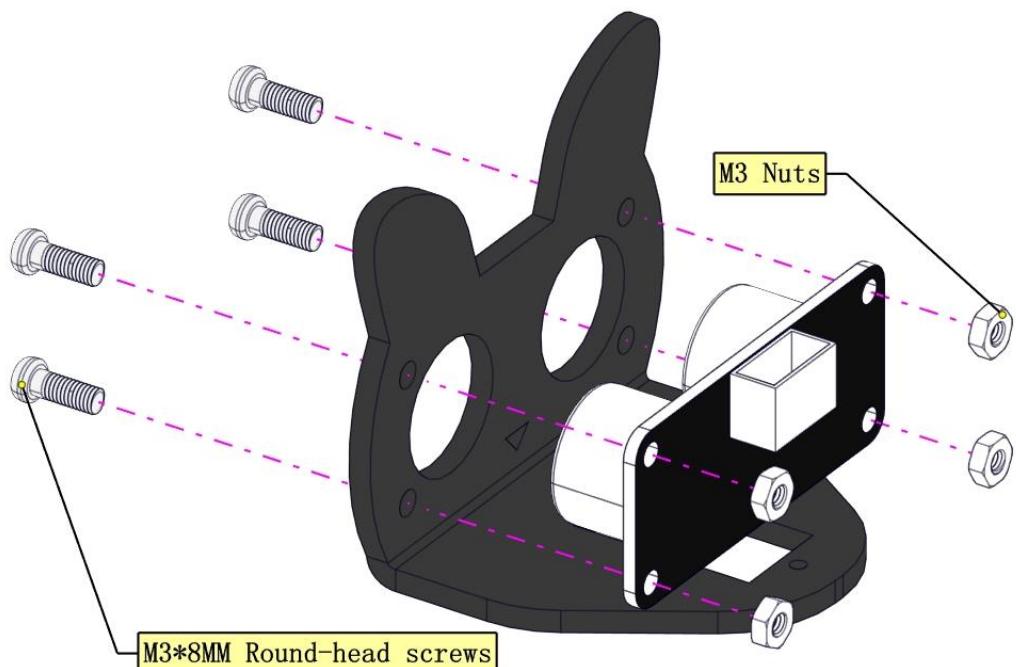
Part 1	
Components Needed	
	 <b>Acrylic Boards</b> ×1

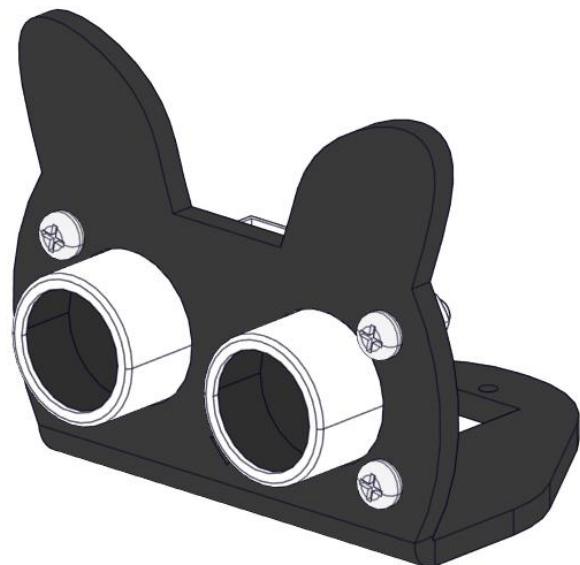
 <b>Ultrasonic Sensor</b> ×1	 <b>M3 Nuts</b> ×4
 <b>M3*8MM Round-head screws</b> ×4	



Installation  
Diagram



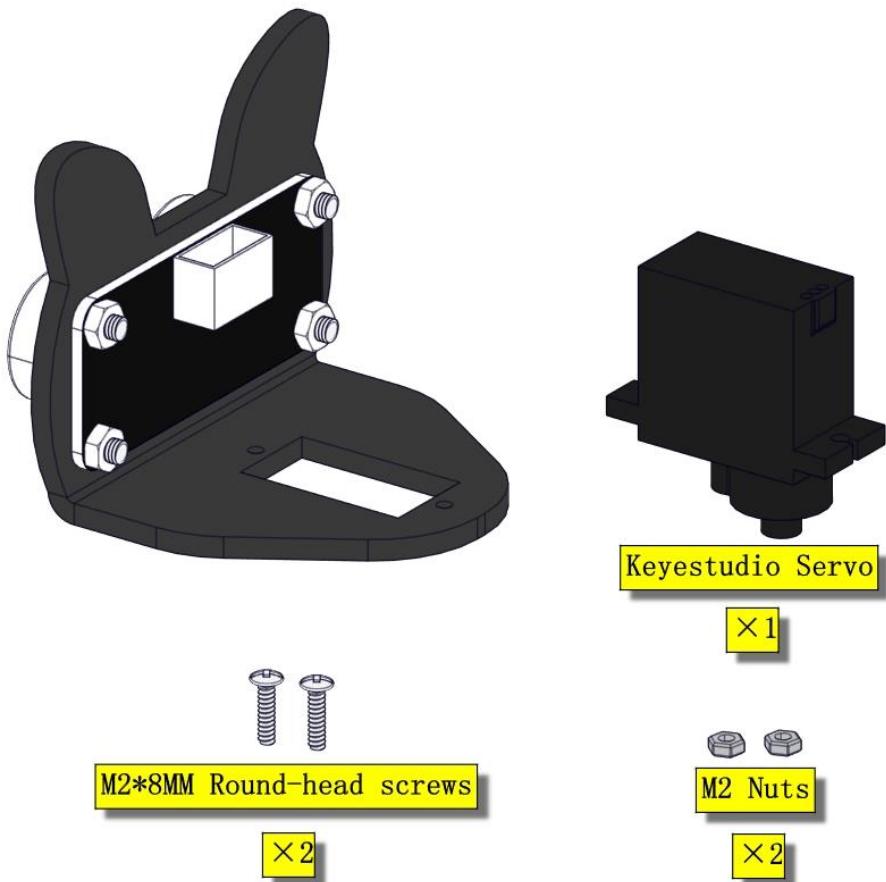
Prototype



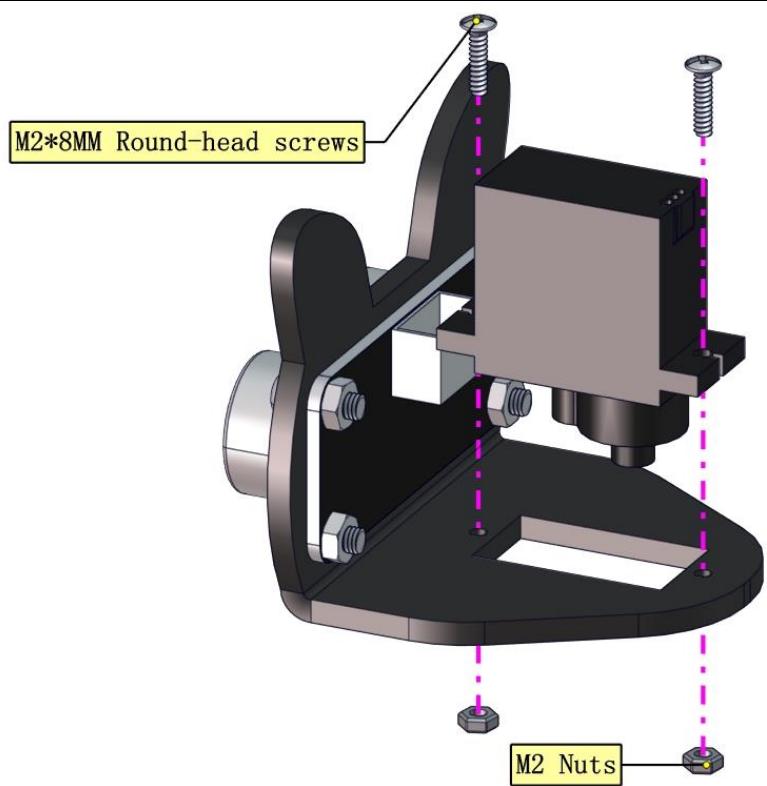
**Part 2**



Components  
Needed

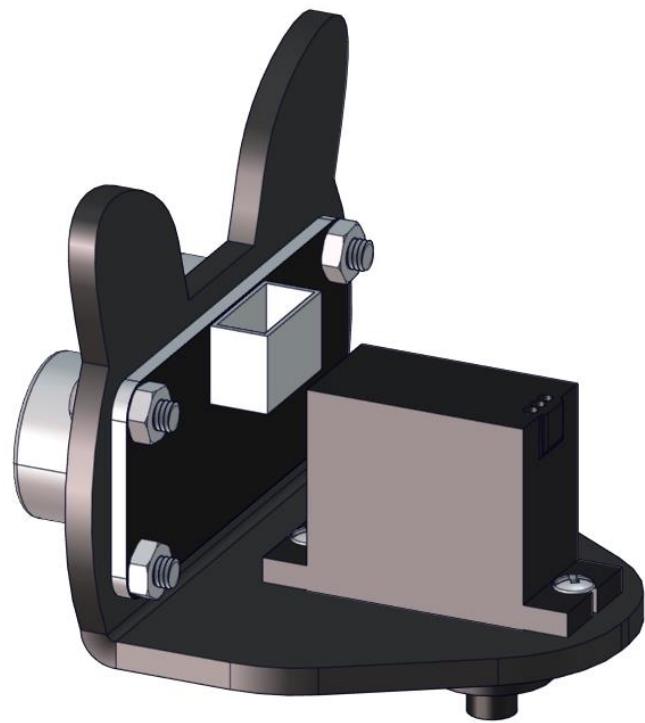


Installation  
Diagram





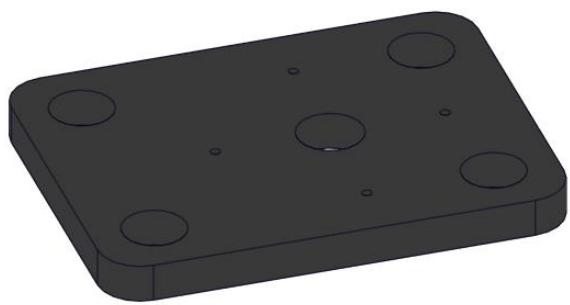
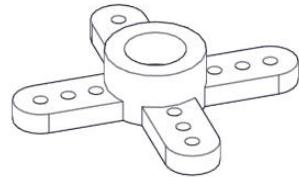
Prototype



### Part 3



## Components Needed



Control horn(belong to servo)

×1

Acrylic Boards

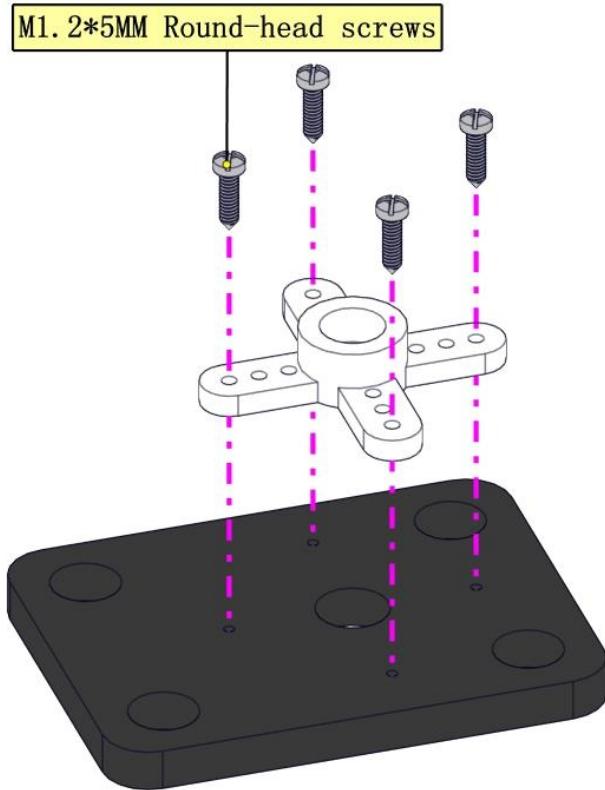
×1



M1.2\*5MM Round-head screws

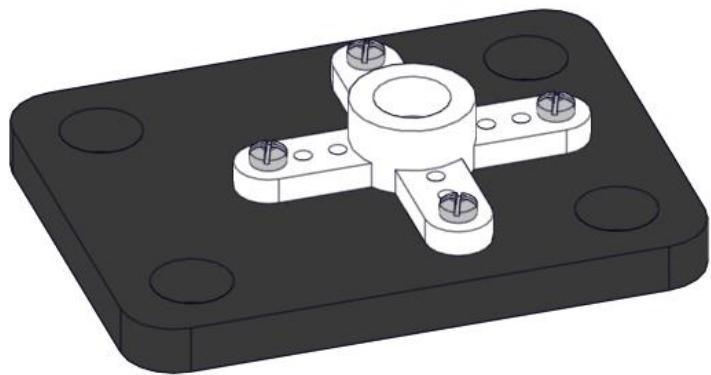
×4

## Installation Diagram





Prototype



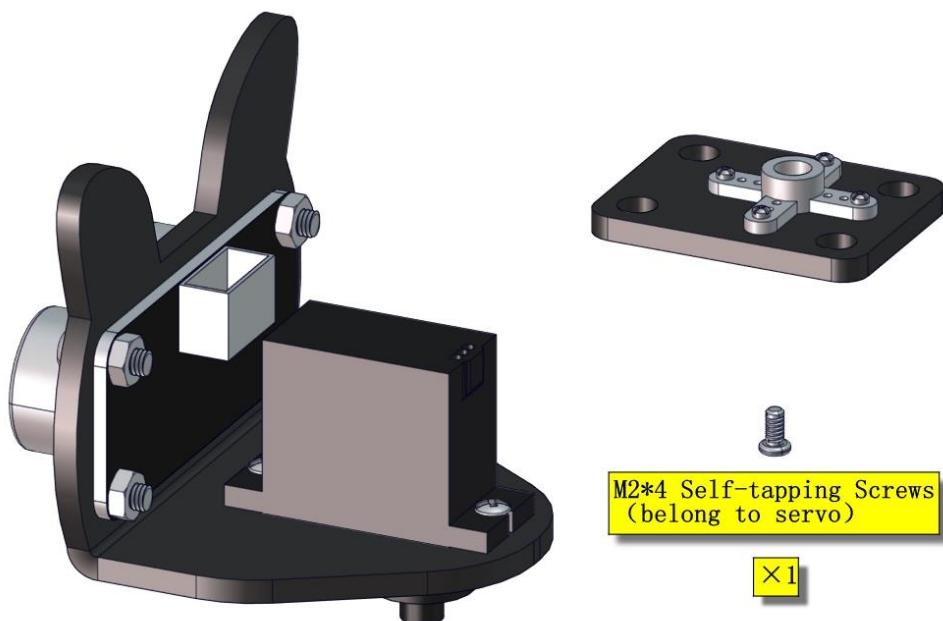
#### Part 4 (adjust the angle of the servo first)

Adjust the angle of the servo to 90 degrees according to the test code in project 8.15

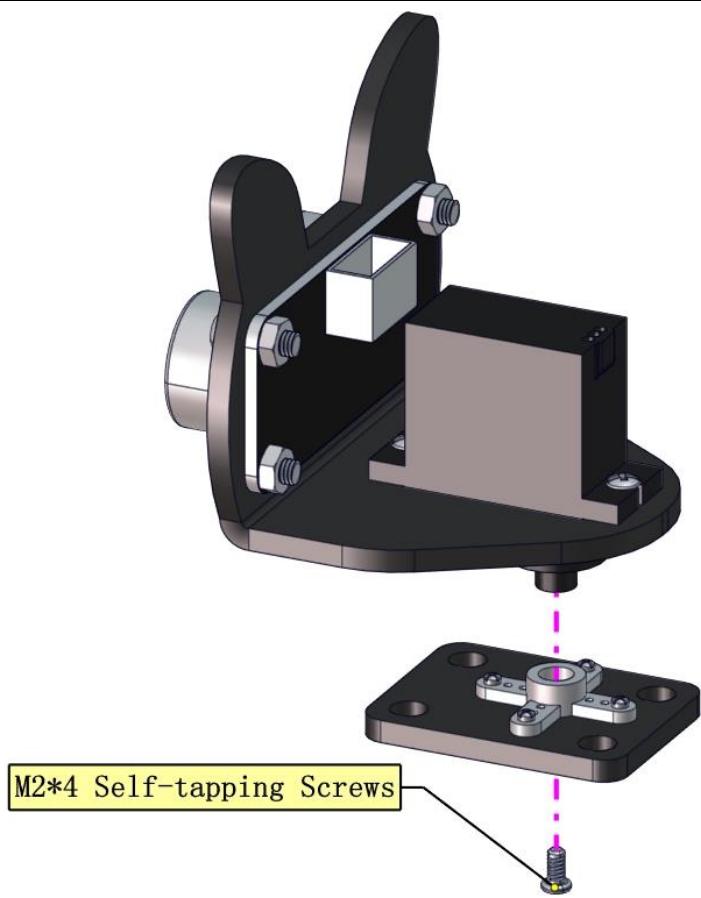




Components  
Needed

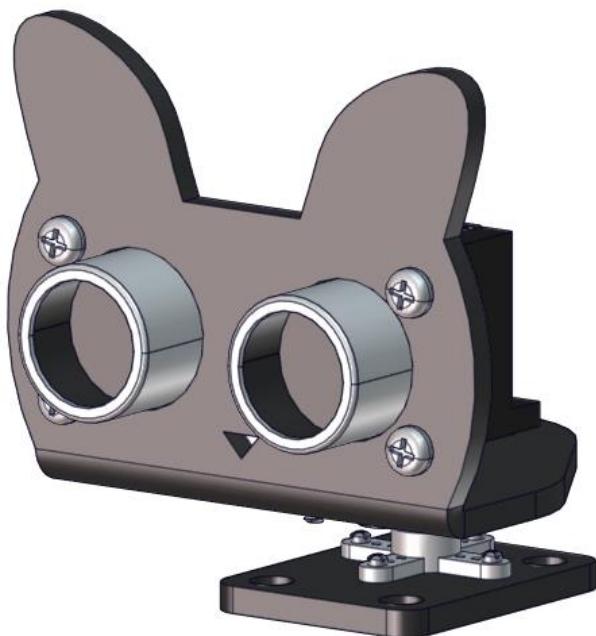


Installation  
Diagram  
(mind the  
installation  
direction)



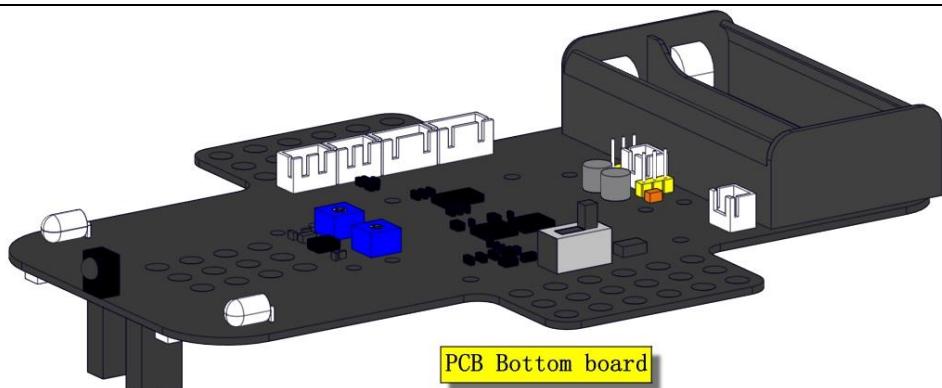


Prototype



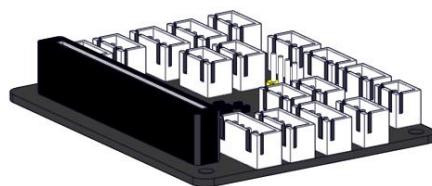
## Part 5

Components  
Needed



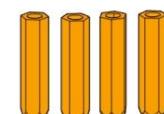
PCB Bottom board

×1



Micro:bit Expansion Board

×1



M3\*20MM Dual-pass  
Copper Pillar

×4

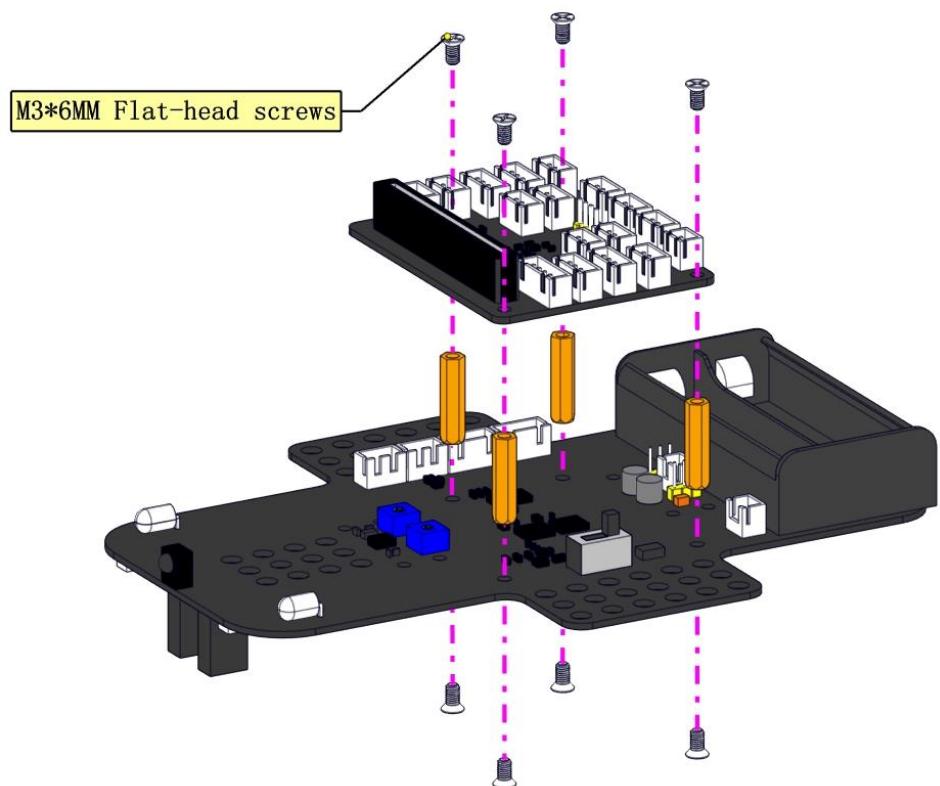


M3\*6MM Flat-head screws

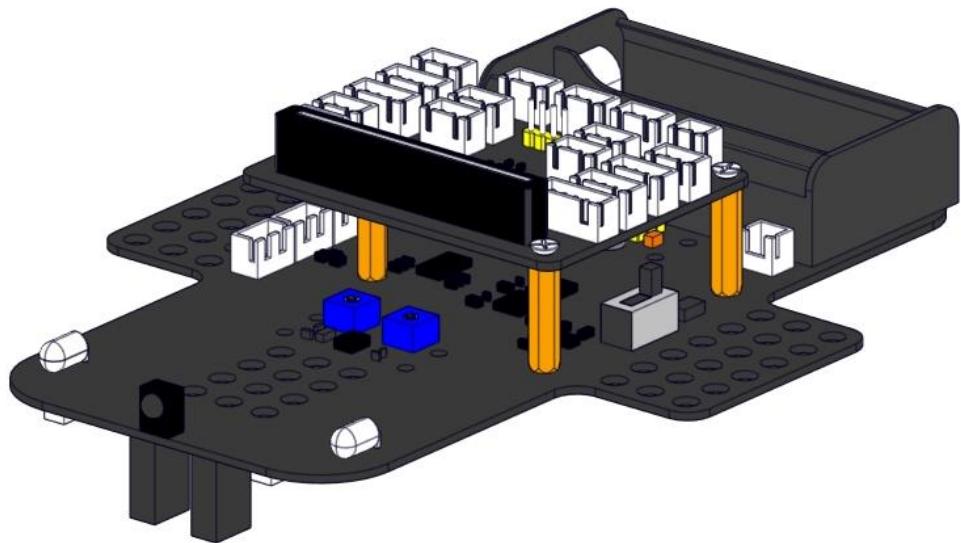
×8



Installation  
Diagram



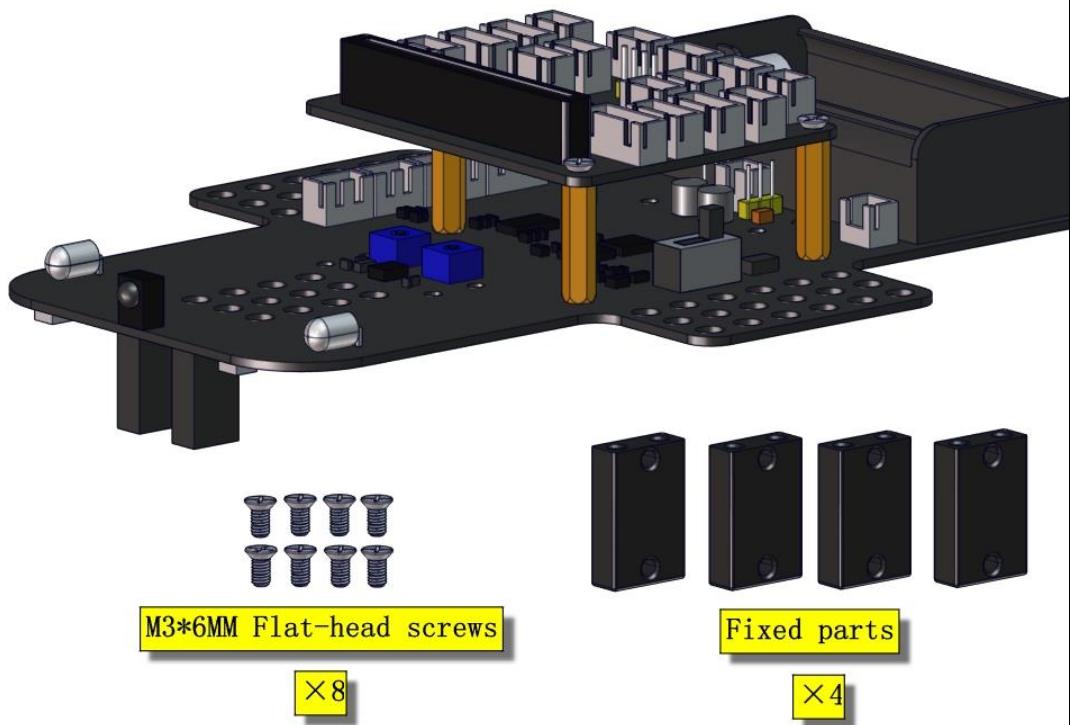
Prototype



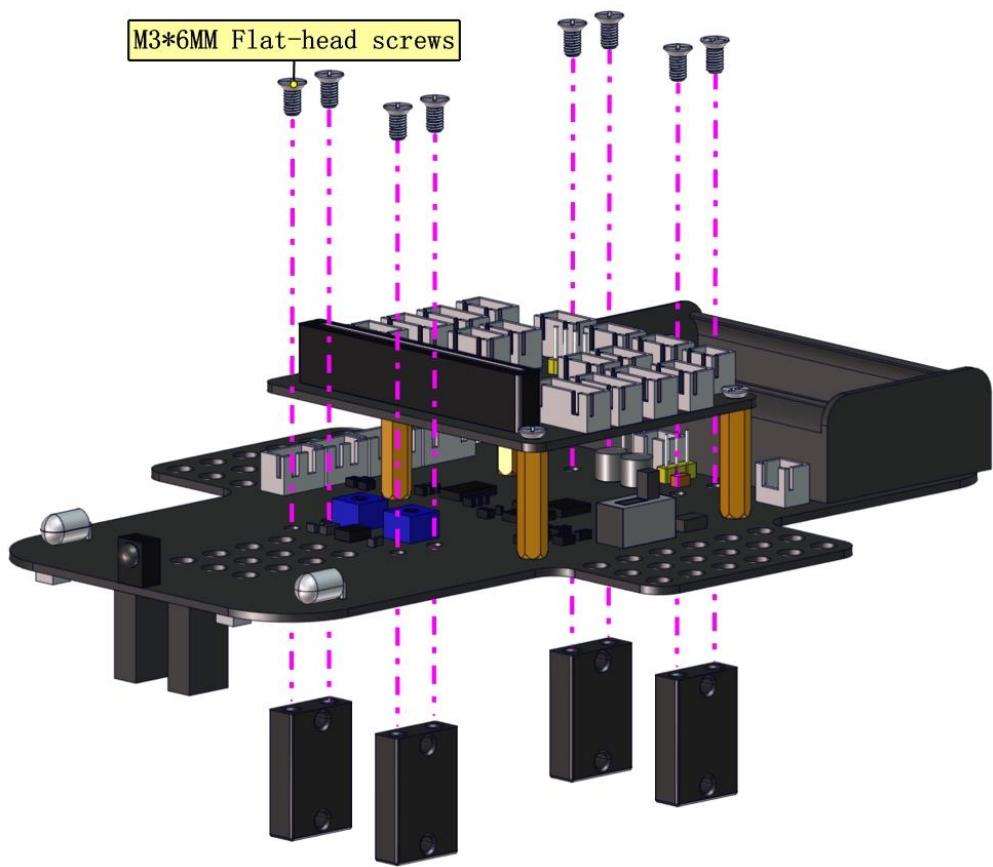
**Part 6**



Components  
Needed

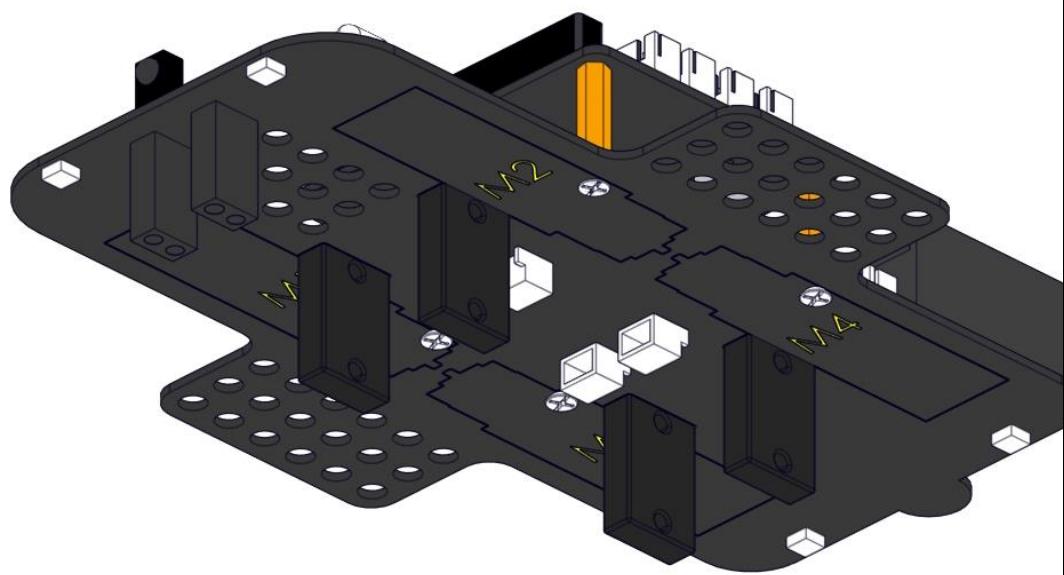


Installation  
Diagram



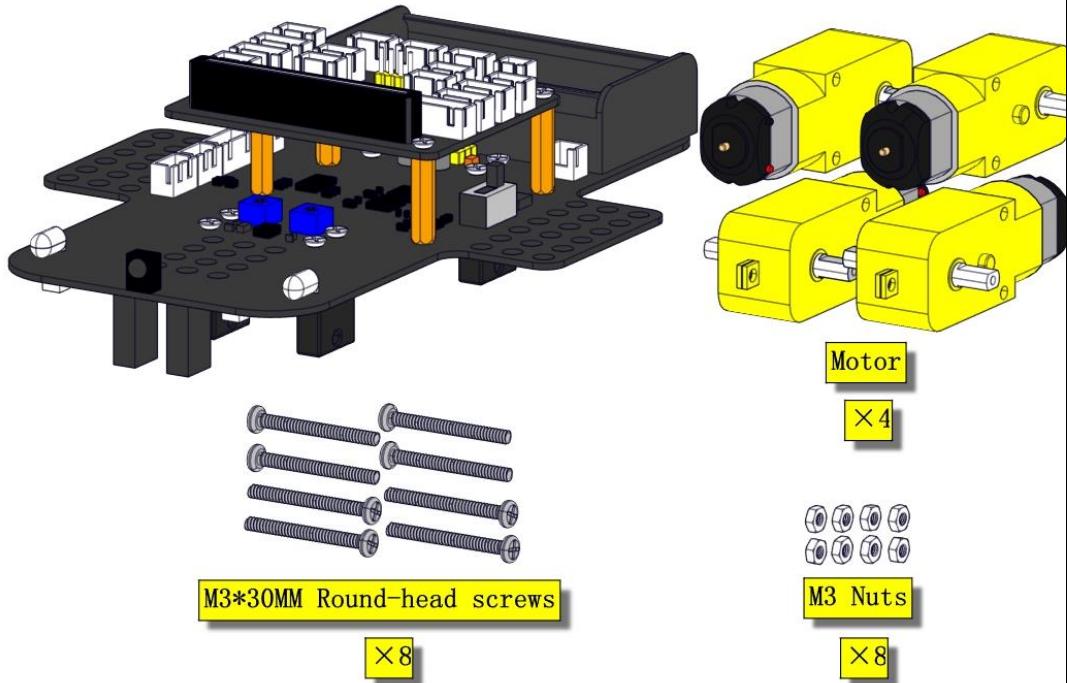


Prototype



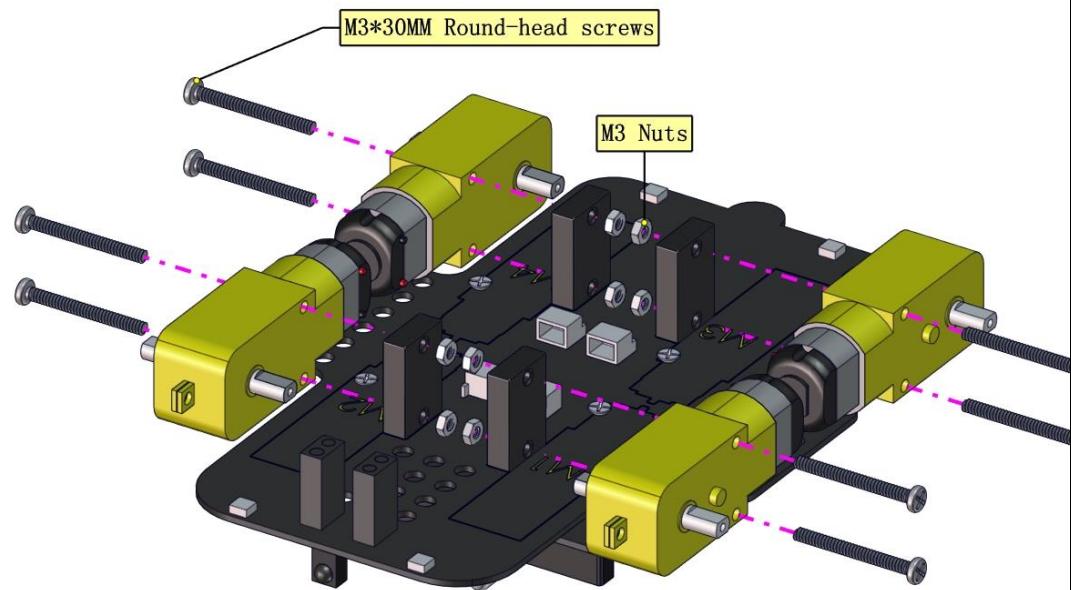
## Part 7

Components  
Needed

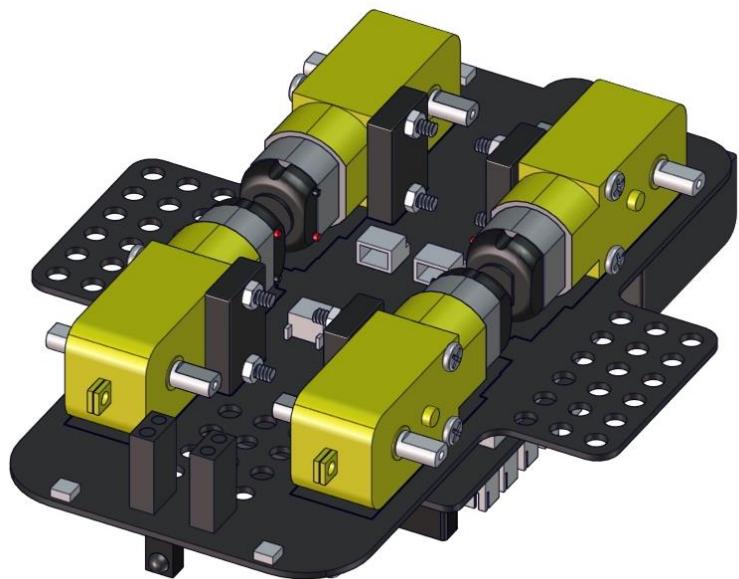




Installation  
Diagram  
(mind the  
direction of  
the motor)



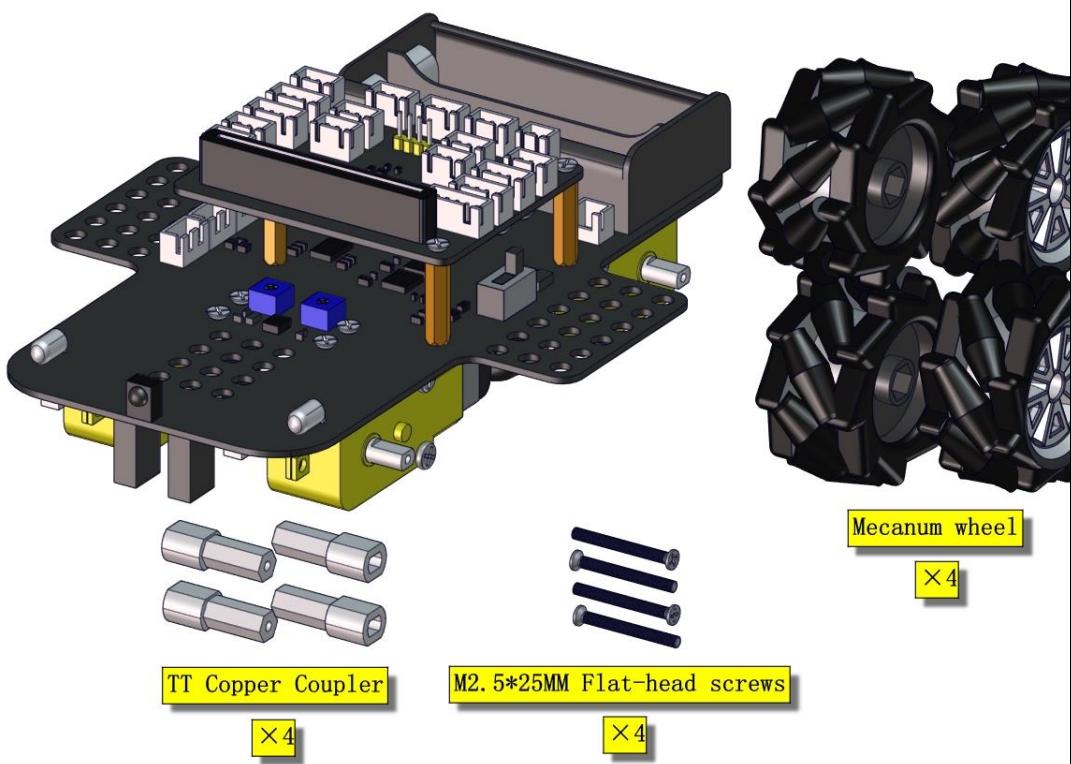
Prototype



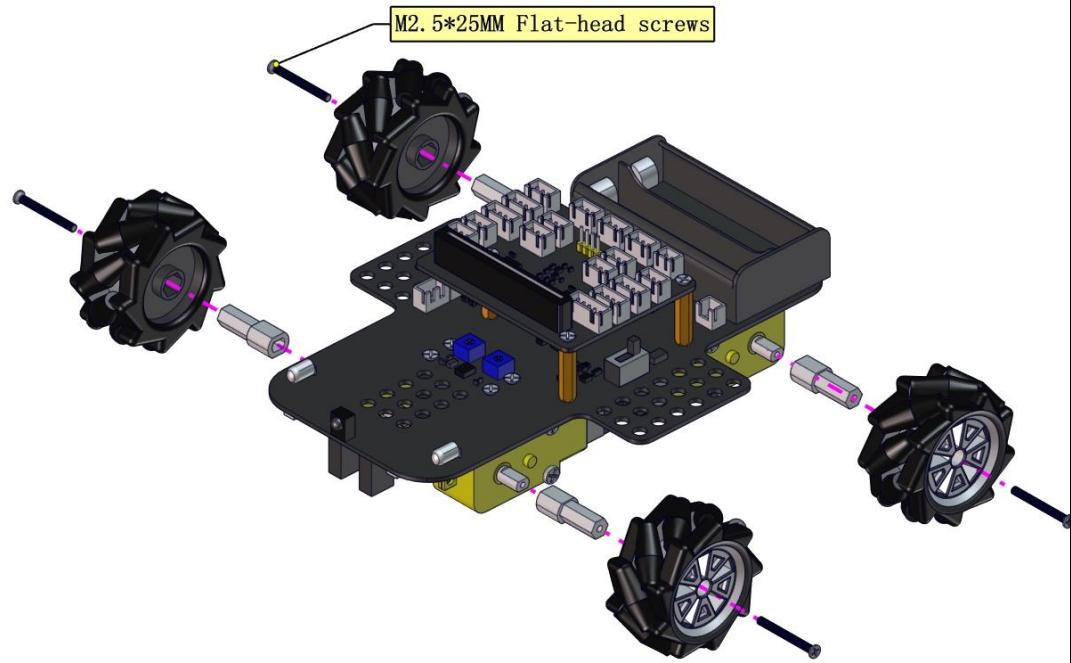
## Part 8



## Components Needed

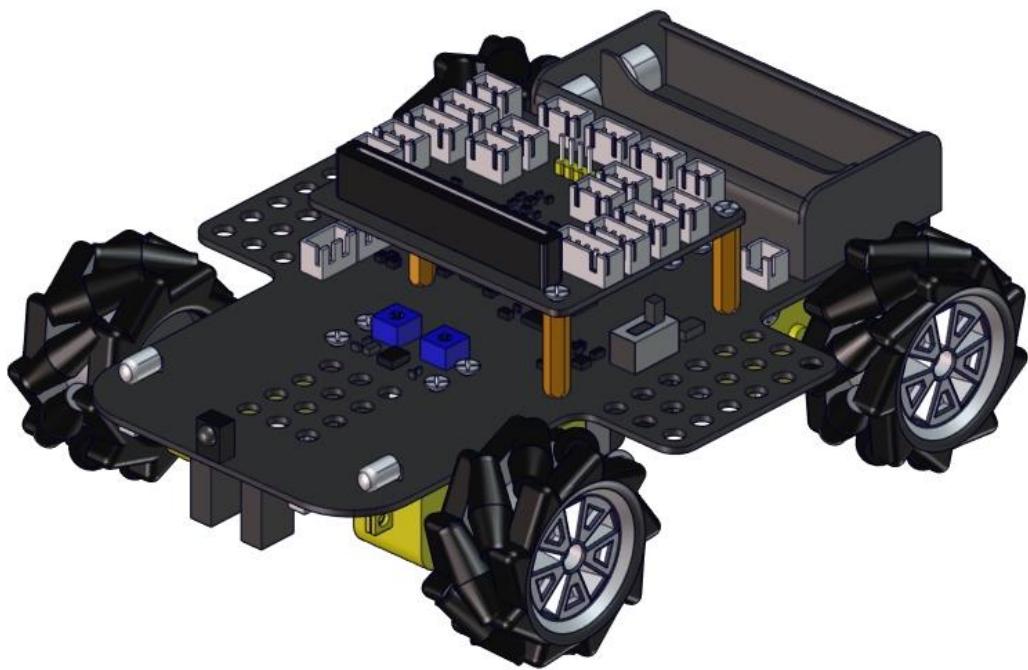


Installation  
Diagram  
(Pay  
attention to  
the  
installation  
direction of  
the  
mecanum  
wheel)



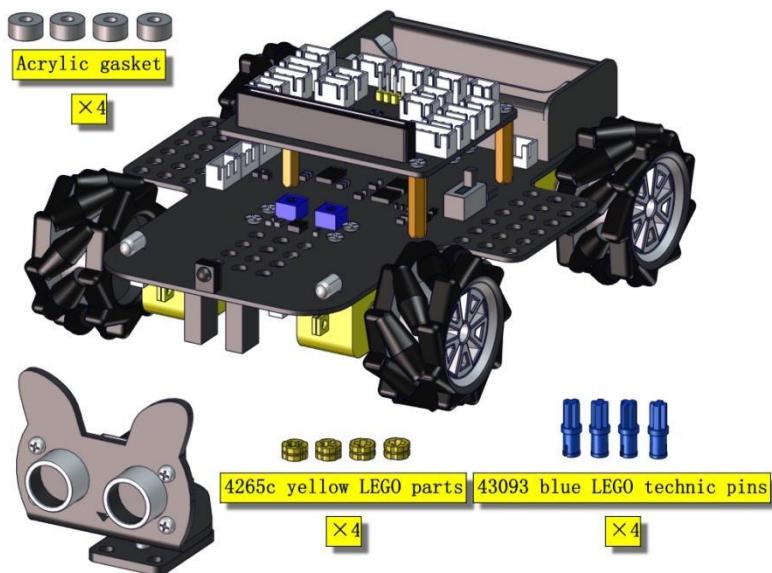


Prototype



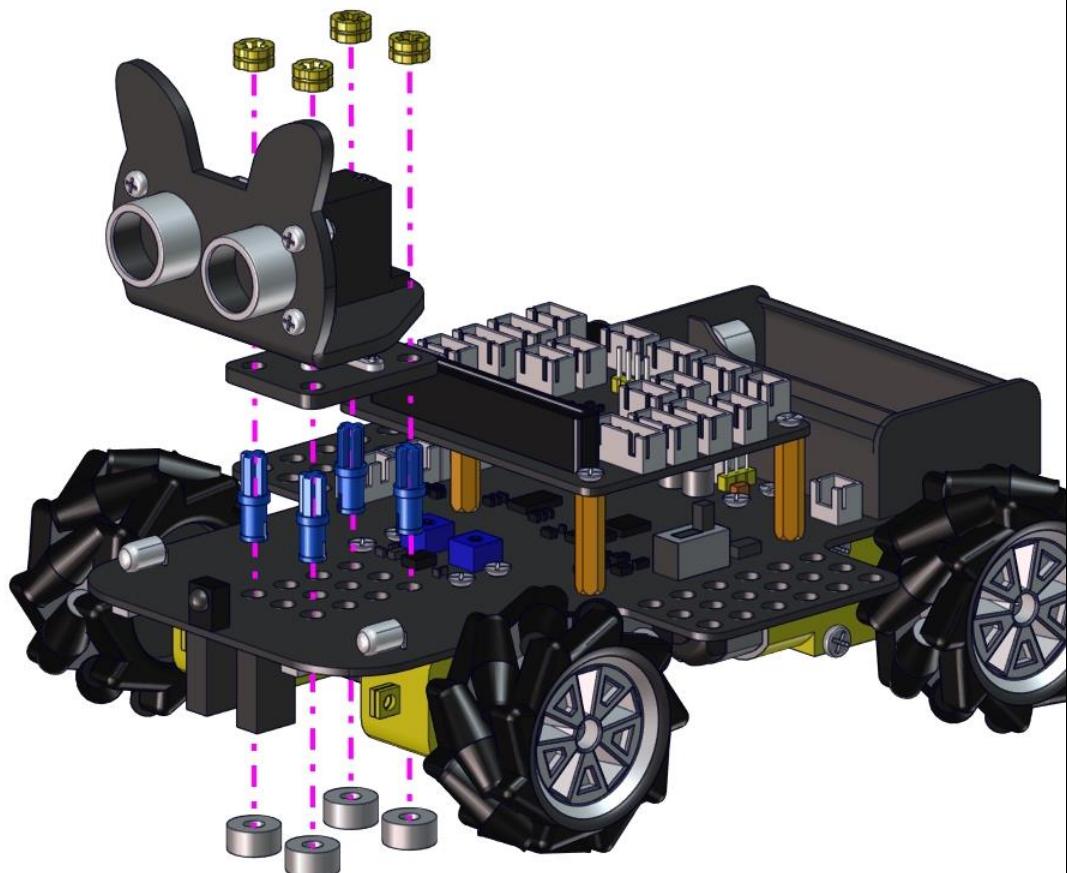
## Part 9

Components  
Needed

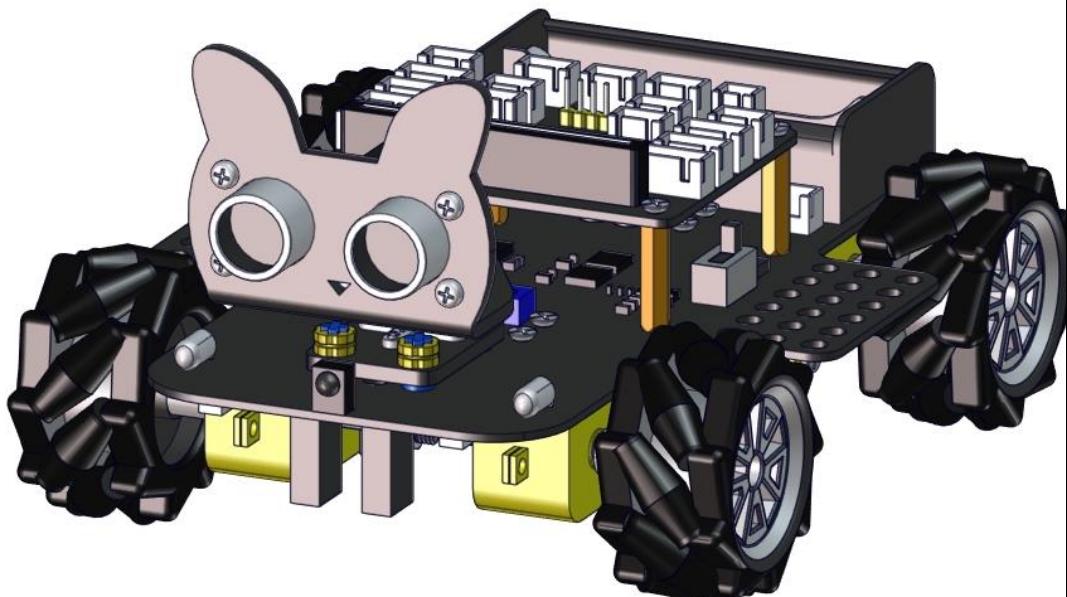




Installation  
Diagram



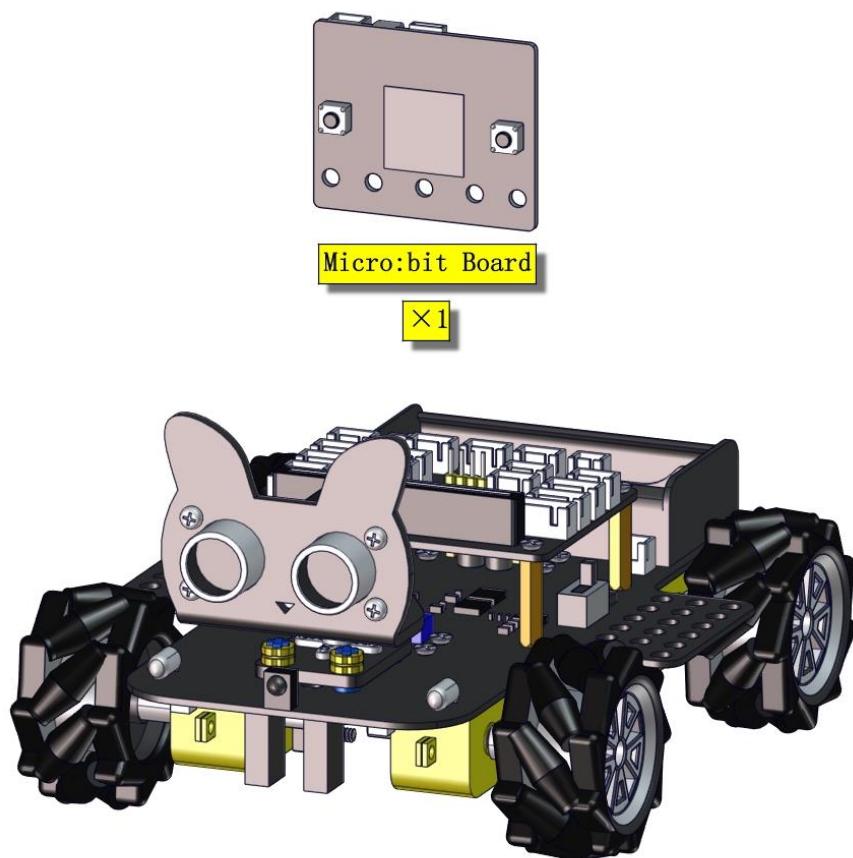
Prototype



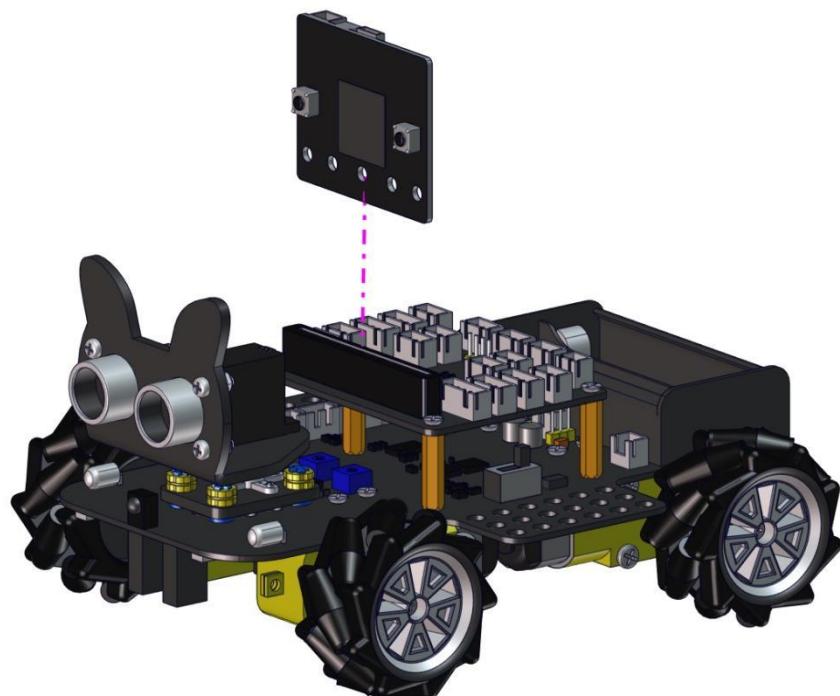
**Part 10**



Components  
Needed

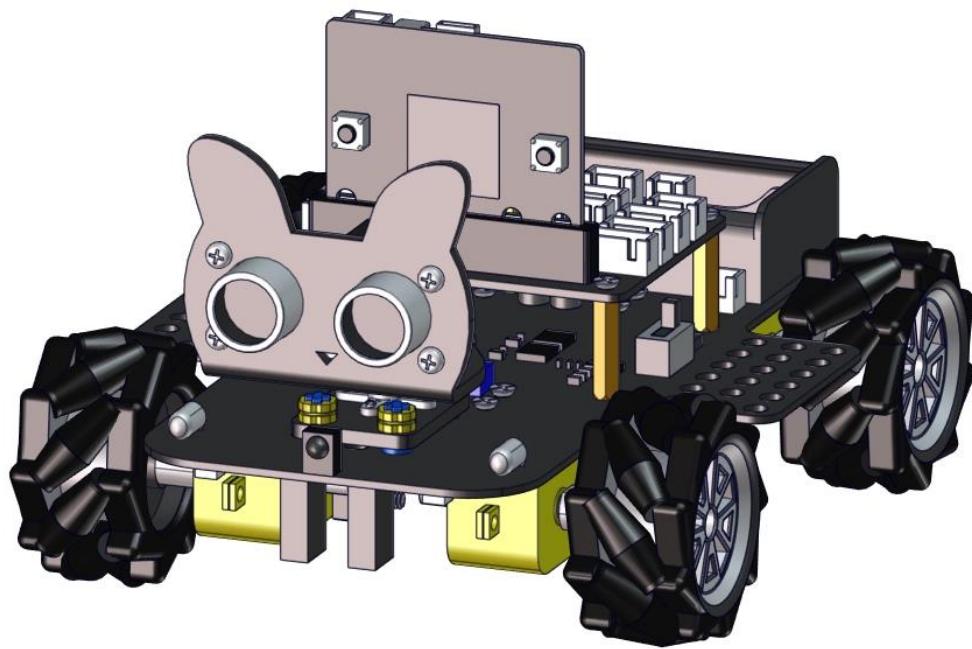


Installation  
Diagram



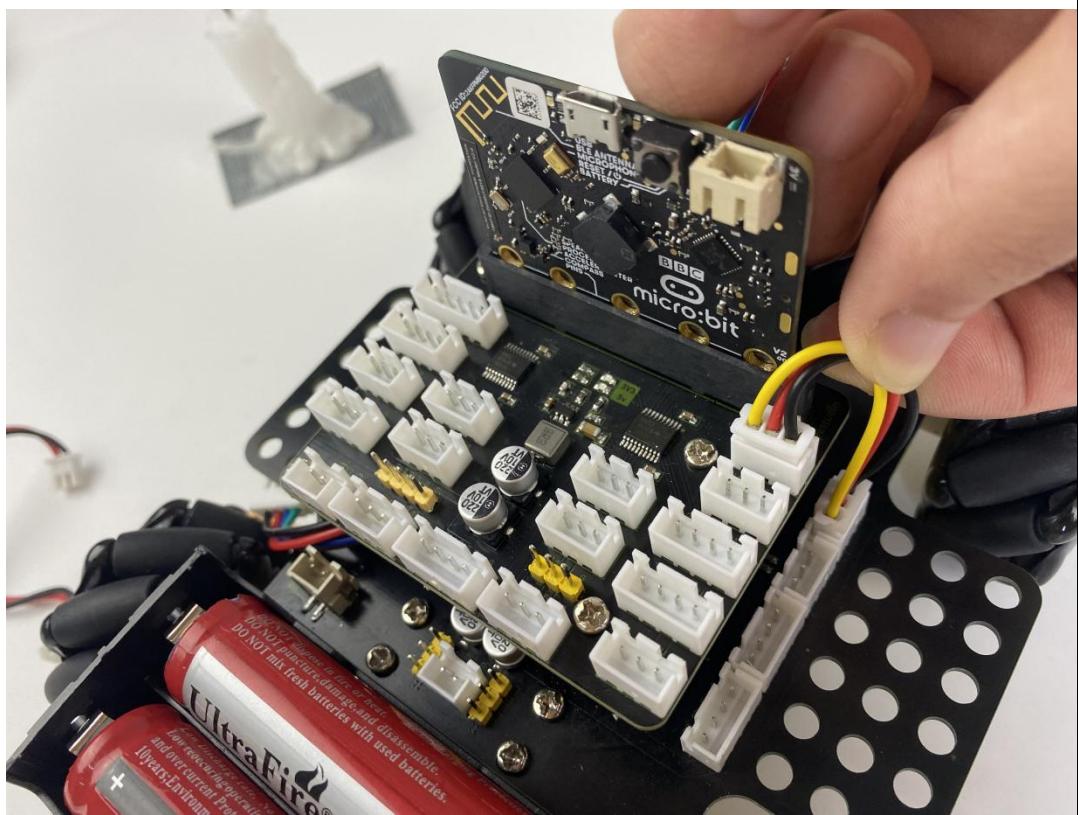


Prototype



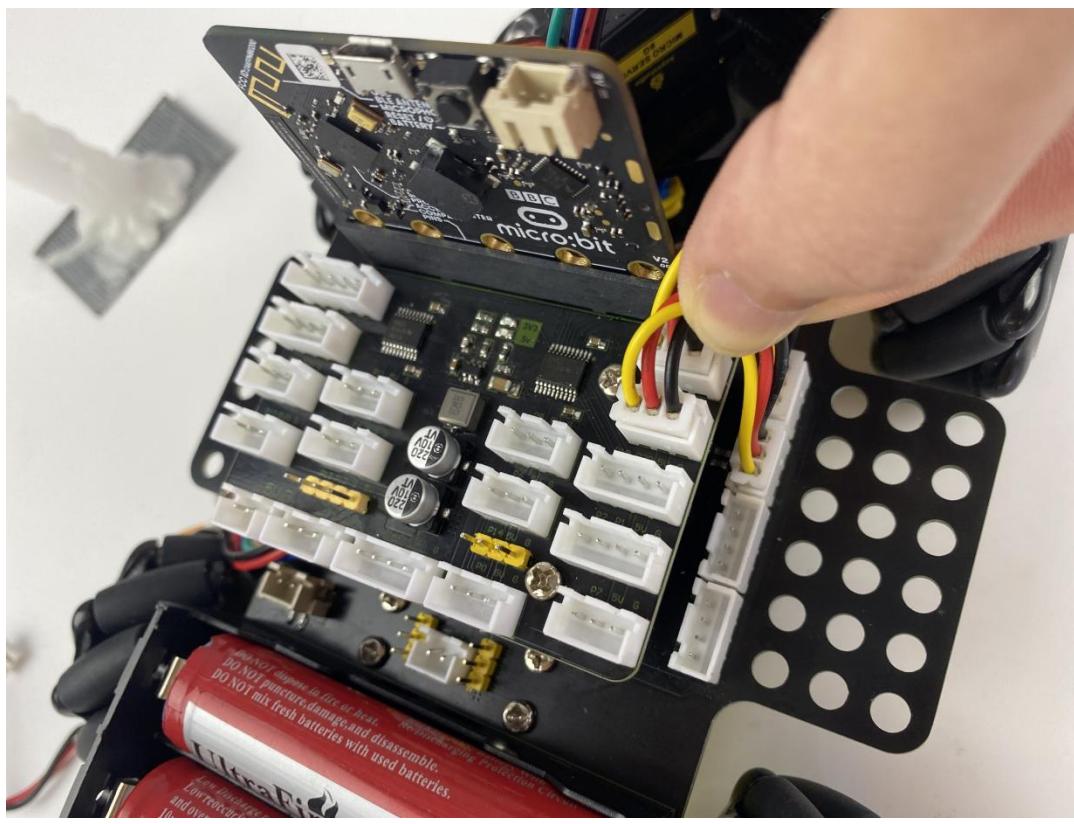
## Start Wiring

The wiring of  
the RGB  
lights

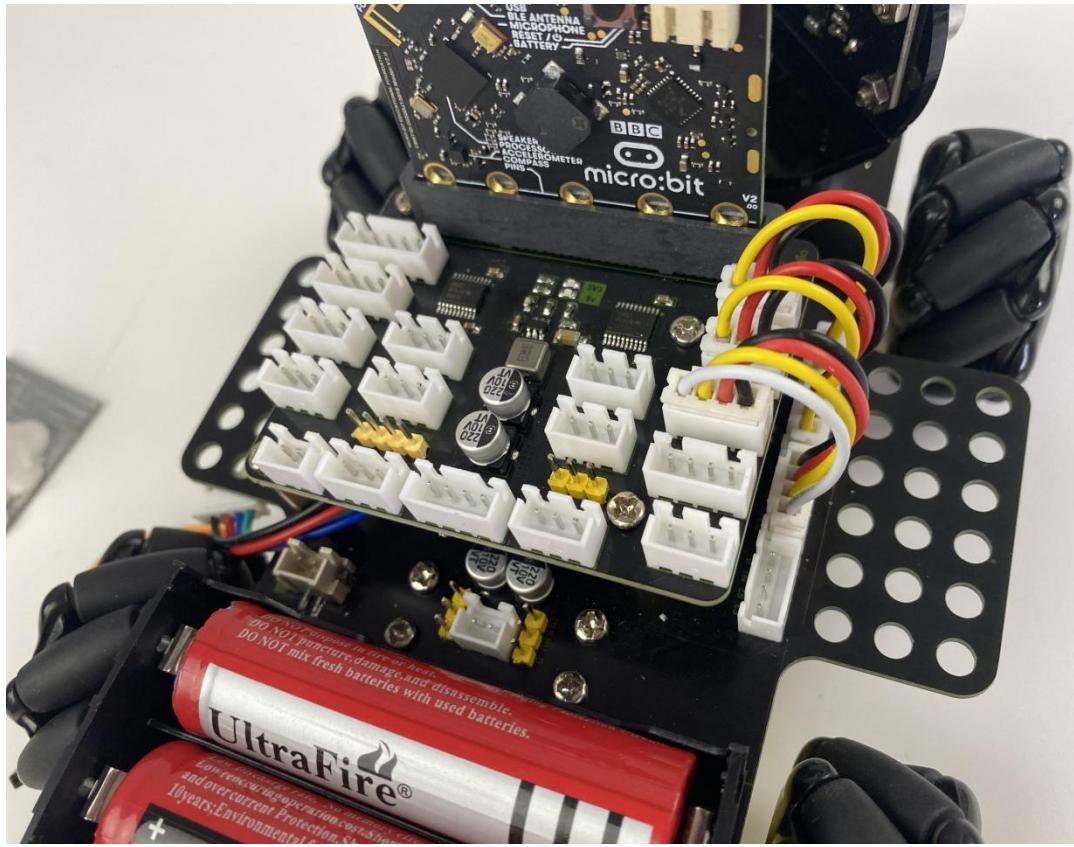




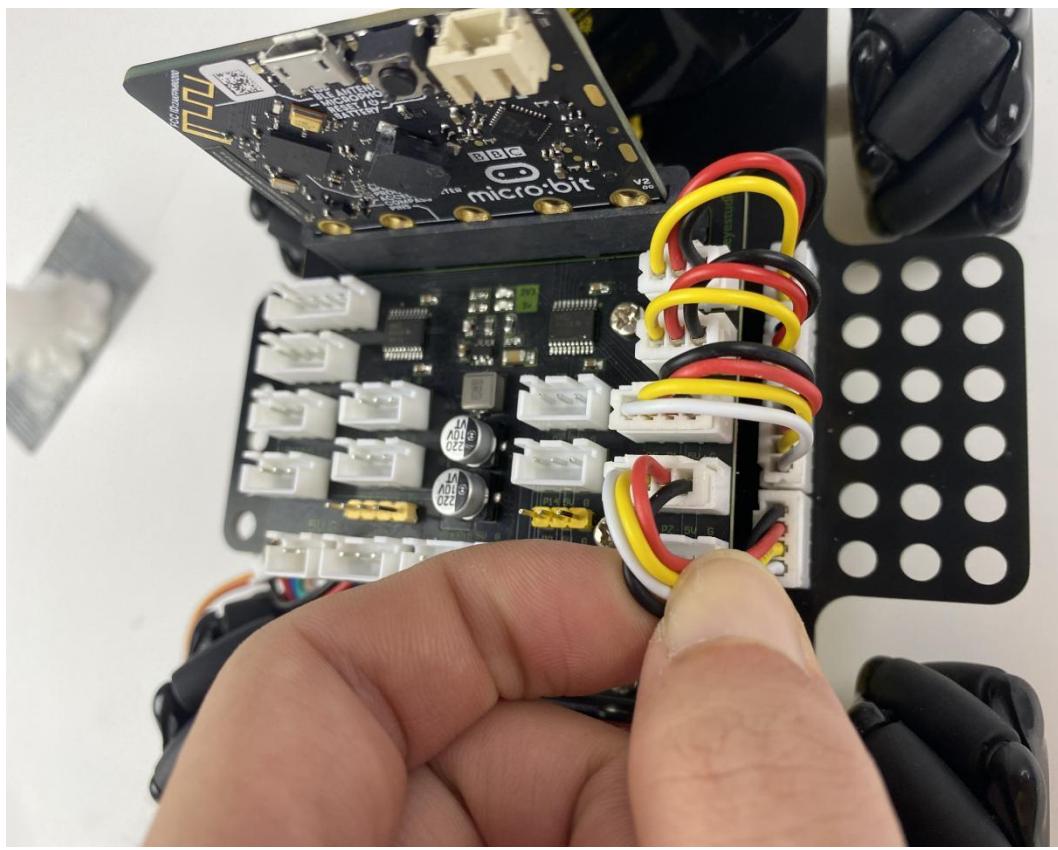
The wiring of  
the infrared  
receiver  
module



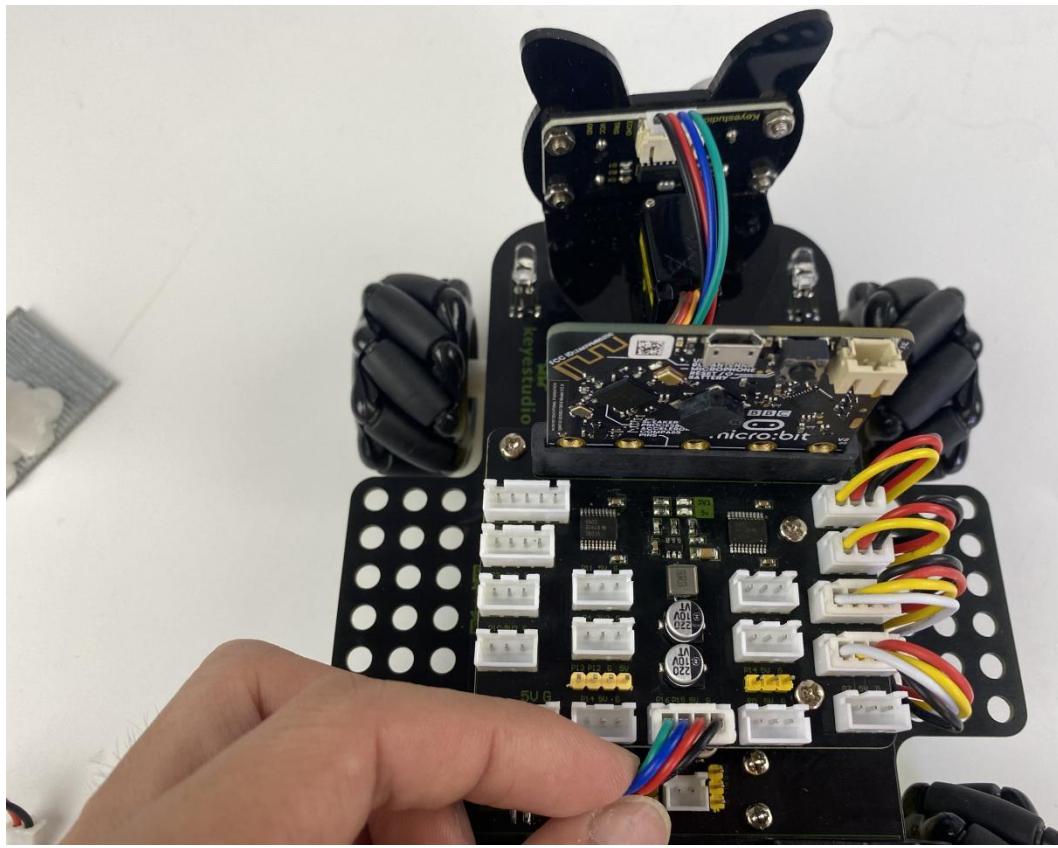
The wiring of  
the motor  
and colorful  
lights



The wiring of  
the  
line-tracking  
sensor

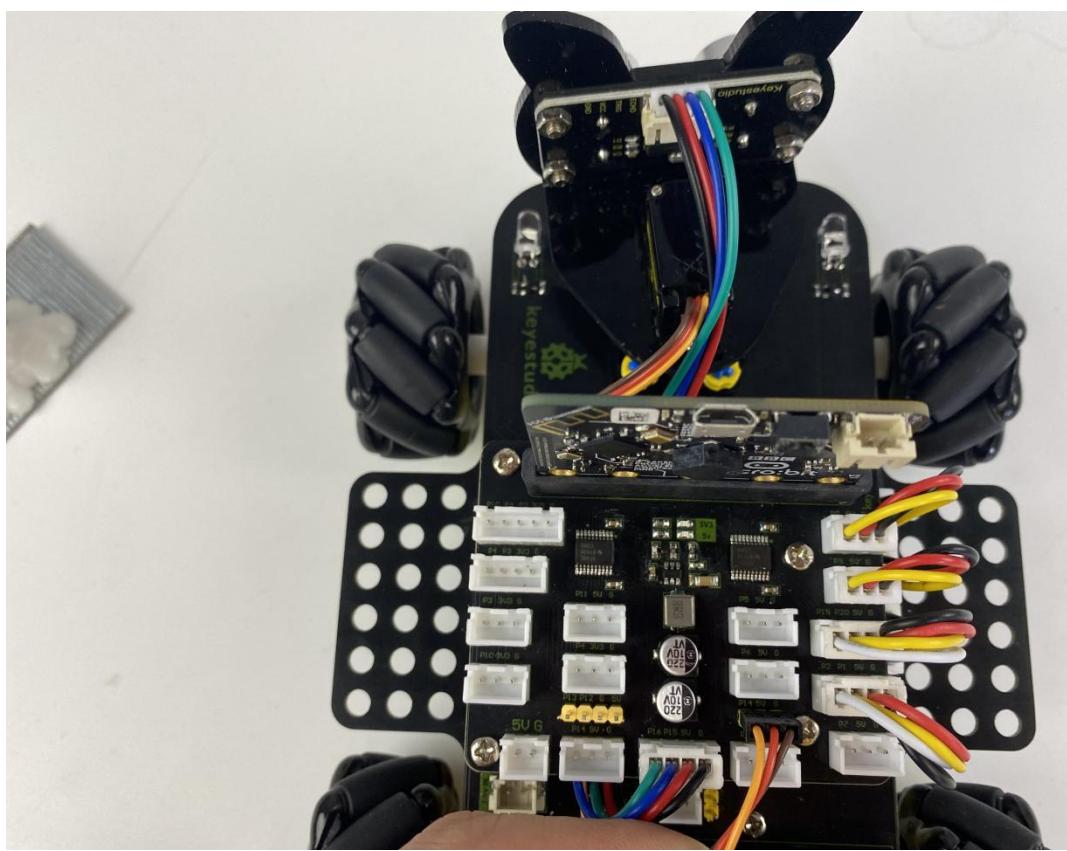


The wiring of  
the  
ultrasonic  
sensor

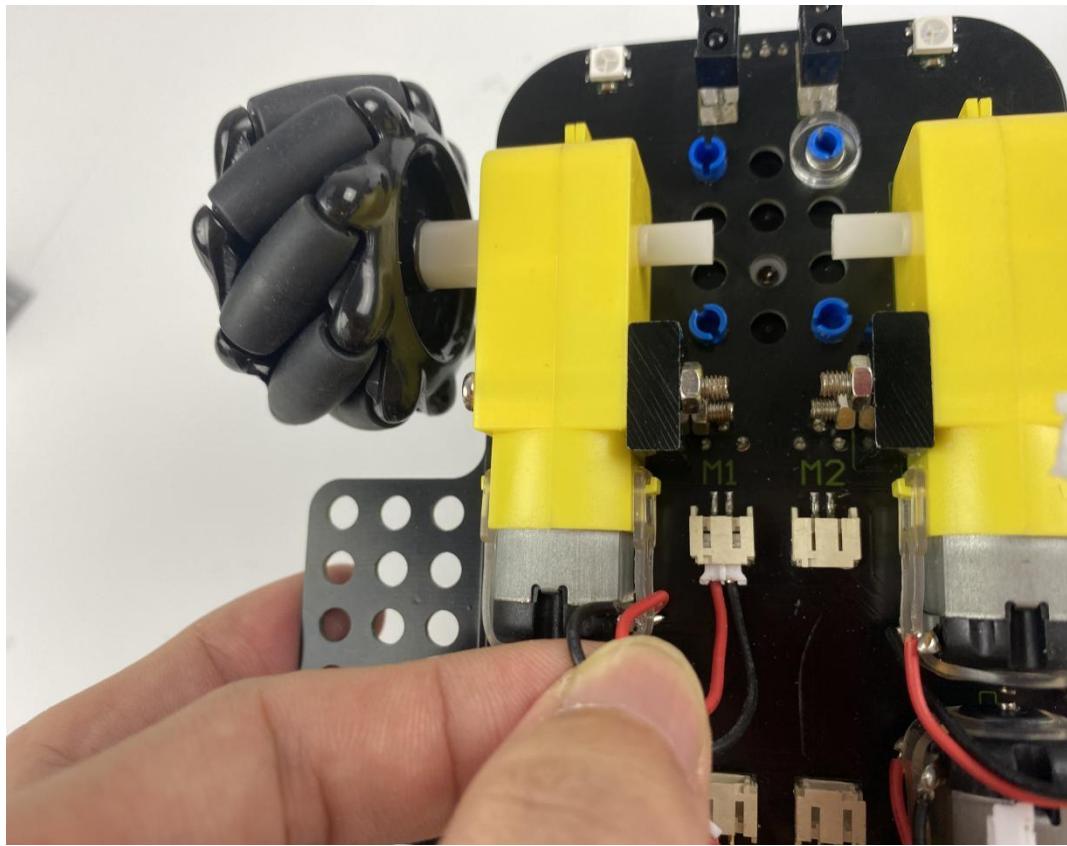




The wiring of  
the servo

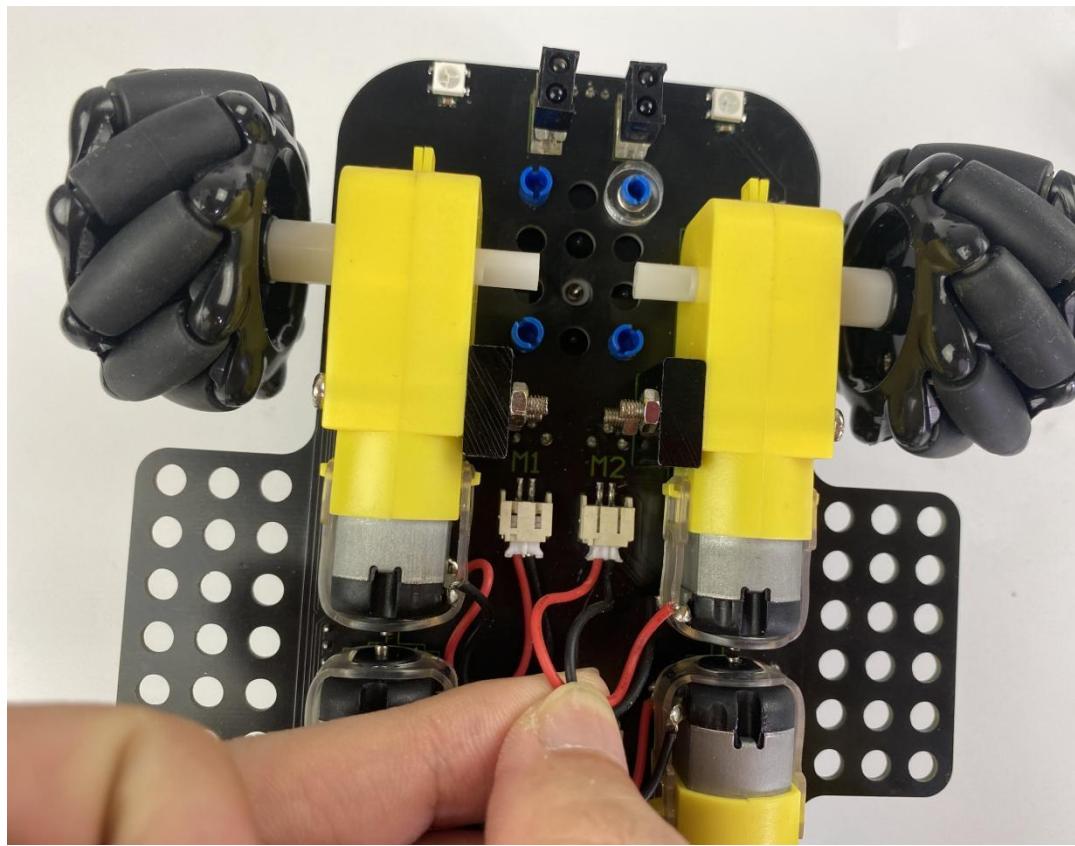


The wiring of  
the  
M1 motor

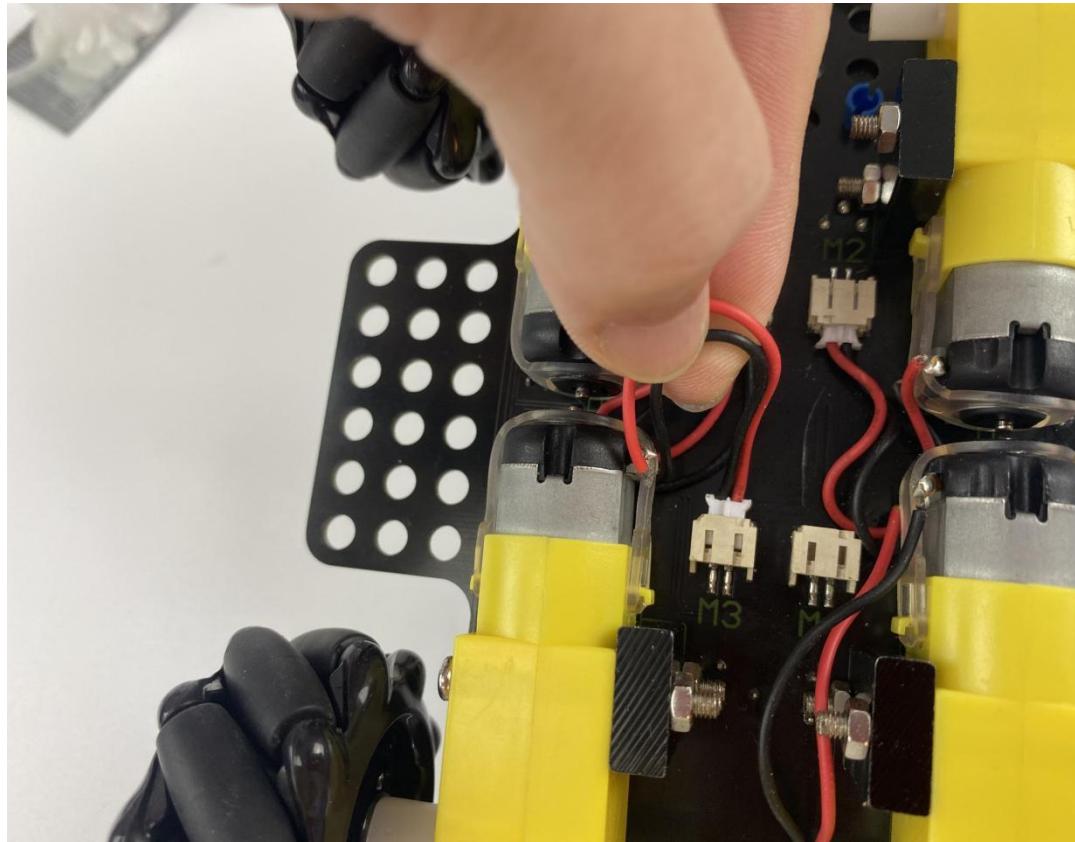




The wiring of  
the  
M2 motor

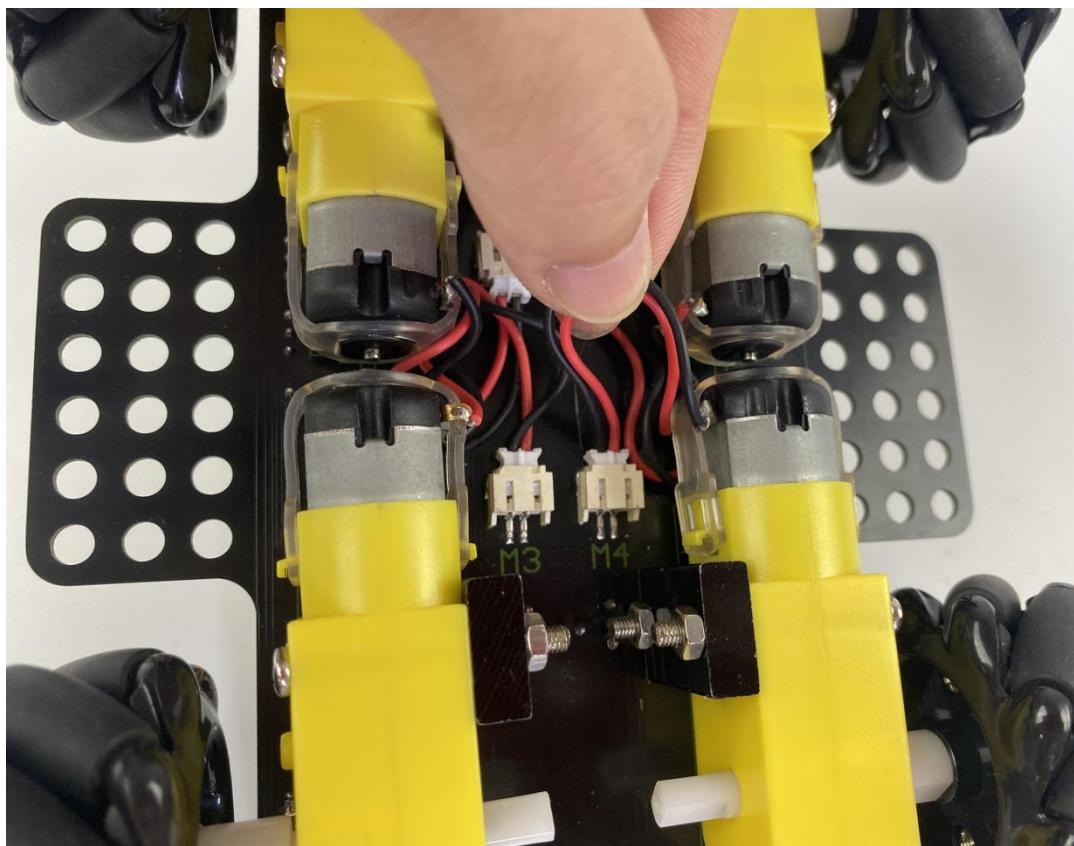


The wiring of  
the  
M3 motor

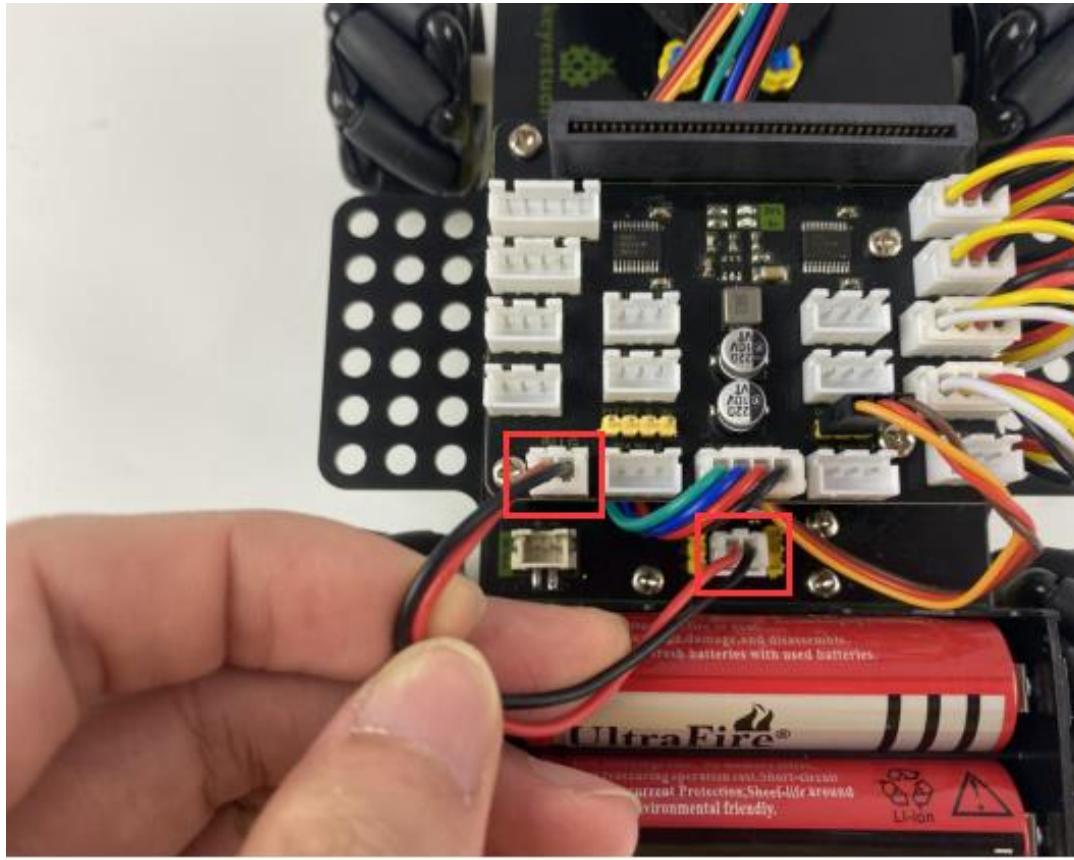




The wiring of  
the  
M4 motor



The wiring of  
the power  
supply (the  
5V is  
connected to  
the shield)





---

## 7.Get Started with Micro:bit

The following instructions are applied for Windows system but can also serve as a reference if you are using a different system.

### 7.1 Write code and program:

This chapter describes how to write program and load the program to the Micro: Bit main board V2.

You are recommended to browse the official website of Micro:bit for more details, and the link is attached below:

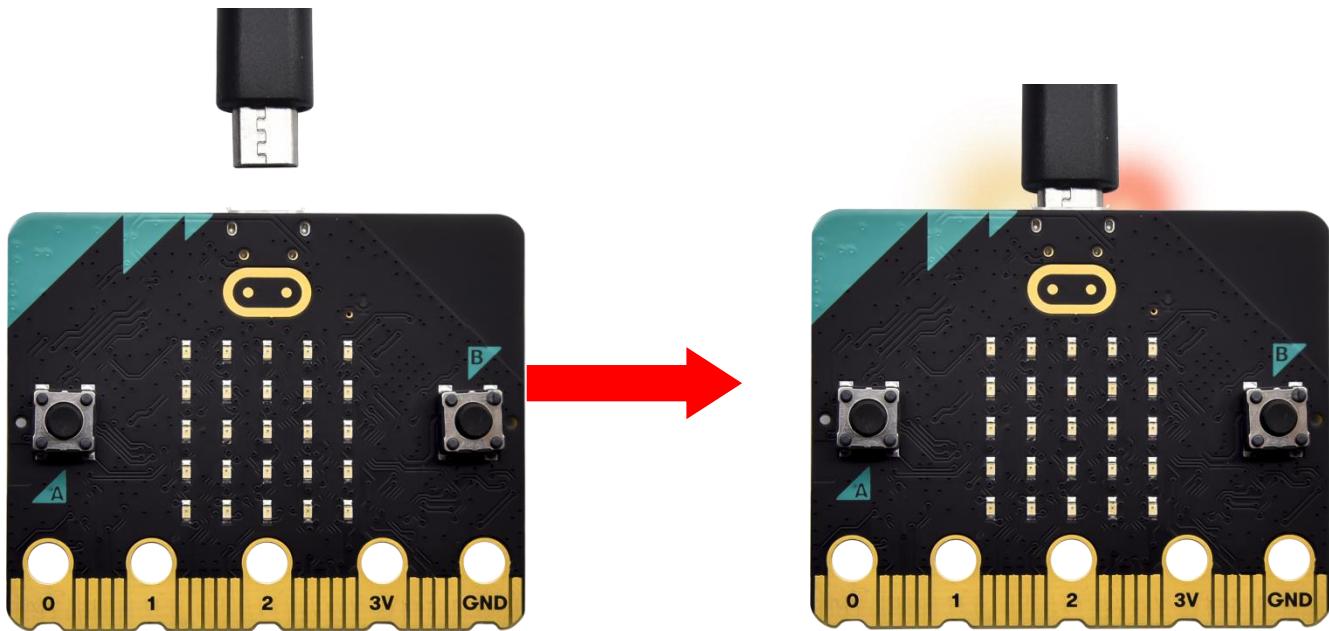
<https://microbit.org/guide/quick/>

#### **Step 1: connect the Micro: Bit main board with your computer**

Firstly, link the Micro: Bit main board with your computer via the USB cable. Macs, PCs, Chromebooks and Linux (including Raspberry Pi) systems are all compatible with the Micro: Bit main board.

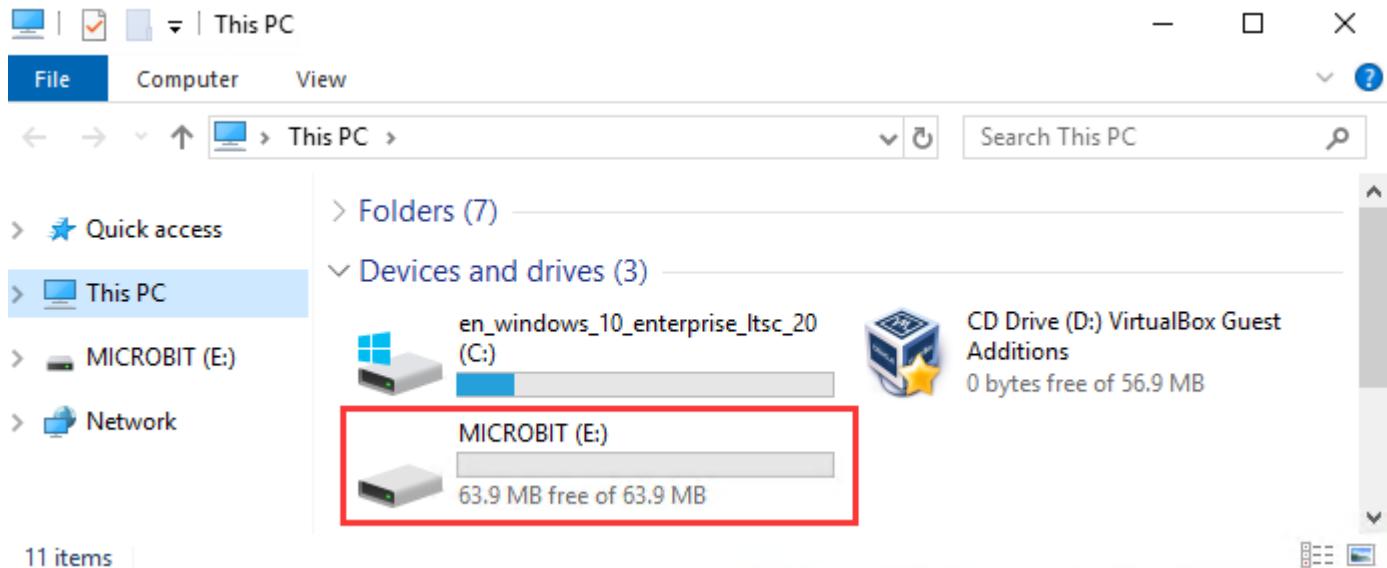
**Note that if you are about to pair the board with your phone or tablet, please refer to this link:**

<https://microbit.org/get-started/user-guide/mobile/>



Secondly, if the red LED on the back of the board is on, that means the board is powered. When your computer communicates with the main board via the USB cable, the yellow LED on it will flash. For example, it will flicker when you burn a “hex” file.

Then Micro: bit main board will appear on your computer as a driver named “MICROBIT(E:)” . Please note that it is not an ordinary USB disk as shown below.



## Step 2: write programs

View the link <https://makecode.microbit.org/> in your browser;

Click 'New Project' ;

The dialog box 'Create a Project' appears, fill it with 'heartbeat' and click 'Create ✓' to edit.

(If you are running Windows 10 system, it is also viable to edit on the APP

MakeCode for micro:bit , which is exactly like editing in the website. And

the link to the APP is

<https://www.microsoft.com/zh-cn/p/makecode-for-micro-bit/9pjc7sv48lcx>

[?ocid=badgep&rtc=1&activetab=pivot:overviewtab \)](https://www.microsoft.com/zh-cn/p/makecode-for-micro-bit/9pjc7sv48lcx?ocid=badgep&rtc=1&activetab=pivot:overviewtab)



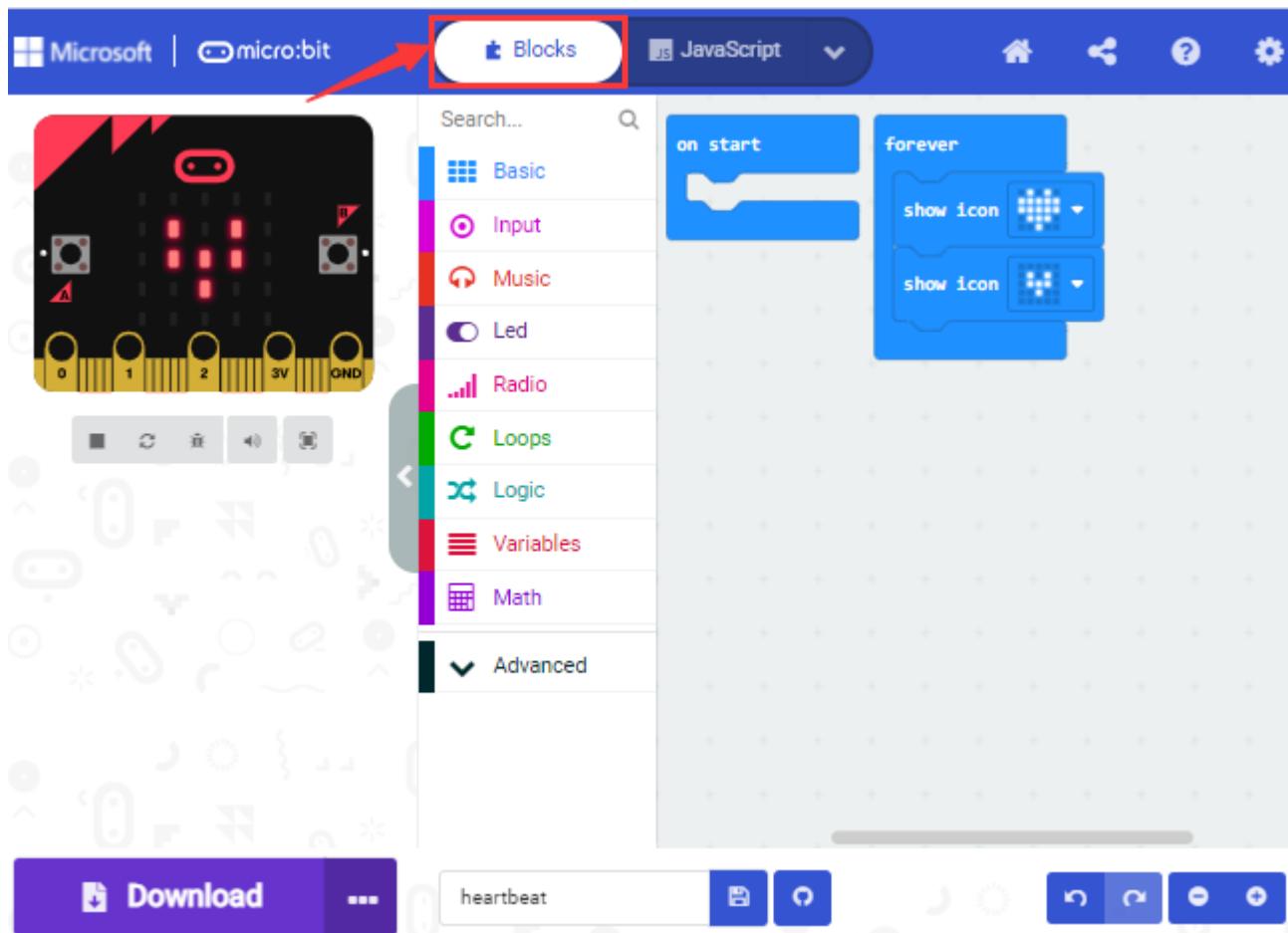
The image shows two screenshots of the Microsoft micro:bit code editor interface.

**Screenshot 1 (Top):** The main screen displays a "New? Start here!" section with a "Start Tutorial" button and a "My Projects" section. A red arrow points from the "New Project" button (which has a red border) to the "Create a Project" dialog in the second screenshot.

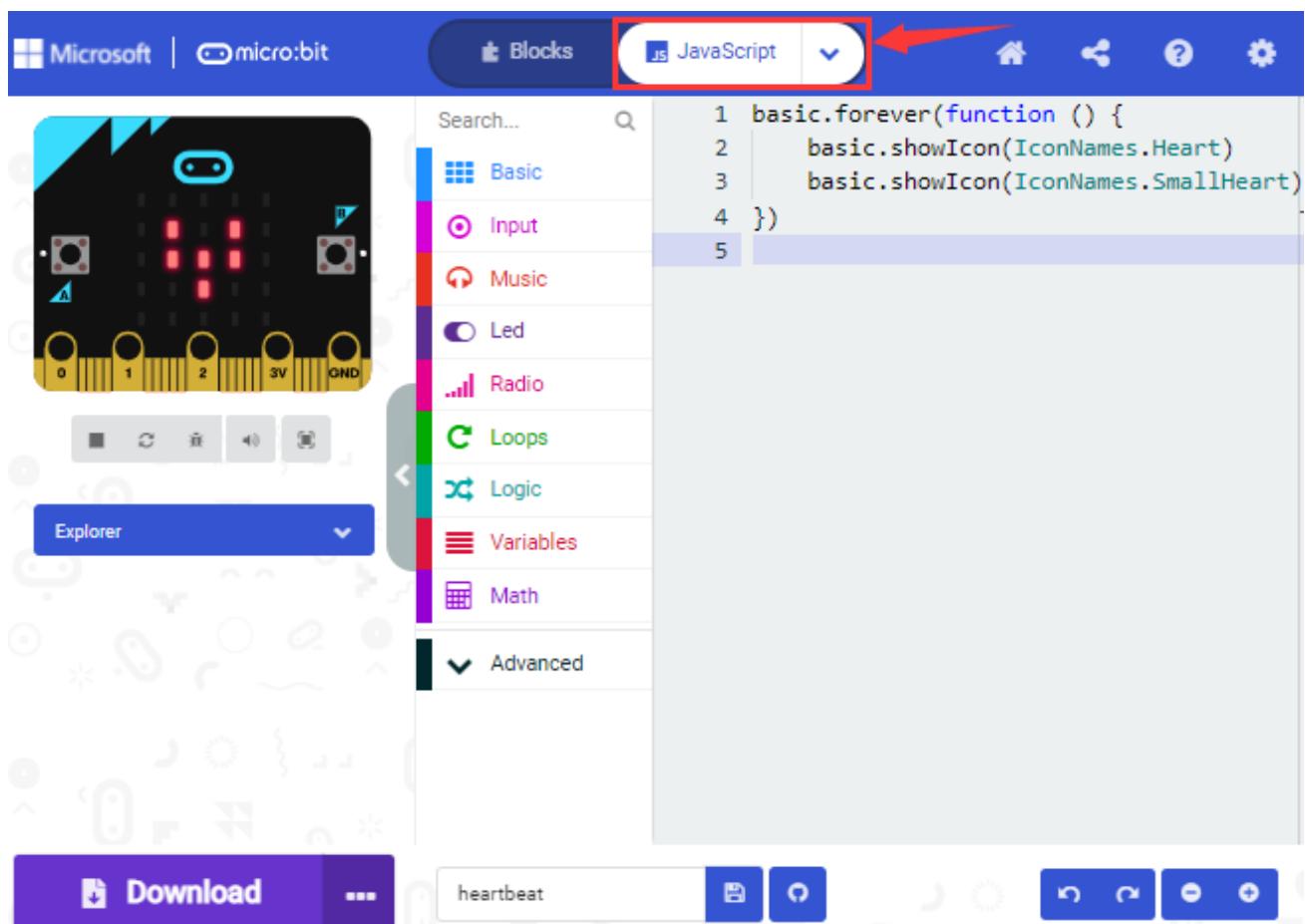
**Screenshot 2 (Bottom):** The "Create a Project" dialog is open. It prompts the user to "Give your project a name." with a text input field containing the text "heartbeat". A red arrow labeled "1" points to this input field. A red arrow labeled "2" points to the green "Create" button at the bottom right of the dialog, which also has a red border.



Write a set of micro:bit code. You can drag some modules in the Blocks to the editing area and then run your program in Simulator of MakeCode editor as shown in the picture below which demonstrates how to edit ‘heartbeat’ program .



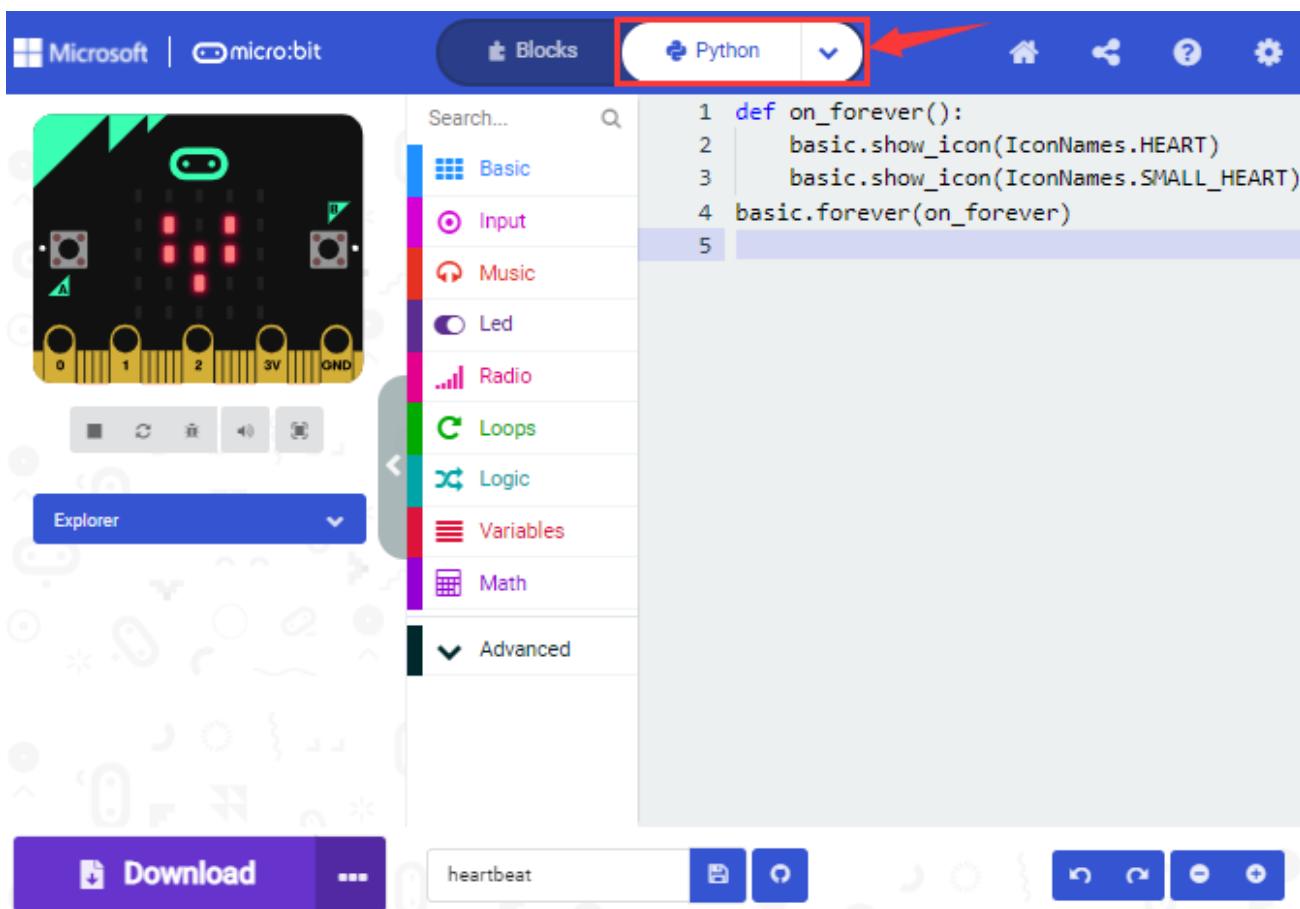
Click the arrow behind “JS JavaScript” to choose between “JavaScript” or “Python” and you will find the corresponding program in JavaScript language or Python language as shown below:



The screenshot shows the Microsoft MakeCode interface for the micro:bit. At the top, there are tabs for "Blocks" and "JavaScript". A red arrow points to the "JavaScript" tab, which is currently selected. Below the tabs is a preview area showing a micro:bit board with red LED patterns. To the right of the preview is a code editor window containing the following JavaScript code:

```
1 basic.forever(function () {
2     basic.showIcon(IconNames.Heart)
3     basic.showIcon(IconNames.SmallHeart)
4 })
5
```

On the left side of the interface, there is a sidebar with categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, and Advanced. At the bottom, there is a "Download" button and a search bar labeled "heartbeat".



### Step 3: download code

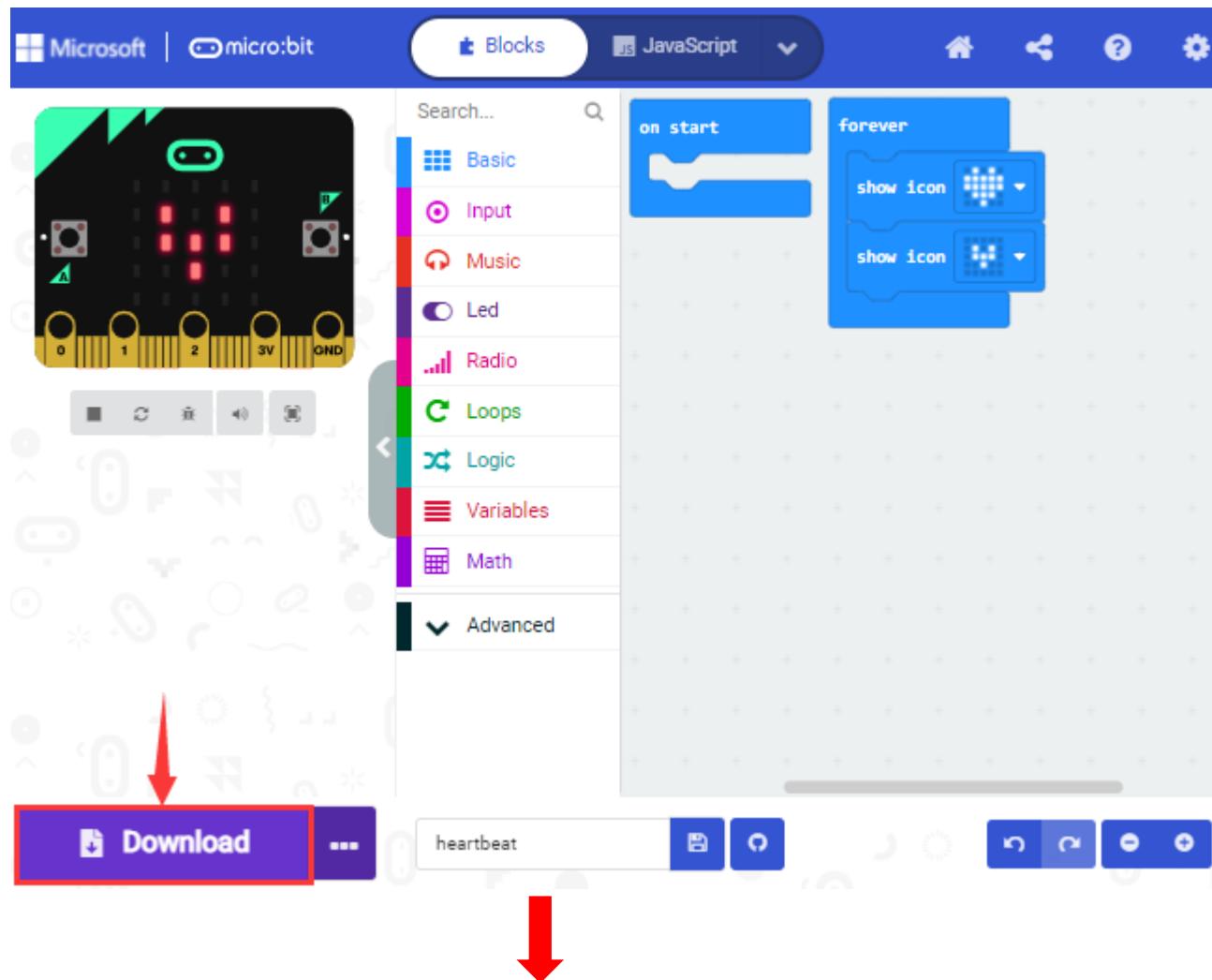
If your computer is Windows 10 and you have downloaded the APP MakeCode for micro:bit to write program, what you will have to do to download the program to your Micro: Bit main board is merely clicking the 'Download' button, then all is done.

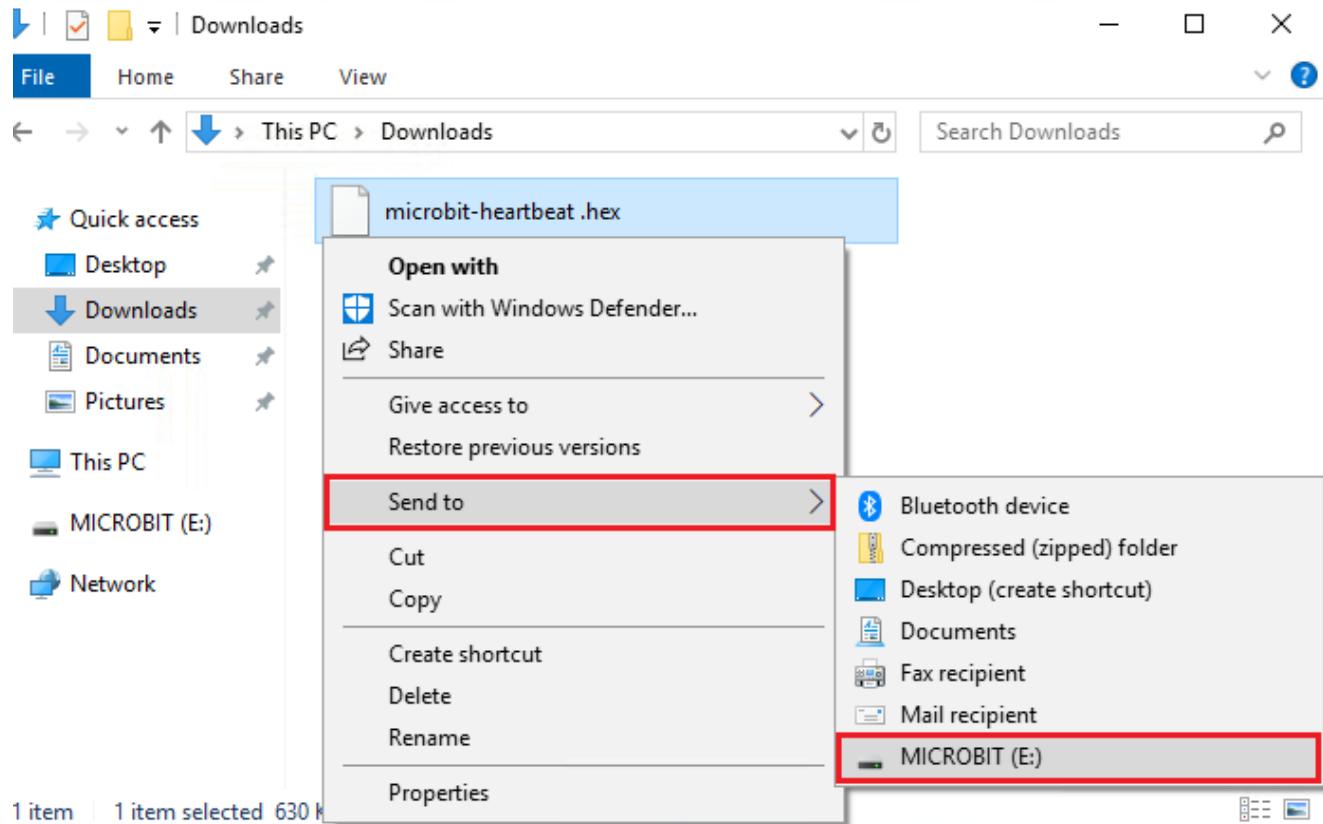
If you are writing program through the website, following these steps:

Click the 'Download' in the editor to download a "hex" file, which is a compact program format that the Micro: Bit main board can read. Once the hexadecimal file is downloaded, copy it to your board just like the

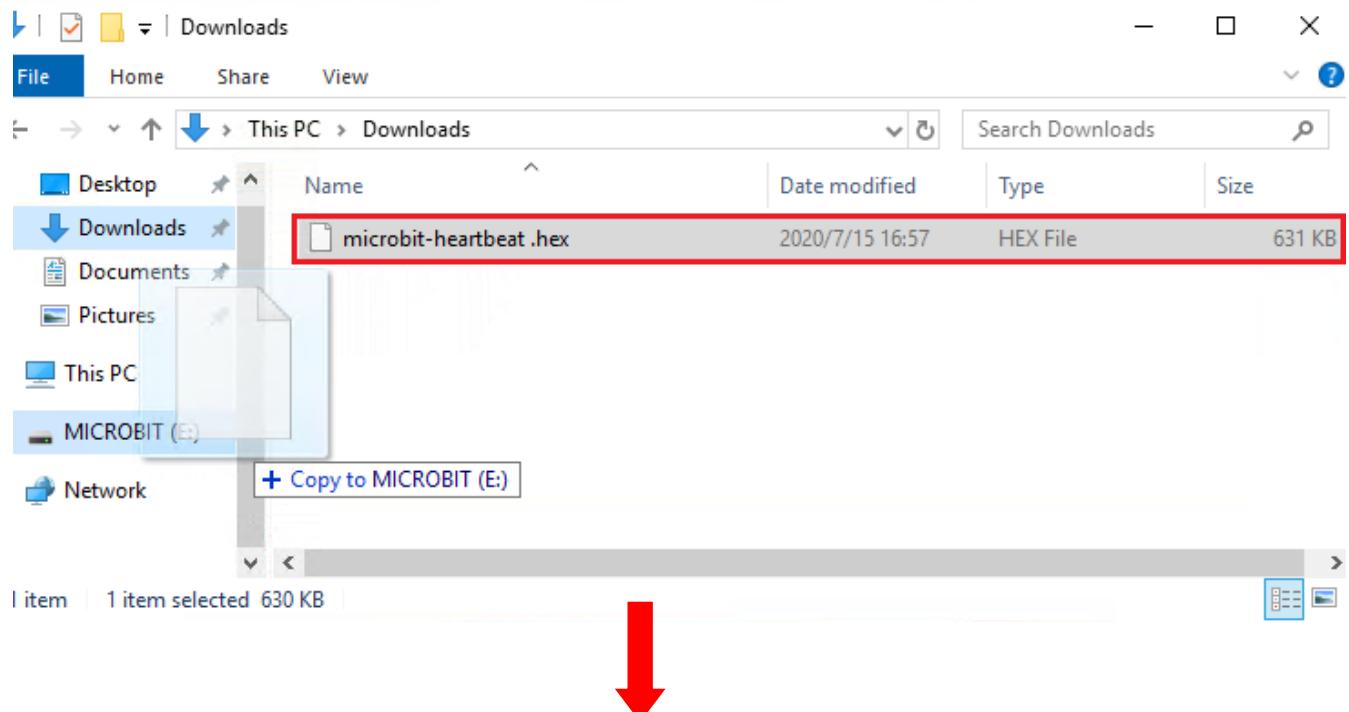


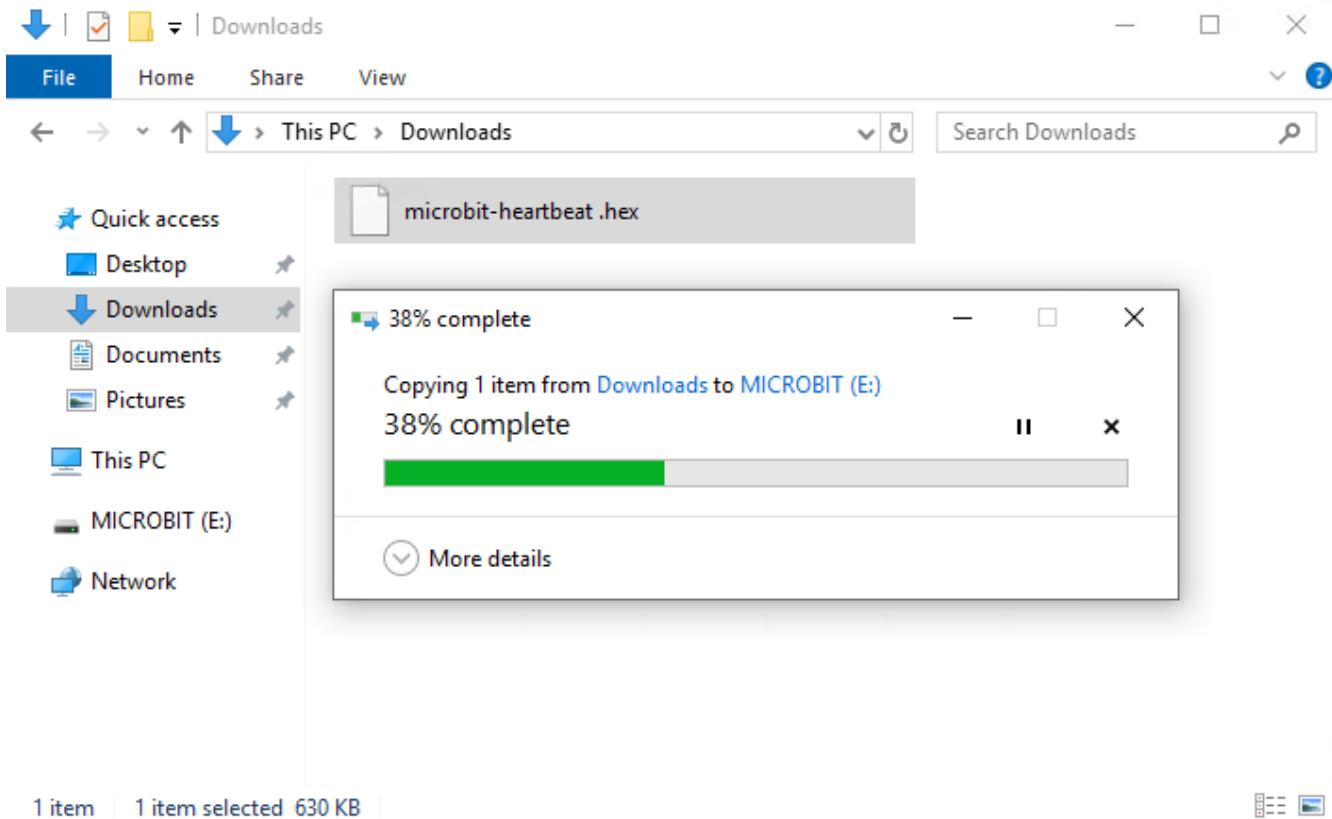
process that you copy the file to the USB driver. If you are running Windows system, you can also right-click and select 'Send to → MICROBIT(E):' to copy the hex file to the Micro: Bit main board.





You can also directly drag the "hex" file onto the MICROBIT (E:) disk.





During the process of copying the downloaded hex file to the Micro: bit main board, the yellow signal light on the back side of the board flashes. When the copy is completed, the yellow signal light will stop flashing and remain on.

#### **Step 4: run the program:**

After the program is uploaded to the Micro: bit main board, you could still power it via the USB cable or change to via an external power. The 5 x 5 LED



dot matrix on the board displays the heartbeat pattern.

	
Power via USB cable	Power via external power (3V)

### **Caution:**

When you programs each time, the driver of Micro: bit will automatically eject and return and your hexadecimal files will disappear . And the board can only have access to hexadecimal files (hex) and save no other files.

### **Step 5: about other programming languages**

This chapter has described how to use the Micro:bit main board.

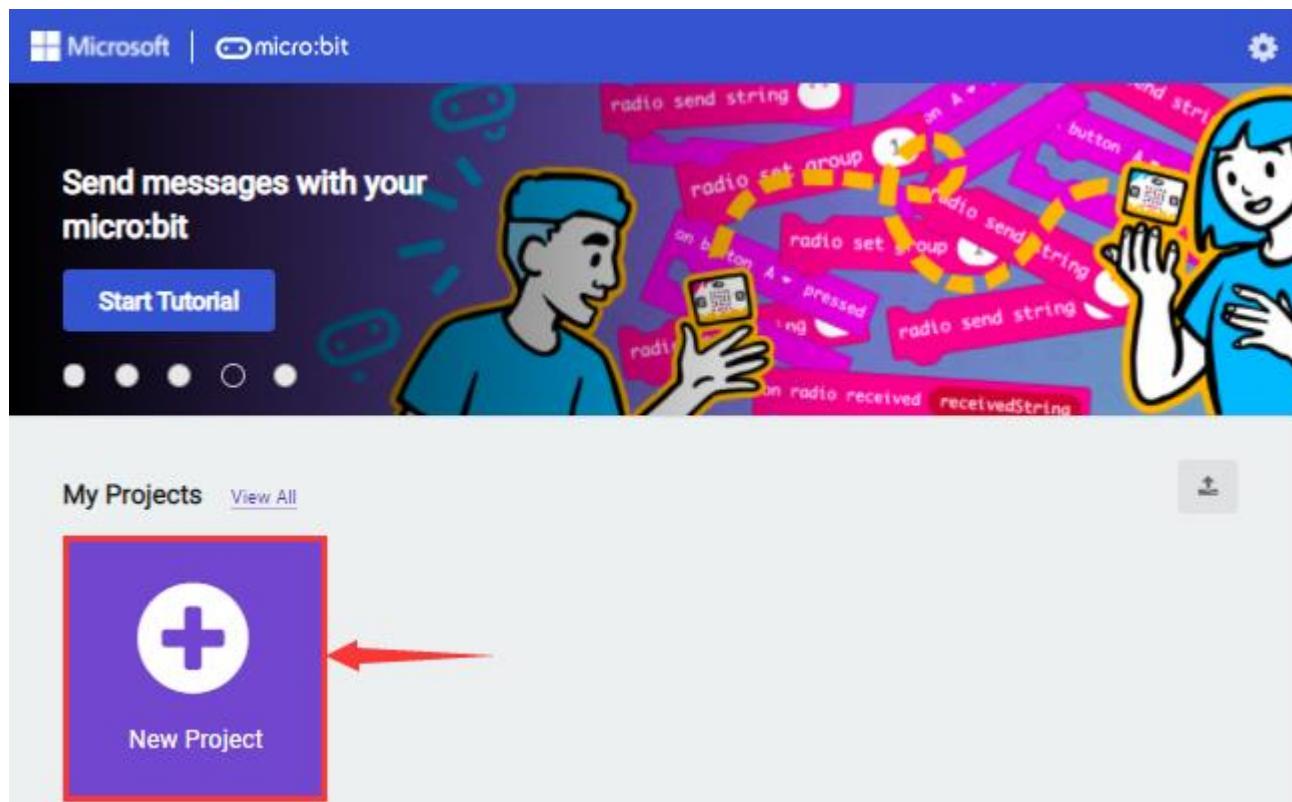
But except for the Makecode graphical programming introduced you can also write Micro:bit programs in other languages. Go to the link: <https://microbit.org/code/> to know about other programming languages , or view the link: <https://microbit.org/projects/>, to find something you want



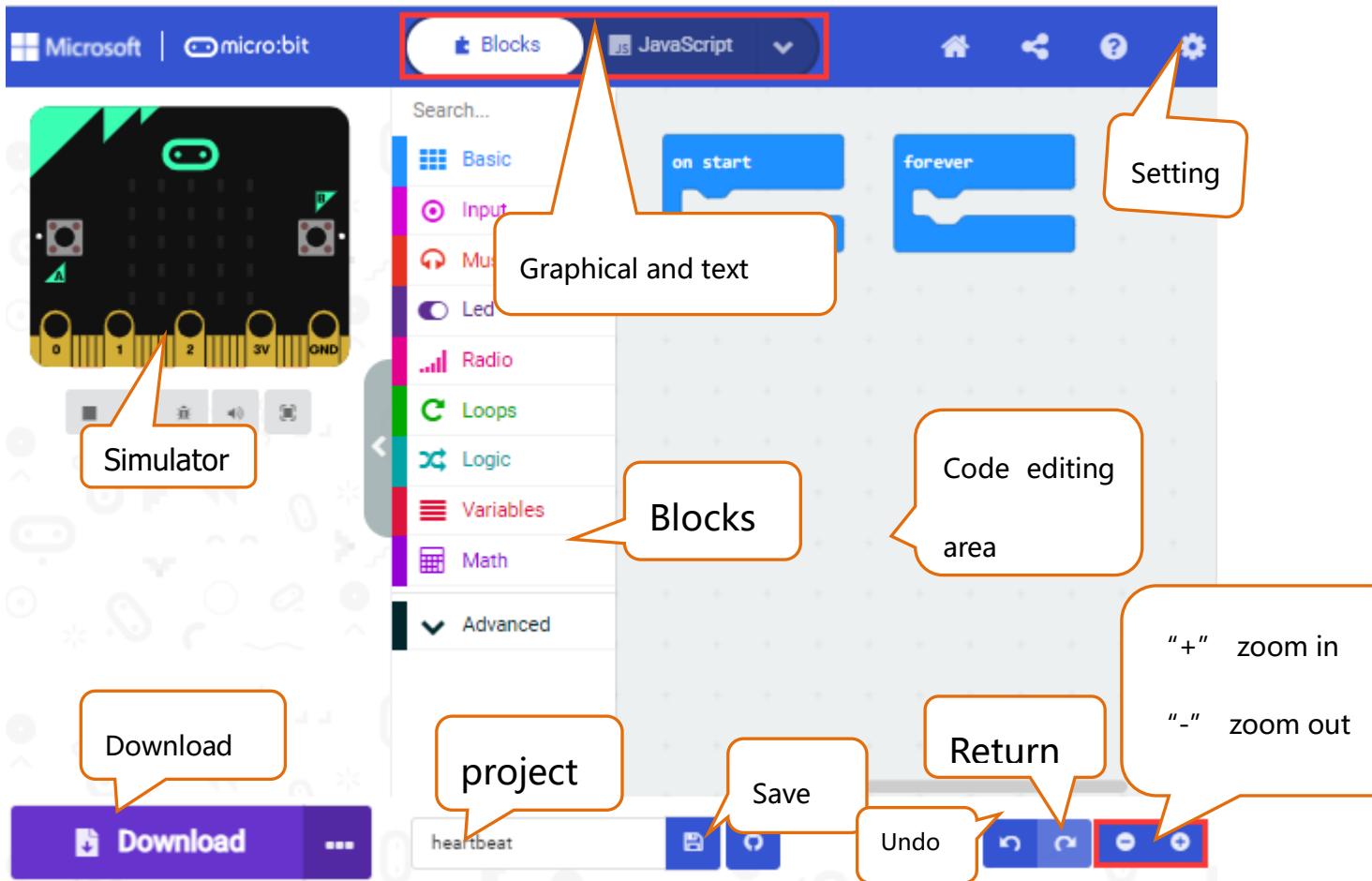
to have a go.

## 7.2. Makecode:

Browse <https://makecode.microbit.org/> and enter Makecode online editor or open the APP MakeCode for micro:bit of Windows 10.



Click “New Project” , and input “heartbeat” , then click “create √” to enter Makecode editor, as shown below:



There are blocks “on start” and “forever” in the code editing area.

When the power is plugged or reset, “on start” means that the code in the block only executes once, while “forever” implies that the code runs cyclically.

### 7.3 Quick Download

As mentioned before, if your computer is Windows 10 and you have downloaded the APP MakeCode for micro:bit to write programs, the



---

program written can be quickly downloaded to the Micro: Bit main board by selecting ‘Download’ .

While it is a little more trickier if you are using a browser to enter Makecode. However, if you use Google Chrome, suitable for Linux, macOS and Windows 10, the process can be quicker too.

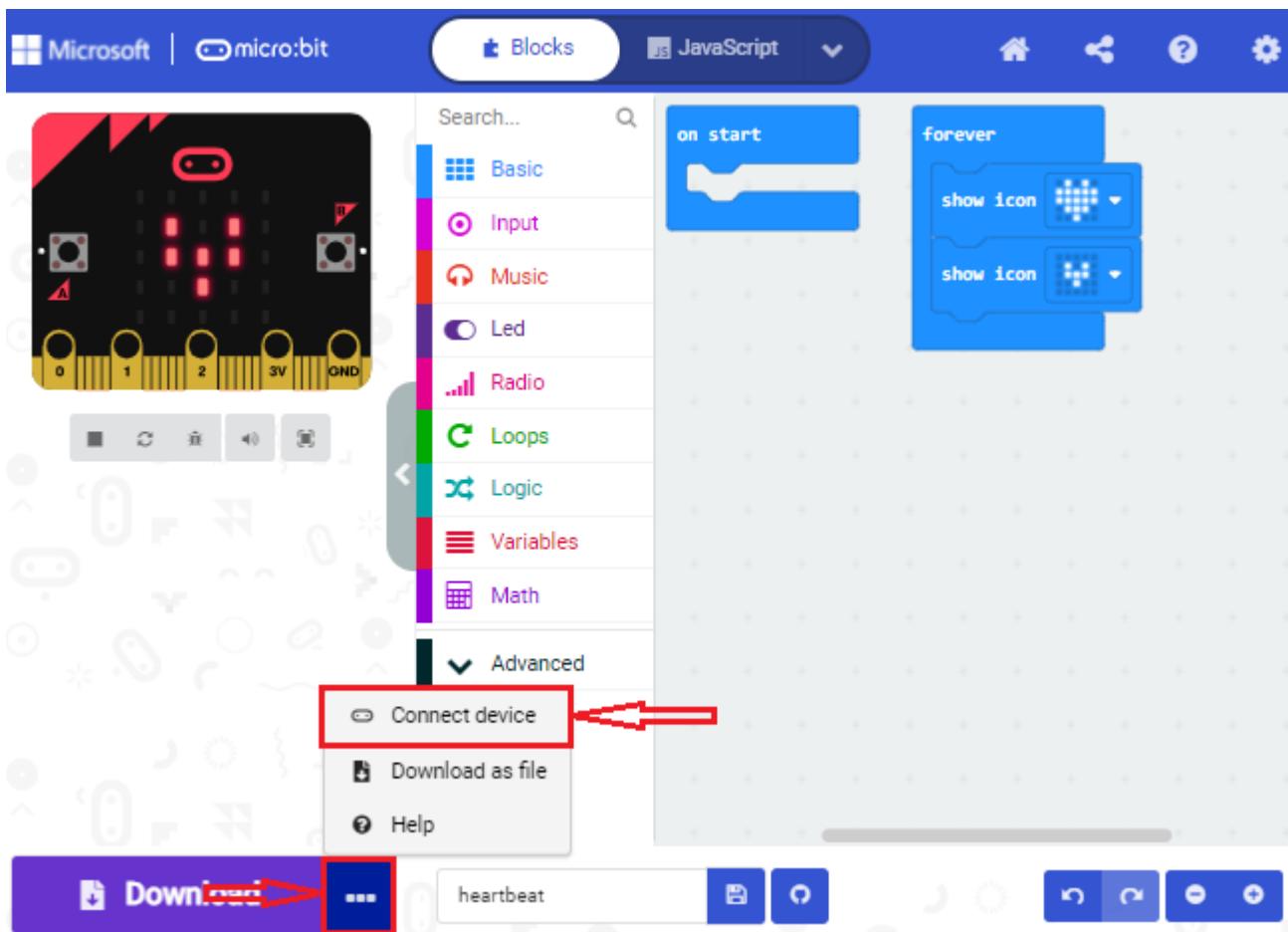
We use the webUSB function of Chrome to allow the internet page to access the hardware device connected USB.

You could refer to the following steps to connect and pair devices.

### **Device pairing:**

Connect micro:bit to your computer by USB cable.

Click “...” beside “Download” and tap “Connect device” ;



Click "Next" ;



Microsoft | micro:bit

Blocks JavaScript

Search... Basic Input

on start forever

Connect your micro:bit...

First, make sure your micro:bit is connected to your computer with a USB cable.

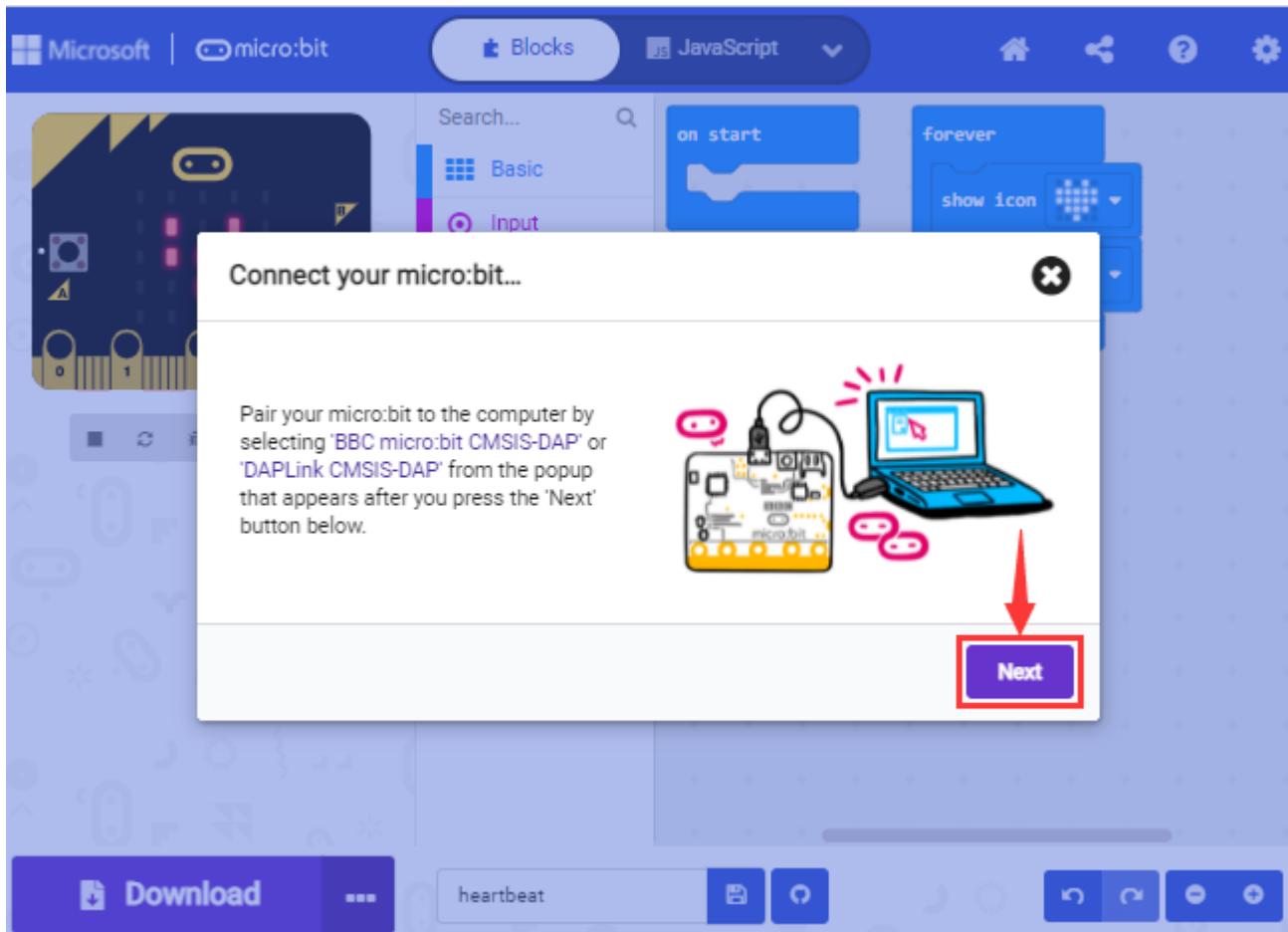
Next

Download ...

heartbeat

5

Click another “Next” ;



Then select the corresponding device and click “Connect” . If no devices shows up for selection, please refer to:

<https://makecode.microbit.org/device/usb/webusb/troubleshoot>

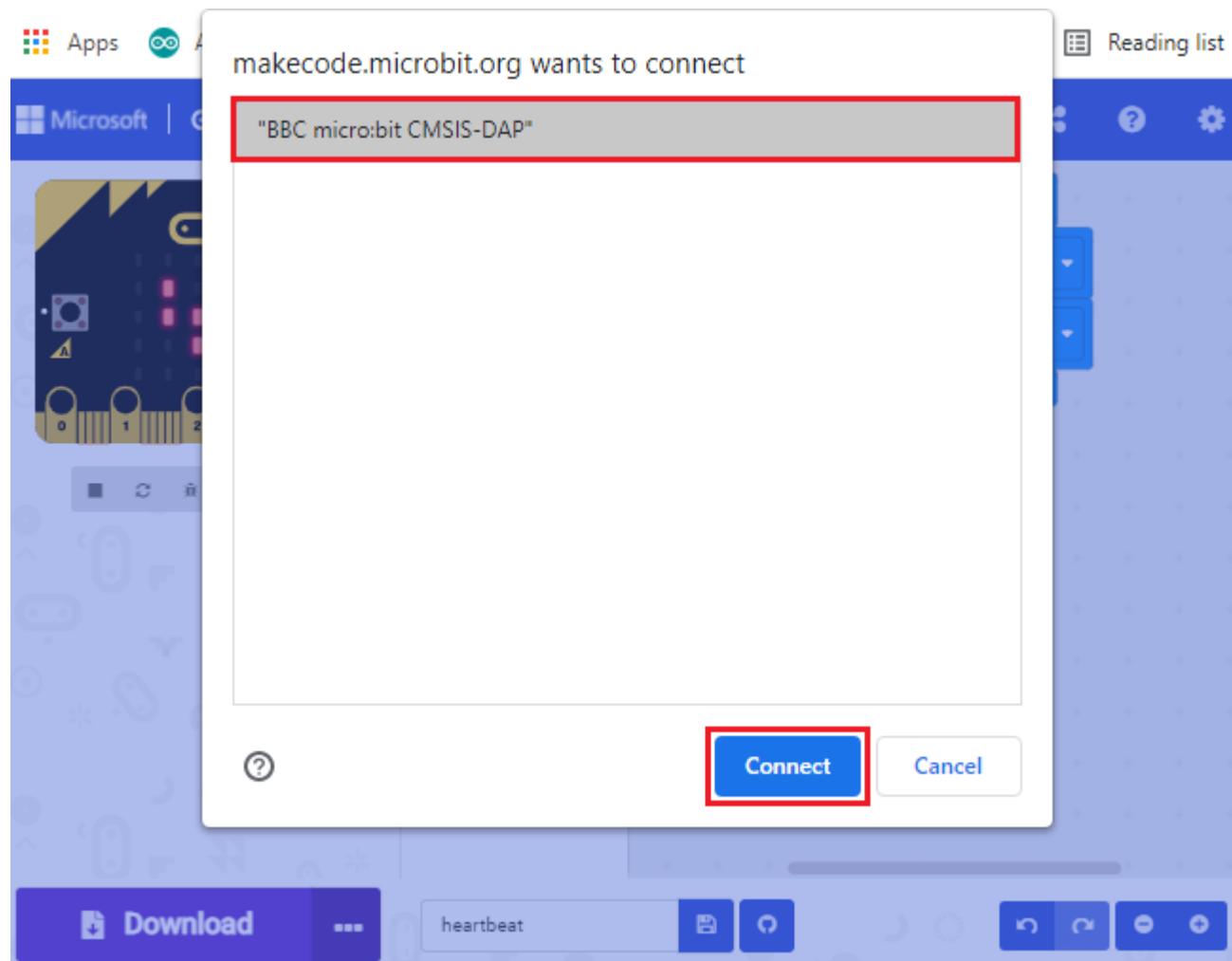
And for updating the firmware of the Micro:bit:  
<https://microbit.org/guide/firmware/> .

If the links are too troublesome for you , then you can also turn to our ‘Troubleshooting Downloads with WebUSB’ and “upload the firmware”

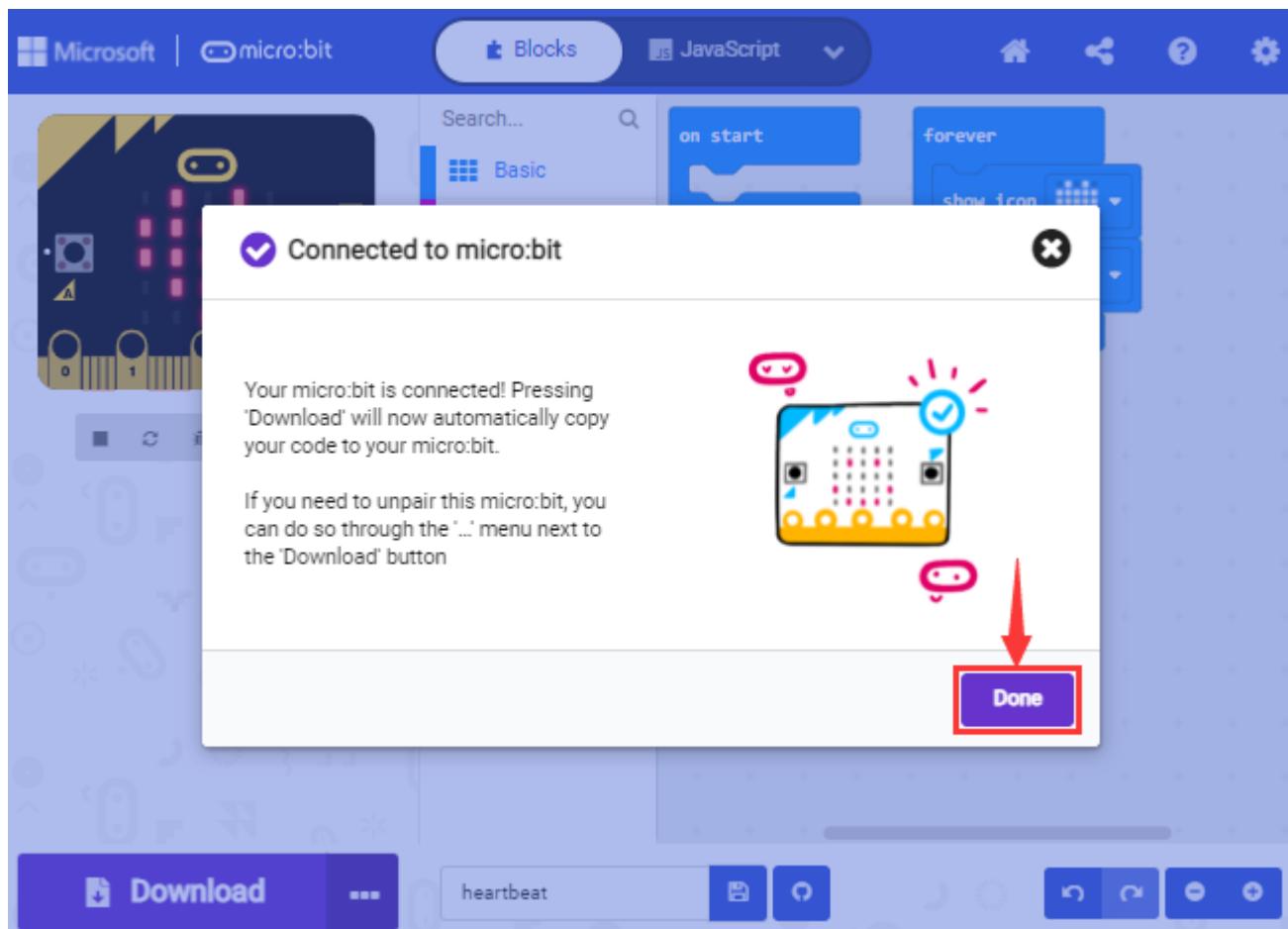


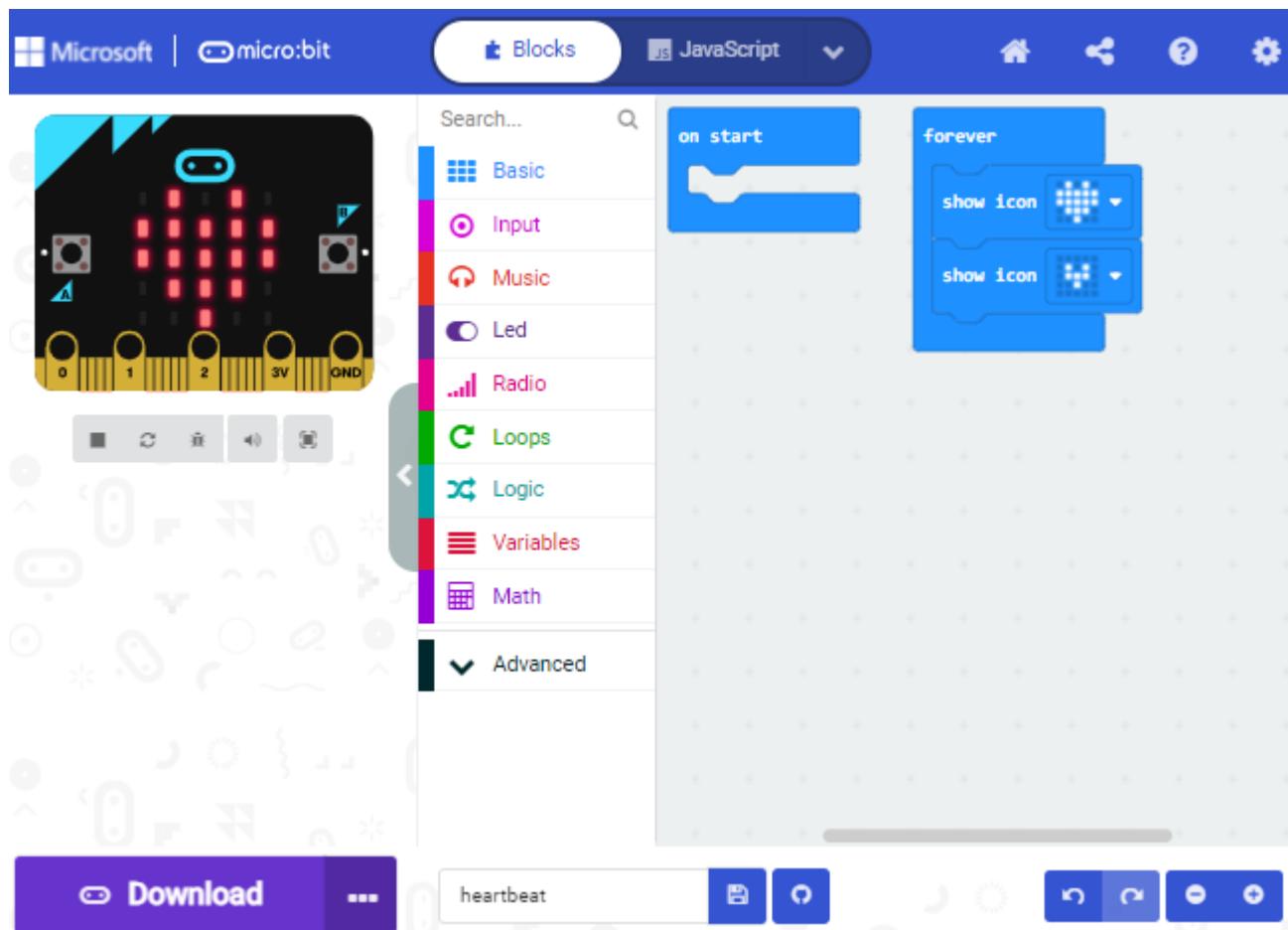
in the folder we provided in the link:

<https://fs.keyestudio.com/KS4031-4032>



Click "Done" to finish the pairing.

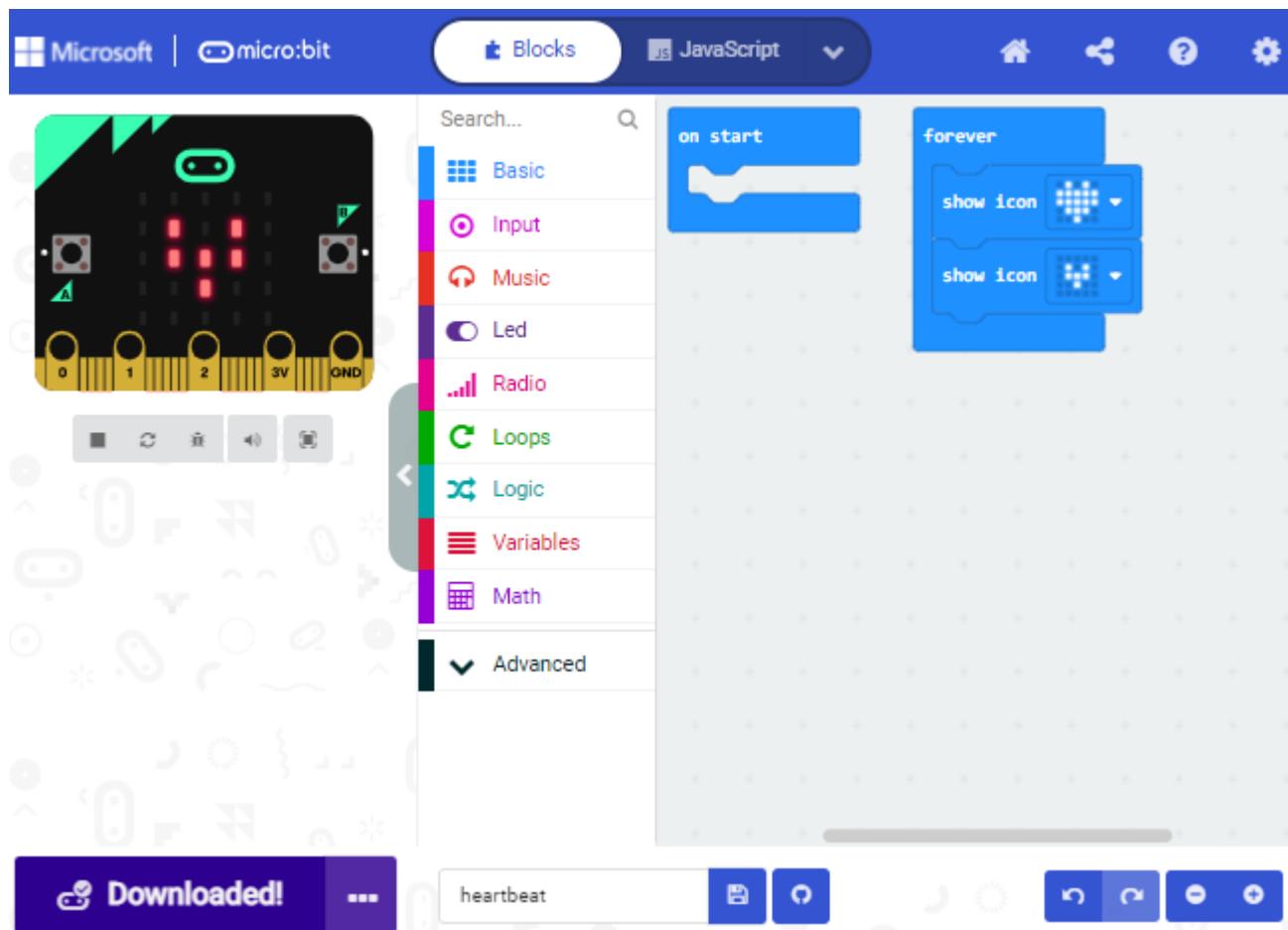




## Download program:

After the pairing, click “download” to directly download the program to the board. If it is successfully downloaded, the icon

will shift to



## 7.4. Makecode extension library:

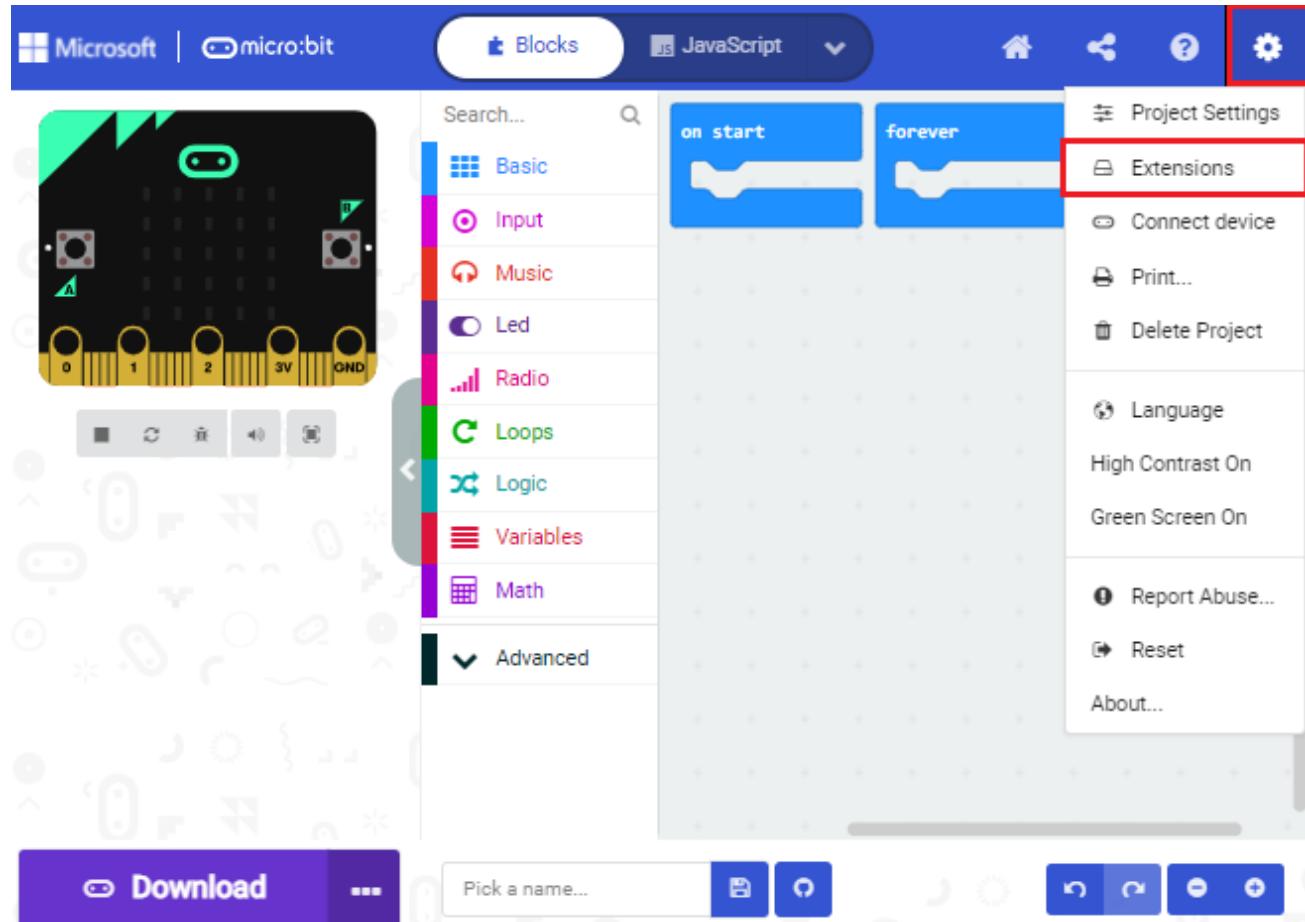
For your convenience, we have made a makecode extension library for this smart home kit.

### Add smart home extension library:

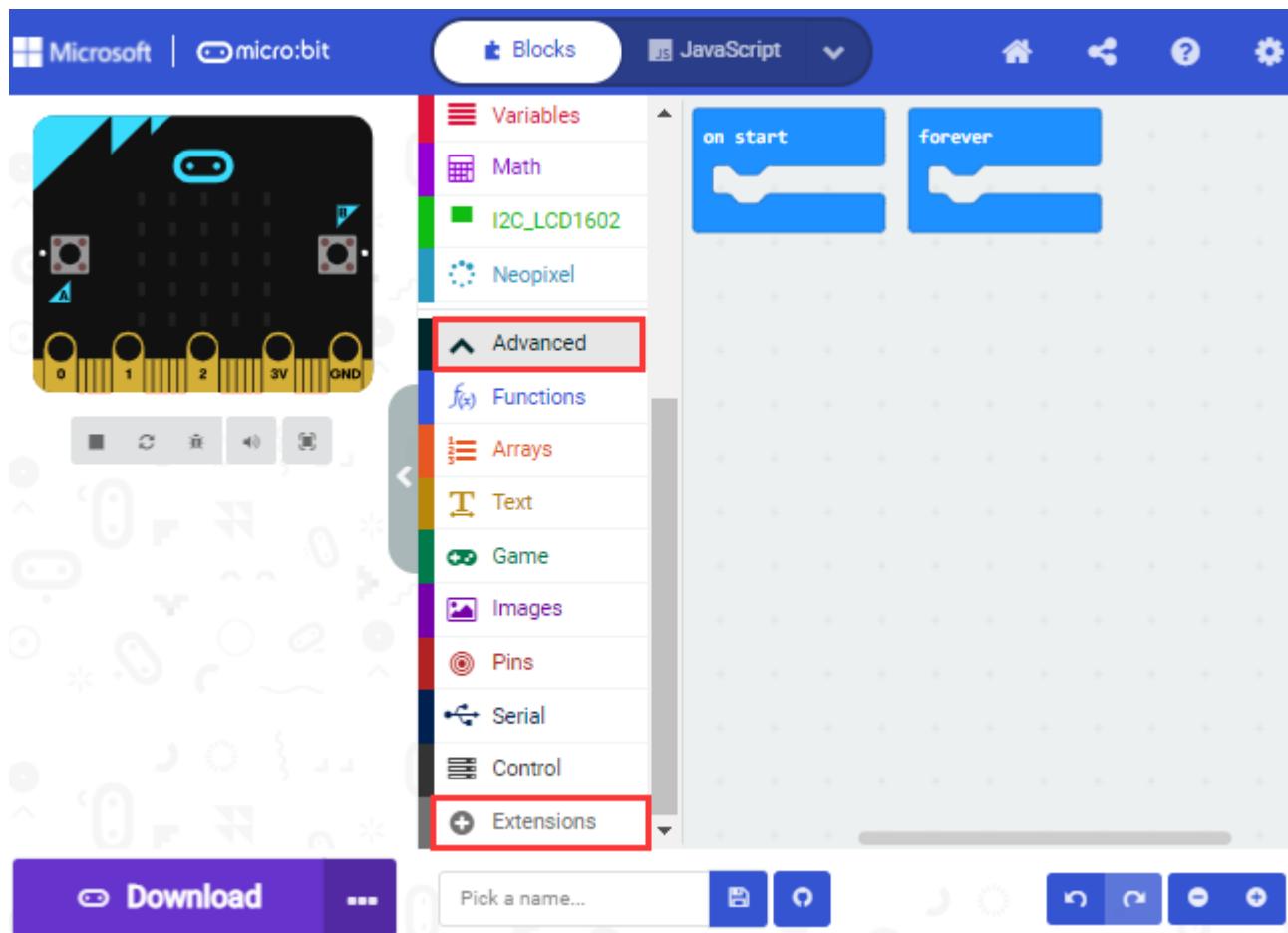
Please follow the following steps to add extension files:



Open Makecode to enter a certain project→click the gear-shaped icon(for setting) in the upper right corner→choose “Extensions” ;



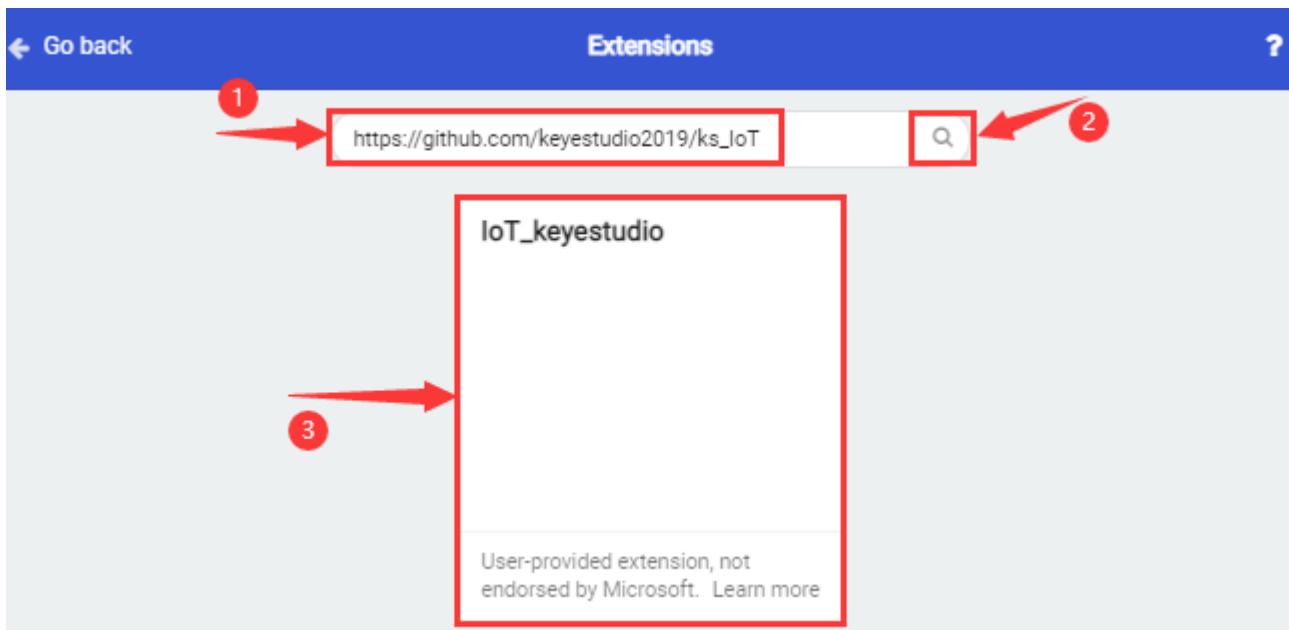
Or click “Advanced” to select “Extensions” as shown below:



Input the link [https://github.com/keyestudio2019/ks\\_IoT](https://github.com/keyestudio2019/ks_IoT) to search;

Tap the searching result “IoT\_keyestudio” to download and install it;

This process may take a few seconds.



After the installation, you can find the extension files DHT11/DHT22 and I2C\_LCD1602 on the left side.

And extension file Neopixel is also installed.



The screenshot shows the Microsoft MakeCode interface for the micro:bit. The top navigation bar includes 'Microsoft' and 'micro:bit' tabs, along with 'Blocks' (selected), 'JavaScript', and various icons for home, refresh, help, and settings.

The central workspace displays a micro:bit board with pins labeled A, 0, 1, 2, 3V, and GND. Below the board are several block categories: Basic, Input, Music, Led, DHT11/DHT22 (highlighted with a red box), Radio, Loops, Logic, Variables, Math, I2C\_LCD1602 (highlighted with a red box), Neopixel, and Advanced.

Two script blocks are visible in the workspace:

- 'on start' block
- 'forever' block

At the bottom, there's a purple 'Download' button, a text input field 'Pick a name...', and a set of circular controls for saving, loading, and navigating scripts.

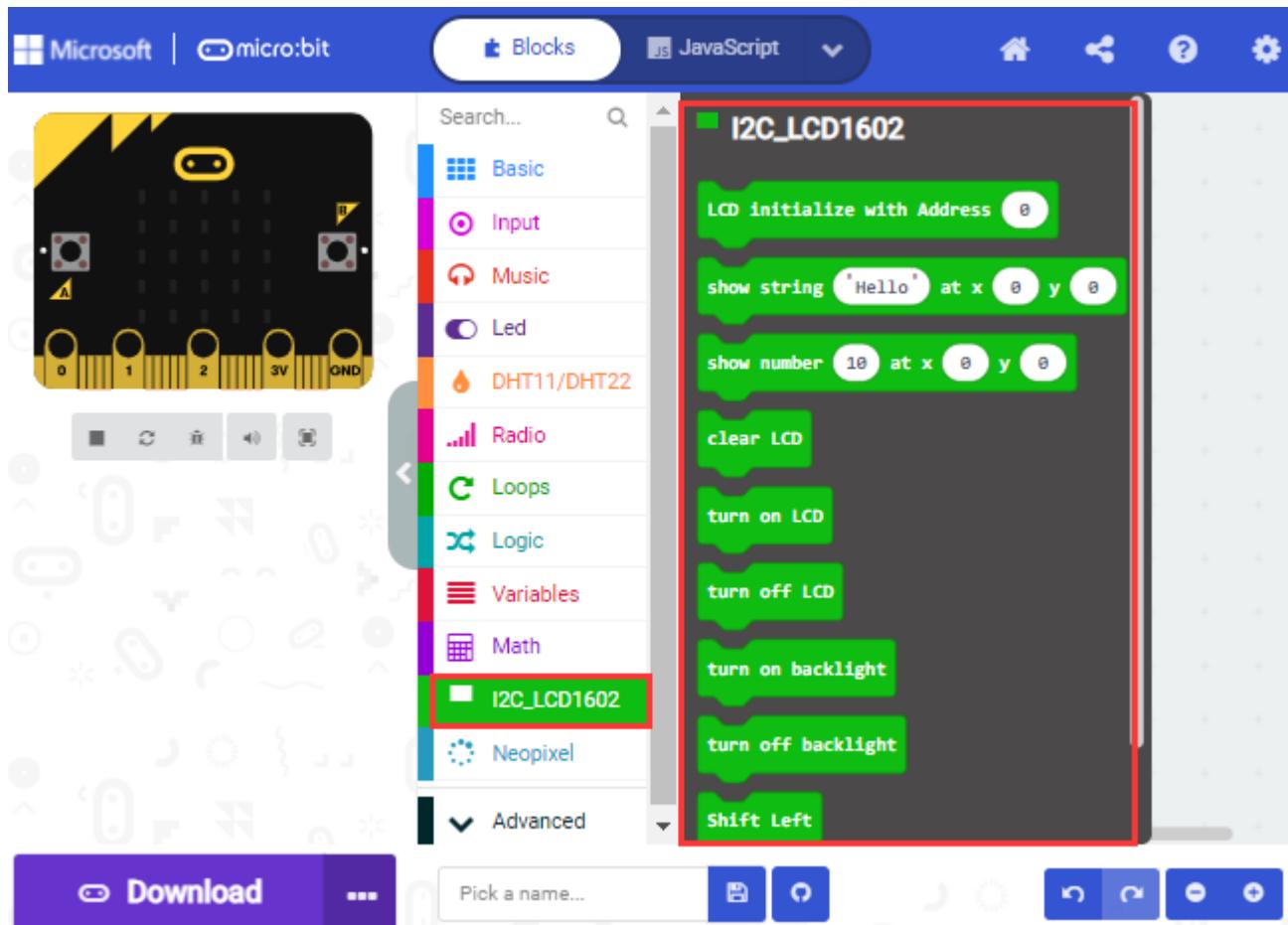


The screenshot shows the Microsoft MakeCode interface for the micro:bit. The top navigation bar includes 'Microsoft' and 'micro:bit' tabs, along with 'Blocks' (selected), 'JavaScript', and various icons for home, help, and settings.

The workspace displays a script for reading DHT11/DHT22 sensor data. The script starts with a conditional block 'Last query successful?' followed by a 'Read humidity' block. This is followed by a series of configuration blocks: 'Query DHT11', 'Data pin P0', 'Pin pull up true', 'Serial output false', and 'Wait 2 sec after query true'. The entire script is highlighted with a red border.

The left sidebar shows the micro:bit board with pins 0, 1, 2, 3V, and GND labeled. Below the board are various block categories: Basic, Input, Music, Led, DHT11/DHT22 (which is currently selected and highlighted in orange), more, Radio, Loops, Logic, Variables, Math, I2C\_LCD1602, and Neopixel. A search bar is also present in the sidebar.

At the bottom, there is a purple 'Download' button, a text input field 'Pick a name...', and several blue circular icons for saving, undoing, redoing, and deleting code blocks.

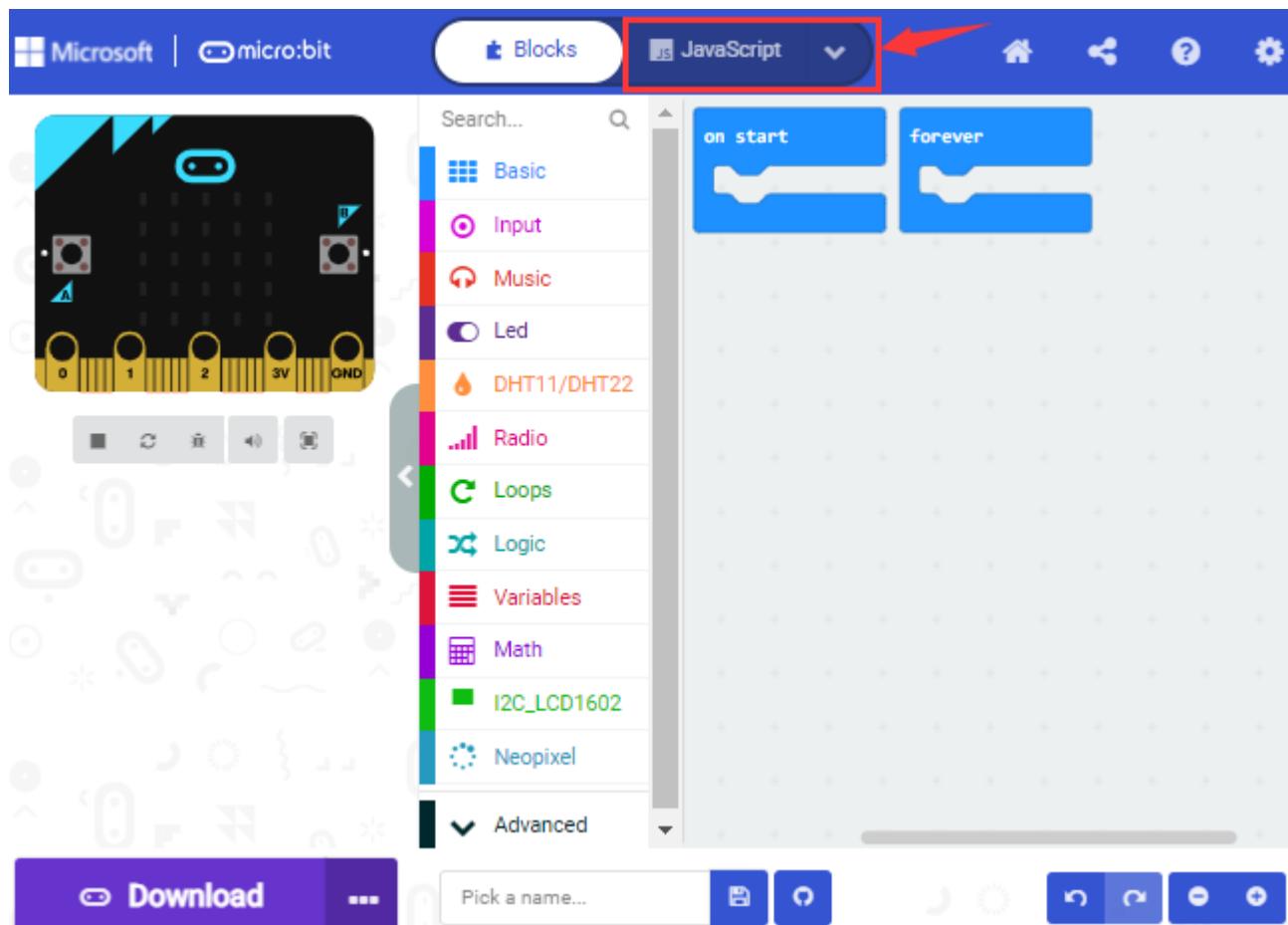


Note: the extension files added are only available for this project. Therefore, when you create a new **IoT\_keyestudio** project, you will need to add these extension files again.

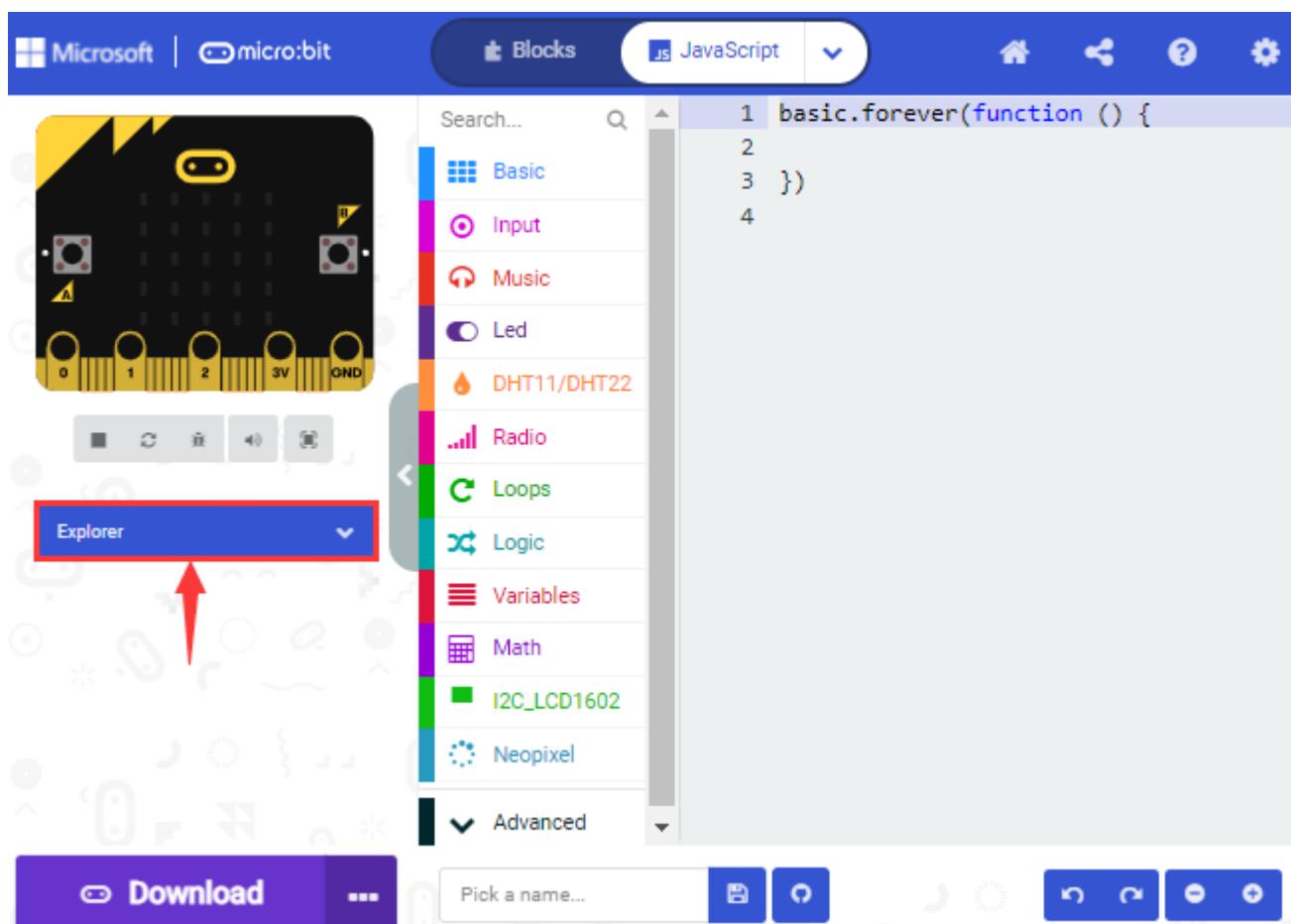
### Update or delete the IoT\_keyestudio extension files:

Please follow the following steps to update or delete extension files:

Click "Js JavaScript" to change to textual version:



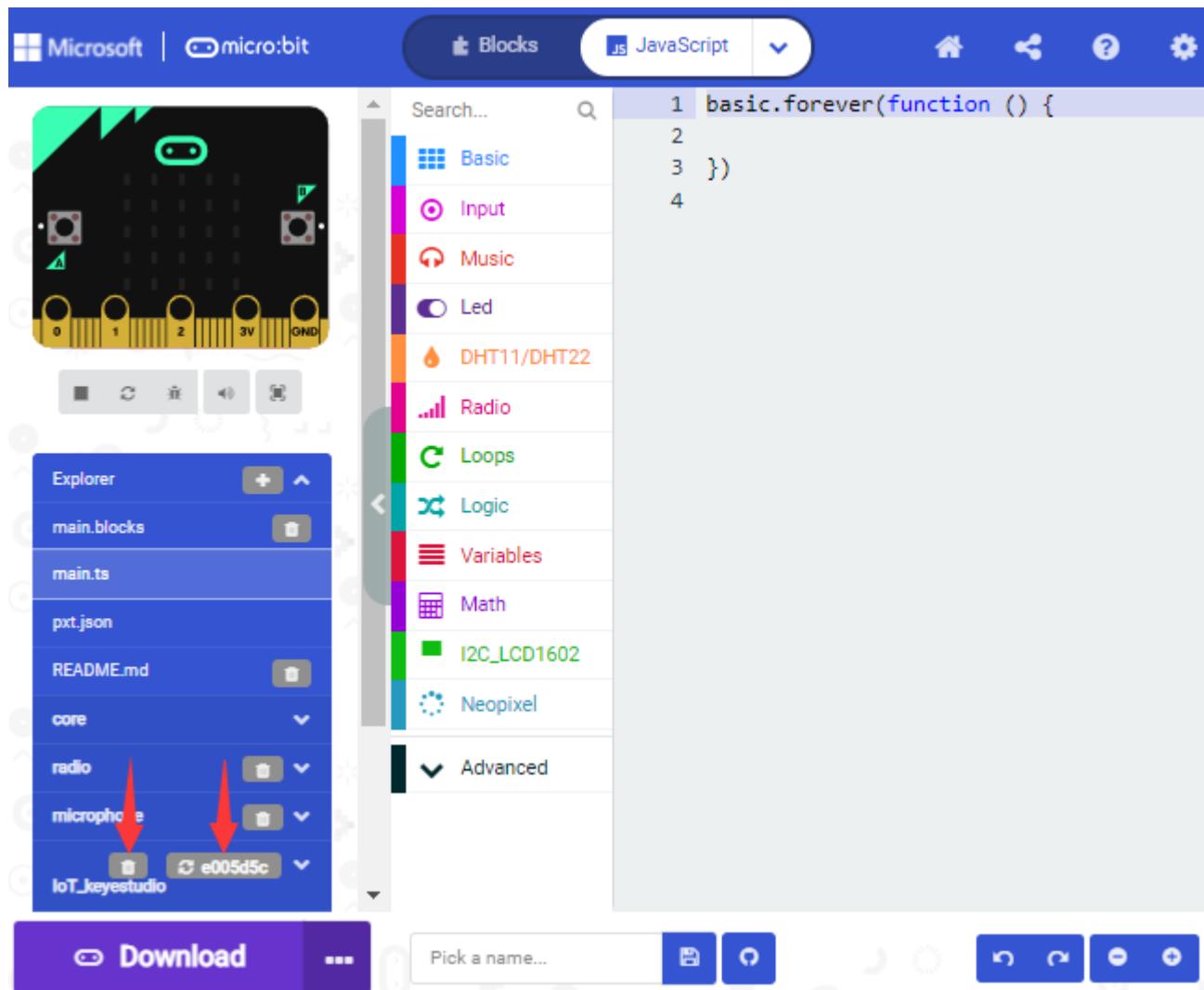
Click the “Explorer” on the left side:



You can find these added files in the list;

Click the dustbin icon beside the file to delete the corresponding file;

Tap the refresh icon to update the corresponding IoT\_keyestudio extension file.



## 7.4.Resources and test code

We also provide a link: <https://fs.keyestudio.com/KS4031-4032>

containing the information of the product from relevant tools to test codes, tutorials and troubleshooting methods as well, as shown in the figure below:



## Name ↑

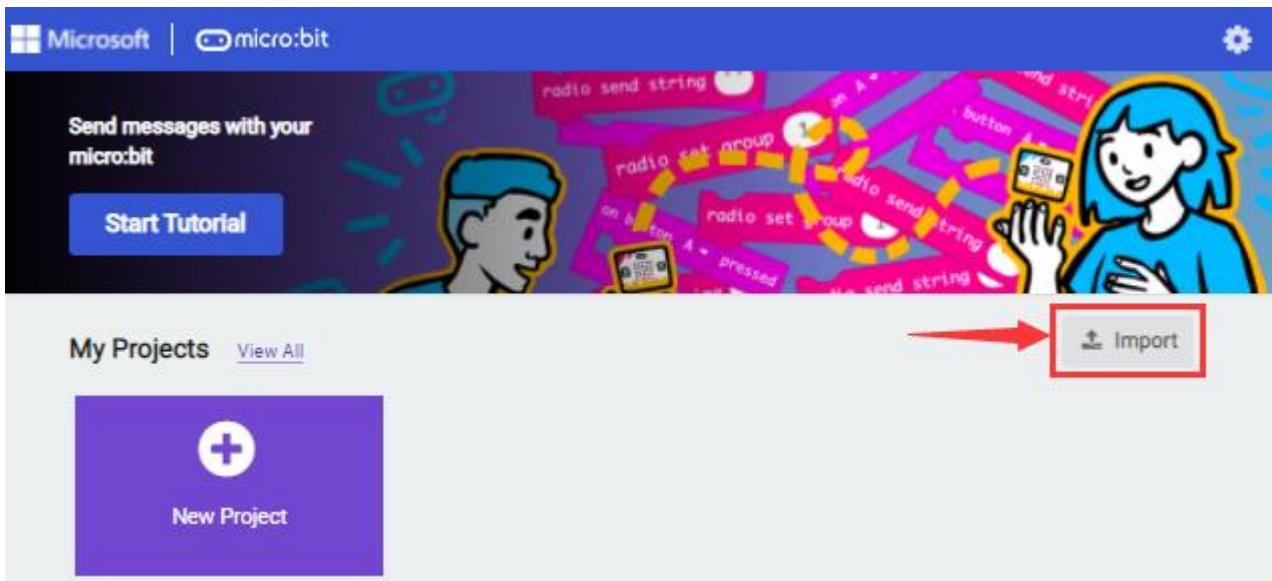
-  1. Install Microbit Driver
-  2. Makecode Tutorial
-  3. Python Tutorial
-  4. How to Update the Firmware
-  5. Troubleshooting-MAINTENANCE Mode
-  6. Troubleshooting-WebUSB
-  7. Cool Term Download

## 7.5. Input test code

We provide hexadecimal code files (project files) for each project. The file contains all the contents of the project and can be imported directly, or you can manually drag the code blocks to complete the program for each project. For simple projects, dragging a block of code to complete the program is recommended. For complex projects, it is recommended to conduct the program by importing the hexadecimal code file we provide.

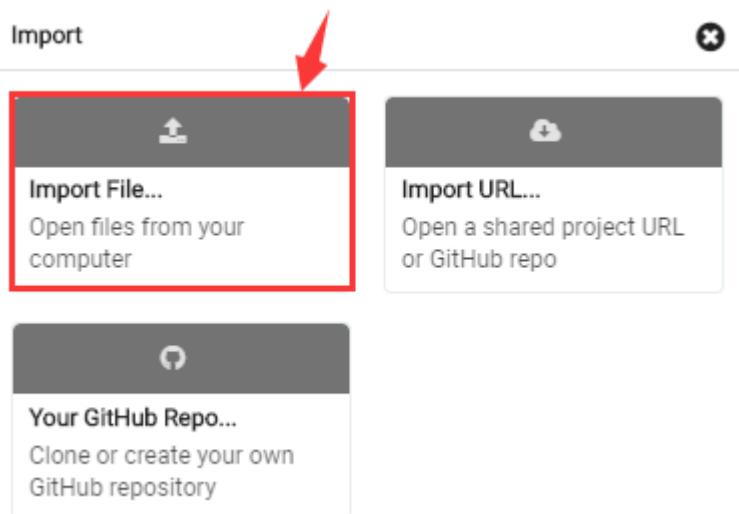
Let's take the "Heatbeat" project as an example to show how to load the code.

Open the Web version of Makecode or the Windows 10 App version of Makecode;



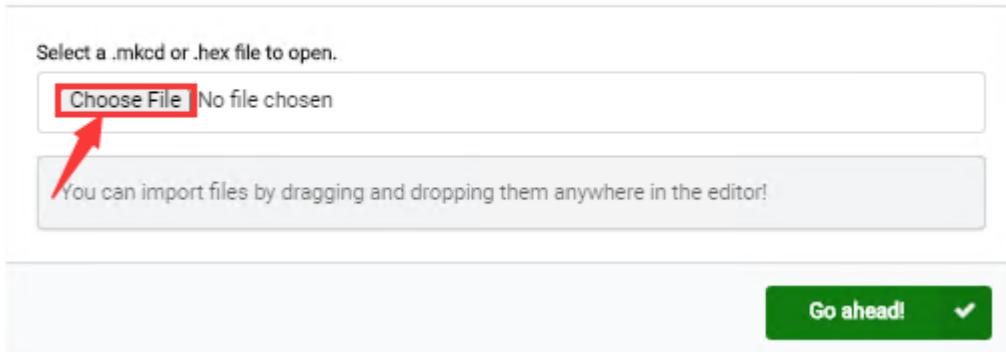
The screenshot shows the Microsoft micro:bit web editor interface. At the top, there's a header with the Microsoft logo and the text "Microsoft | micro:bit". Below the header, there's a section titled "Send messages with your micro:bit" with a "Start Tutorial" button. To the right, there's a colorful illustration of two characters interacting with micro:bit boards. In the bottom left, there's a purple "New Project" button. On the right side, there's a "Import" button with a red arrow pointing to it.

Click "Import File" ;



The screenshot shows the "Import" dialog box. It has three main options: "Import File..." (highlighted with a red box and a red arrow), "Import URL...", and "Your GitHub Repo...".

Open .mkcd or .hex file

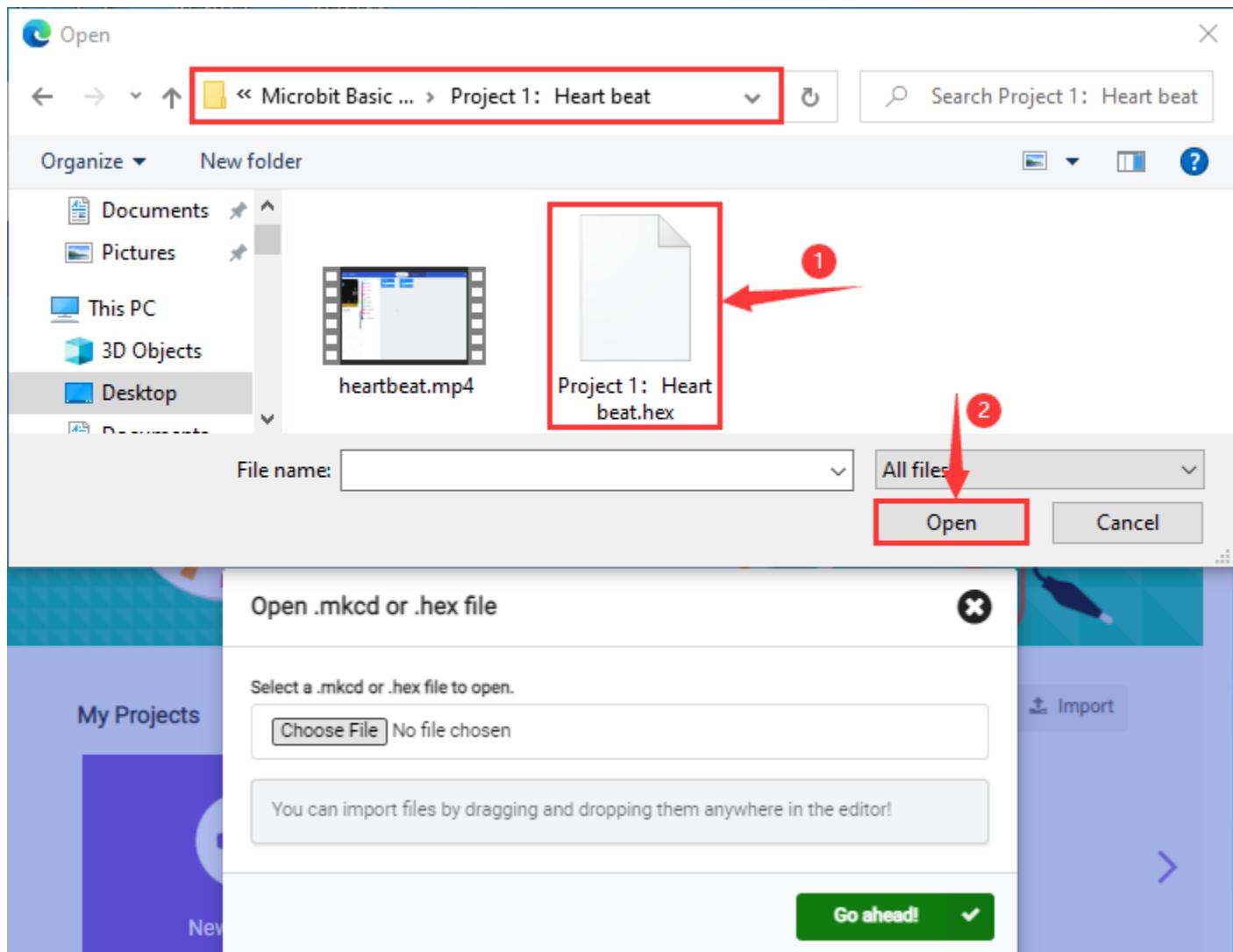


The screenshot shows the "Open .mkcd or .hex file" dialog box. It includes a message "Select a .mkcd or .hex file to open.", a "Choose File" button (highlighted with a red box and a red arrow), and a note "You can import files by dragging and dropping them anywhere in the editor!". At the bottom right, there's a green "Go ahead!" button with a checkmark.



Select “..../Makecode Code/Project 1\_ Heart beat/Project 1\_ Heart beat.hex”

Then click “Go ahead” .





## Open .mkcd or .hex file



Select a .mkcd or .hex file to open.

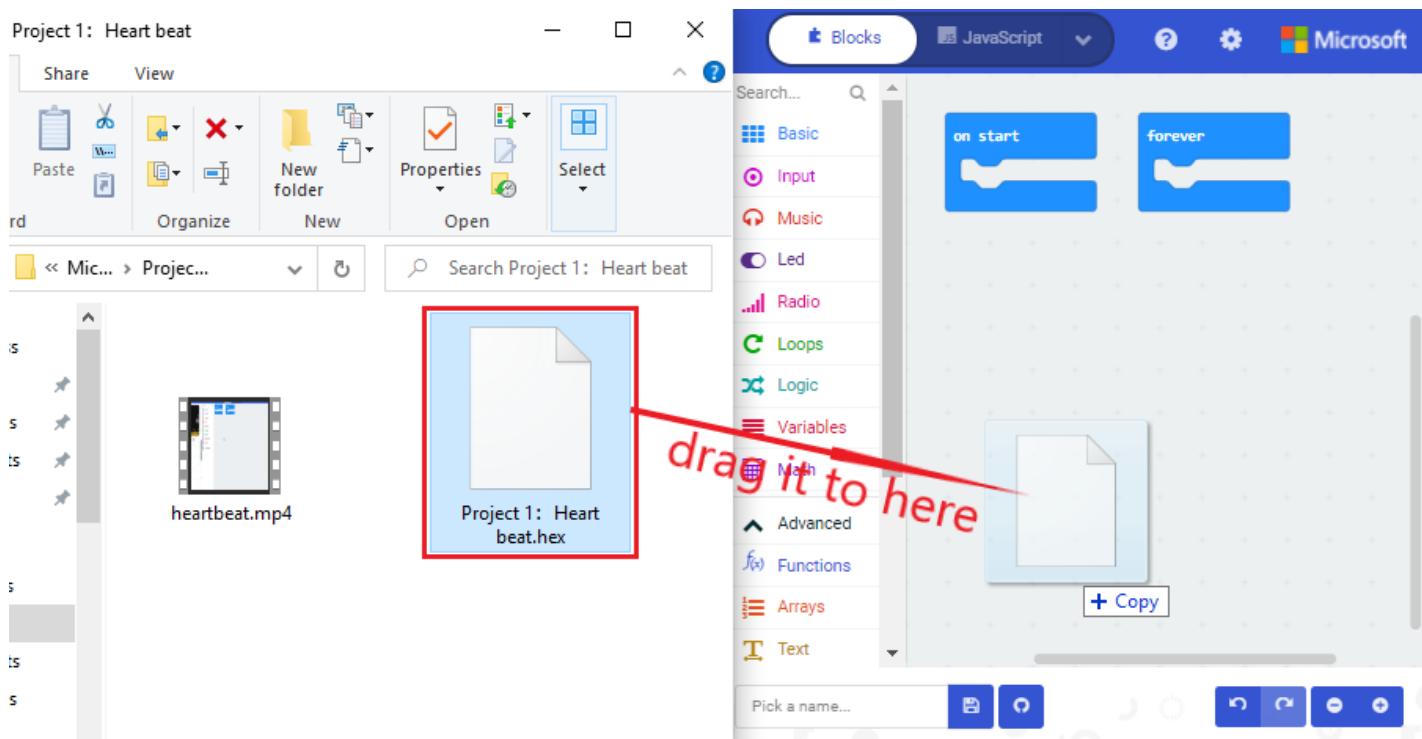
Choose File Project 1\_ Heart beat.hex

You can import files by dragging and dropping them anywhere in the editor!

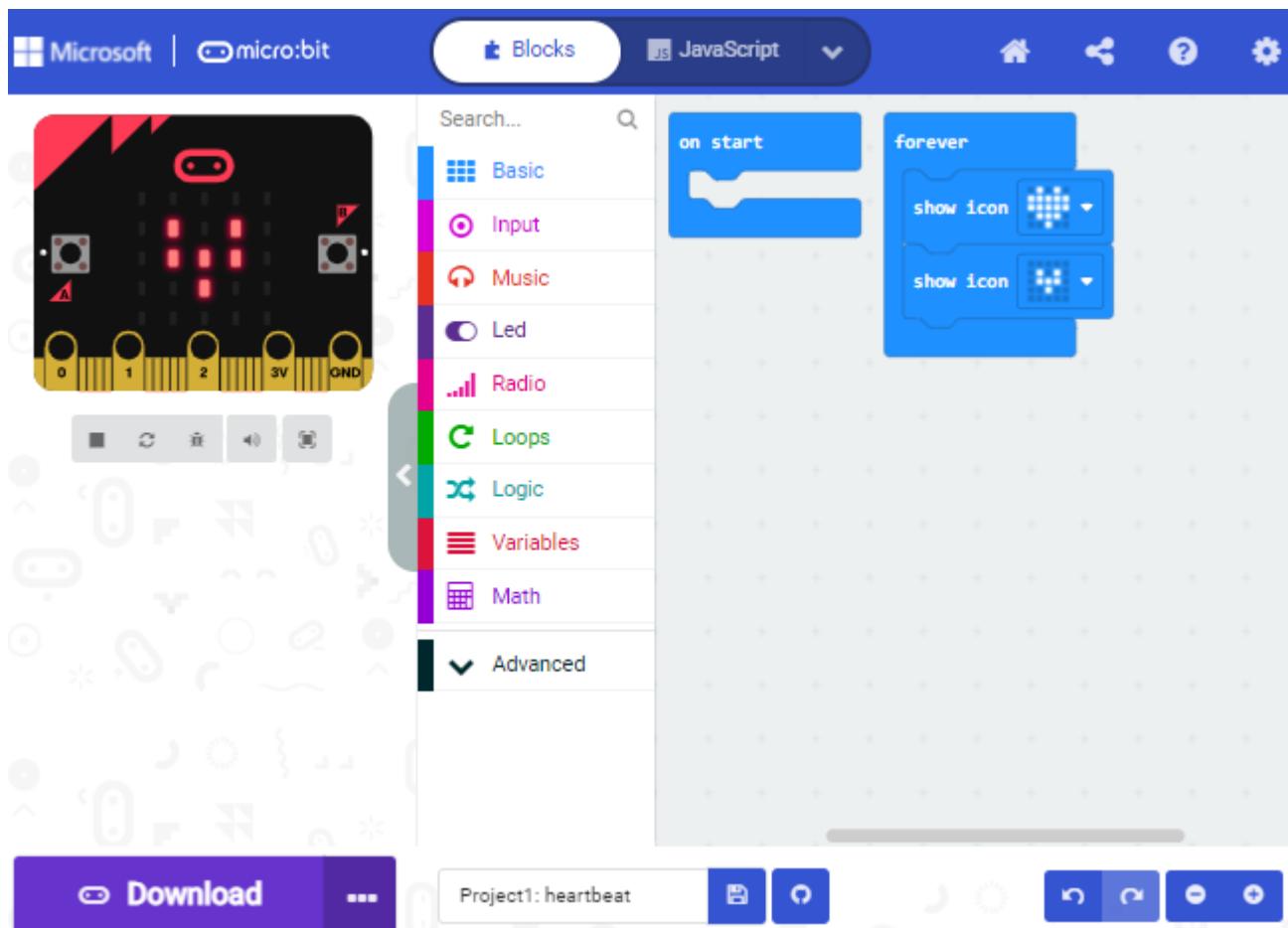
Go ahead! ✓



In addition to importing the test code file provided into the Makecode compiler above, you can also drag the the test code file provided into the code editing area of the Makecode compiler, as shown in the figure below:



After a few seconds, it is done.



Note: if your computer system is Windows 7 or 8 instead of Windows 10, the pairing cannot be done via Google Chrome. Therefore, digital signal or analog signal of sensors and modules cannot be shown on the serial port simulator. However, you need to read the corresponding digital signal or analog signal. So what can we do? You can use the CoolTerm software to read the serial port data of the microbit. Next chapter is about how to install CoolTerm.



## 7.6. Install CoolTerm:

CoolTerm program is used to read the data on serial port.

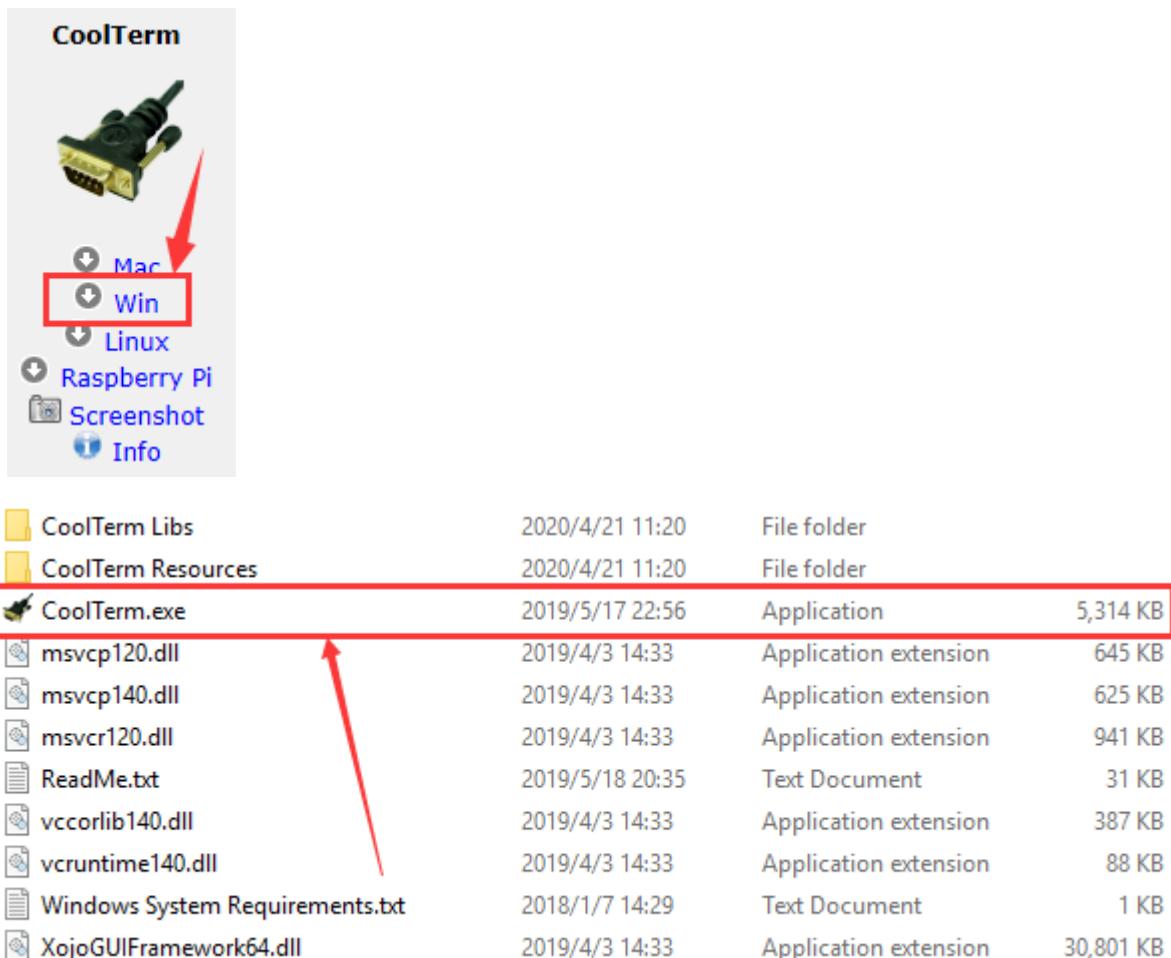
Download CoolTerm program:

<https://freeware.the-meiers.org/>

After the download, we need to install **CoolTerm program file**, below is PC Window system taken as an example.

- (1) Choose "win" to download the zip file of CoolTerm
- (2) Unzip file and open it. **(also suitable for Mac and Linux system)**

(1)



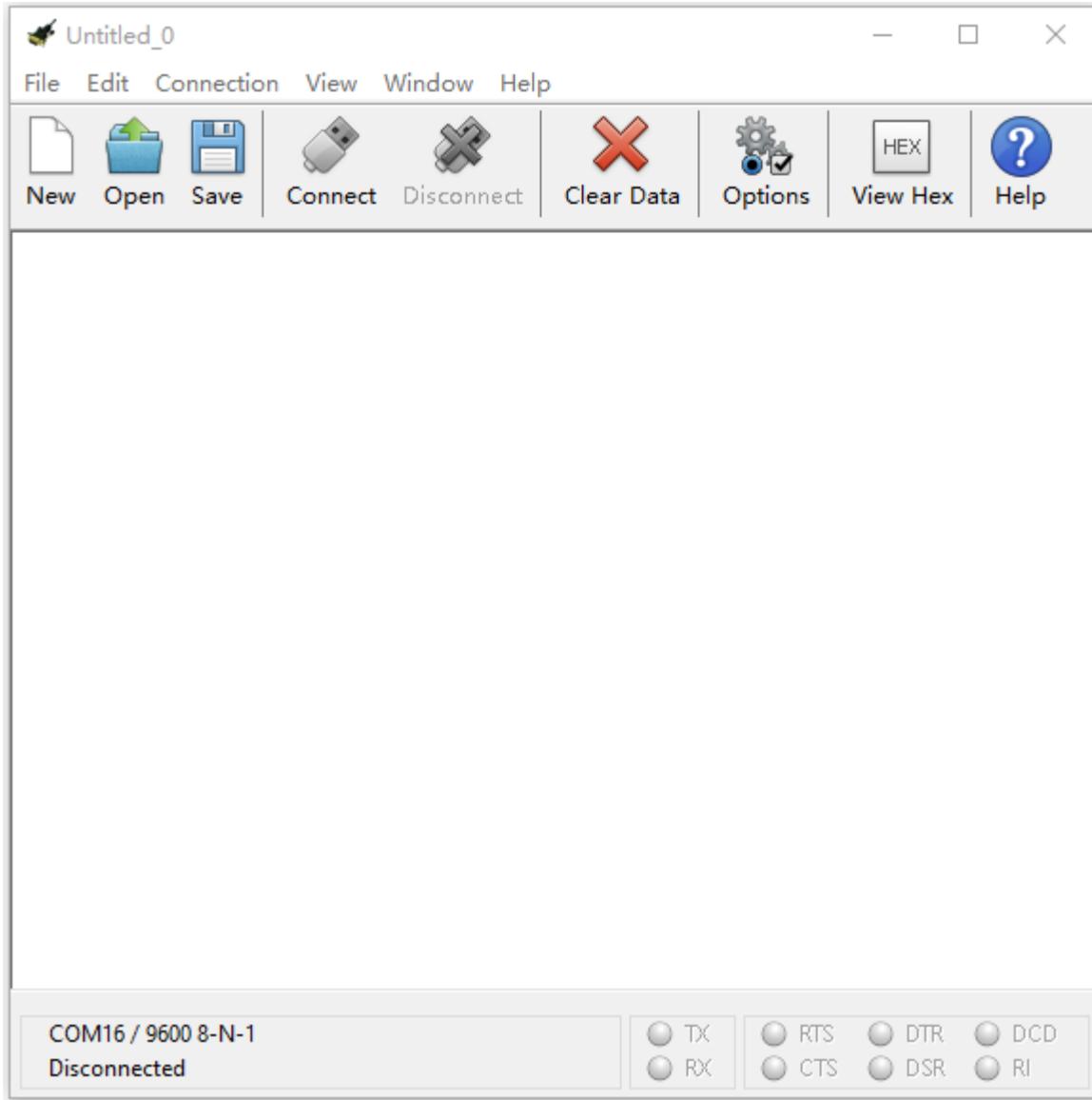
The screenshot shows the CoolTerm download page. At the top, there's a navigation bar with links for Mac, Win (which is highlighted with a red box), and Linux. Below this are links for Raspberry Pi, Screenshot, and Info. The main content area shows a list of files:

File	Modified	Type	Size
CoolTerm Libs	2020/4/21 11:20	File folder	
CoolTerm Resources	2020/4/21 11:20	File folder	
<b>CoolTerm.exe</b>	2019/5/17 22:56	Application	5,314 KB
msvcp120.dll	2019/4/3 14:33	Application extension	645 KB
msvcp140.dll	2019/4/3 14:33	Application extension	625 KB
msvcr120.dll	2019/4/3 14:33	Application extension	941 KB
ReadMe.txt	2019/5/18 20:35	Text Document	31 KB
vccorlib140.dll	2019/4/3 14:33	Application extension	387 KB
vcruntime140.dll	2019/4/3 14:33	Application extension	88 KB
Windows System Requirements.txt	2018/1/7 14:29	Text Document	1 KB
XojoGUILFramework64.dll	2019/4/3 14:33	Application extension	30,801 KB

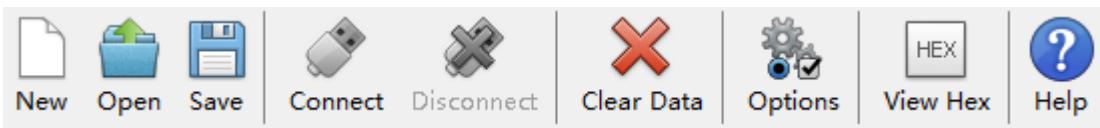
A red arrow points from the "Win" link in the navigation bar down to the "CoolTerm.exe" file in the file list, indicating the selected download.



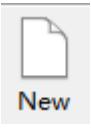
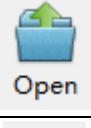
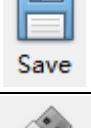
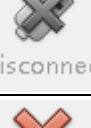
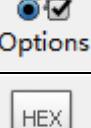
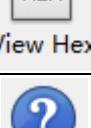
(3) Double-click  CoolTerm.exe. (please make sure that the driver of Micro:bit is installed and the main board is connected with the computer.)



The functions of each button on the Toolbar are listed below:





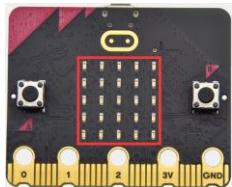
	Open up a new Terminal
	Open a saved Connection
	Save the current Connection to disk
	Open the Serial Connection
	Close the Serial Connection
	Clear the Received Data
	Open the Connection Options Dialog
	Display the Terminal Data in Hexadecimal Format
	Display the Help Window

## 8.Projects

(Note: project 8.1 to 8.12 will be conducted with the built-in sensors and LED dot matrix of the Micro:bit main board V2)



## Project 1: Heartbeat



### (1) Project Description

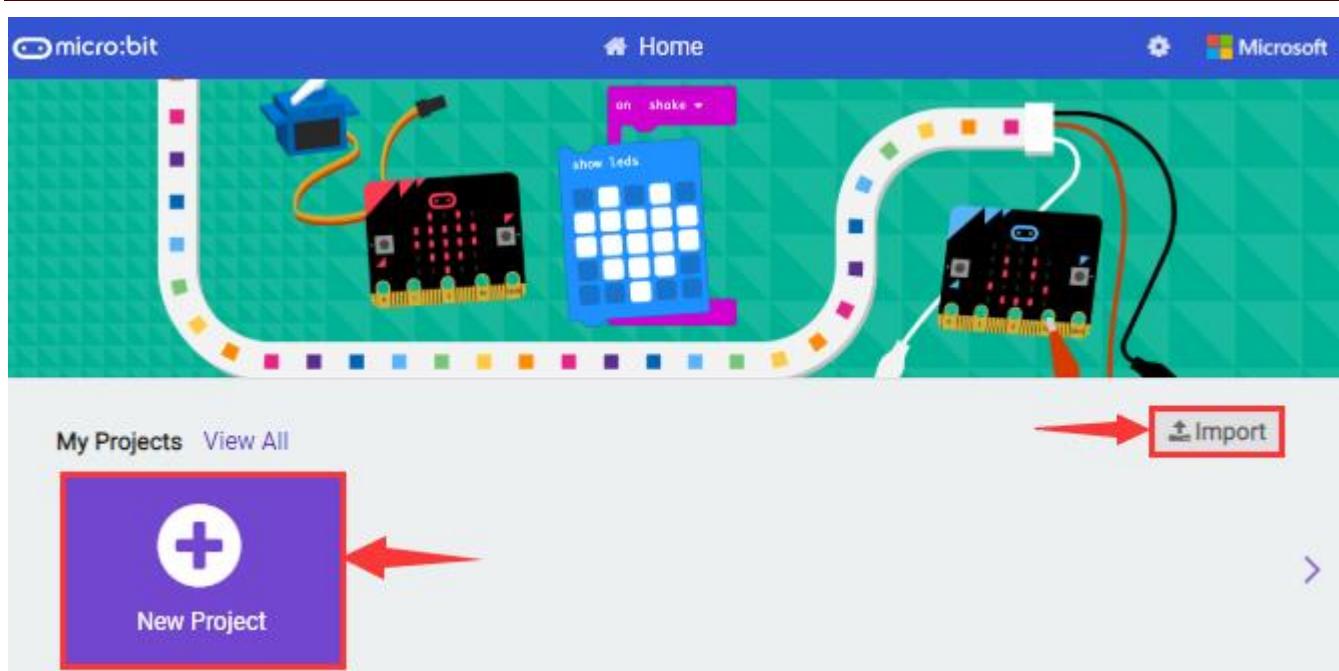
This project is easy to conduct with a micro:bit V2 main board, a Micro USB cable and a computer. The micro:bit LED dot matrix will display a relatively big heart-shaped pattern and then a smaller one. This alternative change of this pattern is like heart beating. This experiment serves as a starter for your entry to the programming world.

### (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor

Import Hex profile (How to import?)

Or click “New Project” and drag blocks step by step



### (3) Test Code

The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makemode Tutorial/Makemode Code/Project Code/Project 1: Heartbeat	Project 1: Heartbeat.hex

Or you could edit code step by step in the editing area.

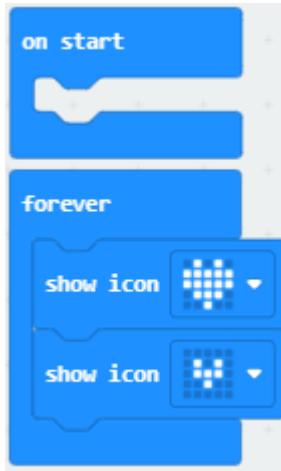
Go to "Basic" → "show icon" .

Copy it again and place into "forever" block.



Click “

Complete Program:



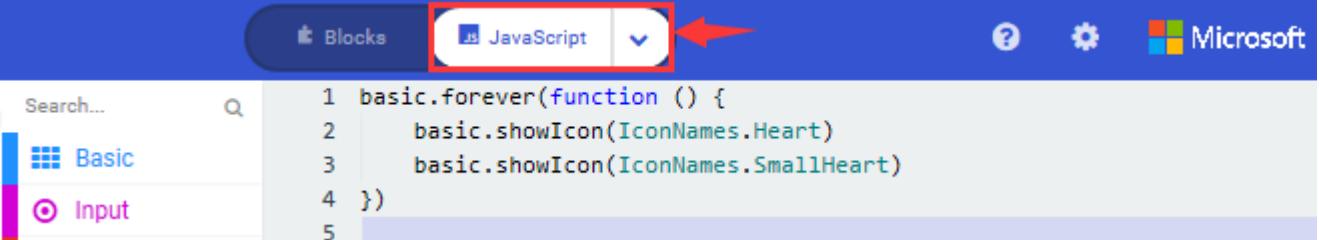
“on start” : command block runs once to start program.

The program under the block “forever” runs cyclically.

LED dot matrix displays “

LED dot matrix shows “

Click “JavaScript” to view the corresponding JavaScript code:



```
1 basic.forever(function () {
2     basic.showIcon(IconNames.Heart)
3     basic.showIcon(IconNames.SmallHeart)
4 })
5
```

8



## (4) Test Results

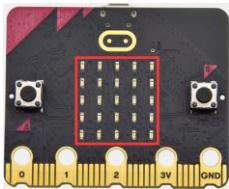
Download code to micro:bit and keep USB cable connected. The LED dot

matrix will display  and  ceaselessly.

([How to download?](#) [How to quick download?](#))

If the download is not success, try to disconnect micro:bit from your computer and then reconnect them and reopen Makecode to try again.

## Project 2: Light A Single LED



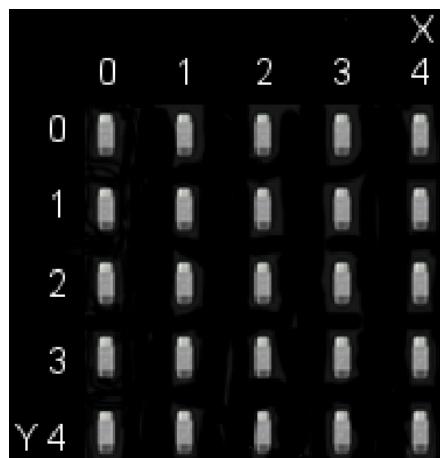
### (1) Project Description

The LED dot matrix consists of 25 LEDs arranged in a 5 by 5 square. In order to locate these LEDs quickly, as the figure shown below, we can regard this matrix as a coordinate system and create two axes by marking those in rows from 0 to 4 from top to bottom, and the ones in columns from 0 to 4 from the left to the right. Therefore, the LED at the second of the first line is (1,0) and the LED positioned in the fifth of the fourth column is (3,4)



---

and others likewise.



## (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor

Import Hex profile (How to import?)

Or click "New Project" and drag blocks step by step

## (3) Test Code

The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 2: Light A Single	Project 2: Light A Single LED.hex



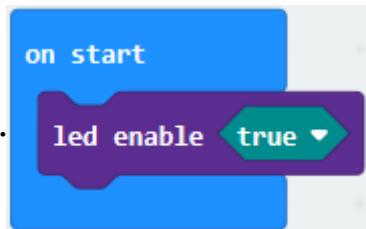
	LED	
--	-----	--

Or you could edit code step by step in the editing area.

A. Click "Led" → "more" → "led enable false"

B. Put it into the "on start" block, and click the drop-down triangle button

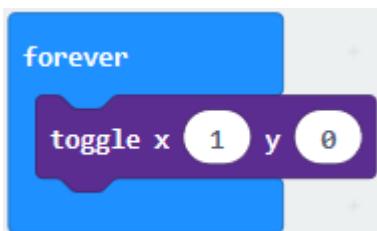
to select "true".



\*\*\*\*\*

(2) A. Enter "Led" → "toggle x 0 y 0" block;

B. Combine it with "forever" , alter "x 0" into "x 1" .

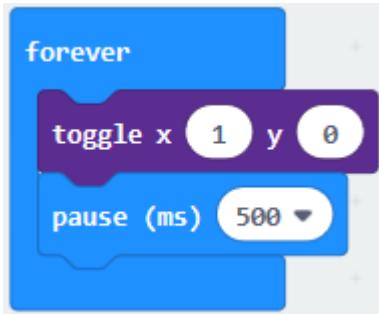


\*\*\*\*\*

(3) A. Enter "Basic" → "pause (ms) 100" from "



B. Then move it below the “toggle x1 y0” block, and set to 500ms.



(4) Duplicate code string once and place it into “forever”

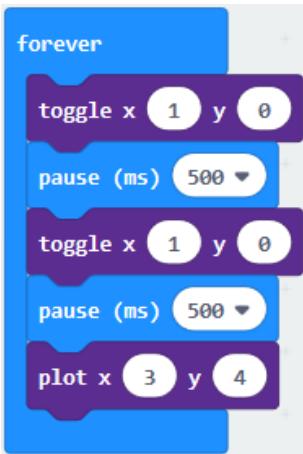
block.



\*\*\*\*\*

A. Enter “Led” → “plot x 0 y 0”

B. Keep it beneath block “pause(ms)500” , then set to “plot x 3 y 4” :



\*\*\*\*\*

Replicate “pause (ms) 500” once and keep it below the block “plot x3y4”



Click “Led” → “unplot x 0 y 0” and set to “unplot x3 y 4” ;

Lay down it beneath “pause (ms) 500” block

Copy “pause (ms) 500” block once, and keep it below the “unplot x3 y 4”



block.





Click "JavaScript" to switch into corresponding JavaScript code:

Complete Program:



"on start": command block only runs once to start program.

Turn on LED dot matrix.

The program under the block "forever" runs cyclically.

Toggle the LED brightness at coordinate point "x 1 y 0".

Toggle the LED brightness at coordinate point "x 1 y 0".

Turn on the LED at coordinate point "x3, y4".

Delay in 500ms

Turn off the LED at coordinate point "x3 y4".



The image shows the Scratch interface. At the top, there are three tabs: 'Blocks', 'JavaScript' (which is selected and highlighted with a red arrow), and a dropdown menu. Below the tabs is a search bar labeled 'Search...'. To the right of the search bar is a vertical sidebar with categories: Basic, Input, Music, Led, Radio, Loops, and Logic. The 'Basic' category is currently selected. On the right side of the interface, the script editor contains the following JavaScript code:

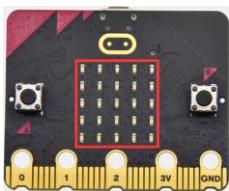
```
1 led.enable(true)
2 basic.forever(function () {
3     led.toggle(1, 0)
4     basic.pause(500)
5     led.toggle(1, 0)
6     basic.pause(500)
7     led.plot(3, 4)
8     basic.pause(500)
9     led.unplot(3, 4)
10    basic.pause(500)
11 })
12
```

#### (4) Test Results:

After uploading test code to micro:bit main board V2 and powering the main board via the USB cable, the LED in (1,0) lights up for 0.5s and the one in (3,4) shines for 0.5s and repeat this sequence.

([How to download?](#)   [How to quick download?](#))

### Project 3: LED Dot Matrix





## (1)Project Description

Dot matrices are very commonplace in daily life. They have found wide applications in LED advertisement screens, elevator floor display, bus stop announcement and so on.

The LED dot matrix of Micro: Bit main board V2 contains 25 LEDs in a grid. Previously, we have succeeded in controlling a certain LED to light by integrating its position value into the test code. Supported by the same theory, we can turn on many LEDs at the same time to showcase patterns, digits and characters.

What's more, we can also click "show" icon " to choose the pattern we like to display. Last but not the least, we can design patterns by ourselves.

## (2)Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor

Import Hex profile (How to import?)

Or click "New Project" and drag blocks step by step

## (3)Test Code

### Code 1:



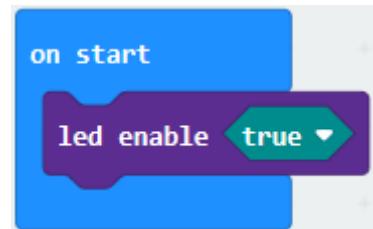
The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 3: LED Dot Matrix-1	Project 3: LED Dot Matrix-1

Or you could edit code step by step in the editing area.

A. Enter “Led” → “more” → “led enable false”

Click the drop-down triangle button to select “true”



Combine it with “on start” block

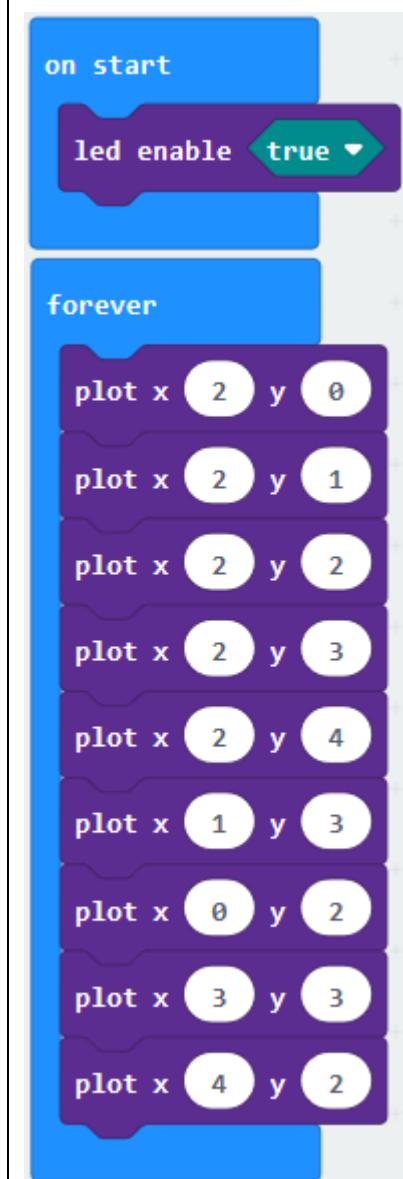
\*\*\*\*\*

Click “Led” to move “plot x 0 y 0” into “forever” , then replicate “plot x 0 y 0” for 8 times, respectively set to “x 2” y 0” , “x 2” y 1” , “x 2” y 2” , “x 2” y 3” , “x 2” y 4” , “x 1” y 3” “x 0” y 2” , “x 3” y 3” , “x 4” y 2” .



```
forever
  plot x 2 y 0
  plot x 2 y 1
  plot x 2 y 2
  plot x 2 y 3
  plot x 2 y 4
  plot x 1 y 3
  plot x 0 y 2
  plot x 3 y 3
  plot x 4 y 2
```

Complete Program:



“on start” : command block only runs once to start program.

Turn on LED dot matrix.

The program under the block “forever” runs cyclically.

Toggle the LED brightness at coordinate point “x 2, y 0” , “x 2, y 1” , “x 2, y 2” , “x 2, y 3” , “x 2, y 4” , “x 1, y 3” , “x 0, y 2” , “x 3, y 3” and “x 4, y 2”

Select “JavaScript” and “Python” to switch into JavaScript and Python language code:

**Code 2:**



The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 3: LED Dot Matrix-2	Project 3: LED Dot Matrix-2

Or you could edit code step by step in the editing area.

A. Enter “Basic” → “show number 0” block,

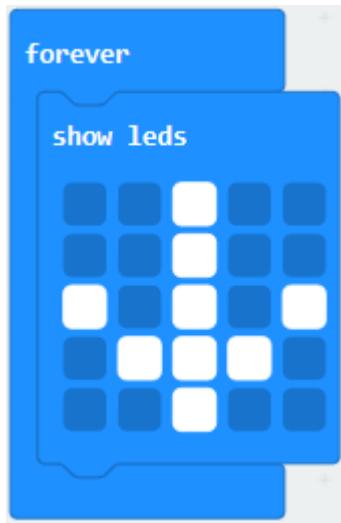
Duplicate it for 4 times, then separately set to “show number 1” , “show number 2” , “show number 3” , “show number 4” , “show number 5” .



\*\*\*\*\*

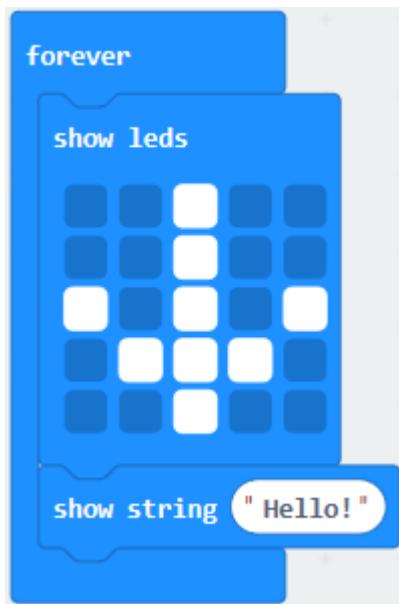


Click “Basic” → “show leds”, then put it into “forever” block, tick blue boxes to light LED and generate “↓” pattern.

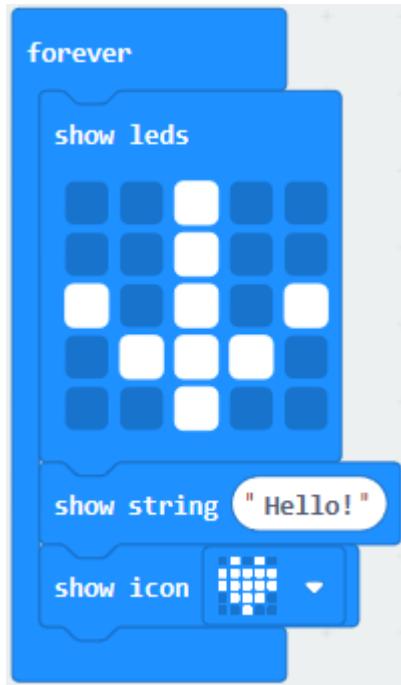


\*\*\*\*\*

(1) Move out the block “show string” from “Basic” block, and leave it beneath the “show leds” block

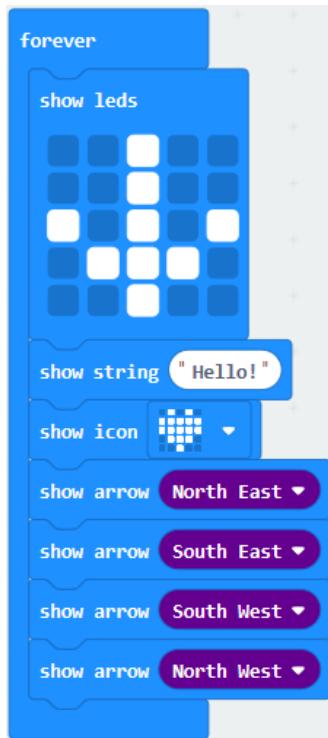


Choose “show icon” from “Basic” block, and leave it beneath the block “show string “Hello!” block

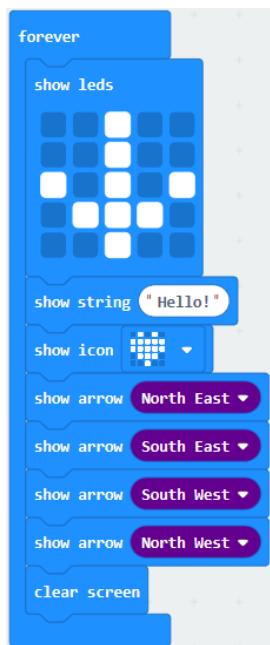


\*\*\*\*\*

- (2) A. Enter “Basic” → “show arrow North” ;  
B. Leave it into “forever” block, replicate “show arrow North” for 3 times, respectively set to “North East” , “South East” , “South West” , “North West” .



(3) Click “Basic” to get block “clear screen” then remain it below the block “show arrow North West”

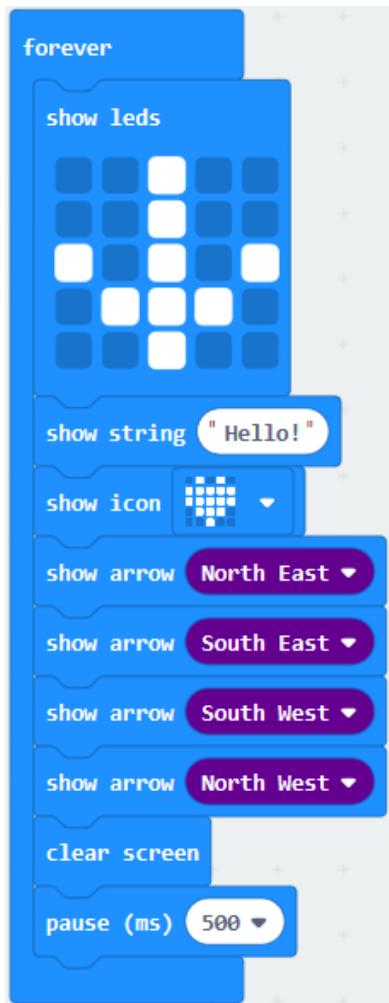


\*\*\*\*\*

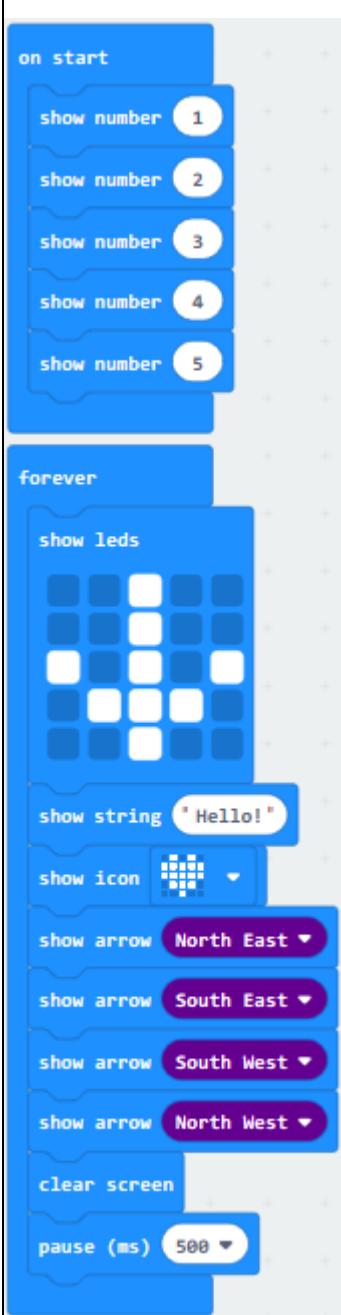
(5) Drag “pause (ms) 100” block from “Basic” block and set to 500ms, then



leave it below "clear screen" block.



Complete Program:



“on start”: command block only runs once to start program.

LED dot matrix displays 1,2,3,4,5

Under the block “forever”, program runs cyclically.

Dot matrix shows the “↓” pattern

Dot matrix scrolls to show “Hello!”

“” is shown on dot matrix

LED dot matrix displays “North East” arrow.

The “South East” arrow shows up on LED dot matrix

The “South West” arrow appears up on LED dot matrix

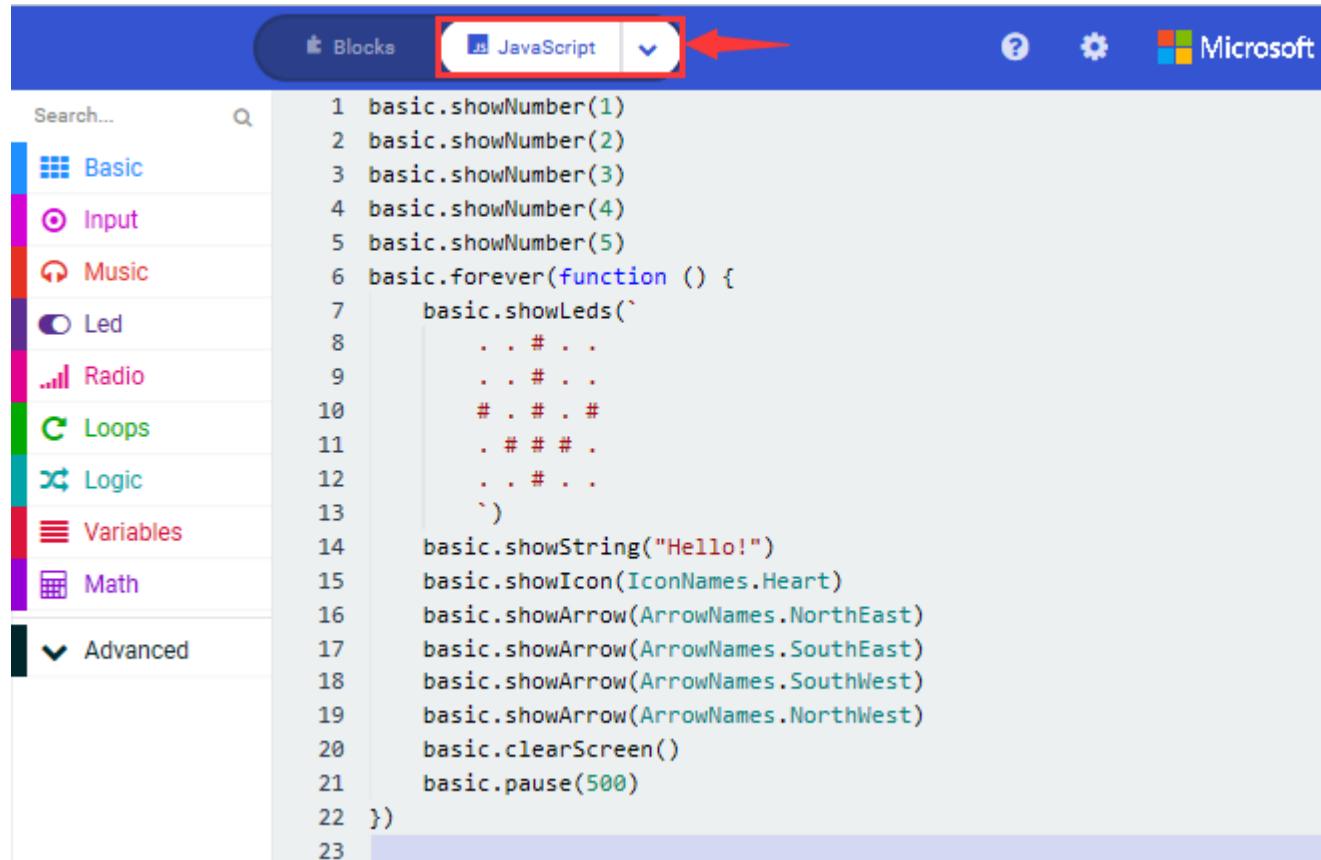
The “North West” arrow is displayed on LED dot matrix

Clear the screen

Delay in 500ms



Select "JavaScript" and "Python" to switch into JavaScript and Python language code:

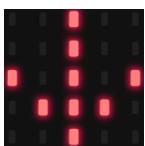


The screenshot shows the Scratch programming environment. At the top, there's a blue header bar with tabs for "Blocks" (selected), "JavaScript" (highlighted with a red arrow pointing to it), and "Python". To the right of the tabs are icons for help, settings, and Microsoft. Below the header is a search bar labeled "Search..." with a magnifying glass icon. On the left, a sidebar lists categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, and Advanced. The "Basic" category is currently selected. The main workspace contains a script consisting of 23 numbered lines of code:

```
1 basic.showNumber(1)
2 basic.showNumber(2)
3 basic.showNumber(3)
4 basic.showNumber(4)
5 basic.showNumber(5)
6 basic.forever(function () {
7     basic.showLeds(`
8         . . # .
9         . . # .
10        # . # . #
11        . # # # .
12        . . # .
13    `)
14    basic.showString("Hello!")
15    basic.showIcon(IconNames.Heart)
16    basic.showArrow(ArrowNames.NorthEast)
17    basic.showArrow(ArrowNames.SouthEast)
18    basic.showArrow(ArrowNames.SouthWest)
19    basic.showArrow(ArrowNames.NorthWest)
20    basic.clearScreen()
21    basic.pause(500)
22 })
23
```

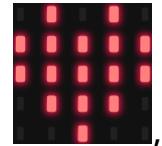
#### (4) Test Results:

Upload code 1 and power the board , we will see the icon





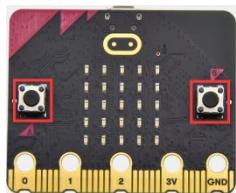
Upload code 2 and plug micro:bit in power, Micro: bit starts showing



number 1, 2, 3, 4, and 5, then cyclically display , "Hello!" ,  patterns.

([How to download?](#) [How to quick download?](#))

## Project 4: Programmable Buttons



### (1) Project Description



Buttons can be used to control circuits. In an integrated circuit

with a push button, the circuit is connected when pressing the button and it is open the other way around.

Micro: Bit main board boasts three push buttons, two are programmable



---

buttons(marked with A and B), and the one on the other side is a reset button. By pressing the two programmable buttons can input three different signals. We can press button A or B alone or press them together and the LED dot matrix shows A,B and AB respectively. Let's get started.

## (2)Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor

Import Hex profile (How to import?)

Or click "New Project" and drag blocks step by step

## (3)Test Code

### Code 1:

Press buttons on micro:bit, micro:bit will display character strings.

The route to get test code ([How to load?](#))

File Type	Path	File Name



Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 4: Programmable Buttons-1	Project 4: Programmable Buttons-1
----------	--	---

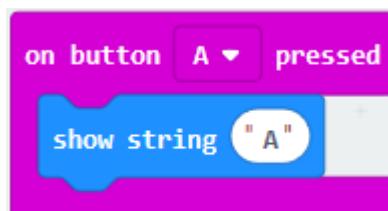
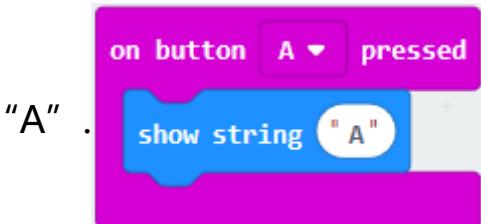
Or you could edit code step by step in the editing area.

(1) Delete “on start” and “forever” firstly, then click “Input” → “on button A pressed”

\*\*\*\*\*

(1) A. Click “Basic” → “show string” ;

B. Then place it into “on button A pressed” block, change “Hello!” into



(2) Copy code string once, tap the drop-down button



“A” to select “B” and modify character “A” into “B” .



\*\*\*\*\*



(3) Copy once , and set to "on button A+B pressed" and "show string "AB""



\*\*\*\*\*

Complete Code:



```
on button [A] pressed
  show string [A]
on button [B] pressed
  show string [B]
on button [A+B] pressed
  show string [AB]
```

Press button A on Micro: bit  
main board

Show the character "A"

Press button B on Micro: bit  
main board

Show the character "B"

Press button A and B at same  
time

Display the character "AB"

Select "JavaScript" and "Python" to switch into JavaScript and Python  
language code:

The screenshot shows the Microsoft MakeCode interface for micro:bit. At the top, there's a navigation bar with 'Blocks' (selected), 'JavaScript' (highlighted with a red arrow), and 'Python'. Below the navigation bar is a search bar labeled 'Search...'. On the left, there's a sidebar with categories: Basic, Input, Music, Led, Radio, and Loops. The main area displays the following JavaScript code:

```
1 input.onButtonPressed(Button.A, function () {
2   basic.showString("A")
3 }
4 input.onButtonPressed(Button.AB, function () {
5   basic.showString("AB")
6 }
7 input.onButtonPressed(Button.B, function () {
8   basic.showString("B")
9 })
```

**Code 2:**



The route to get test code ([How to load?](#))

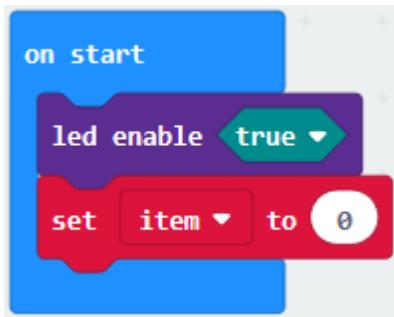
File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 4: Programmable Buttons-2	Project 4: Programmable Buttons-2

Or you could edit code step by step in the editing area.

- A. Click "Led" → "more" → "led enable false" ,
- B. Put it into the block "on start" , click drop-down triangle button to select



- \*\*\*\*\*
- A. Tap "Variables" → "Make a Variable..." → "New variable name: "
  - B. Enter "item" in the dialog box and click "OK" , then variable "item" is produced. And move "set item to 0" into "on start" block



- A. Click "Input" → "on button A pressed".
- B. Go to "Variables" → "change item by 1"
- C. Place it into "on button A pressed" and 1 is modified into 5.



\*\*\*\*\*

Duplicate  code string once, click the drop-down button to select "B", then set "change item by -5".



\*\*\*\*\*

- A. Enter "Led" → "plot bar graph of 0 up to 0"



B. Keep it into “forever” block

C. Go to “Variables” to move “item” into 0 box, change 0 into 25.



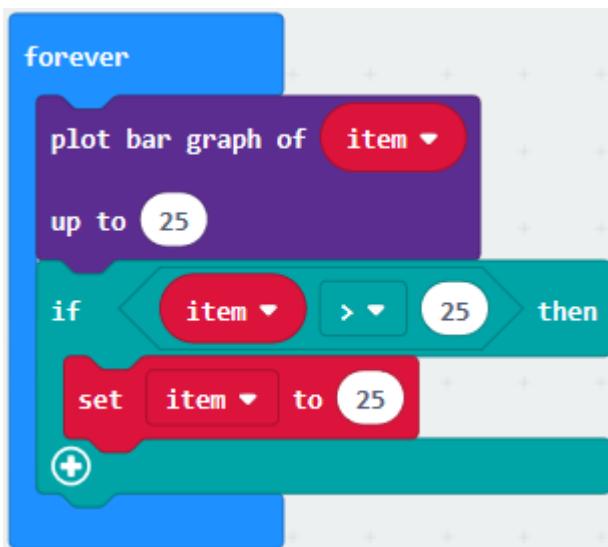
\*\*\*\*\*

A. Go to “Logic” to move out “if...true...then...” and “=” blocks,

B. Keep “=” into “true” box and set to “>”

C. Select “item” in the “Variables” and lay it down at left box of “>”,  
change 0 into 25;

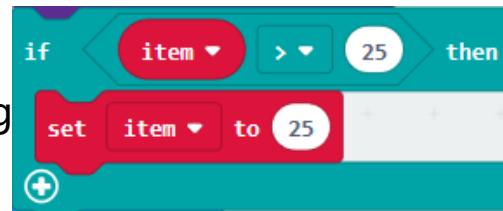
D. Enter “Variables” to drag “set item to 0” block into “if...true..then...”, alter  
0 into 25.



\*\*\*\*\*

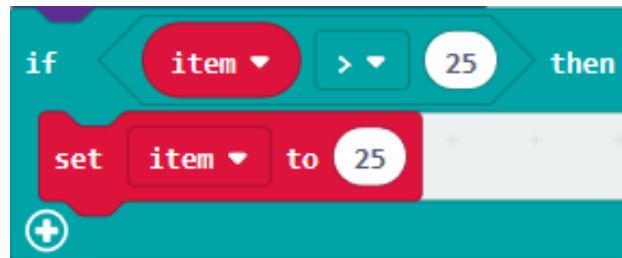


(7) A. Replicate code string



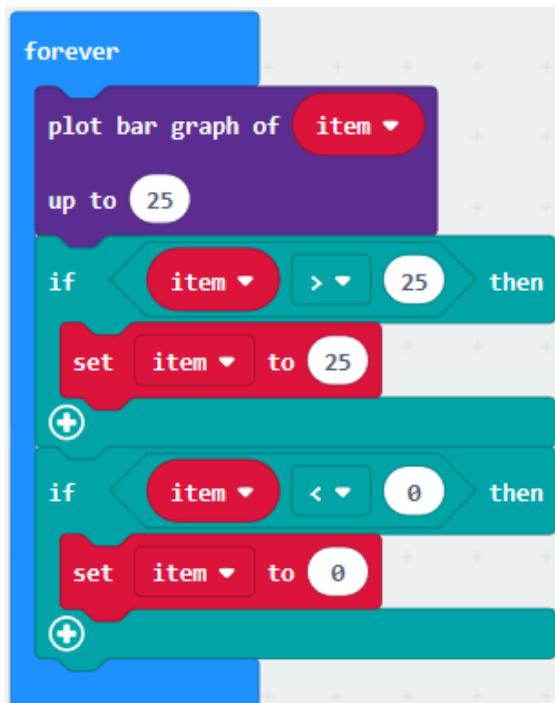
once

B. ">" is modified into "<" and 25 is changed into 0,

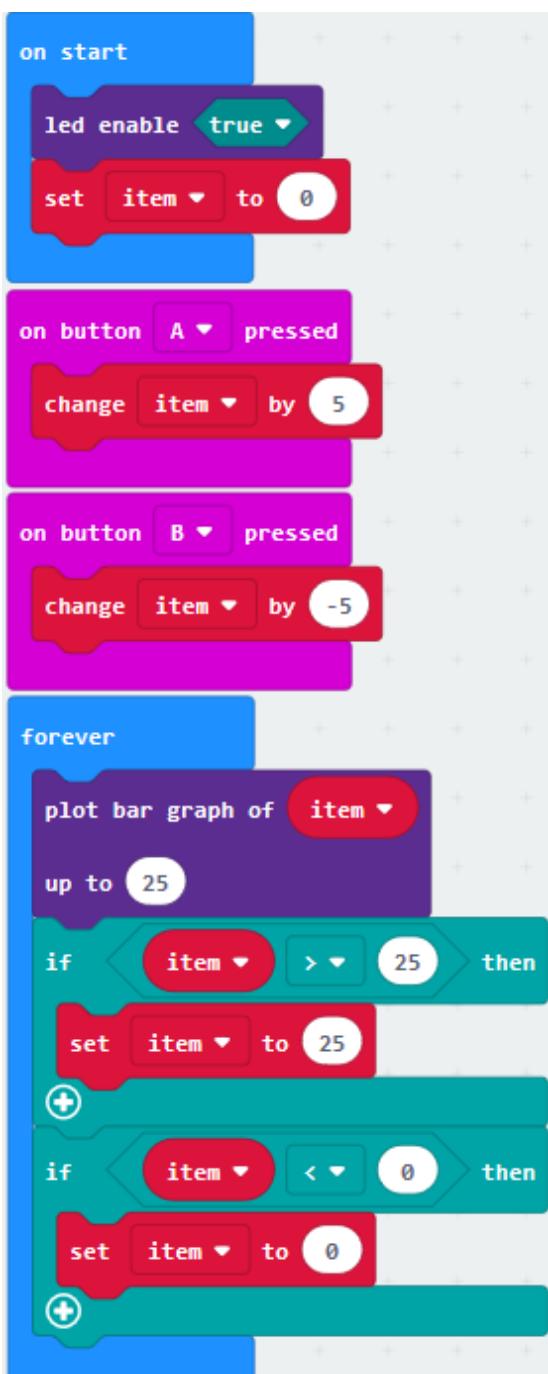


C. Leave it beneath

code string.



Complete Program:



"on start": command block runs once to start program.

Turn on LED dot matrix

Set the initial value of item to 0

Press button A on Micro:bit board

Change item by 5

Press button B on Micro:bit board

Change item by -5

The program under the block "forever" runs cyclically. Light on LED in dot matrix to draw bar graph, light up to 25 LEDs

If item is greater than 25

Then set item to 25

If item is less than 0

Then set item to 0

Click "JavaScript" to switch into JavaScript code:



The screenshot shows the Microsoft MakeCode interface for the micro:bit. At the top, there's a navigation bar with 'Blocks' and 'JavaScript' tabs, where 'JavaScript' is highlighted with a red arrow pointing to it. To the right of the tabs are help, settings, and Microsoft logos. Below the navigation bar is a search bar labeled 'Search...'. On the left, a vertical palette lists categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, and Advanced. The 'Advanced' category is currently expanded. The main area contains the following JavaScript code:

```
1 input.onButtonPressed(Button.A, function () {
2     item += 5
3 })
4 input.onButtonPressed(Button.B, function () {
5     item += -5
6 })
7 let item = 0
8 led.enable(true)
9 basic.forever(function () {
10     led.plotBarGraph(
11         item,
12         25
13     )
14     if (item > 25) {
15         item = 25
16     }
17     if (item < 0) {
18         item = 0
19     }
20 })
21
```

#### (4) Test Results:

After uploading test code 1 to micro:bit main board V2 and powering the main board via the USB cable, the 5\*5 LED dot matrix shows A if button A is pressed, B if button B pressed, and AB if button A and B pressed together.

After uploading test code 2 to micro:bit main board V2 and powering the main board via the USB cable, when pressing the button A the LEDs turning red increase by 5 while when pressing the button B the LEDs turning red reduce.



([How to download?](#) [How to quick download?](#))

## Project 5: Temperature Measurement

### (1) Project Description

The Micro:bit main board V2 is not equipped with a temperature sensor, but uses the temperature sensor built into NFR52833 chip for temperature detection. Therefore, the detected temperature is more closer to the temperature of the chip, and there maybe deviation from the ambient temperature.

Note: the temperature sensor of Micro:bit main board is shown below:



### (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor



Import Hex profile (How to import?)

Or click "New Project" and drag blocks step by step.

### (3) Test Code

#### Code 1:

Micro:bit detects temperature

The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 5: Temperature Measurement-1	Project 5: Temperature Measurement-1

Or you could edit code step by step in the editing area.

Go to "Advanced" → "Serial" → "serial redirect to USB"

Place it into "on start"



\*\*\*\*\*

Click "Serial" to drag out "serial write value x=0"

Move it into "forever" block



Go to "Input" → "temperature(°C)"

Place it into 0 box

Change x into Temperature

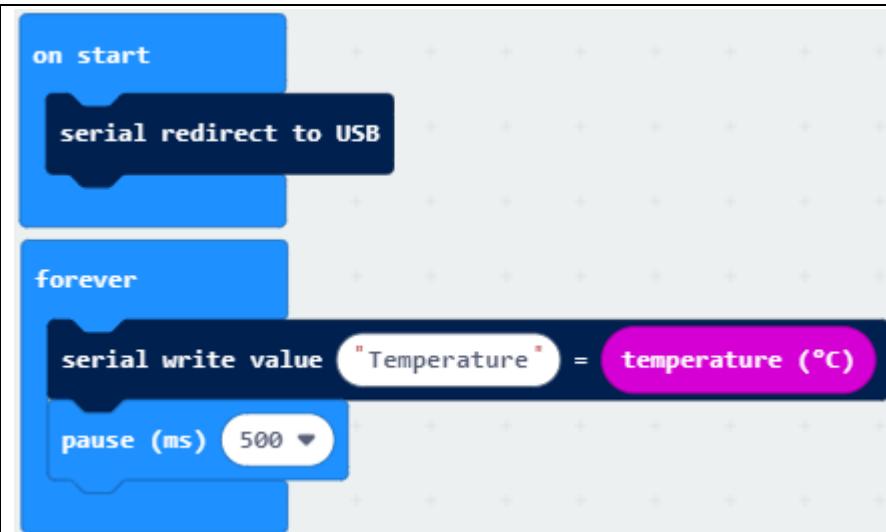


\*\*\*\*\*

Move "pause (ms) 100" from "Basic" block and place it under block "serial write....temperature(°C)"



Complete Program:



"on start" : command block runs once to start program.

Serial redirect to USB

The program under the block "forever" runs cyclically.

Serial writes Temperature

Delay in 500ms

Click "JavaScript" to view the corresponding JavaScript code:

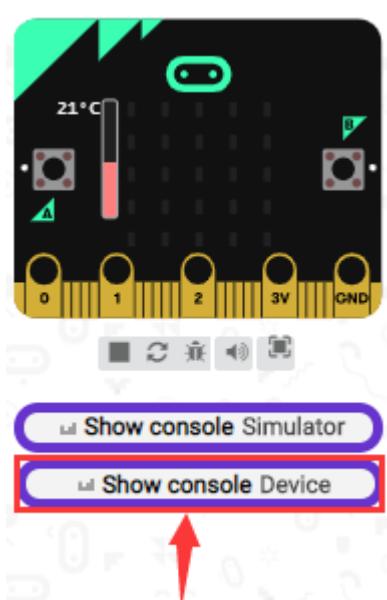


The image shows the Scratch interface with the code editor open. The language dropdown menu at the top is set to "JavaScript". A red arrow points to this dropdown menu. The code in the script is:

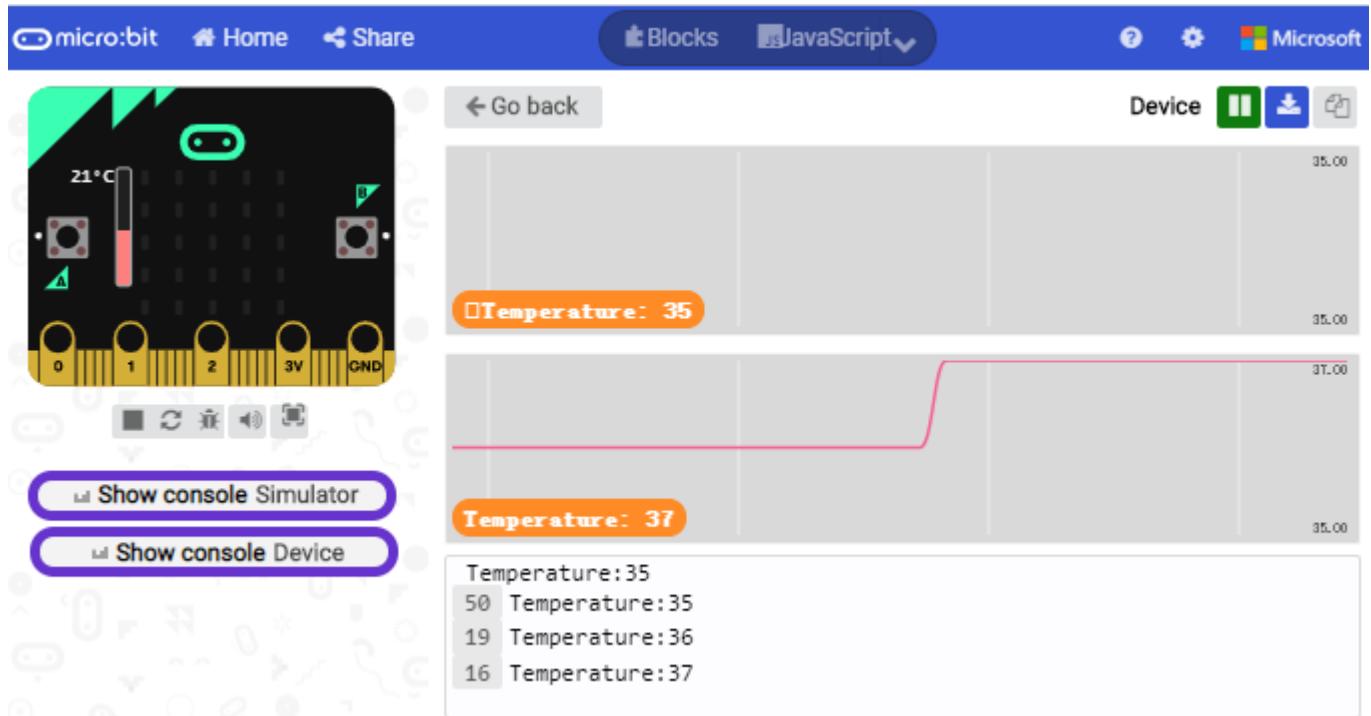
```
1 serial.redirectToUSB()
2 basic.forever(function () {
3     serial.writeValue("Temperature", input.temperature())
4     basic.pause(500)
5 })
6
```

Download code 1 to micro:bit board and keep USB cable connected, then tap button :

( How to quick download?)



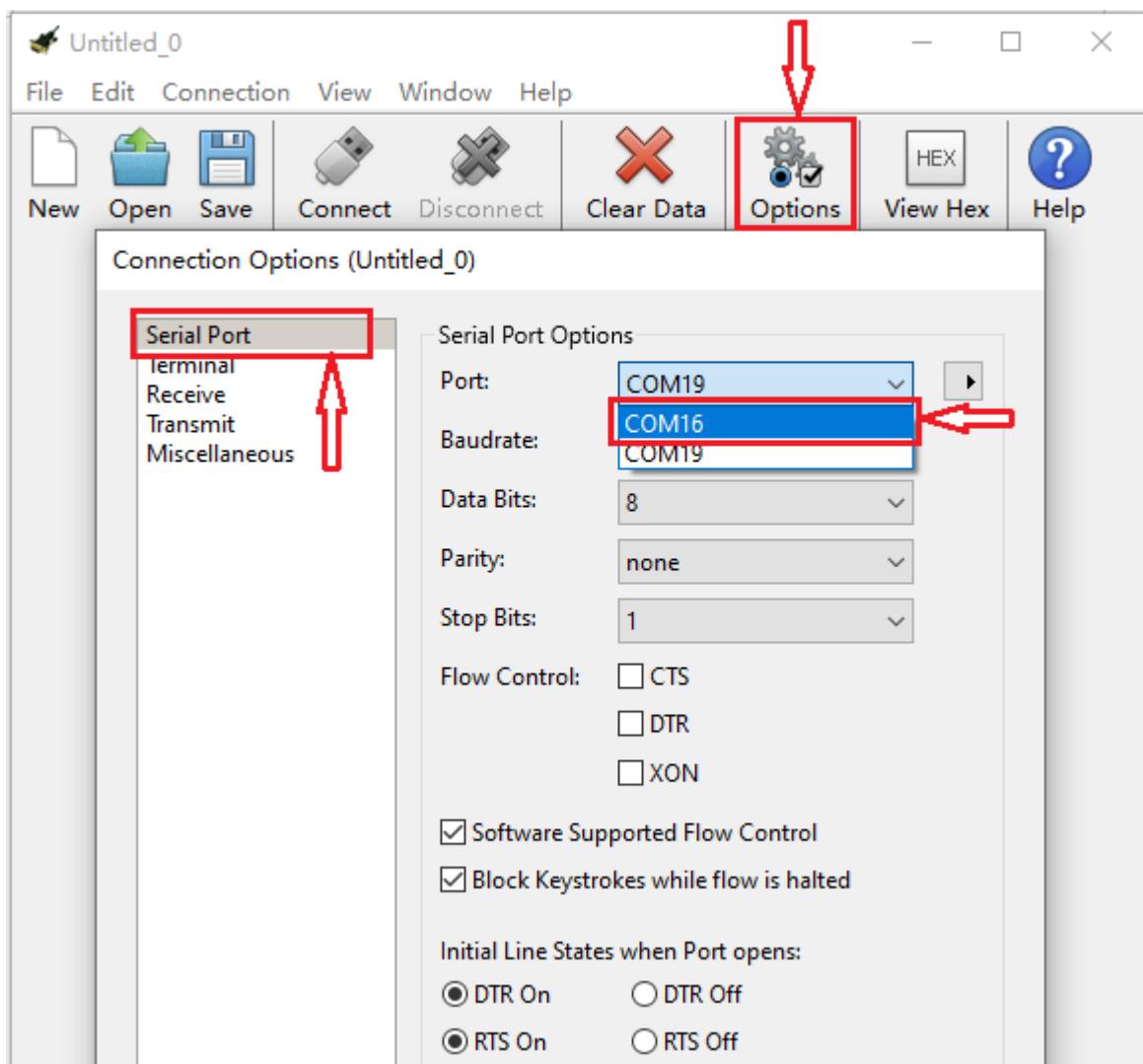
Temperature data is shown below:

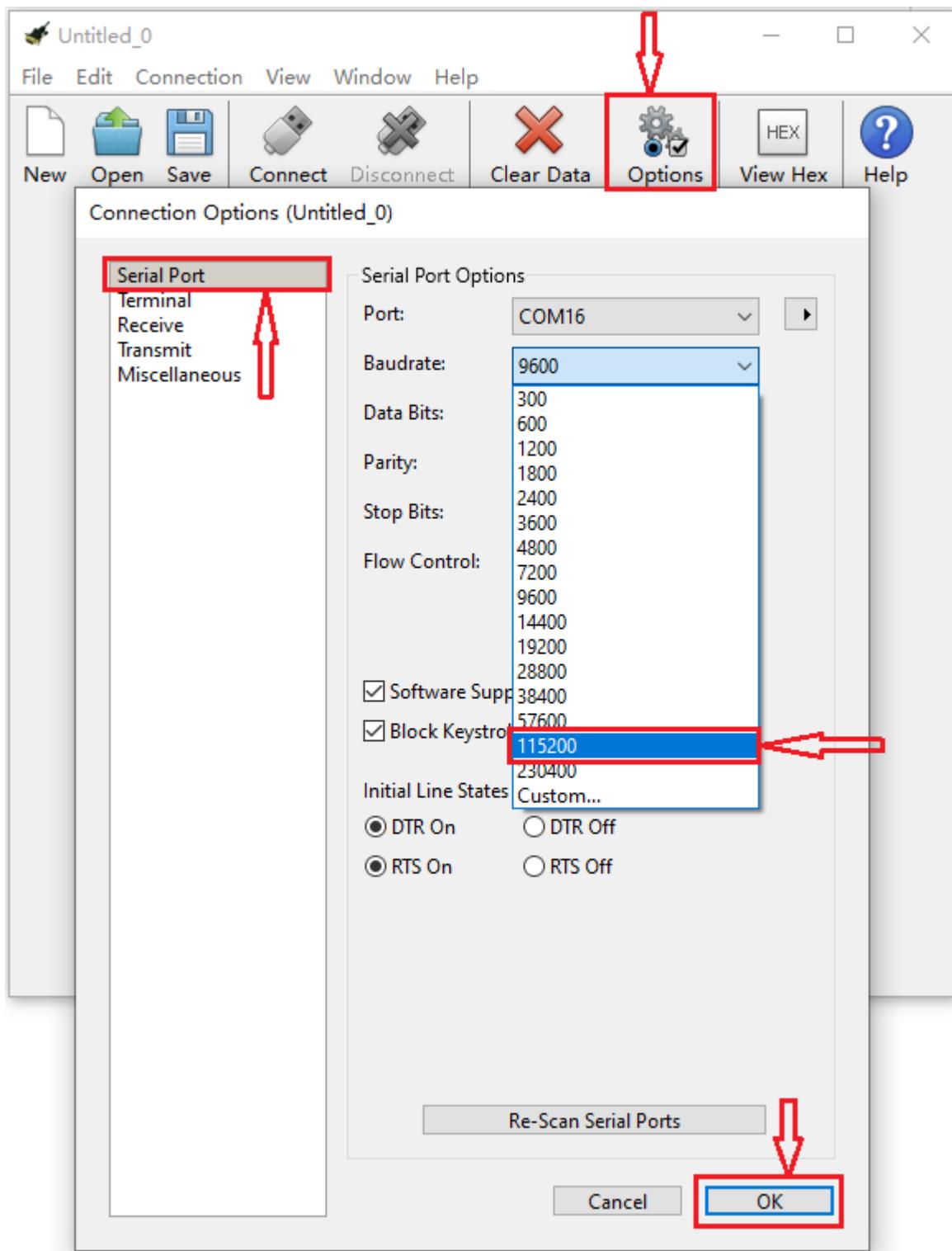


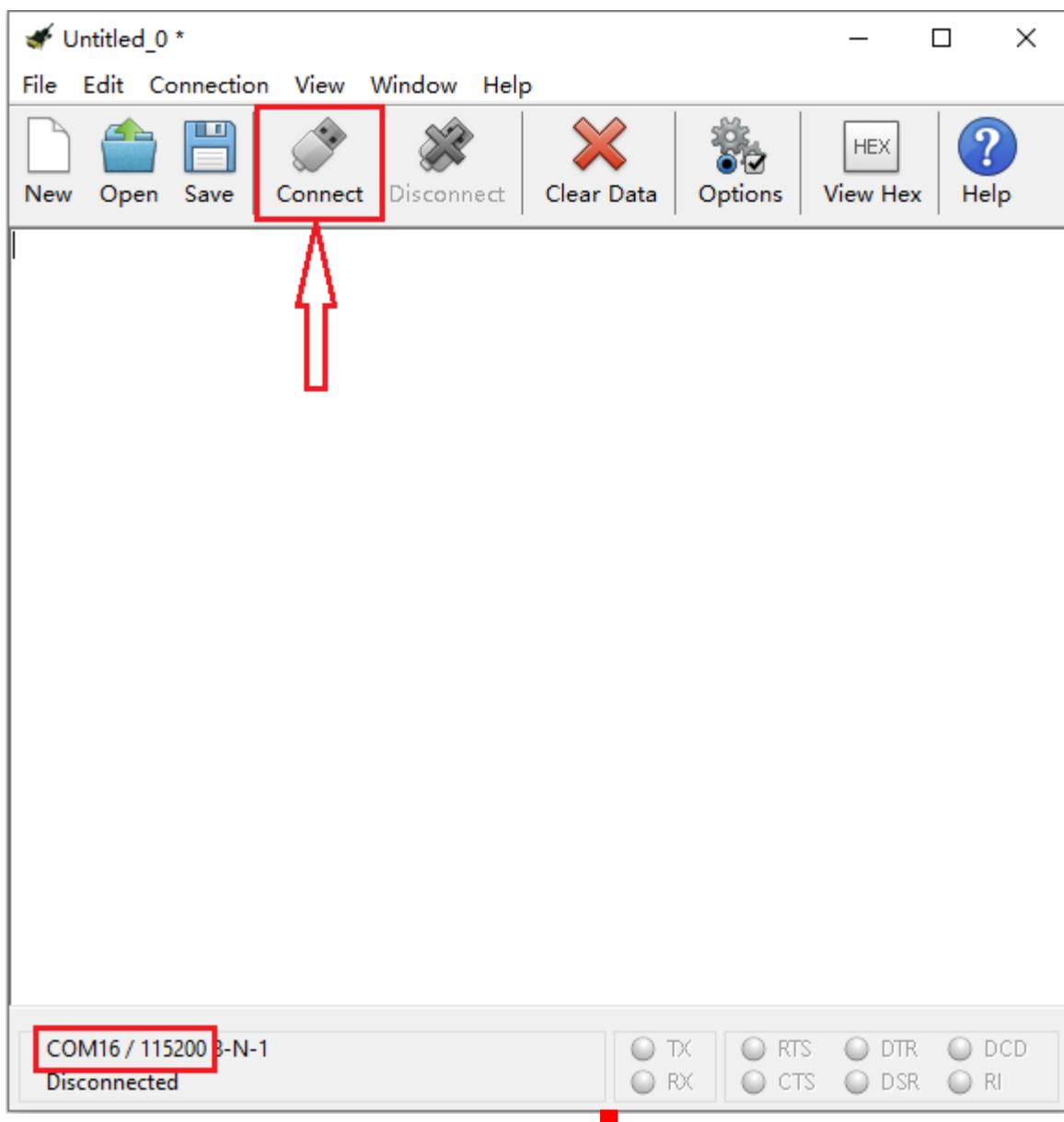
Through the test, the room temperature is 35 °C when touching the NFR51822 chip of micro:bit; however, the temperature rises to 37°C when it touches water cup.

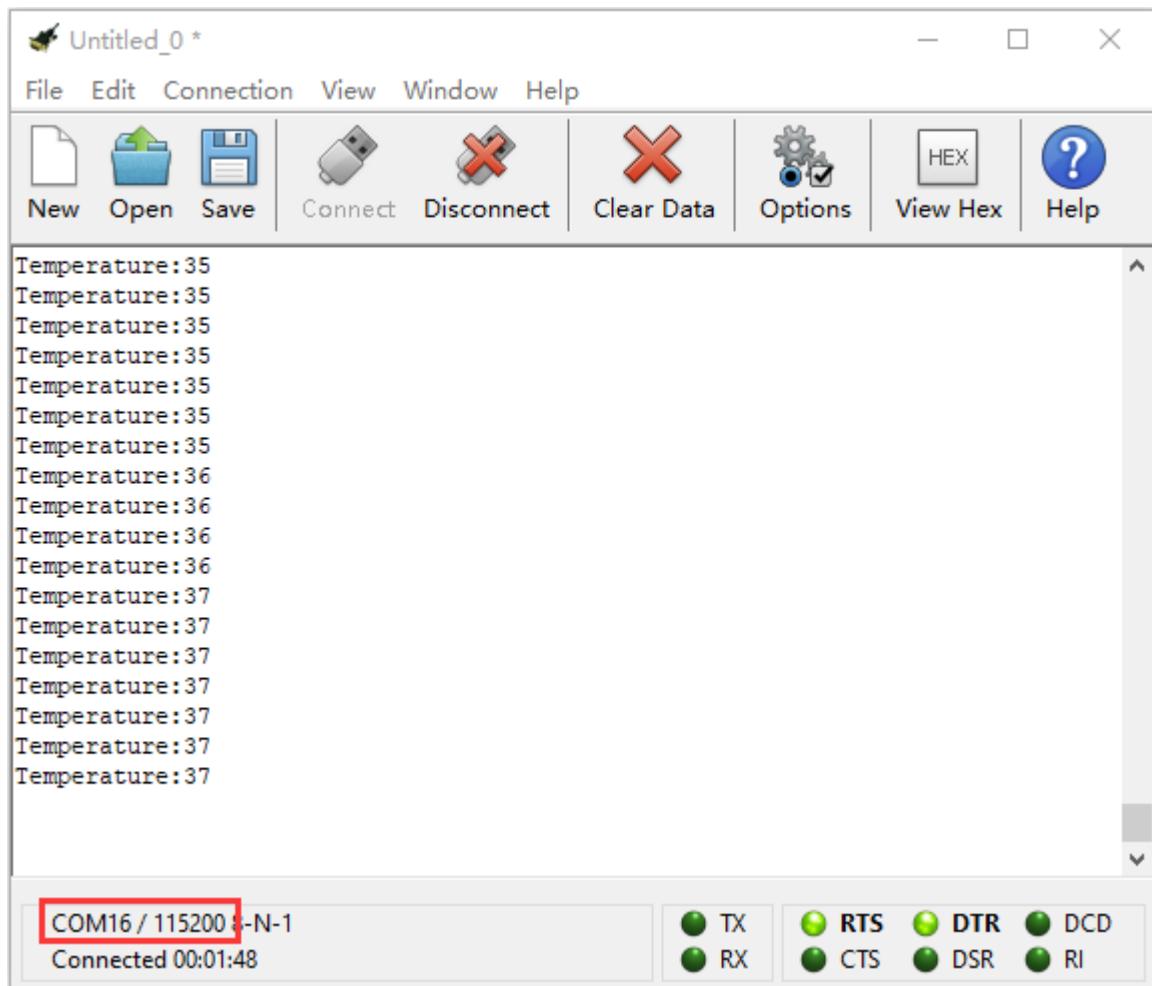
Open CoolTerm, click Options to select SerialPort. Set COM port and 115200 baud rate(the baud rate of USB serial communication of Micro:bit is 115200 through the test). Click “OK” and “Connect” .

The serial monitor shows the current ambient temperature value, as shown below:









## Code 2:

Micro:bit display different pictures by temperature(the temperature value in the code could be adjusted).

The route to get test code ([How to load?](#))

File Type	Path	File Name



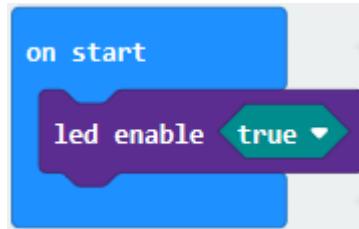
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 5: Temperature Measurement-2	Project 5: Temperature Measurement-2
----------	---	--------------------------------------

Or you could edit code step by step in the editing area.

You could set temperature based on real situation.

Click “Led” → “more” → “led enable false” into “on start” , click drop-down

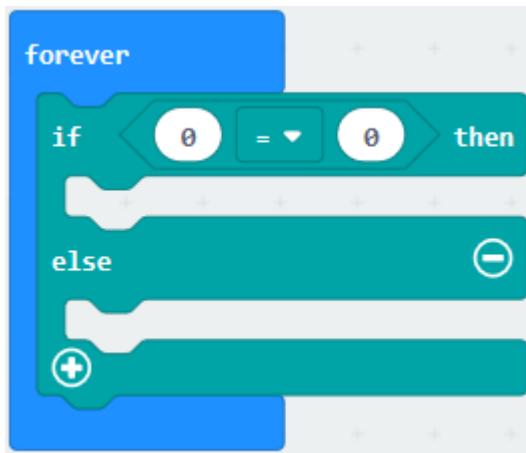
triangle button to select “true”



\*\*\*\*\*

A. Go to “Logic” → “if..true...then...else” and “=” block;

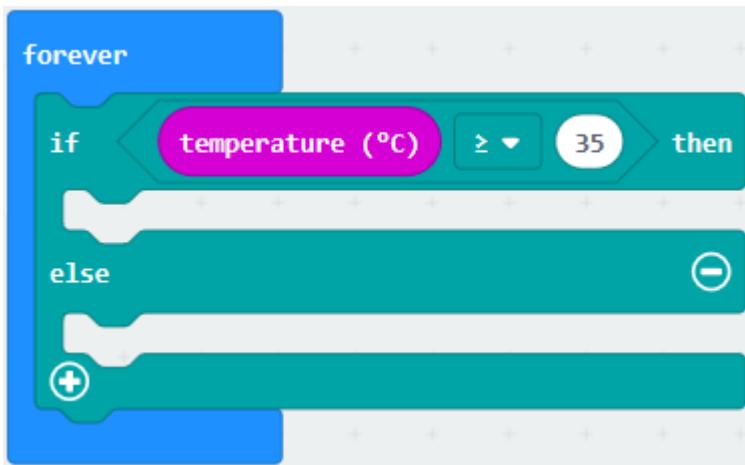
B. Move “if..true...then...else” into “forever” block, then place “=” into “true” box.



\*\*\*\*\*



- A. Change “=” into “≥”
- B. Go to “Input” → “temperature(°C)” and move it into left 0 box;
- C. Change 0 into 35.



\*\*\*\*\*

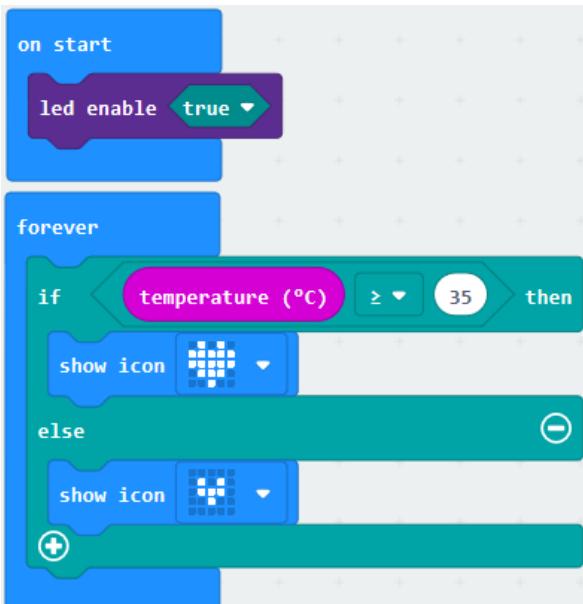
Tap “Basic” → “show icon” , copy it once and lay down them under the “if ...then” and else blocks, then click the drop-down triangle button to

select “” .





## Complete Program:



“on start” : command block runs once to start program.

Turn on LED dot matrix

Under the block “forever” , program runs cyclically.

If the detected temperature  $\geq 35^\circ$ , the next program will be executed.

Dot matrix shows “”

Click “JavaScript”, the corresponding JavaScript code is shown below:

```
1 led.enable(true)
2 basic.forever(function () {
3     if (input.temperature() >= 35) {
4         basic.showIcon(IconNames.Heart)
5     } else {
6         basic.showIcon(IconNames.SmallHeart)
7     }
8 })
9
```



#### (4) Test Results:

Upload the Code 1 and plug in power. And 5\*5LED displays the ambient temperature. When pressing the temperature sensor, the temperature will grow on dot matrix.

Upload the code 2 plug in micro:bit via USB cable, when the ambient



temperature is less than 35 °C, 5\*5LED will show [red LED pattern]. When the



temperature is equivalent to or greater than 35°C, the pattern [red LED pattern] will appear.

([How to download?](#) [How to quick download?](#))

## Project 6: Geomagnetic Sensor



#### (1) Project Description

This project mainly introduces the use of the Micro:bit's compass. In addition to detecting the strength of the magnetic field, it can also be used to determine the direction, an important part of the heading and attitude reference system (AHRS) as well.



---

It uses FreescaleMAG3110 three-axis magnetometer. Its I2C interface communicates with the outside, the range is  $\pm 1000\mu\text{T}$ , the maximum data update rate is 80Hz. Combined with accelerometer, it can calculate the position. Additionally, it is applied to magnetic detection and compass blocks.

Then we could read the value detected by it to determine the location. We need to calibrate the Micro:bit board when magnetic sensor works.

The correct calibration method is to rotate the Micro:bit board.

In addition, the objects nearby may affect the accuracy of readings and calibration.

## (2)Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor

Import Hex profile ([How to import?](#))

Or click “New Project” and drag blocks step by step

## (3)Test Code

### Code 1:

Press A on micro:bit, the value of compass is shown.

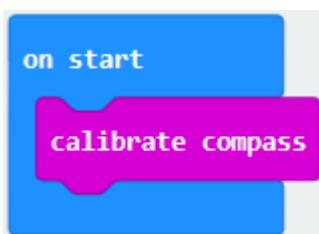
The route to get test code ([How to load?](#))



File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 6: Geomagnetic Sensor-1	Project 6: Geomagnetic Sensor-1

Or you could edit code step by step in the editing area.

- Click "Input" → "more" → "calibrate compass"
- Lay down it into block "on start" .



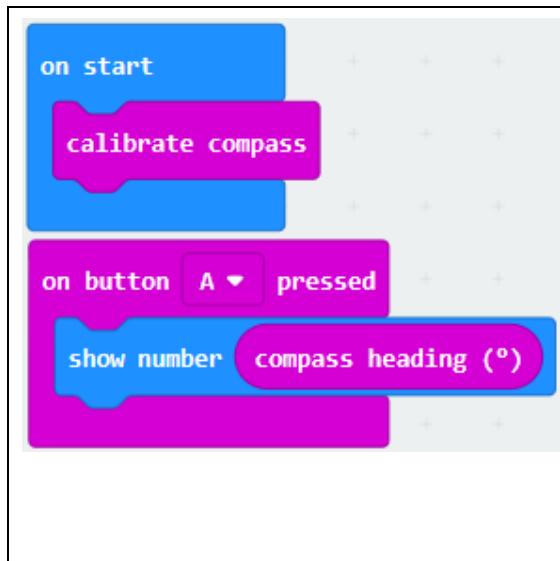
- Go to "Input" → "on button A pressed" .
- Enter "Basic" → "show number" , put it into "on button A pressed" block;
- Tap "Input" → "compass heading(°C)" , and place it into "show number"



\*\*\*\*\*



## Complete Program:

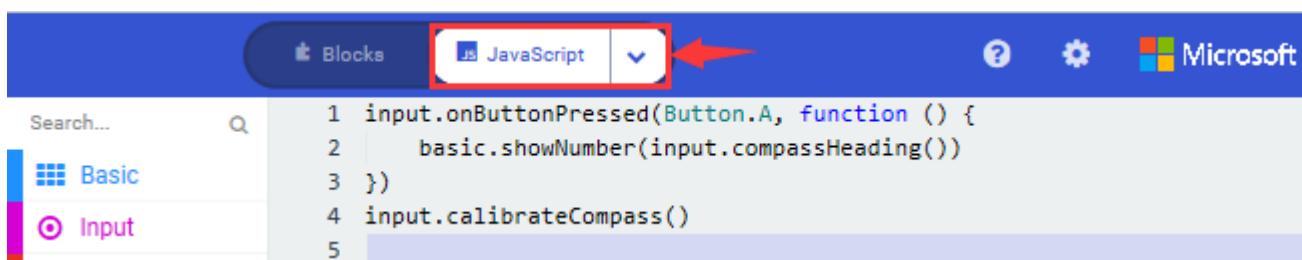


The Scratch program consists of two scripts:

- on start:** A blue command block followed by a magenta **calibrate compass** block.
- on button A pressed:** A purple control block followed by a blue **show number compass heading (°)** block.

① "on start": command block only runs once to start program.  
② Calibrate compass  
③ Press button A on Micro:bit main board  
④ Dot matrix shows the direction of compass heading

Select "JavaScript" and "Python" to switch into JavaScript and Python language code:



The Scratch interface shows the following code in the JavaScript tab:

```
1 input.onButtonPressed(Button.A, function () {
2     basic.showNumber(input.compassHeading())
3 }
4 input.calibrateCompass()
5 }
```

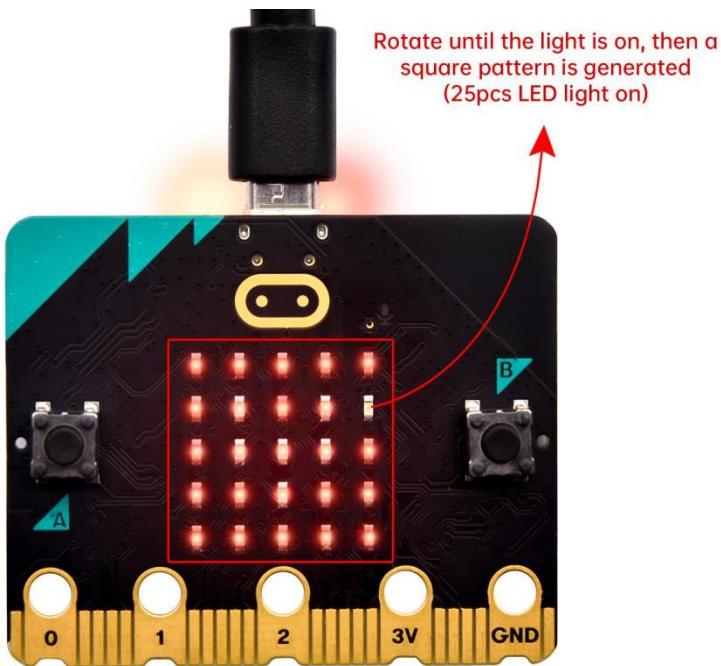
## Code Description:

Upload the code 1, plug in micro:bit via USB cable.

As the button A is pressed, LED dot matrix indicates that "TILT TO FILL SCREEN" then enter the calibration interface. The calibration method: rotate the micro:bit to make LED dot matrix draw a square (25 LEDs are on), as shown in the following figure:



(How to download? How to quick download?)



The calibration will be finished until you view the smile pattern



appear.

The serial monitor will show  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$  and  $270^\circ$  when pressing A.

## Code 2:

Make micro: bit board point to the north, south, east and west horizontally ,  
LED dot matrix displays the corresponding direction patterns

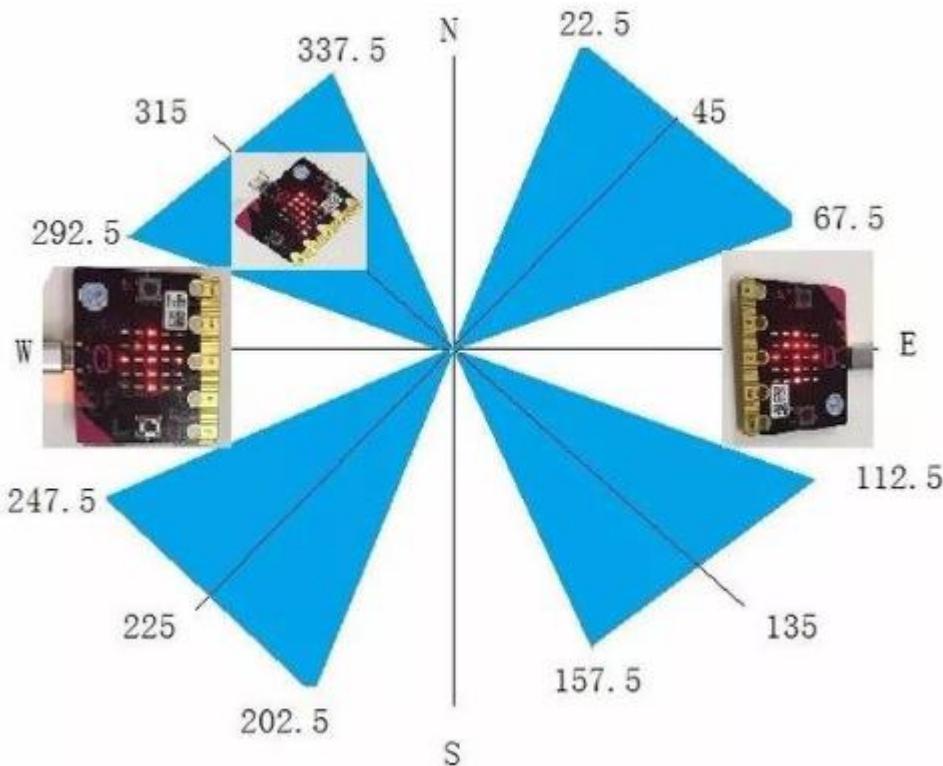
The route to get test code ([How to load?](#))



File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 6: Geomagnetic Sensor-2	Project 6: Geomagnetic Sensor-2



This module can keep reading data to determine direction, so does point to the current magnetic North Pole by arrow.

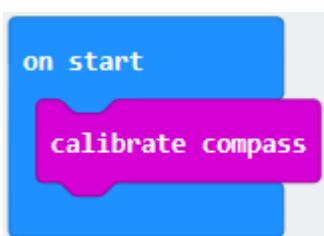


For the above picture, the arrow pointing to the upper right when the value ranges from 292.5 to 337.5. Because 0.5 can't be input in the code, the values we get are 293 and 338.

Link computer with micro:bit board by micro USB cable, and program in MakeCode editor:

Enter "Input" → "more" → "calibrate compass"

Move "calibrate compass" into "on start"





\*\*\*\*\*

- A. Click "Variables" → "Make a Variable..." → "New variable name: "
- B. Input "x" in the blank box and click "OK" , and the variable "x" is generated.
- C. Drag out "set x to" into "forever" block



- A. Go to "Input" → "compass heading(°C)" , and keep it into "0" box



Tap "Logic" → "if...then...else" , leave it below block "sex x to compass heading" , then click  icon for 6 times.

\*\*\*\*\*

- A. Place "and" into "true" block
- B. Then move "=" block to the left box of "and"
- C. Click "Variables" to drag "x" to the left "0" box, change 0 into 293 and set to " $\geq$ " ;
- D. Then copy " $x \geq 293$ " once and leave it to the right "0" box and set to " $x < 338$ "



```
forever
  set [x v] to [compass heading (°)]
  if [x ≥ 293 and x ≤ 338] then
    [ ]
  else if [ ] then
    [ ]
  else
    [ ]
  end
```

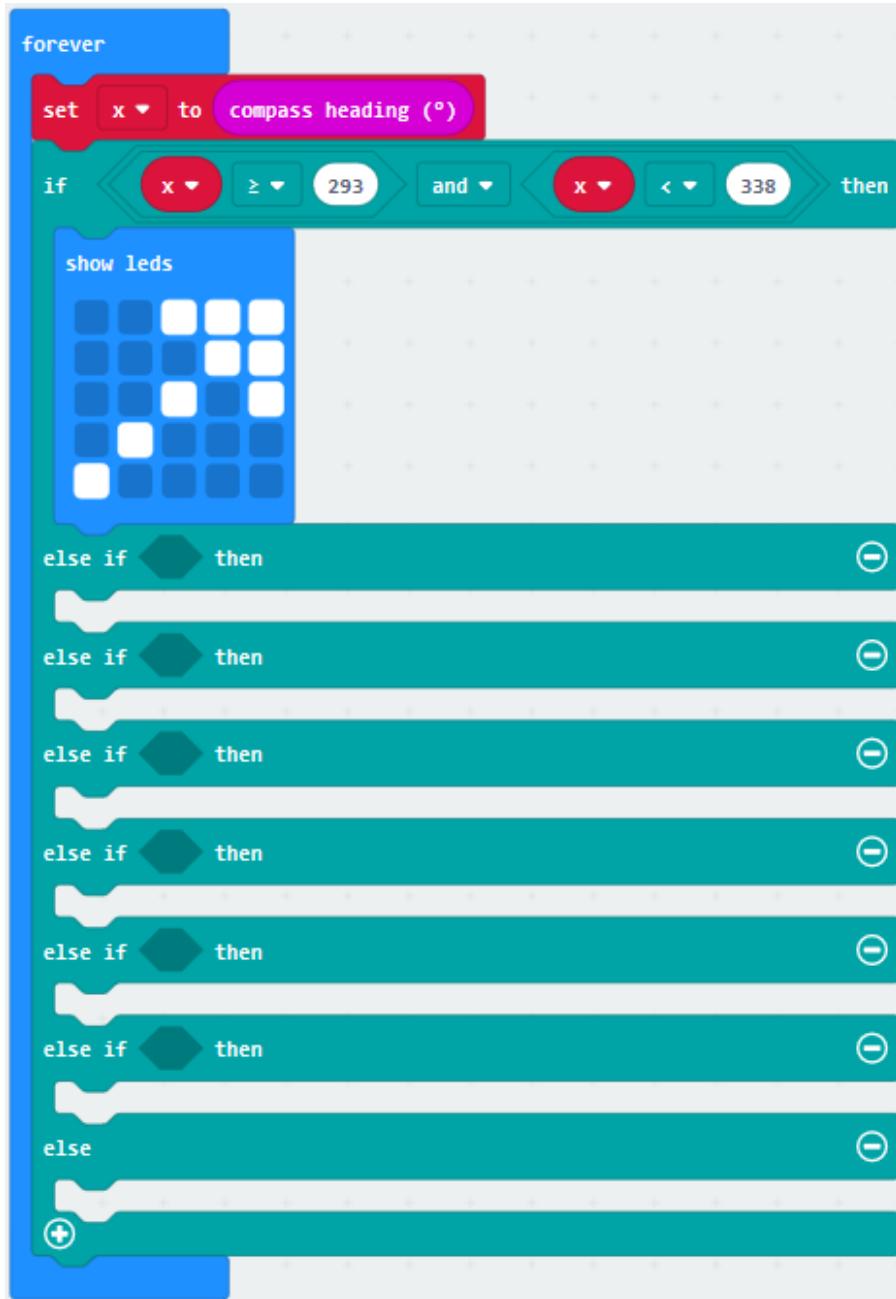
\*\*\*\*\*

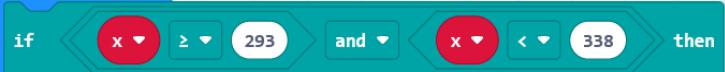
A. Go to "Basic" → "show leds"

B. Lay it down beneath



click "show leds" and the pattern appears.



A. Duplicate  for 6 times.

B. Separately leave them into the blank boxes behind "else if" .

C. Set to "x≥23 and x<68" , "x≥68 and x<113" , "x≥113 and x<158" ,



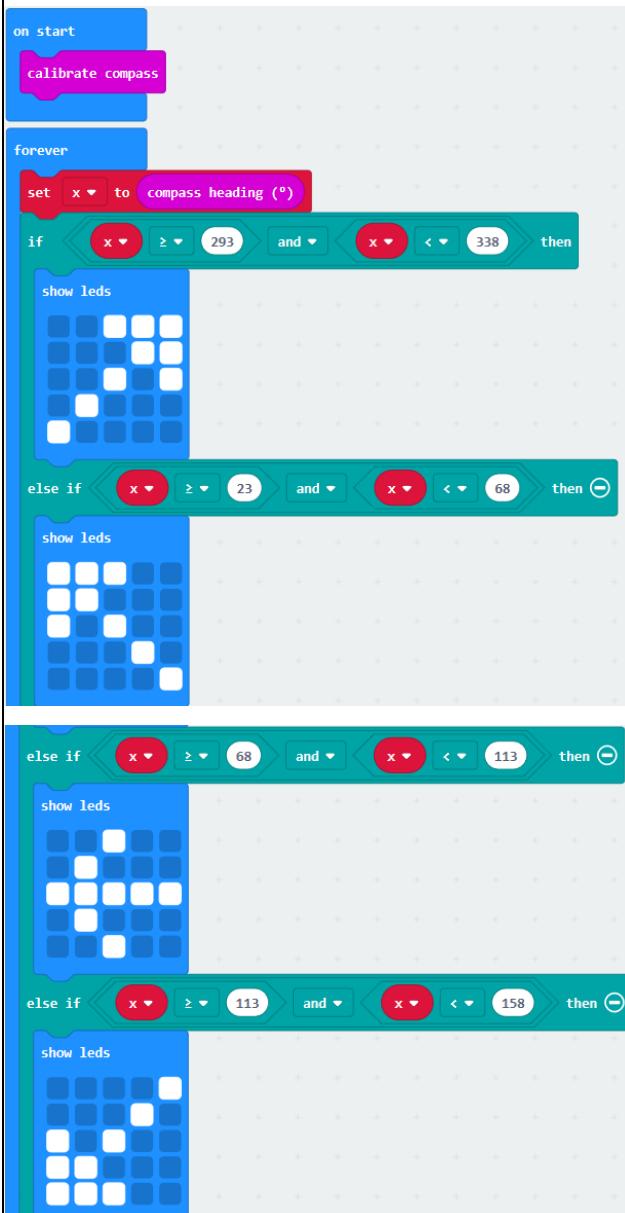
" $x \geq 158$  and  $x < 203$ " , " $x \geq 203$  and  $x < 248$ " , " $x \geq 248$  and  $x < 293$ " respectively.

D. Then copy "show leds" for 7 times and keep them below the "else if.....then" block respectively.

E. Click the blue boxes to form the pattern "" , "" , "" , "" , "" , "" and "".

\*\*\*\*\*

Complete Program:



"on start": command block only runs once to start program.

Calibrate compass

The program under the block "forever" runs cyclically.

Store the angle of the compass heading into the variable **x**

When  $293 \leq x < 338$ , the next program will be executed



appears on the dot matrix

When  $23 \leq x < 68$ , the next program will be executed



is displayed on dot matrix

When  $68 \leq x < 113$ , the next program will be executed

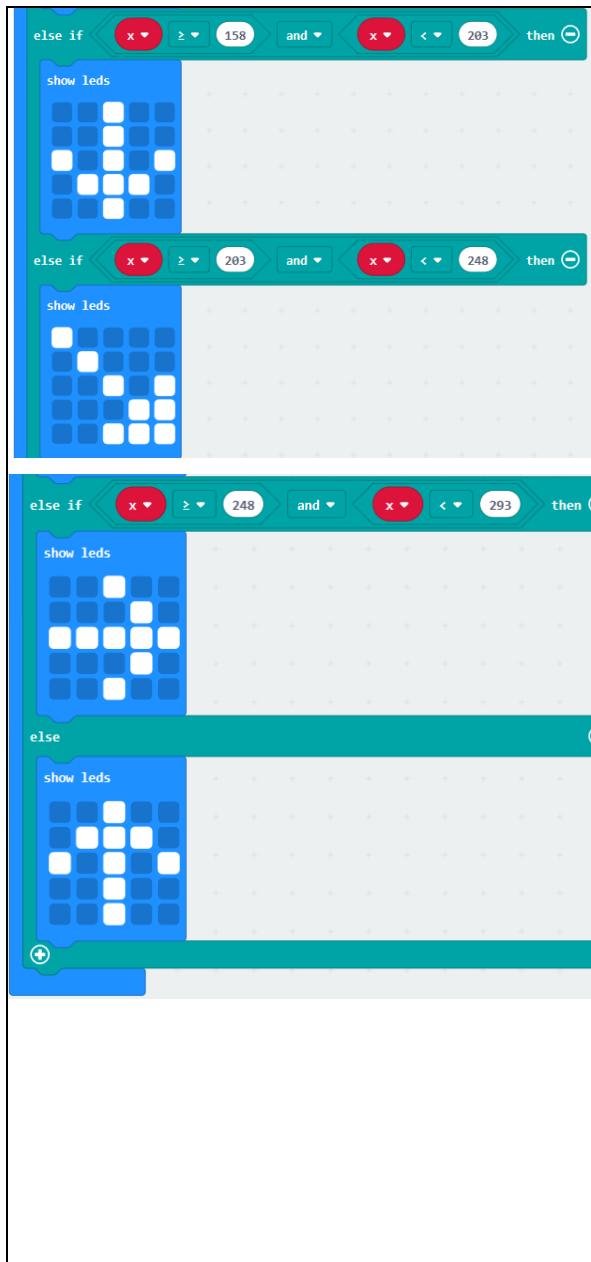


is shown on dot matrix

When  $113 \leq x < 158$ , the next program will be executed



pattern appears



```
else if [x >= 158 and x < 203] then
    show leds [dot matrix v1]
else if [x >= 203 and x < 248] then
    show leds [dot matrix v2]
else
    show leds [dot matrix v3]
```

When  $158 \leq x < 203$ , the next program will be executed.

Dot matrix shows 

When  $203 \leq x < 248$ , the next program will be executed.

Dot matrix displays 

When  $248 \leq x < 293$ , the next program will be executed.

Dot matrix shows 

When  $x$  is not among the above rang, the next program will be executed under else block

Select "JavaScript" and "Python" to switch into JavaScript and Python language code:



Blocks     JavaScript    

Search...    

Basic
Input
Music
Led
Radio
Loops
Logic
Variables
Math
Advanced

```
1 let x = 0
2 input.calibrateCompass()
3 basic.forever(function () {
4     x = input.compassHeading()
5     if (x >= 293 && x < 338) {
6         basic.showLeds(`
7             . . # # #
8             . . . # #
9             . . # . #
10            . # . . .
11            # . . . .
12            `)
13     } else if (x >= 23 && x < 68) {
14         basic.showLeds(`
15             # # # . .
16             # # . . .
17             # . # . .
18             . . . # .
19             . . . . #
20             `)
21     } else if (x >= 68 && x < 113) {
22         basic.showLeds(`
23             . . # . .
24             . # . . .
25             # # # # #
26             . # . . .
27             . . # . .
28             `)
29     } else if (x >= 113 && x < 158) {
30         basic.showLeds(`
31             . . . . #
32             . . . # .
33             # . # . .
34             # # . . .
35             # # # . .
36             `)
```



```
37     } else if (x >= 158 && x < 203) {
38         basic.showLeds(`
39             ...
40             ...
41             # . # . #
42             . # # # .
43             ...
44             `)
45     } else if (x >= 203 && x < 248) {
46         basic.showLeds(`
47             #
48             ...
49             .
50             ...
51             ...
52             `)
53     } else if (x >= 248 && x < 293) {
54         basic.showLeds(`
55             ...
56             ...
57             # # # # #
58             ...
59             ...
60             `)
61     } else {
62         basic.showLeds(`
63             ...
64             ...
65             # . # . #
66             ...
67             ...
68             `)
69     }
70 })
71 }
```

#### (4) Test Results:

Upload code 2 and plug micro:bit into power. After calibration, tilt micro:bit board, and the LED dot matrix displays the direction signs.

([How to download?](#) [How to quick download?](#))



## Project 7: Accelerometer



### (1) Project Description

The micro:bit board has a built-in Freescale MMA8653FC three-axis acceleration sensor (accelerometer). Its I<sub>2</sub>C interface works on external communication, the range can be set to  $\pm 2g$ ,  $\pm 4g$ , and  $\pm 8g$ , and the maximum data update rate can reach 800Hz.

When the Micro:bit is stationary or moving at a constant speed, the accelerometer only detects the gravitational acceleration; when the Micro:bit is slightly shaken, the acceleration detected is much smaller than the gravitational acceleration and can be ignored. Therefore, in the process of using Micro:bit, the main purpose is to detect the changes of the gravitational acceleration on the x, y, and z axes when the attitude changes.

For this project, we will introduce the detection of several special postures by the accelerometer.

### (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor



Import Hex profile (How to import?)

Or click "New Project" and drag blocks step by step

### (3) Test Code

**Code 1:**

The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 7: Accelerometer-1	Project 7: Accelerometer-1

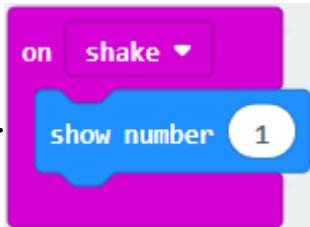
Or you could edit code step by step in the editing area.

(1) A. Enter "Input" → "on shake" ,

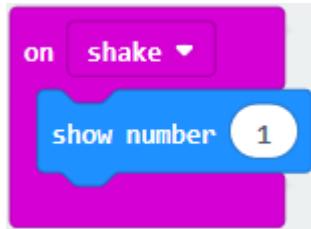
B. Click "Basic" → "show number", place it into "on shake" block, then change



0 into 1.



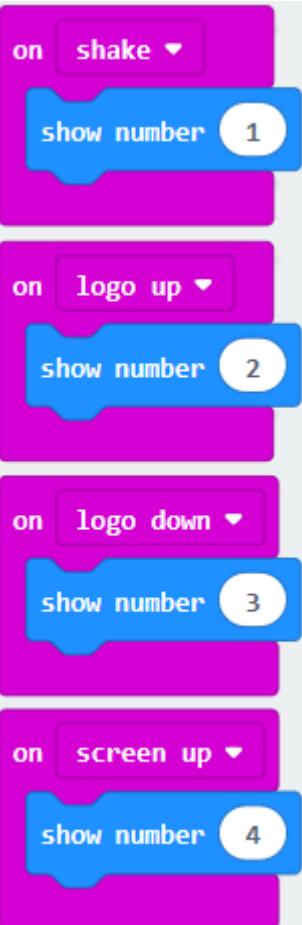
(2) A. Copy code string



for 7 times;

separately click the triangle button to select “logo up”, “logo down”, “screen up”, “screen down”, “tilt left”, “tilt right” and “free fall”, then respectively change 1 into 2, 3, 4, 5, 6, 7, 8.

\*\*\*\*\*  
Complete Program:



Shake the Micro:bit board

LED dot matrix displays 1

The log is up

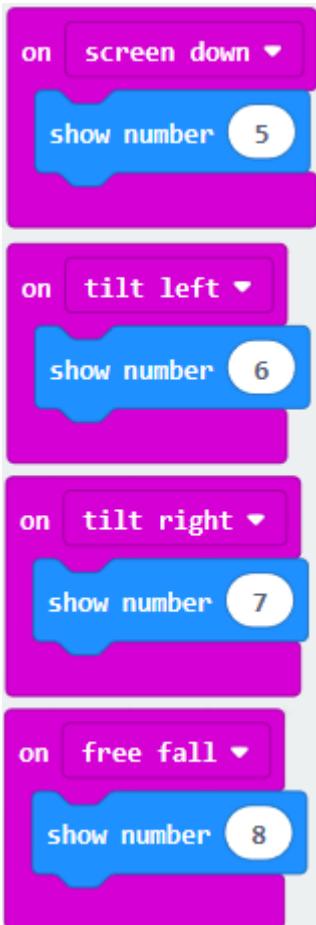
LED dot matrix displays 2

The logo is down

LED dot matrix displays 3

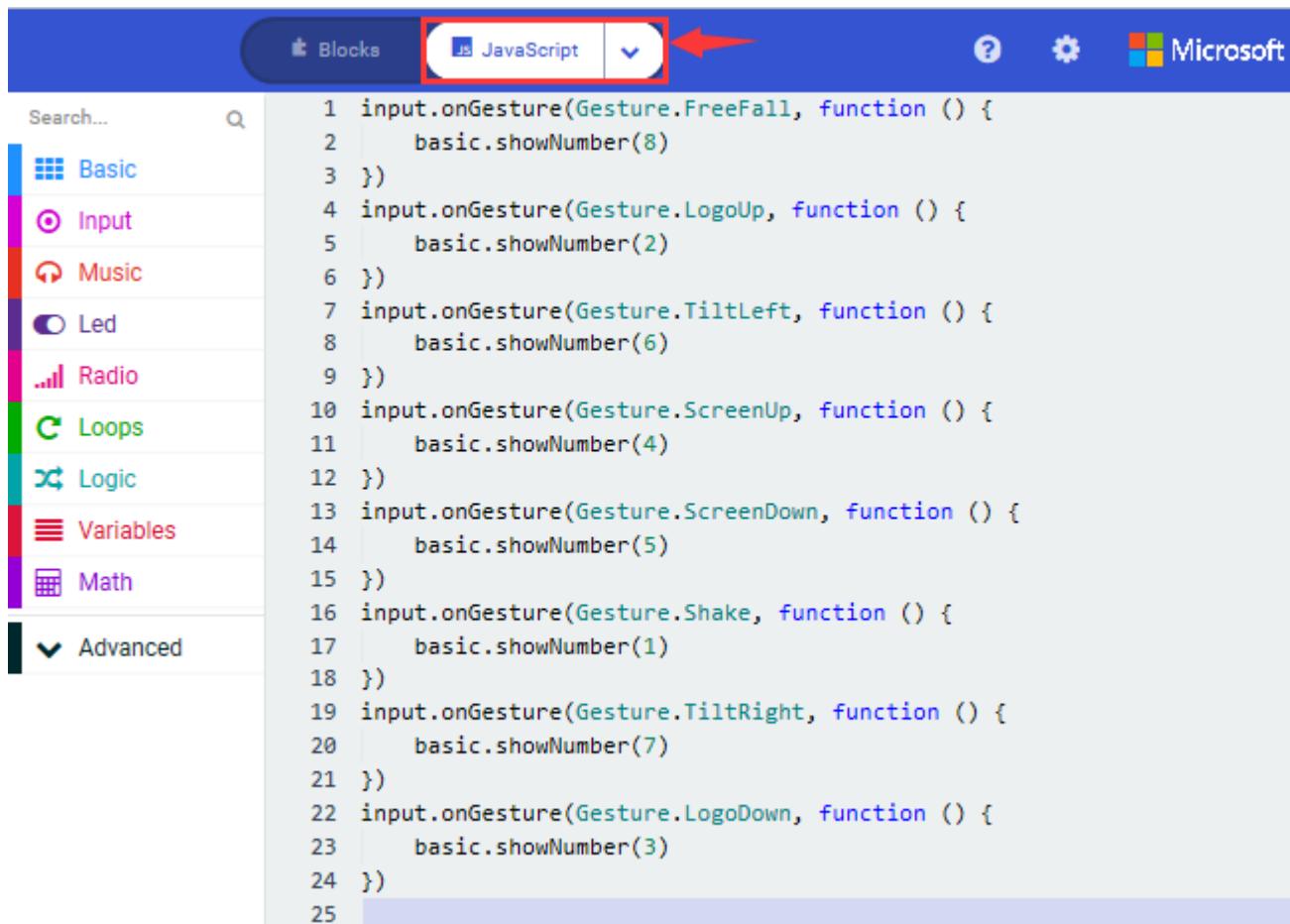
The screen is up

LED dot matrix displays 4



The screen is down  
Number 5 is shown  
The Micro:bit board is tilt to the left  
Number 6 is displayed  
The Micro:bit board is tilt to the right  
Number7 is displayed  
When the Micro:bit board is free fall  
LED dot matrix shows 8

Click "JavaScript", you will view the corresponding JavaScript code:



The screenshot shows the Microsoft Makecode interface. At the top, there are tabs for 'Blocks' and 'JavaScript'. A red arrow points to the 'JavaScript' tab, which is highlighted. Below the tabs is a search bar labeled 'Search...' with a magnifying glass icon. To the right of the search bar is a gear icon for settings. On the far right, the Microsoft logo is visible. The main area contains the following JavaScript code:

```
1 input.onGesture(Gesture.FreeFall, function () {
2     basic.showNumber(8)
3 }
4 input.onGesture(Gesture.LogoUp, function () {
5     basic.showNumber(2)
6 }
7 input.onGesture(Gesture.TiltLeft, function () {
8     basic.showNumber(6)
9 }
10 input.onGesture(Gesture.ScreenUp, function () {
11     basic.showNumber(4)
12 })
13 input.onGesture(Gesture.ScreenDown, function () {
14     basic.showNumber(5)
15 })
16 input.onGesture(Gesture.Shake, function () {
17     basic.showNumber(1)
18 })
19 input.onGesture(Gesture.TiltRight, function () {
20     basic.showNumber(7)
21 })
22 input.onGesture(Gesture.LogoDown, function () {
23     basic.showNumber(3)
24 })
25
```

## Code 2:

Detect the value of acceleration speed at x, y and z axis

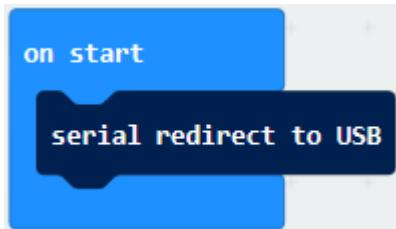
The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 7: Accelerometer-2	Project 7: Accelerometer-2



Or you could edit code step by step in the editing area.

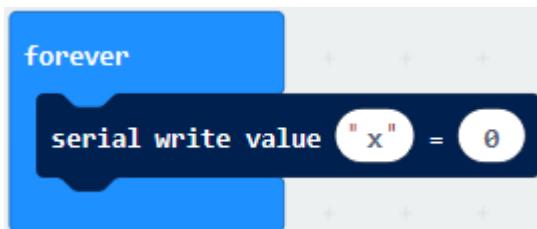
- A. Go to "Advanced" → "Serial" → "serial redirect to USB"
- B. Drag it into "on start"



\*\*\*\*\*

- A. Enter "Serial" → "serial write value x =0"

- B. Leave it into "forever" block



\*\*\*\*\* A.

Click "Input" → "acceleration(mg) x" ;

- B. Keep it into "0" box and capitalize the "x"



\*\*\*\*\*

Go to "Basic" and move out " pause (ms) 100 " below the block

**serial write value [X] = [acceleration (mg) x]**, then set to 100ms.



```
forever
  serial write value "X" = acceleration (mg) x ▾
  pause (ms) 100 ▾
```

\*\*\*\*\*

Replicate code string

```
serial write value "X" = acceleration (mg) x ▾
pause (ms) 100 ▾
```

for 3 times and keep them into “forever” block, separately set the whole code string as follows:

```
forever
  serial write value "X" = acceleration (mg) x ▾
  pause (ms) 100 ▾
  serial write value "Y" = acceleration (mg) y ▾
  pause (ms) 100 ▾
  serial write value "Z" = acceleration (mg) z ▾
  pause (ms) 100 ▾
```



## Complete Program:



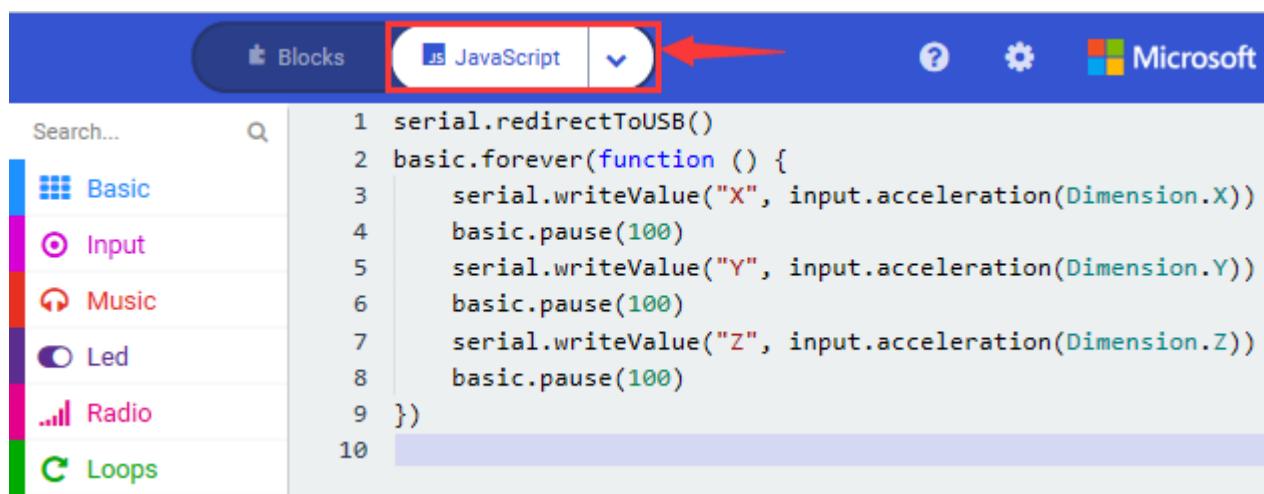
"on start": command block runs once to start program.  
Serial redirects to USB  
The program under the block "forever" runs cyclically.

Serial write value "X"=acceleration value on x axis

Serial write value "Y"=acceleration value on y axis

Serial write value "Z"=acceleration value on z axis

Click "JavaScript" to view the corresponding JavaScript code:



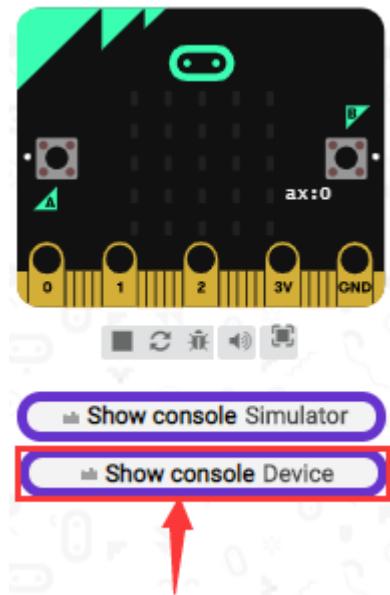
```
1 serial.redirectToUSB()
2 basic.forever(function () {
3     serial.writeValue("X", input.acceleration(Dimension.X))
4     basic.pause(100)
5     serial.writeValue("Y", input.acceleration(Dimension.Y))
6     basic.pause(100)
7     serial.writeValue("Z", input.acceleration(Dimension.Z))
8     basic.pause(100)
9 })
10
```

Download code 1 to micro:bit board, keep USB cable connected and click

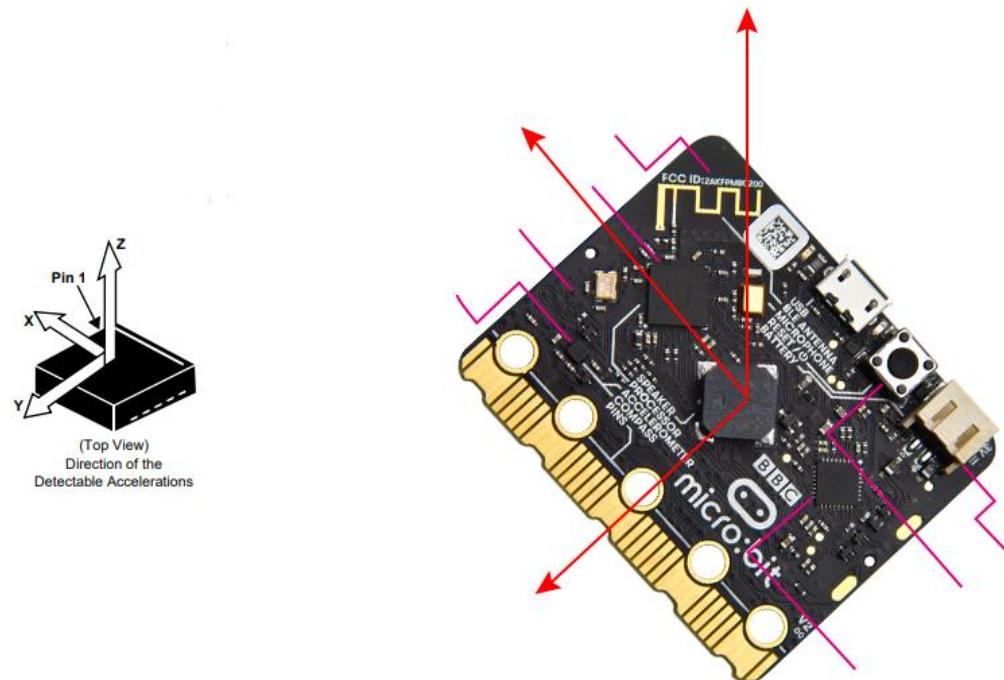
Show console Device



## (How to quick download?)



After referring to the MMA8653FC data manual and the hardware schematic diagram of the Micro: Bit main board V2, the accelerometer coordinate of the Micro: Bit V2 motherboard are shown in the figure below:



The following interface shows the decomposition value of acceleration in X axis, Y axis and Z axis respectively, as well as acceleration synthesis (acceleration synthesis of gravity and other external forces).

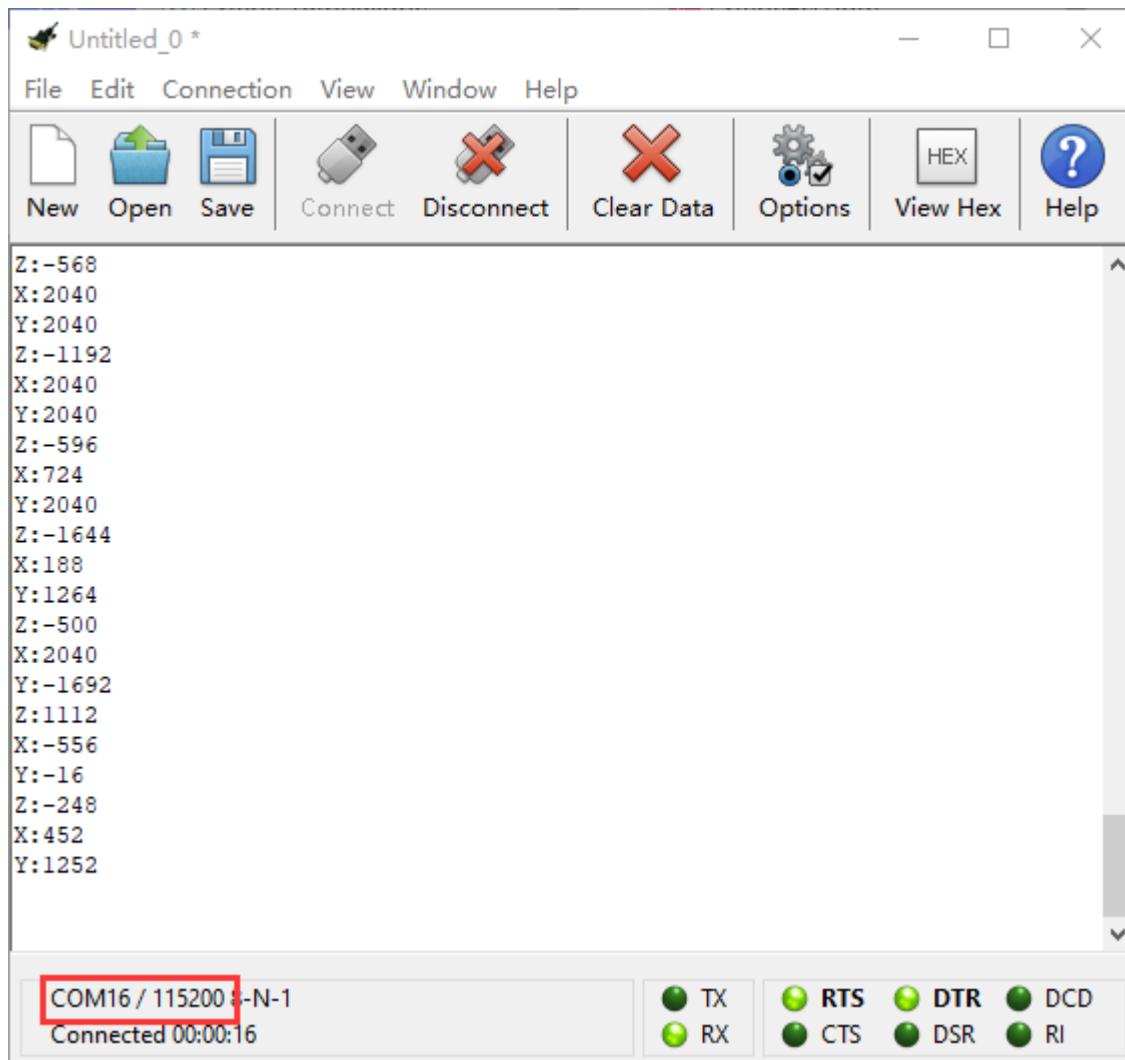


If you're running Windows 7 or 8 instead of Windows 10, via Google Chrome won't be able to match devices. You'll need to use the CoolTerm serial monitor software to read data.

You could open CoolTerm software, click Options, select SerialPort, set COM port and put baud rate to 115200 (after testing, the baud rate of USB



SerialPort communication on Micro: Bit main board V2 is 115200), click OK, and Connect. The CoolTerm serial monitor shows the data of X axis, Y axis and Z axis , as shown in the figures below :



#### (4)Test Results:

After uploading the test code 1 to micro:bit main board V2 and powering the board via the USB cable, if we shake the Micro: Bit main board V2. no matter at any direction, the LED dot matrix displays the digit “1” .

([How to download?](#) [How to quick download?](#))



---

When it is kept upright (make its logo above the LED dot matrix) , the number 2 shows.



When it is kept upside down( make its logo below the LED dot matrix) , it shows as below.



When it is placed still on the desk, showing its front side, the number 4 appears.



When it is placed still on the desk, showing its back side, the number 5 exhibits.

When the board is tilted to the left , the LED dot matrix shows the number



---

6 as shown below.



When the board is tilted to the right , the LED dot matrix displays the number 7 as shown below

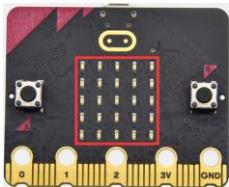


When the board is knocked to the floor, this process can be considered as a free fall and the LED dot matrix shows the number 8. (please note that this test is not recommended for it may damage the main board.)

Attention: if you'd like to try this function, you can also set the acceleration to 3g, 6g or 8g. But still ,we do not recommend.



## Project 8: Light Detection



### (1) Project Description

In this project, we focus on the light detection function of the Micro: Bit main board V2. It is achieved by the LED dot matrix since the main board is not equipped with a photoresistor.

When the light irradiates the LED matrix, the voltage change will be produced. Therefore, we could determine the light intensity by voltage change.

### (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor

Import Hex profile (How to import?)

Or click "New Project" and drag blocks step by step

### (3) Test Code



The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 8: Light Detection	Project 8: Light Detection

Or you could edit code step by step in the editing area.

(1) A. Enter “Advanced” → “Serial” → “serial redirect to USB” ;



B. Drag it into “on start” block.

\*\*\*\*\*  
(2) A. Go to “Serial” → “serial write value x =0” ;



B. Move it into “forever”

A. Click “Input” → “acceleration(mg) x”

B. Put “acceleration(mg) x” in the “0” box and change “x” into “Light intensity” .



\*\*\*\*\*

- A. Click "Basic" → "pause (ms) 100" ;
- B. Lay it down into "forever" and set to 100ms.



\*\*\*\*\*

Complete Program:



“on start”: command block runs once to start program.

Serial redirects to USB

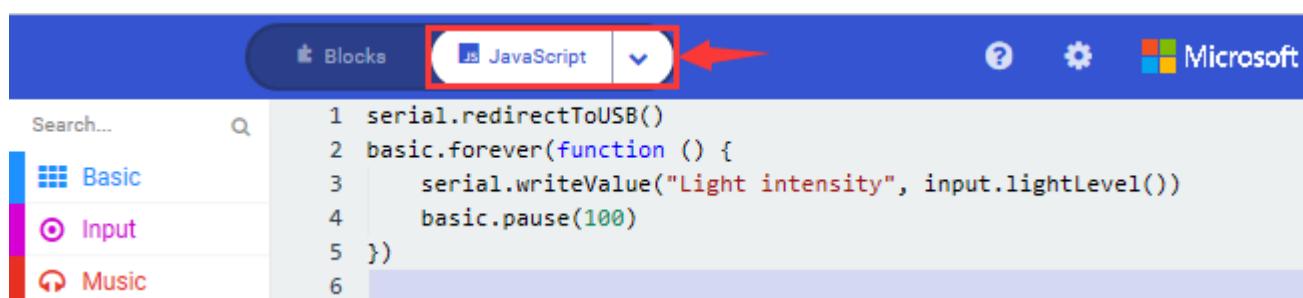
The program under the block “forever” runs cyclically.

Serial write value “Light intensity”

= light level

Delay in 100ms

Click “JavaScript” to switch into the corresponding JavaScript code:



```
1 serial.redirectToUSB()
2 basic.forever(function () {
3   serial.writeValue("Light intensity", input.lightLevel())
4   basic.pause(100)
5 })
6
```

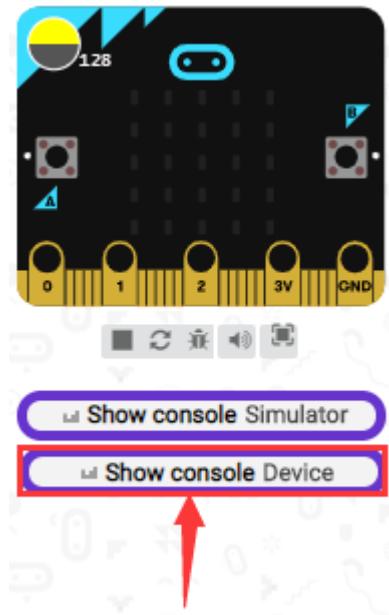


## (4) Test Results:

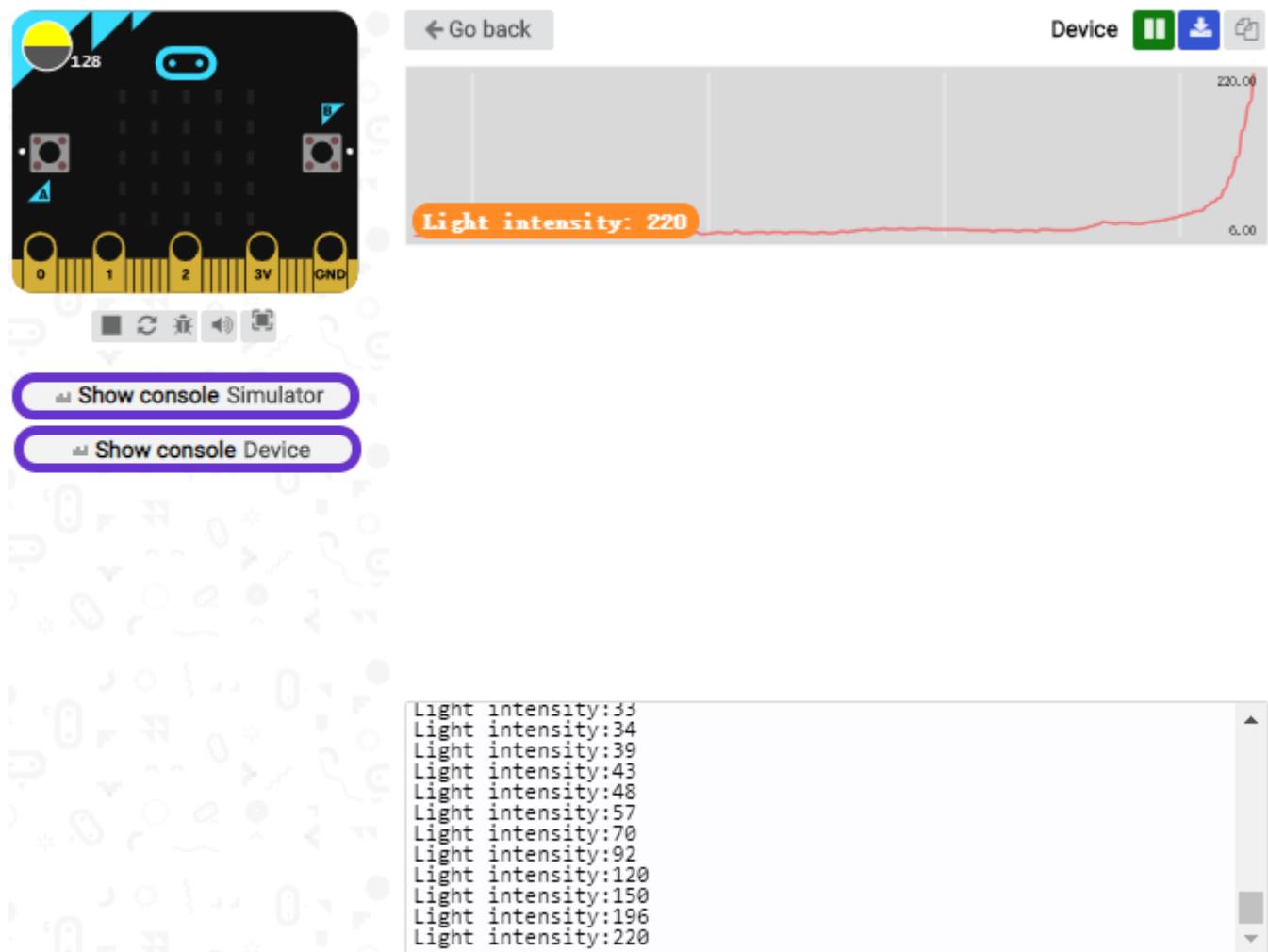
Download code to micro:bit board don't plug off USB cable and click

Show console Device

(How to quick download?)



The intensity value is 0 when covering LED dot matrix. And the value varies with the light intensity. When placing micro:bit under the sunlight, the stronger the light is, the larger the intensity value is. As shown below:



Open “CoolTerm” , click “Options” to select “SerialPort” , and set “COM” port and 115200 baud rate(the baud rate of USB serial communication of micro:bit is 115200 through the test).

Then click “OK” and “Connect” .

The light intensity value is shown below:



Untitled\_0 \*

File Edit Connection View Window Help

New Open Save Connect Disconnect Clear Data Options HEX Help

```
Light intensity:0
Light intensity:1
Light intensity:1
Light intensity:3
Light intensity:3
Light intensity:5
Light intensity:9
Light intensity:11
Light intensity:13
Light intensity:16
Light intensity:19
Light intensity:21
Light intensity:25
Light intensity:27
Light intensity:29
Light intensity:32
Light intensity:40
Light intensity:50
Light intensity:58
Light intensity:70
Light intensity:78
Light intensity:86
Light intensity:93
Light intensity:101
Light intensity:108
```

COM16 / 115200 8-N-1  
Connected 00:02:34

TX RX RTS DTR DCD  
CTS DSR RI

## Project 9: Speaker





## (1) Project Description

The Micro: Bit main board V2 has an built-in speaker, which makes adding sound to the programs easier. We can program the speaker to air all kinds of tones, like playing the son *Ode to Joy*.

## (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor

Import Hex profile (How to import?)

Or click “New Project” and drag blocks step by step

## (3) Test Code:

The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project Code/Project 9: Speaker	Project 9: Speaker



Or you could edit code step by step in the editing area.

Enter “Basic” module to find “show icon” and drag it into “on start” block;

Click the little triangle to find “

\*\*\*\*\*

(2) Enter “Music” module to find and drug “play sound giggle until done” into “forever” block;

Enter “Basic” module to find and drug “pause(ms) 100” into “forever” block ;

Change 100 into 1000;



( 3 ) Copy



three times and place it into



“forever” block ;

Click the little triangle to select “happy” , “hello” , “yawn” ;



\*\*\*\*\*



## Complete Program:



```
on start
  show icon [grid v]

forever
  play sound [giggle v] until done
  pause (ms) [1000 v]
  play sound [happy v] until done
  pause (ms) [1000 v]
  play sound [hello v] until done
  pause (ms) [1000 v]
  play sound [yawn v] until done
  pause (ms) [1000 v]
```

Select "JavaScript" and "Python" to switch into JavaScript and Python language code:



The screenshot shows the Microsoft MakeCode interface for a micro:bit. The top bar has tabs for 'Blocks' (selected), 'JavaScript' (highlighted with a red box and arrow), and 'Python'. Below the tabs is a search bar and a sidebar with categories: Basic, Input, Music, Led, Radio, Loops, Logic, and Variables. The main area contains the following JavaScript code:

```
1 basic.showIcon(IconNames.EighthNote)
2 basic.forever(function () {
3     soundExpression.giggle.playUntilDone()
4     basic.pause(1000)
5     soundExpression.happy.playUntilDone()
6     basic.pause(1000)
7     soundExpression.hello.playUntilDone()
8     basic.pause(1000)
9     soundExpression.yawn.playUntilDone()
10    basic.pause(1000)
11 })
12
```

The screenshot shows the Microsoft MakeCode interface for a micro:bit. The top bar has tabs for 'Blocks' (selected), 'Python' (highlighted with a red box and arrow), and 'JavaScript'. Below the tabs is a search bar and a sidebar with categories: Basic, Input, Music, Led, Radio, Loops, Logic, and Variables. The main area contains the following Python code:

```
1 basic.show_icon(IconNames.EIGHTH_NOTE)
2
3 def on_forever():
4     soundExpression.giggle.play_until_done()
5     basic.pause(1000)
6     soundExpression.happy.play_until_done()
7     basic.pause(1000)
8     soundExpression.hello.play_until_done()
9     basic.pause(1000)
10    soundExpression.yawn.play_until_done()
11    basic.pause(1000)
12 basic.forever(on_forever)
13
```

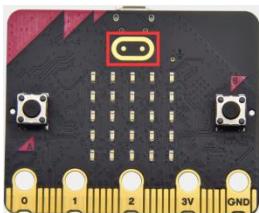
#### (4) Test Results:

After uploading the test code to micro:bit main board V2 and powering the board via the USB cable, the speaker utters sound and the LED dot matrix shows the logo of music.

([How to download?](#) [How to quick download?](#))



## Project 10: Touch-sensitive Logo



### (1) Project Description

The Micro: Bit main board V2 is equipped with a golden touch-sensitive logo, which can act as an input component and function like an extra button.

It contains a capacitive touch sensor that senses small changes in the electric field when pressed (or touched), just like your phone or tablet screen do. When you press it, you can activate the program.

### (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor

Import Hex profile (How to import?)

Or click "New Project" and drag blocks step by step

### (3) Test Code



The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 10: Touch-sensitive Logo	Project 9: Speaker.hex

Or you could edit code step by step in the editing area.

( 1 ) Delete block “on start” and “forever” ;

( 2 ) Enter “Input” module to find and drag “on logo pressed” ;

Click the little triangle to find “touched” ;



( 3 ) Enter module “Variables” →choose “Make a Variable” →input “start” →click “OK”

The variable “start” is established;

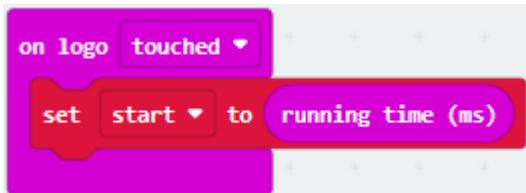
Enter “Variables” module to find and drag “set start to 0” into “on logo



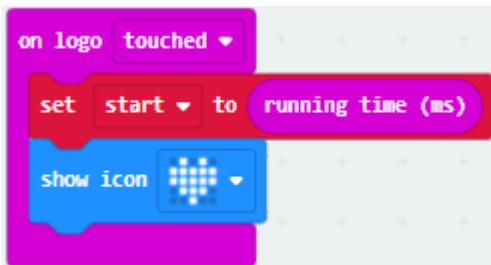
touched” block;



( 4 )Enter “Input” module →click “more” → find and drag “running time(ms)” into the “0” of “set start to 0” block;



( 5 )Enter “Basic” module to find and drag “show icon”  into “on logo touched” block;



( 6 )Enter “Input” module to find and drag “on logo pressed” →choose “released” → establish variable “time” ;

Enter “Variables” module to find and drag “set time to 0” into “on logo pressed” block;

Enter “Math” module to find and drag “0-0” into the “0” of “set start to 0” block;



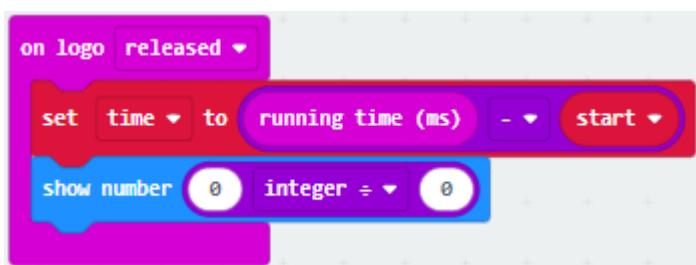
( 7 )Enter “Input” module→ “more” → find and drag “running time(ms)” into “0” on the left side of “0-0” ;

Enter “Variables” module to find and drag “start” into “0” on the right side of “0-0” ;



( 8 )Enter “Basic” module to find and drag “show number” into “on logo released” block;

Enter “Math” module to find and drag “square root 0” into “0” ; Click the little triangle to find “integer÷” ;



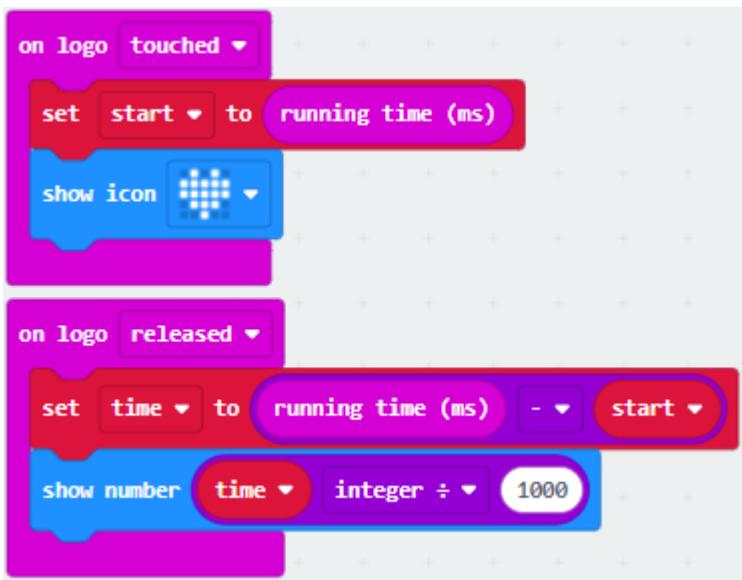
( 9 ) Enter “Variables” module to find and drag “time” into “0” on the left



side of "0-0" and change the "0" on the right side to " 1000" ;

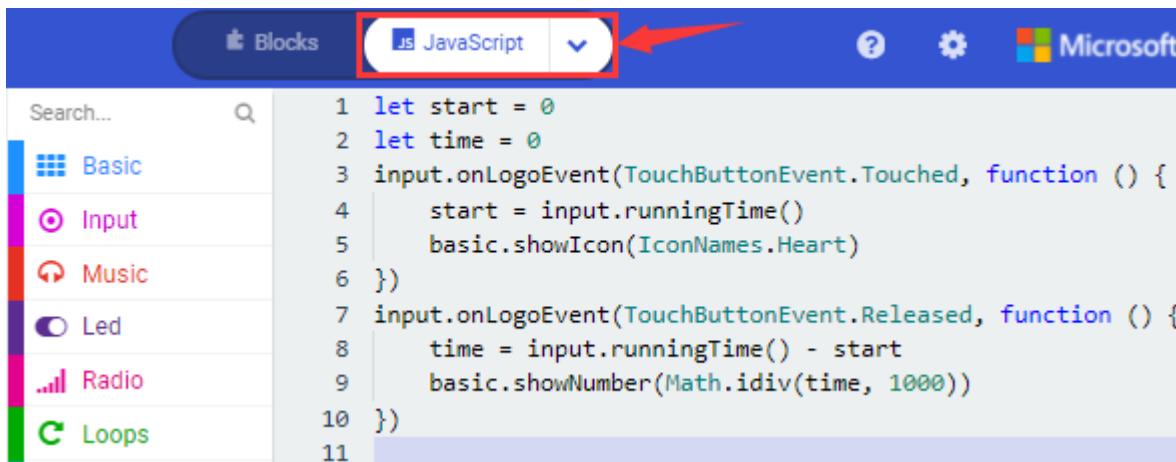


## Complete Program:





Select "JavaScript" and "Python" to switch into JavaScript and Python language code:



The screenshot shows the Scratch interface with the language dropdown set to "JavaScript". A red arrow points to the dropdown menu. The code area contains the following JavaScript code:

```
1 let start = 0
2 let time = 0
3 input.onLogoEvent(TouchButtonEvent.Touched, function () {
4     start = input.runningTime()
5     basic.showIcon(IconNames.Heart)
6 })
7 input.onLogoEvent(TouchButtonEvent.Released, function () {
8     time = input.runningTime() - start
9     basic.showNumber(Math.idiv(time, 1000))
10 })
11
```



The screenshot shows the Scratch interface with the language dropdown set to "Python". A red arrow points to the dropdown menu. The code area contains the following Python code:

```
3
4 def on_logo_touched():
5     global start
6     start = input.running_time()
7     basic.show_icon(IconNames.HEART)
8 input.on_logo_event(TouchButtonEvent.TOUCHED, on_logo_touched)
9
10 def on_logo_released():
11     global time
12     time = input.running_time() - start
13     basic.show_number(Math.idiv(time, 1000))
14 input.on_logo_event(TouchButtonEvent.RELEASED, on_logo_released)
15
```

#### (4) Test Results:

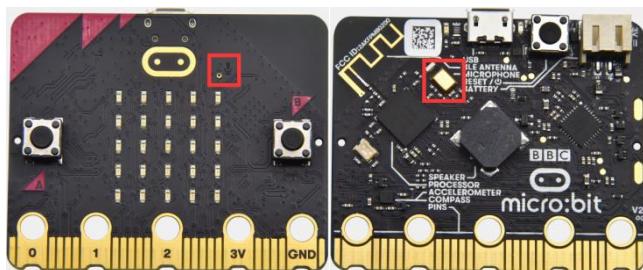
After uploading the test code to micro:bit main board V2 and powering the board via the USB cable, the LED dot matrix exhibits the heart pattern when the touch-sensitive logo is pressed or touched and displays digit



when the logo is released.

([How to download?](#) [How to quick download?](#))

## Project 11: Microphone



### (1) Project Description

The Micro: Bit main board V2 is built with a microphone which can test the volume of ambient environment. When you clap, the microphone LED indicator turns on. Since it can measure the intensity of sound, you can make a noise scale or disco lighting changing with music. The microphone is placed on the opposite side of the microphone LED indicator and in proximity with holes that lets sound pass. When the board detects sound, the LED indicator lights up.

### (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor



---

Import Hex profile (How to import?)

Or click "New Project" and drag blocks step by step

### (3) Test Code

#### Code 1

The route to get test code ([How to load?](#))

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 11: Microphone-1	Project 11: Microphone-1.hex

Or you could edit code step by step in the editing area.

(1 ) Delete block "on start" and "forever" ;

( 2 ) Enter "Input" module to find and drag "on loud sound" ;

Enter "Basic" module to find and drag "show number" into "on loud sound" block ;





( 3 )Copy once;



Click the little triangle of "lond" to choose "quiet" ;

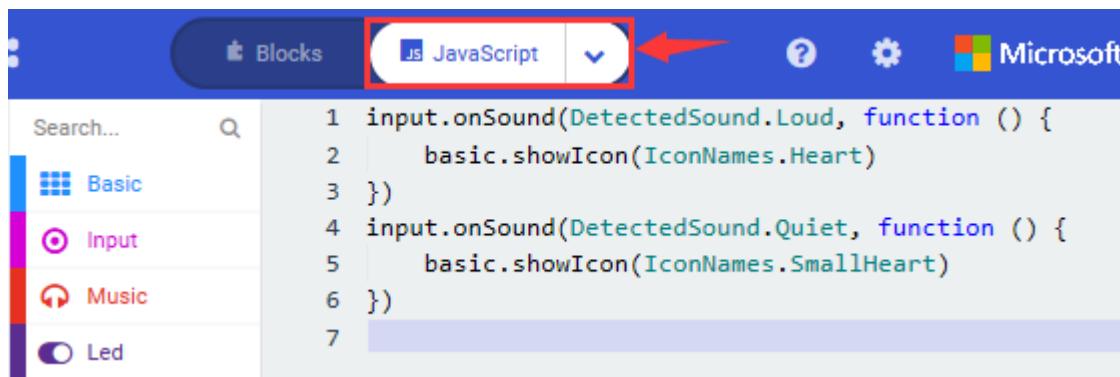
Click the little triangle of "grid icon" to choose "grid" ;



Complete Program:



Select "JavaScript" and "Python" to switch into JavaScript and Python language code:

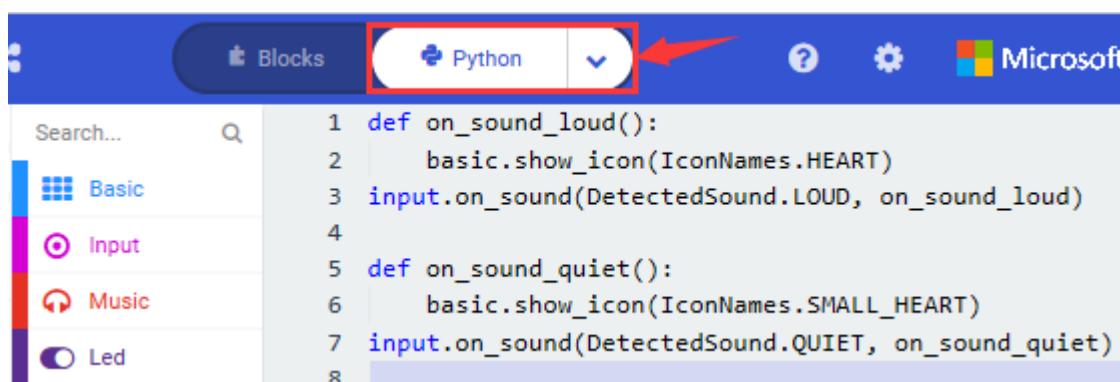


Blocks    JS JavaScript    ?    Microsoft

Search...    Q

Basic    Input    Music    Led

```
1 input.onSound(DetectedSound.Loud, function () {
2     basic.showIcon(IconNames.Heart)
3 })
4 input.onSound(DetectedSound.Quiet, function () {
5     basic.showIcon(IconNames.SmallHeart)
6 })
7
```



Blocks    Python    ?    Microsoft

Search...    Q

Basic    Input    Music    Led

```
1 def on_sound_loud():
2     basic.show_icon(IconNames.HEART)
3     input.on_sound(DetectedSound.LOUD, on_sound_loud)
4
5 def on_sound_quiet():
6     basic.show_icon(IconNames.SMALL_HEART)
7     input.on_sound(DetectedSound.QUIET, on_sound_quiet)
8
```

#### (4) Test Results 1:

After uploading test code to micro:bit main board V2 and powering the board via the USB cable, the LED dot matrix displays pattern  when you clap and pattern  when it is quiet around. ([How to download?](#)  
[How to quick download?](#))

#### Code 2:

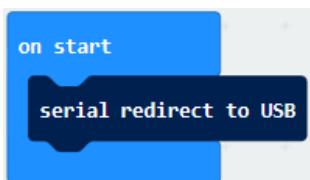
The route to get test code ([How to load?](#))



File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 11: Microphone-2	Project 11: Microphone-2.hex

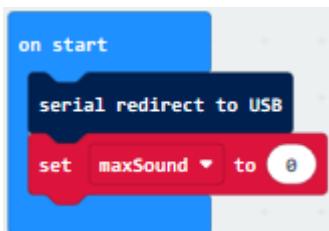
Or you could edit code step by step in the editing area.

Enter “Advanced” module → choose “Serial” to find and drag “serial redirect to USB” into “on start” block ;



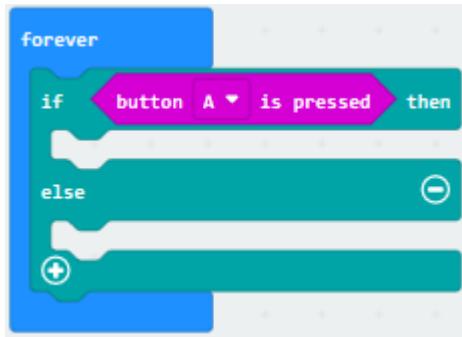
Enter “Variables” module → choose “Make a Variable” → input “maxSound” → click “OK”, variable “maxSound” is established;

Enter “Variables” module to find and drag “set maxSound to 0” into “on start” block ;



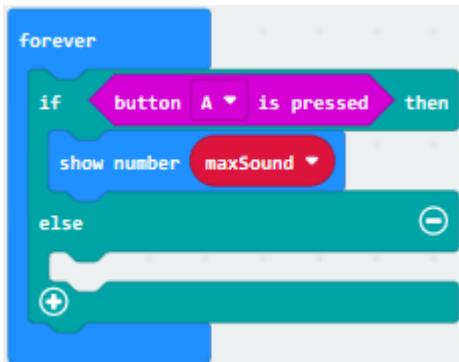
Enter “Logic” module to find and drag “if true then...else” into “forever” block ;

Enter “Input” module to find and drag “button A is pressed” into “then” ;



Enter "Basic" module to find and drag "show number" into "then" ;

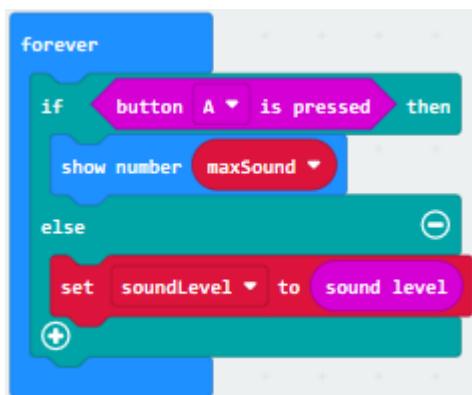
Enter "Variables" module to find and drag "maxSound" into "0" ;



Establish variable "soundLevel" ;

Enter "Variables" module to find and drag "set soundLevel to 0" into "else" ;

Enter "Input" module to find and drag "sound level" into "0" ;



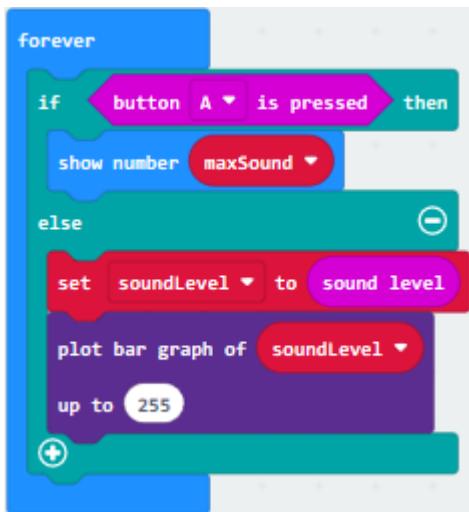


Enter "Led" module to find and drag "plot bar graph of 0 up to 0"

into "else" ;

Enter "Variables" module to find and drag "soundLevel" into the "0" behind "of" ;

Change the "0" behind "up" to "255" ;

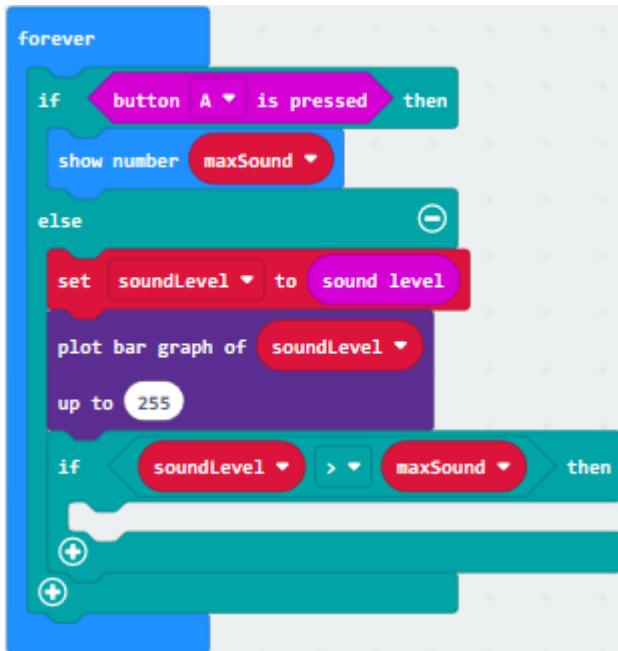


Enter "Logic" module to find and drag "if true then" into "else" block ;

Enter "Logic" module to find and drag "0 > 0" into "then" ;

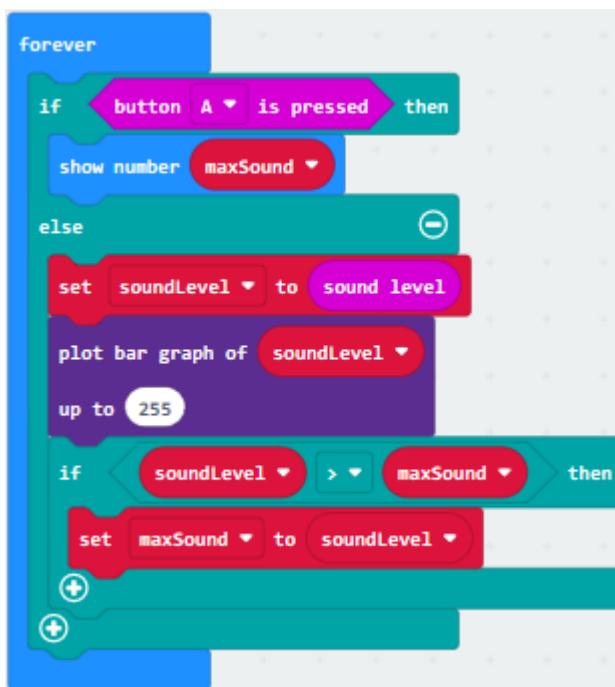
Enter "Variables" module to find and drag "soundLevel" into "0" on the left side of "0-0" ;

Enter "Variables" module to find and drag "maxSound" into "0" on the right side;



Enter “Variables” module to find and drag “set maxSound to 0” into the second “then” ;

Enter “Variables” module to find and drag “soundLevel” into the “0” ;





## Complete Program:

```
on start
  serial redirect to USB
  set maxSound to 0
forever
  if button A is pressed then
    show number maxSound
    else
      set soundLevel to sound level
      plot bar graph of soundLevel up to 255
      if soundLevel > maxSound then
        set maxSound to soundLevel
        next [2 v]
      end
    end
  end
  next [2 v]
```

Select "JavaScript" and "Python" to switch into JavaScript and Python language code:



Blocks    JS JavaScript    ?    Microsoft

Search...    Q

Basic    Input    Music    Led    Radio    Loops    Logic    Variables    Math

Advanced

```
1 let soundLevel = 0
2 serial.redirectToUSB()
3 let maxSound = 0
4 basic.forever(function () {
5     if (input.buttonIsPressed(Button.A)) {
6         basic.showNumber(maxSound)
7     } else {
8         soundLevel = input.soundLevel()
9         led.plotBarGraph(
10            soundLevel,
11            255
12        )
13        if (soundLevel > maxSound) {
14            maxSound = soundLevel
15        }
16    }
17 })
18 }
```

Blocks    Python    ?    Microsoft

Search...    Q

Basic    Input    Music    Led    Radio    Loops    Logic    Variables    Math

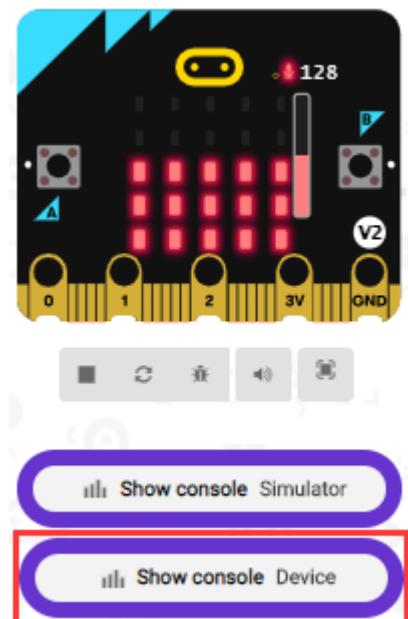
```
1 soundLevel = 0
2 serial.redirect_to_usb()
3 maxSound = 0
4
5 def on_forever():
6     global soundLevel, maxSound
7     if input.button_is_pressed(Button.A):
8         basic.show_number(maxSound)
9     else:
10         soundLevel = input.sound_level()
11         led.plot_bar_graph(soundLevel, 255)
12         if soundLevel > maxSound:
13             maxSound = soundLevel
14 basic.forever(on_forever)
15 
```



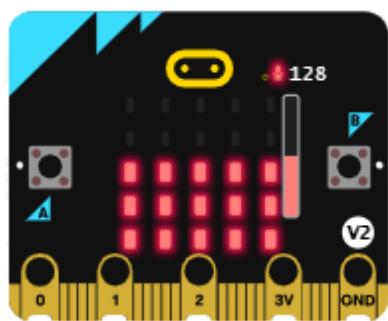
## (5)Test Results 2:

Upload test code to micro:bit main board V2, power the board via the USB cable and click "Show console Device" as shown below.

( [How to quick download?](#))

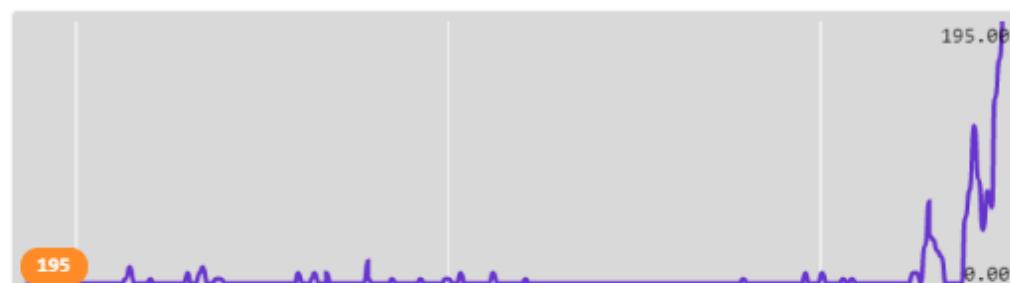


When the sound is louder around, the sound value shows in the serial port is bigger as shown below.



← Go back

Device



Show console Simulator

Show console Device

95
127
150
153
172
187
183
2 187
191
3 195

What's more, when pressing the button A, the LED dot matrix displays the value of the biggest volume( please note that the biggest volume can be reset via the Reset button on the other side of the board ) while when clapping, the LED dot matrix shows the pattern of the sound.



## Project 12: Bluetooth Wireless Communication



### (1) Project Description

The Micro: Bit main board V2 comes with a nRF52833 processor (with a built-in BLE(Bluetooth Low Energy) device Bluetooth 5.1 ) and a 2.4GHz antenna for Bluetooth wireless communication and 2.4GHz wireless communication. With the help of them, the board is able to communicate with a variety of Bluetooth devices, including smart phones and tablets.

In this project, we mainly concentrate on the Bluetooth wireless communication function of this main board. Linked with Bluetooth, it can transmit code or signals. To this end, we should connect an Apple device (a phone or an iPad) to the board.

Since setting up Android phones to achieve wireless transmission is similar to that of Apple devices, no need to illustrate again.

### (2) Experimental Preparation:

- Connect micro:bit to computer with the USB cable
- Open online Makecode editor



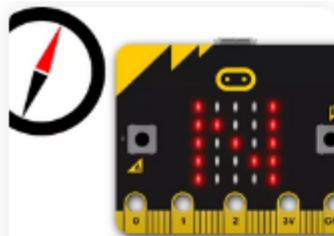
---

Import Hex profile (How to import?)

Or click “New Project” and drag blocks step by step

### (3)Procedures:

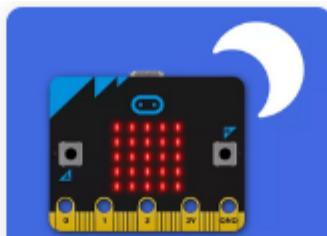
For Apple devices, enter this link <https://www.microbit.org/get-started/user-guide/ble-ios/> with your computer first, and then click “Download pairing HEX file” to download the Micro: Bit firmware to a folder or desk, and upload the downloaded firmware to the Micro: Bit main board V2.



### Compass North

Create a simple compass to show...

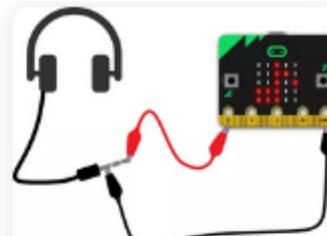
 Intermediate



### Nightlight

Create a light that turns on when it's...

 Intermediate



### Make some noise

Connect headphones or speakers to make...

 Intermediate

## If you need help

If you're having problems flashing code from your iOS device to your micro:bit, download this HEX file and transfer it to your micro:bit from a computer, or visit our support site.

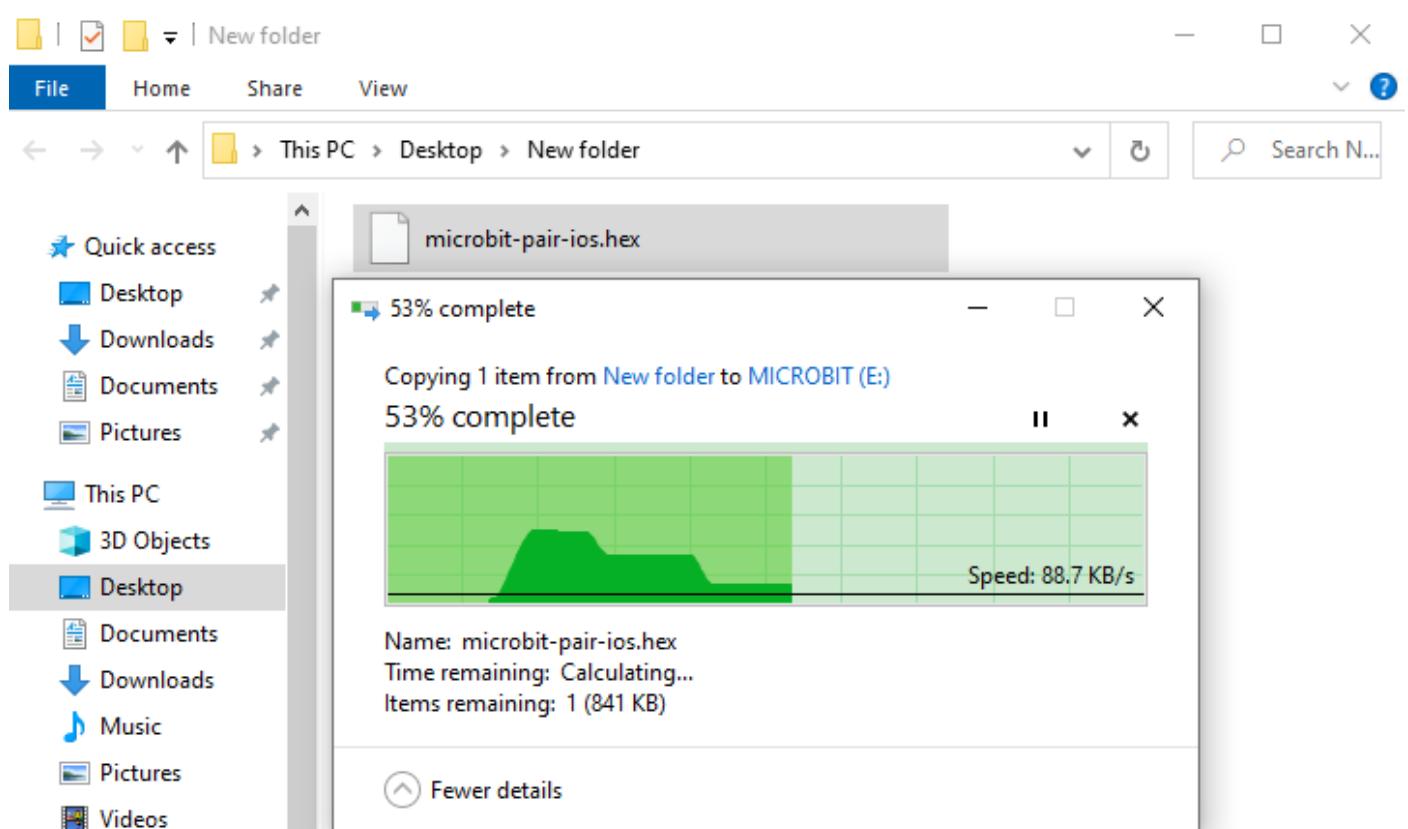
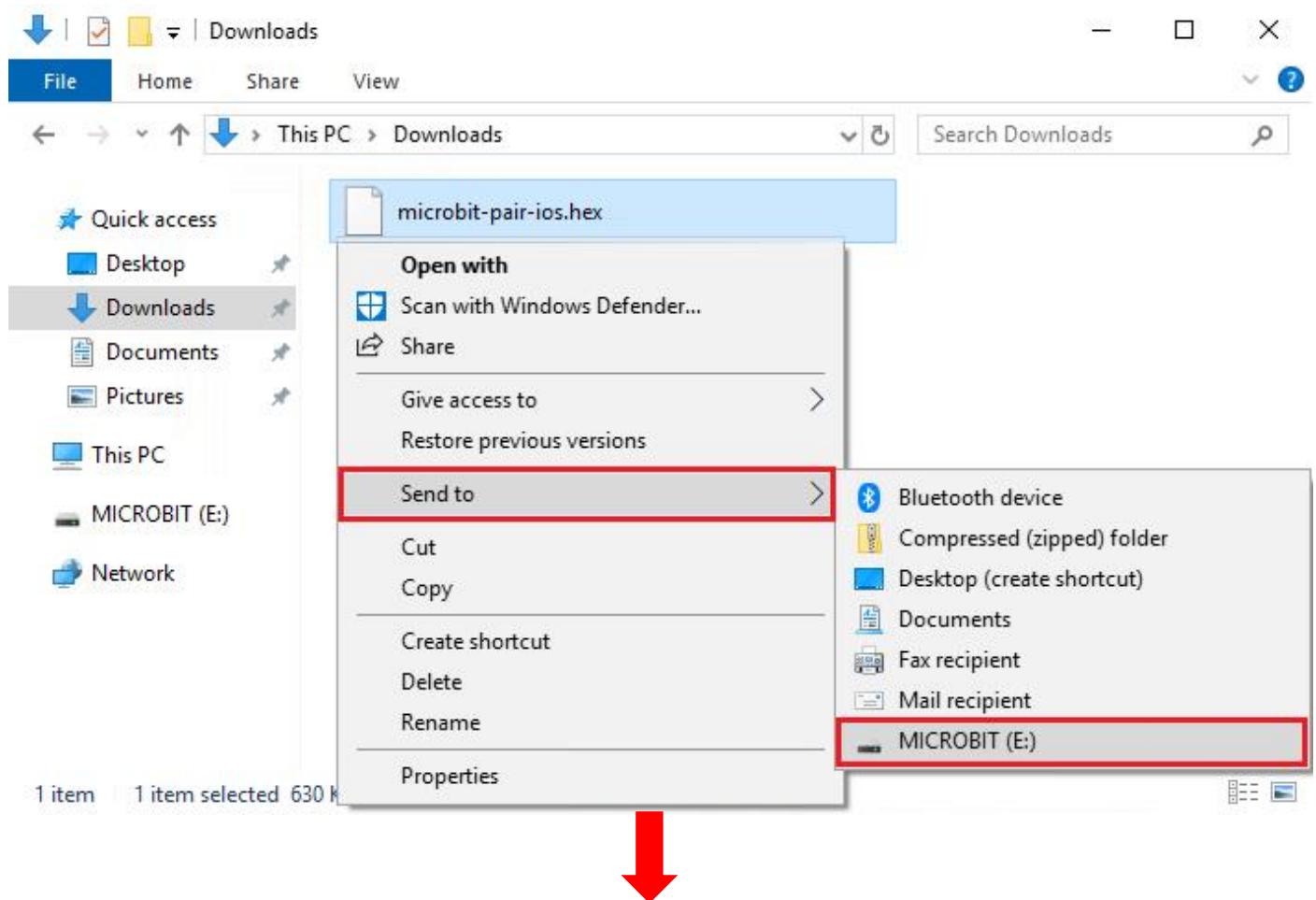
[Download pairing HEX file](#)

[iOS app support](#)

## Monitor and control

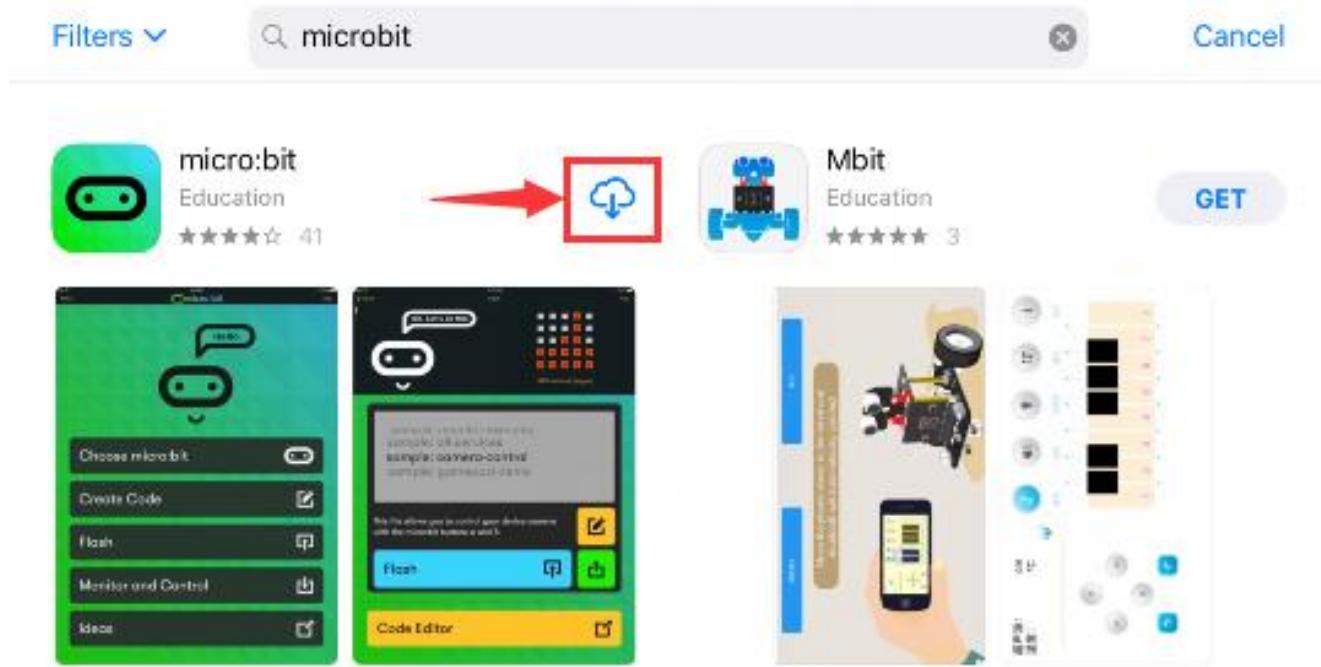
The 'Monitor and control' section of the iOS app allows you to observe real-time data from the micro:bit sensors, send messages directly to the LEDs and control the micro:bit buttons and pins from your iPad or iPhone.







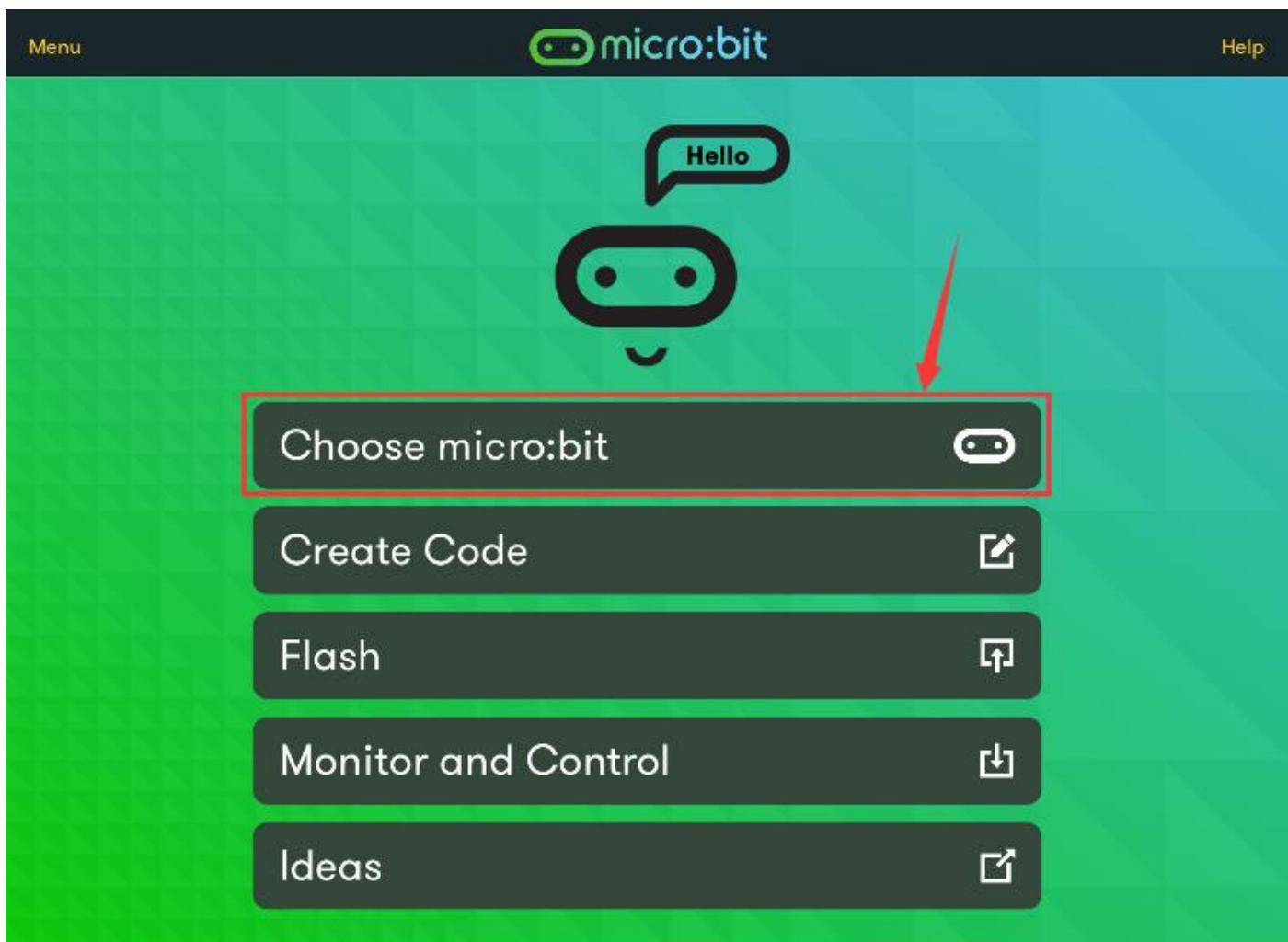
Search “micro bit” in your App Store to download the APP micro:bit.



Connect your Apple device with Micro: Bit main board V2:

Firstly, turn on the Bluetooth of your Apple device and open the APP micro:bit to select item “Choose micro:bit” to start pairing Bluetooth.

Please make sure that the Micro: Bit main board V2 and your computer are still linked via the USB cable.



Secondly, click “Pair a new micro:bit” ;



Home Choose micro:bit Help

## Currently selected micro:bit

None selected

Pair a micro:bit

If you want to use a new micro:bit, tap the button below.

If you want to remove a pairing from a micro:bit, go to the Bluetooth section in your device Settings.

Having problems?  
Try the Help page.

Pair a new micro:bit

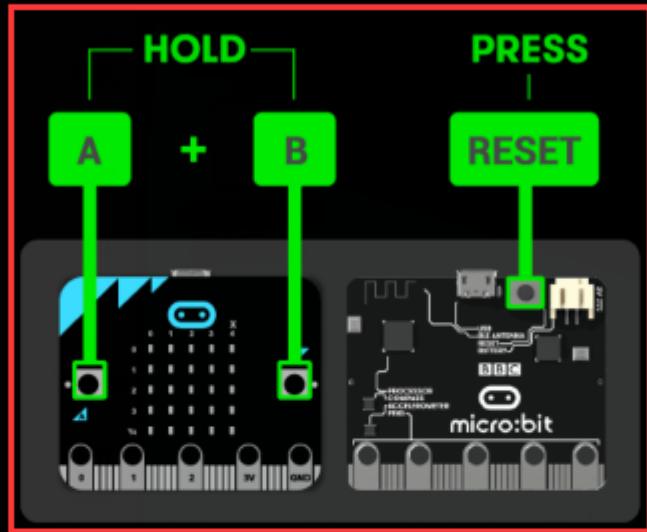


Following the instructions to press button A and B at the same time(do not release them until you are told to) and press Reset & Power button for a few seconds.

Release the Reset & Power button, you will see a password pattern shows on the LED dot matrix. Now , release buttons A and B and click Next.



## How to pair your micro:bit



Let's do this

Step 1

HOLD the A and B buttons and  
PRESS and RELEASE RESET



Cancel

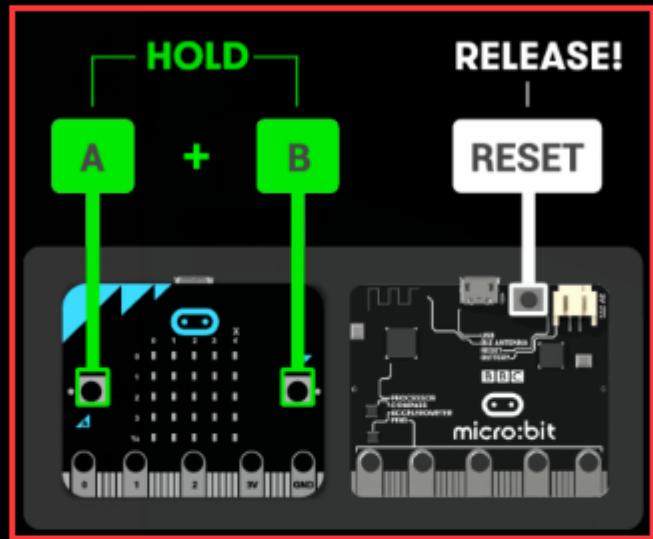


Next





## How to pair your micro:bit



Let's do this

Step 1

HOLD the A and B buttons and  
PRESS and RELEASE RESET



Cancel

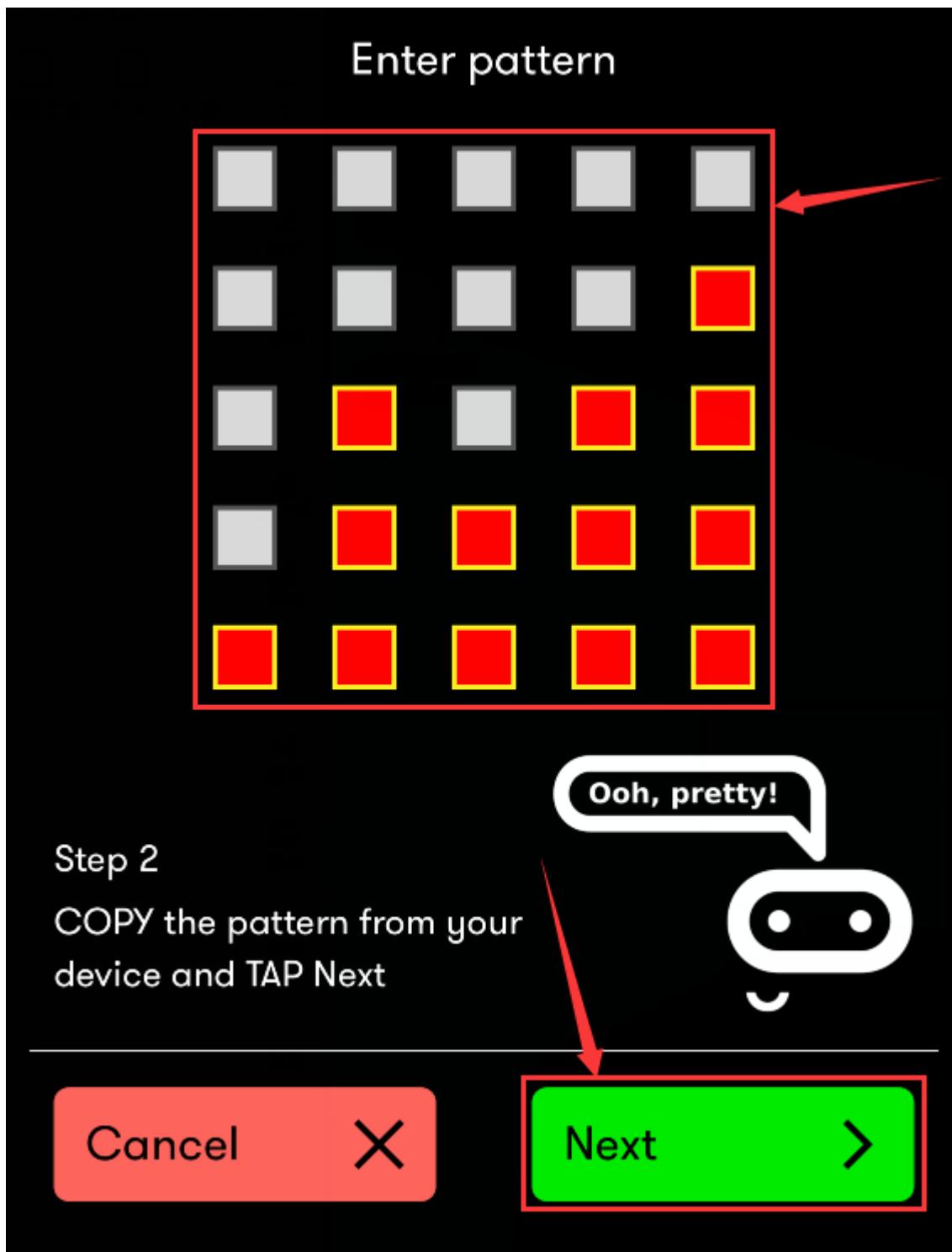


Next





Set the password pattern on your Apple device as the same pattern showed on the matrix and click Next.

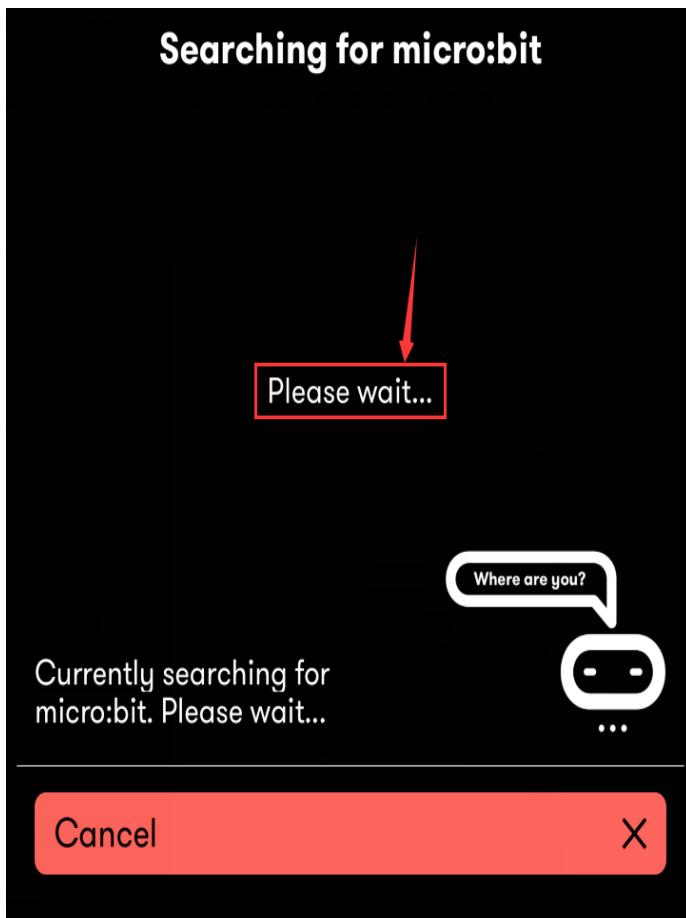
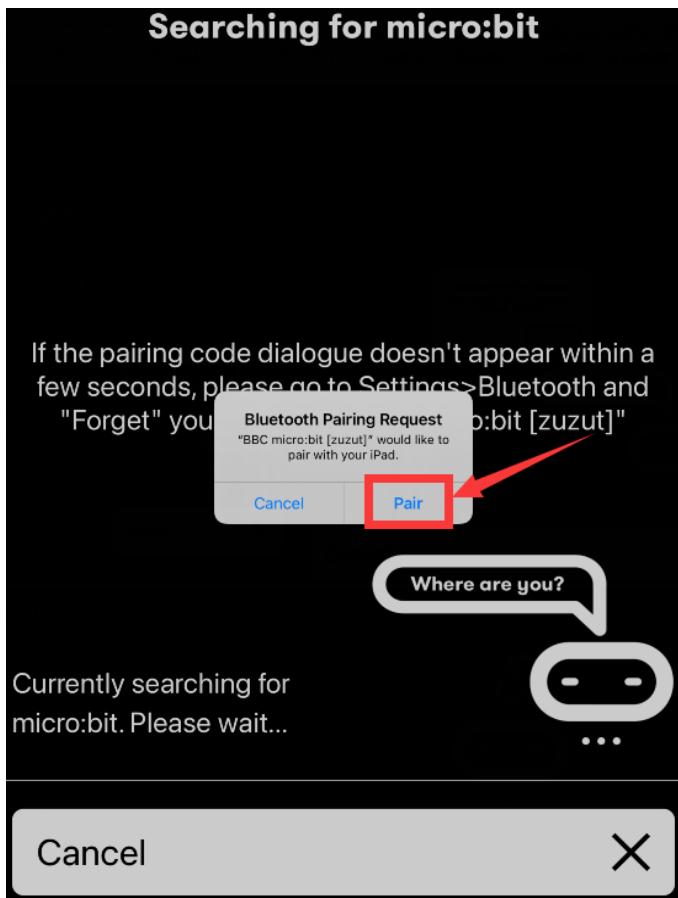


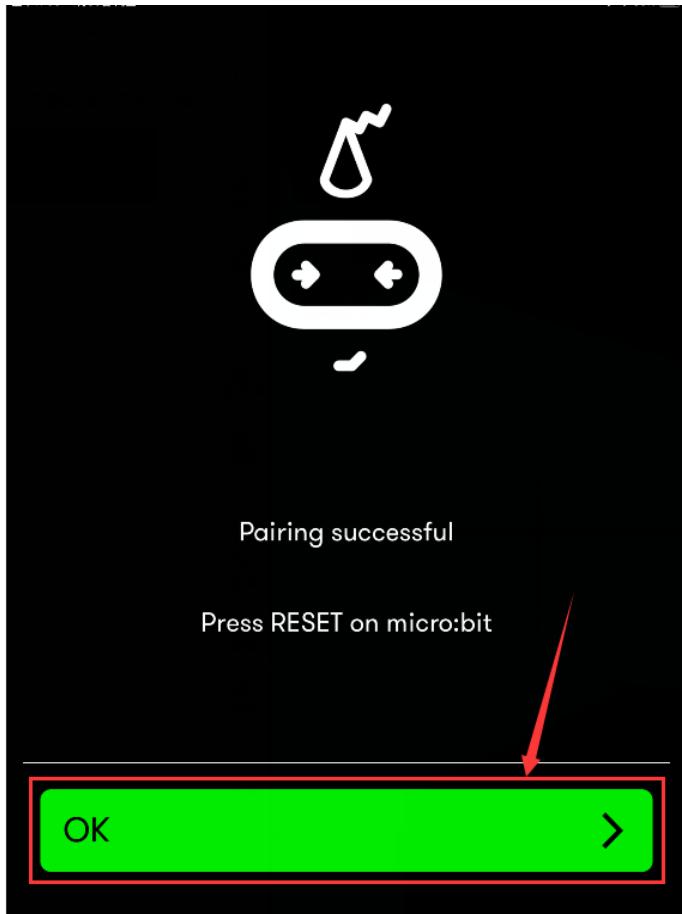
Still click Next and a dialog box props up as shown below. Then click "Pair".



A few seconds later, the match is done and the LED dot matrix displays the "√" pattern.

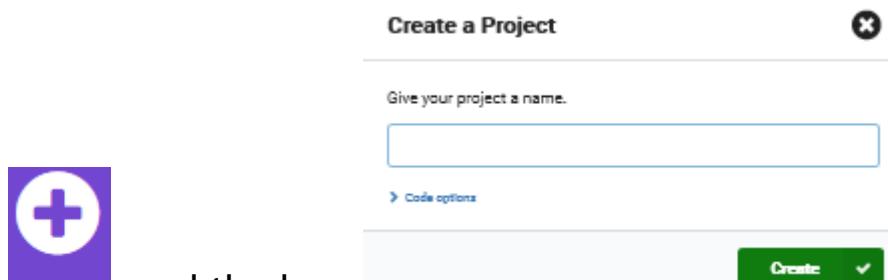




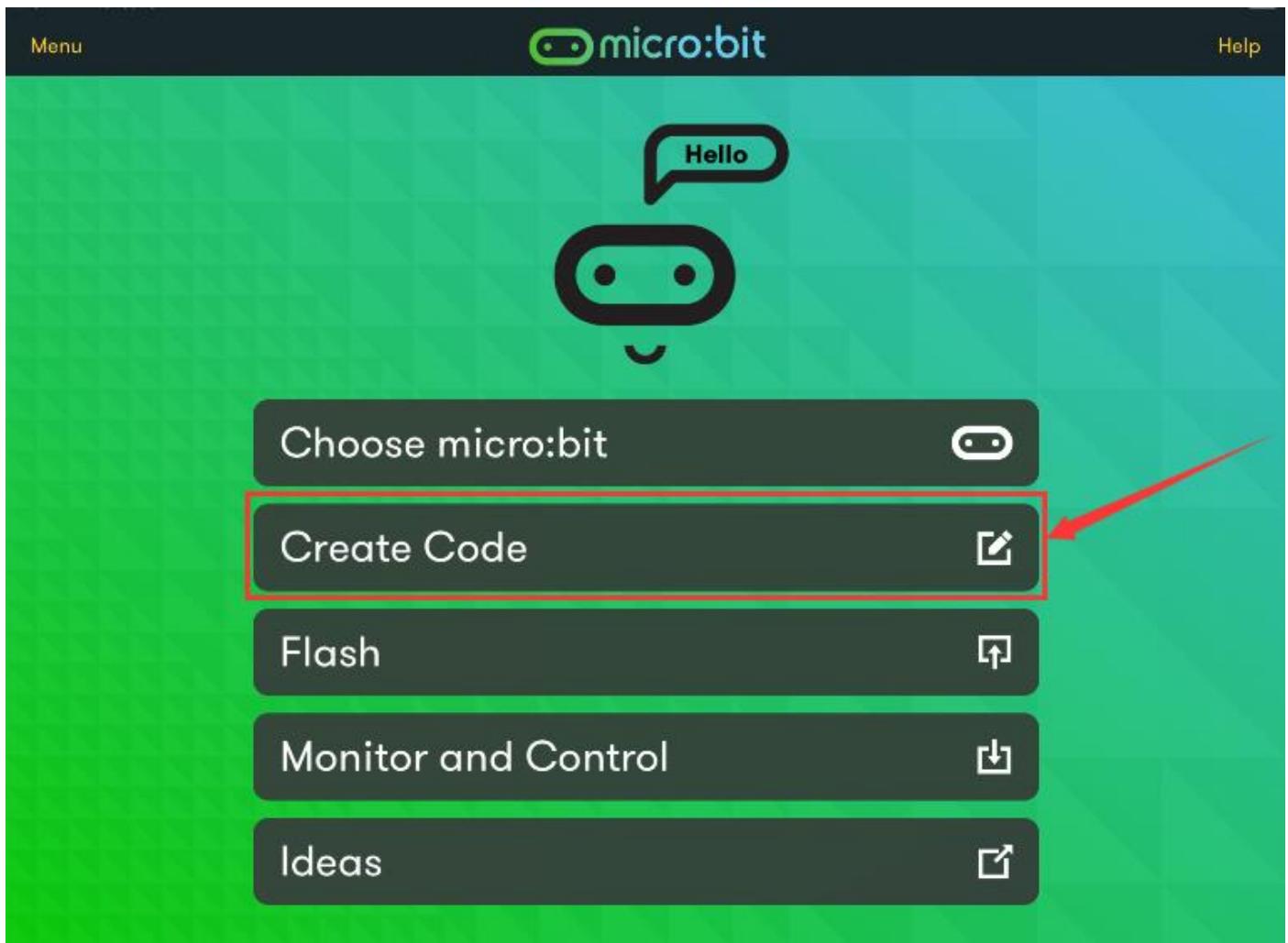


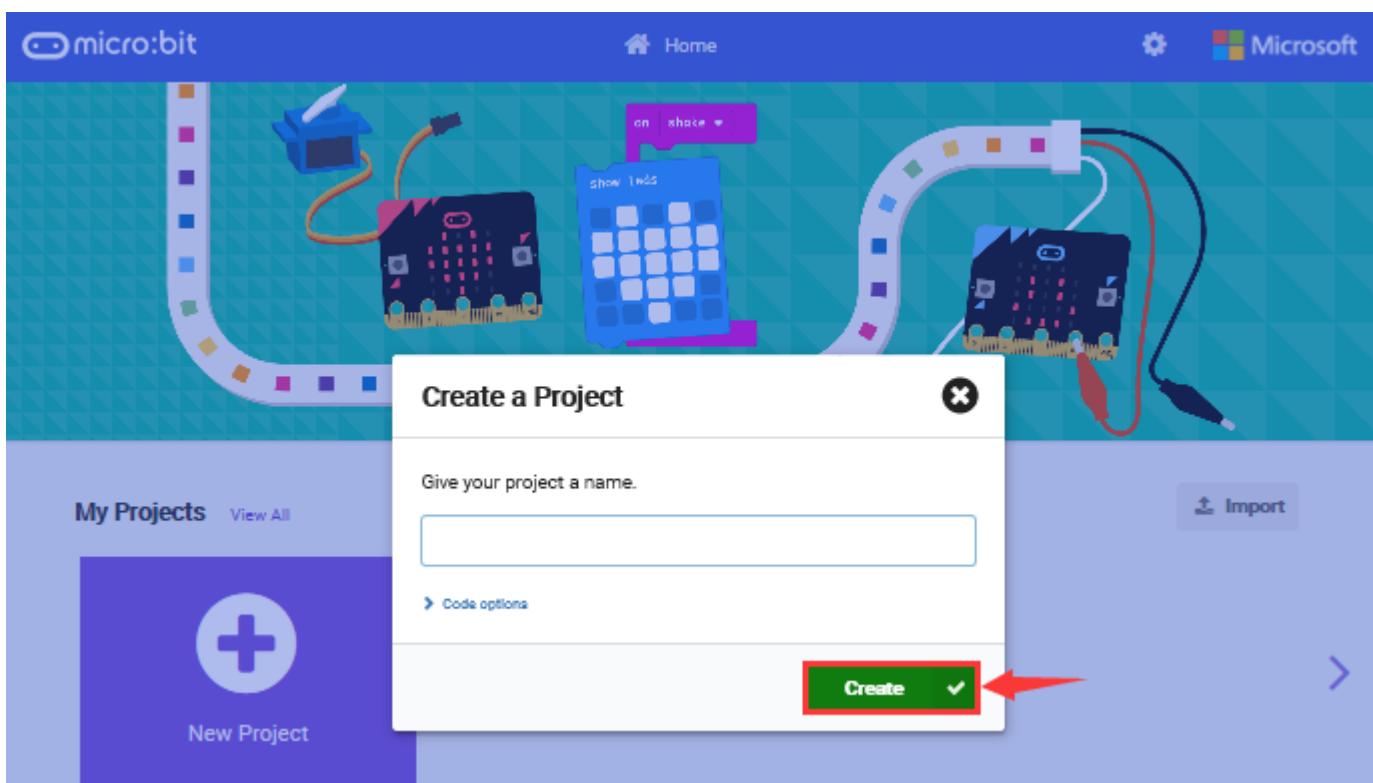
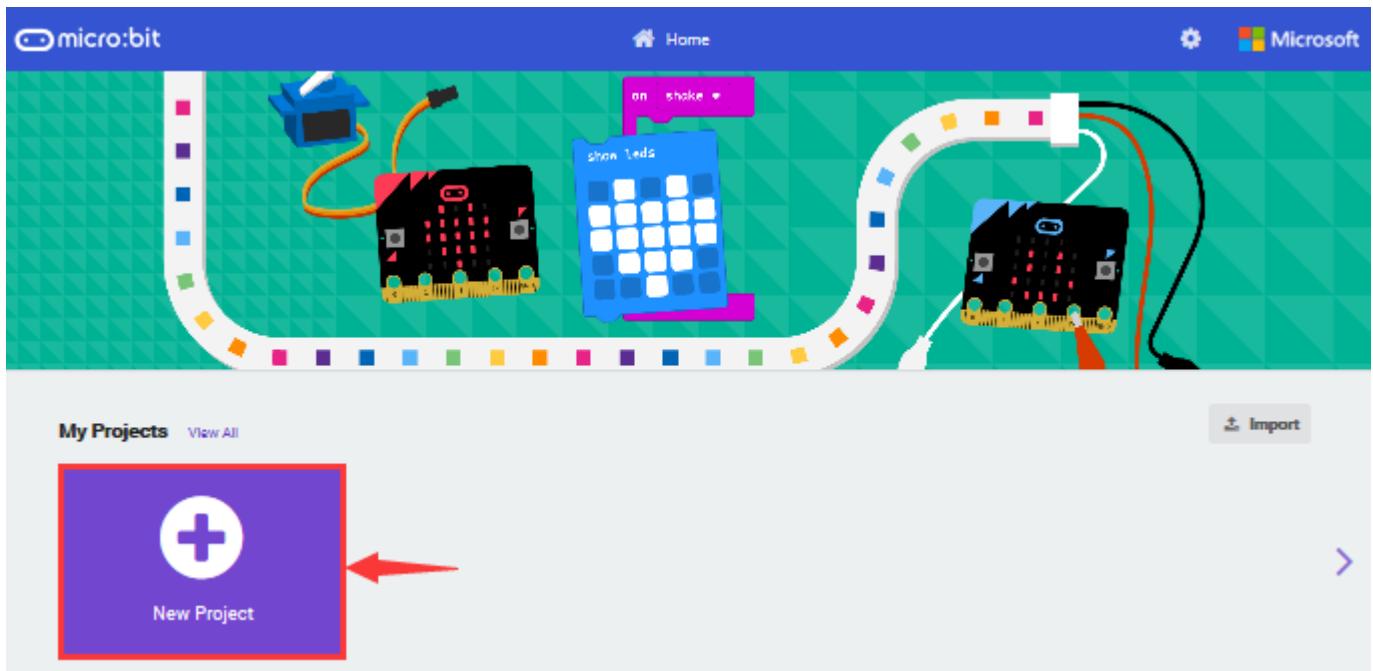
After the match with Bluetooth, write and upload code with the App.

Click “Create Code” to enter the programming page and write code.



Click **New Project** and the box appears, and then select “Create √” .







The screenshot shows the Microsoft micro:bit code editor interface. On the left, there's a preview of the micro:bit board with some LED patterns. The main area has a search bar and a sidebar with categories like Basic, Input, Music, etc. A script is open in the center:

```
on start
  led enable true
forever
  show leds
    [grid of 2x10 LEDs]
    pause (ms) 500
  show leds
    [grid of 2x10 LEDs]
    pause (ms) 500
```

At the bottom, there are download and save buttons, and a save icon highlighted with a red box.

Name the code as "1" and click  to save it.

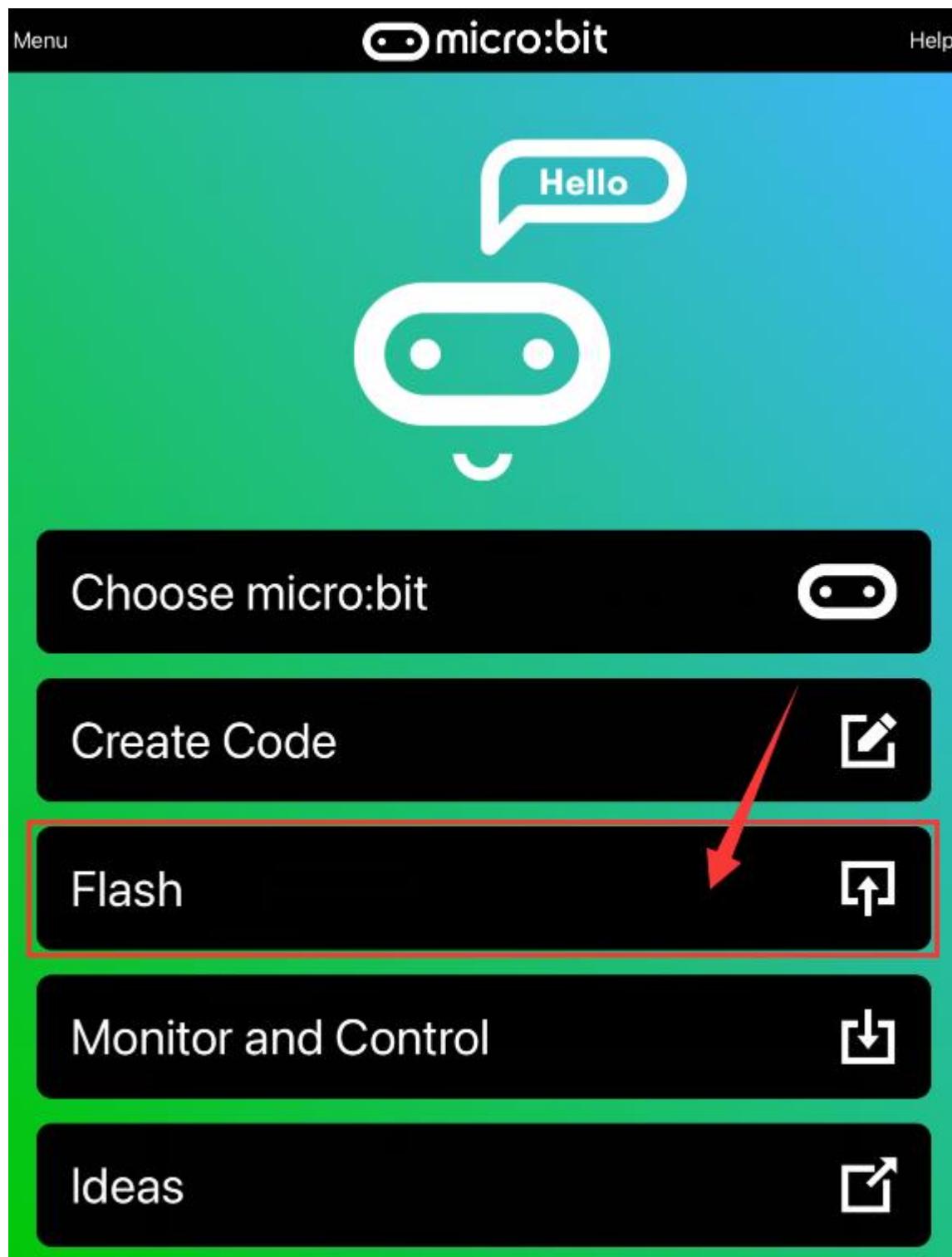


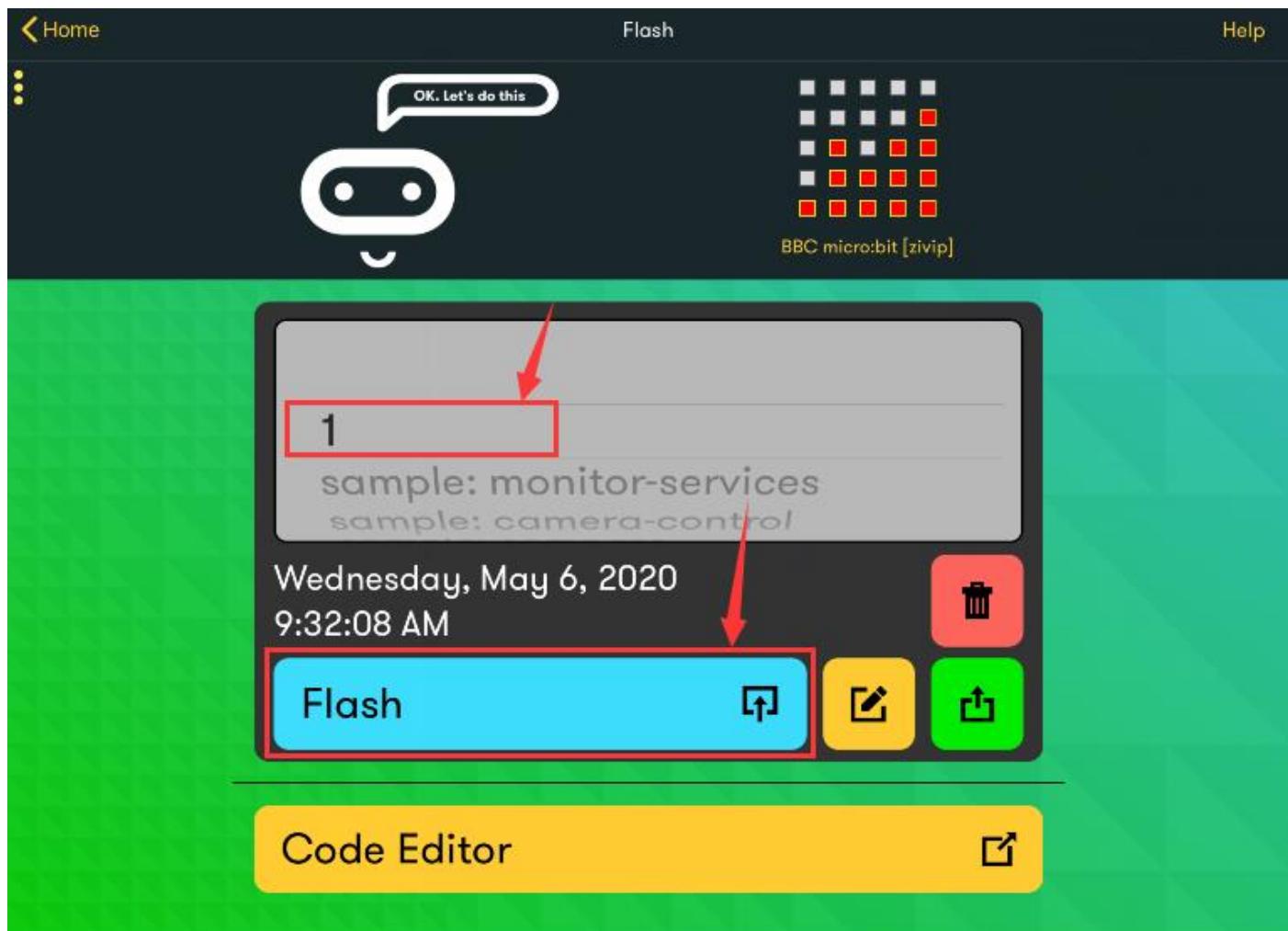
The screenshot shows the Microsoft MakeCode interface for the micro:bit. On the left, there's a preview window displaying a 5x5 grid of red pixels forming a pattern. Below it are various control buttons. The main workspace contains a script of blocks:

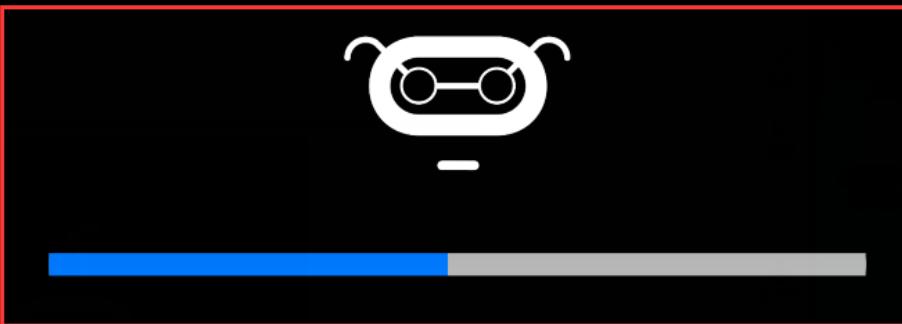
- An **on start** event block with a **led enable** block set to **true**.
- A **forever** loop block containing two **show leds** blocks. Each **show leds** block displays a 5x5 grid of alternating blue and white pixels. Between these two **show leds** blocks is a **pause (ms)** block set to **500**.
  - Below the first **show leds** block is another **show leds** block, also displaying a 5x5 grid of alternating blue and white pixels, with a **pause (ms)** block of **500** ms below it.

On the right, a sidebar lists categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, and Advanced. At the bottom, there are download and upload buttons. A red arrow points to the "Flash" button in the upload area.

Click the third item "Flash" to enter the uploading page. The default code program for uploading is the one saved just now and named "1" and then click the other "Flash" to upload the code program "1".





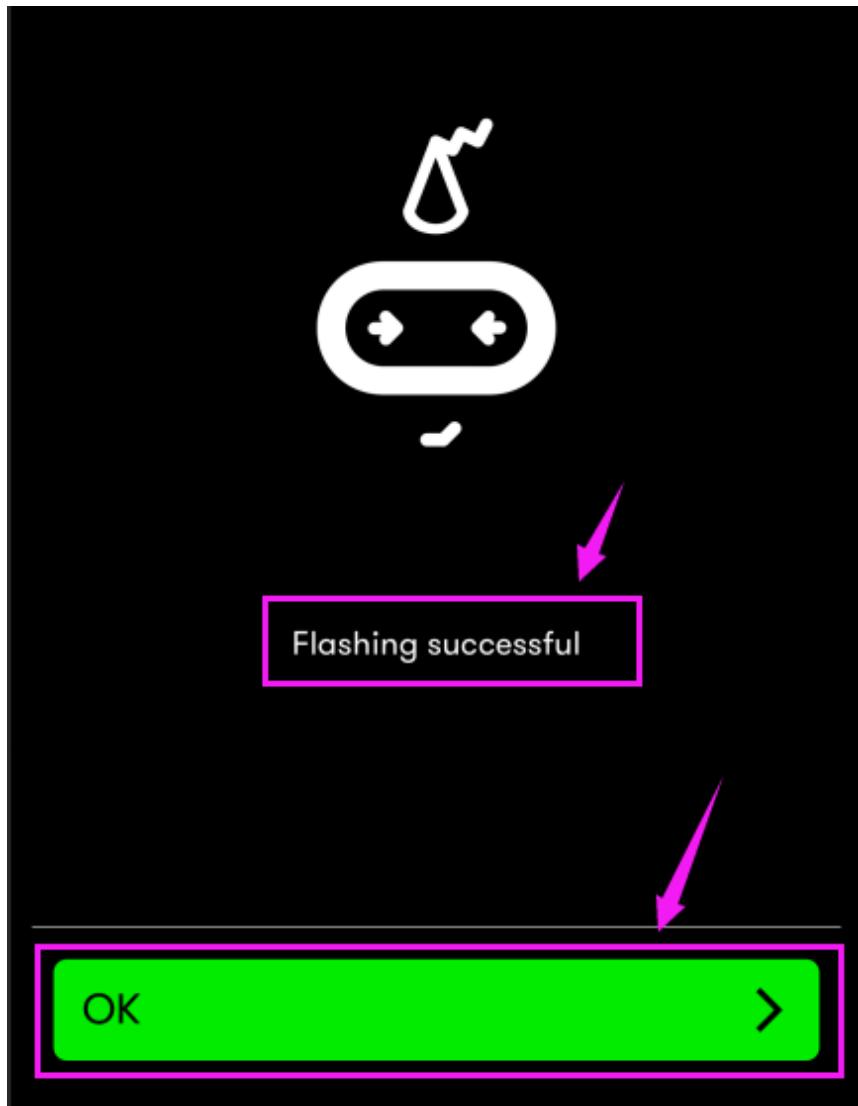


## Flashing code to micro:bit

Please do not interact with micro:bit until  
the process is complete

Sending...

If the code is uploaded successfully a few seconds later, the App will emerge as below and the LED dot matrix of the Micro: Bit main board V2 will exhibit a heart pattern.



Projects above all conduct with the built-in sensors and the LED dot matrix of the main board while the following ones will carry out with the help of external sensors of this turtle car.

**(Attention: to avoid burning the the Micro:bit main board V2, please remove the USB cable and the external power from the board before fix it with the shield of the car; likewise, the USB cable and the external power should be cut from the main board before disconnect the shield from the board.)**



## Project 13: Colorful Lights



### (1) Project Description

This module consists of a commonly used LED with 7 colors but in white appearance. It can automatically flash different colors to create fantastic light effects when high level is input like a normal LED.

### (2) Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- Open online Makecode editor

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

[\*\*\(How to add Mecanum Robot extension?\)\*\*](#)



### (3) Test Code

#### Code1

Make the RGB light flash 7 lights alternatively.

Code path:

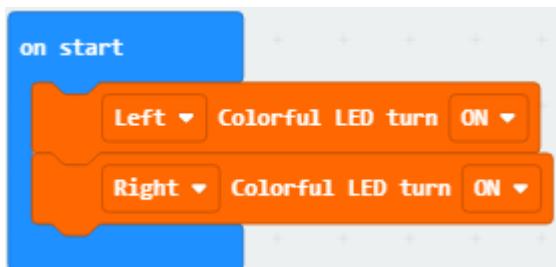
File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 13: Colorful Lights-1.hex	Project 13: Colorful Lights-1.hex

Or you could edit code step by step in the editing area.

(1) Click “MecanumRobot” → find and drag  指 to “on start” ;

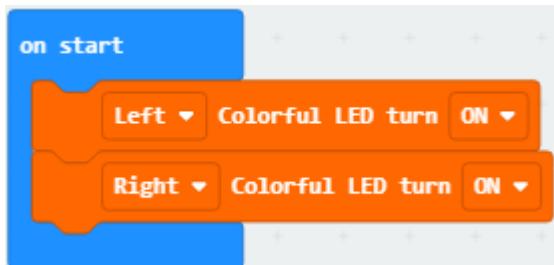
Copy  once;

Click the little triangle behind “Left” to choose “Right” :





## Compete Program:



.....①run "on start" once to start the program

.....②open the colorful light on the left of the car

.....③open the colorful light on the right of the car

Click "JavaScript" to view the corresponding JavaScript code: :



Microsoft | micro:bit

Blocks

JavaScript



Search...



Basic

Input

Music

Led

Radio

Loops

Logic

Variables

Math

MecanumRobot

IrRemote

Neopixel

Advanced

```
1 mecanumRobot.setLed(LedCount.Left, LedState.ON)
2 mecanumRobot.setLed(LedCount.Right, LedState.ON)
3 basic.forever(function () {
4
5 })
6
```

Download

...

Project 13 : microbit-七彩灯



## Code 2:

File Type	Path	File Name



Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 13: Colorful Lights-2.hex	Project 13: Colorful Lights-2.hex
----------	---	-----------------------------------

Or you could edit code step by step in the editing area.

(1) click "MecanumRobot" to find and drag



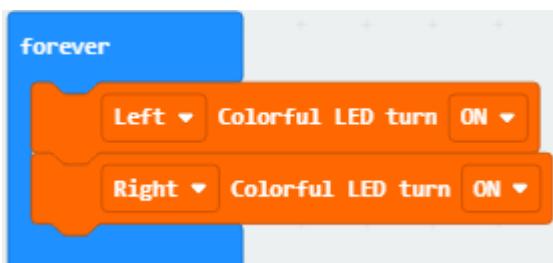
to "forever" ;

Copy



once;

Click the little triangle behind "Left" to choose "Right" :



(2) Click "Basic" to find and drag

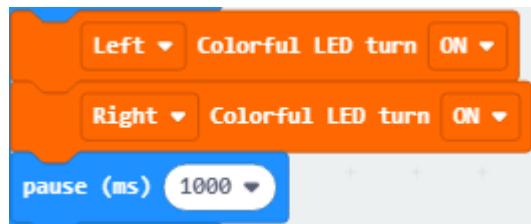


and tap the little triangle

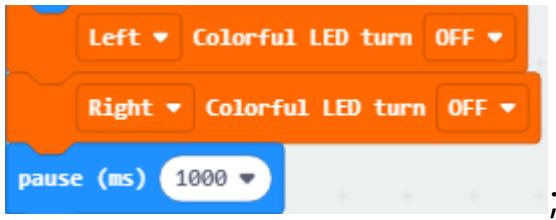
to choose "1 second" ;



Copy



once and choose OFF

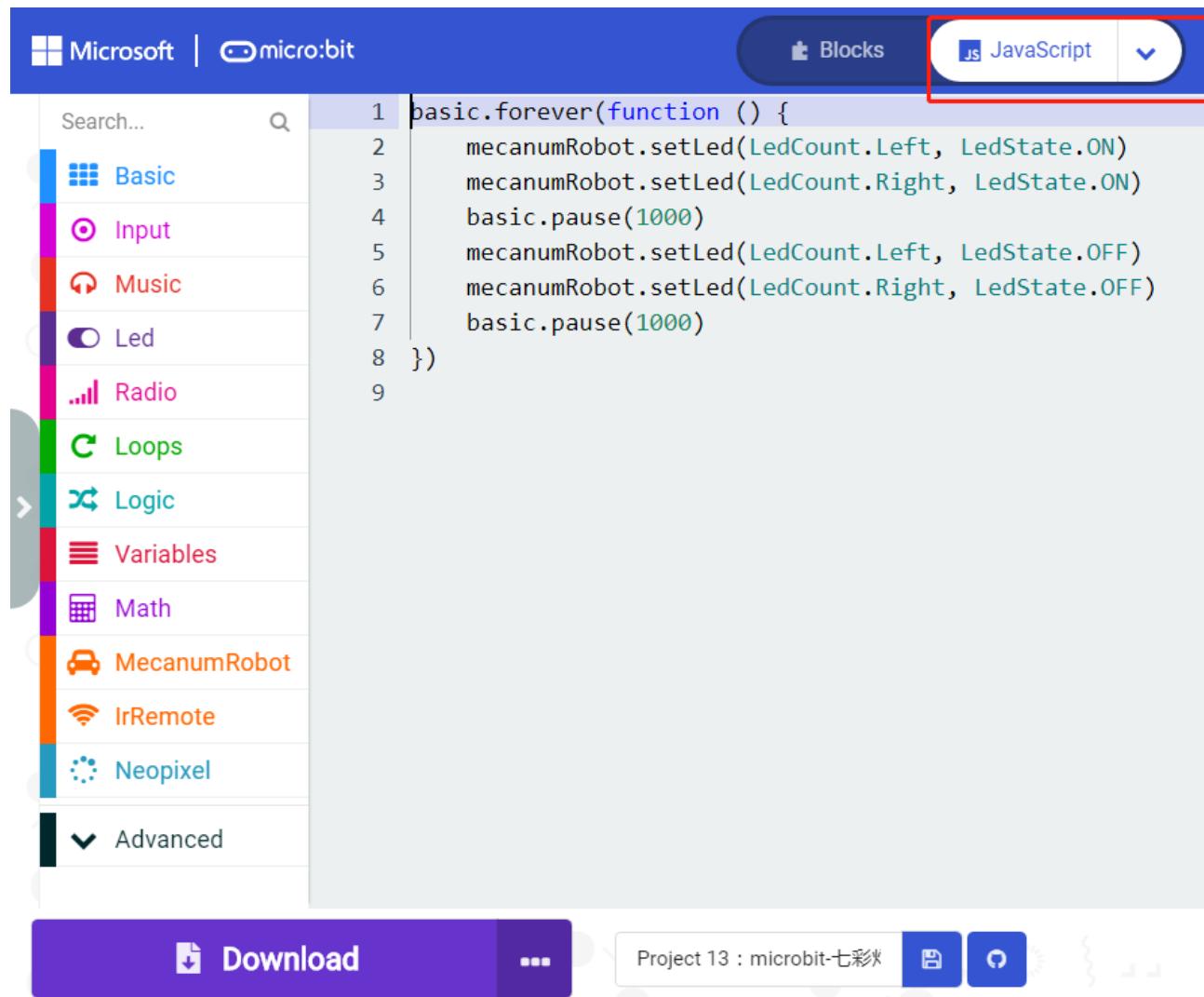


Put them in forever.

### Complete Program:



- ① In the "forever" instruction block, the program runs cyclically.
- ② Turn on the 2 colorful lights of the car.
- ③ Wait for 1 second
- ④ Turn off the 2 colorful lights of the car.
- ⑤ Wait for 1 second



The screenshot shows the Microsoft MakeCode interface for the micro:bit. At the top, there are tabs for "Blocks" and "JavaScript", with "JavaScript" being the active tab and highlighted by a red box. On the left, a sidebar lists categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, MecanumRobot, IrRemote, Neopixel, and Advanced. The main workspace contains the following JavaScript code:

```
1 basic.forever(function () {
2     mecanumRobot.setLed(LedCount.Left, LedState.ON)
3     mecanumRobot.setLed(LedCount.Right, LedState.ON)
4     basic.pause(1000)
5     mecanumRobot.setLed(LedCount.Left, LedState.OFF)
6     mecanumRobot.setLed(LedCount.Right, LedState.OFF)
7     basic.pause(1000)
8 })
```

At the bottom, there is a purple "Download" button with a white icon, and a status bar showing "Project 13 : microbit-七彩".

#### (4) Test Results:

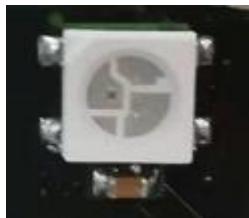
Download code 1 to micro:bit board and dial POWER switch to ON end, 2 RGB lights of smart car emit red, green, blue, indigo, dark red, yellow and white color cyclically.

Download code 2 to micro:bit board, 2 RGB lights show different color cyclically.

([How to download?](#) [How to quick download?](#))



## Project 14: WS2812 RGB LEDs



### (1) Project Description

The driver shield cooperates 4 pcs WS2812 RGB LEDs, compatible with micro:bit board and controlled by P8. In this lesson, we will make RGB LEDs display different colors by P8. In this lesson, 3 sets of test code are provided to make the 4 WS2812 RGB LEDs display different effects.

### (2) Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

#### [\*\*\(How to add Mecanum Robot extension?\)\*\*](#)

### (3) Test Code

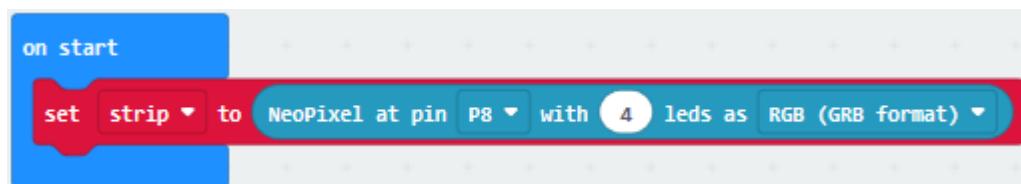
#### Code 1:



File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 14: WS2812 RGB LEDs-1.hex	Project 14: WS2812 RGB LEDs-1.hex

Or you could edit code step by step in the editing area.

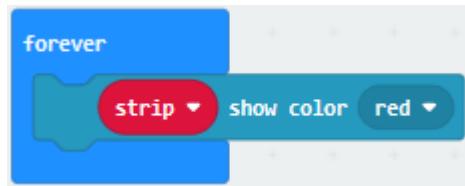
- a. Enter "Neopixel" → "set strip to Neopixel at pin P0 with 24 leds as RGB (GRB format)"
- b. Place it into "on start" block,
- c. Signal end P8 of WS2812 RGB is controlled by P8 of micro:bit . So we set to P8.
- d. Smart car has 4 pcs WS2812 RGB lights, so set to 4 leads



Click "Neopixel" to move block "strip clear" into "on start" block.



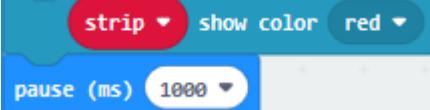
Enter "Neopixel" to move block "strip show color red" into "forever" block



Click "Basic" to move "pause (ms) 100" block into "forever" block

Then set to 1000ms



Copy code string  for eight times, and click red to respectively set to orange, yellow, green, blue, indigo, violet, purple and white.

Tap the triangle icon to select orange, yellow, green, blue, indigo, violet, purple and white.



```
forever
    strip show color red
    pause (ms) 1000
    strip show color orange
    pause (ms) 1000
    strip show color yellow
    pause (ms) 1000
    strip show color green
    pause (ms) 1000
    strip show color blue
    pause (ms) 1000
    strip show color indigo
    pause (ms) 1000
    strip show color violet
    pause (ms) 1000
    strip show color purple
    pause (ms) 1000
    strip show color white
    pause (ms) 1000
```

Complete Code



```
on start
  set strip to NeoPixel at pin P8 with 4 leds as RGB (GRB format)
  strip clear

forever
  strip show color red
  pause (ms) 1000
  strip show color orange
  pause (ms) 1000
  strip show color yellow
  pause (ms) 1000
  strip show color green
  pause (ms) 1000
  strip show color blue
  pause (ms) 1000
  strip show color indigo
  pause (ms) 1000
  strip show color violet
  pause (ms) 1000
  strip show color purple
  pause (ms) 1000
  strip show color white
  pause (ms) 1000
```



. "on start" : command block runs once to start program.

Set strip to Neopixel at pin P8 with 4 leads as RGB

Turn off 4pcs WS2812 RGB lights

The program under the block "forever" runs cyclically.

All RGB lights show red color

Delay in 1000ms

All RGB lights show orange color

Delay in 1000ms

All RGB lights show yellow color

Delay in 1000ms

All RGB lights show green color

Delay in 1000ms

All RGB lights show blue color

Delay in 1000ms

All RGB lights show indigo color

Delay in 1000ms

All RGB lights show violet color

Delay in 1000ms

All RGB lights show purple color

Delay in 1000ms

All RGB lights show white color

Delay in 1000ms

Click "JavaScript" to switch into the corresponding JavaScript code:



The screenshot shows the Makecode IDE interface. At the top, there are tabs for 'Blocks' and 'JavaScript'. A red arrow points from the 'JavaScript' tab towards the search bar. Below the tabs, there is a search bar with placeholder text 'Search...'. To the right of the search bar is a gear icon. On the far right, the Microsoft logo is visible. The main area contains a block of JavaScript code:

```
1 let strip = neopixel.create(DigitalPin.P8, 4, NeoPixelMode.RGB)
2 strip.clear()
3 basic.forever(function () {
4     strip.showColor(neopixel.colors(NeoPixelColors.Red))
5     basic.pause(1000)
6     strip.showColor(neopixel.colors(NeoPixelColors.Orange))
7     basic.pause(1000)
8     strip.showColor(neopixel.colors(NeoPixelColors.Yellow))
9     basic.pause(1000)
10    strip.showColor(neopixel.colors(NeoPixelColors.Green))
11    basic.pause(1000)
12    strip.showColor(neopixel.colors(NeoPixelColors.Blue))
13    basic.pause(1000)
14    strip.showColor(neopixel.colors(NeoPixelColors.Indigo))
15    basic.pause(1000)
16    strip.showColor(neopixel.colors(NeoPixelColors.Violet))
17    basic.pause(1000)
18    strip.showColor(neopixel.colors(NeoPixelColors.Purple))
19    basic.pause(1000)
20    strip.showColor(neopixel.colors(NeoPixelColors.White))
21    basic.pause(1000)
22 })
23
```

On the left side of the code editor, there is a sidebar with categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, TurtleBit, IrRemote, Neopixel, and Advanced. The 'Neopixel' category is currently selected.

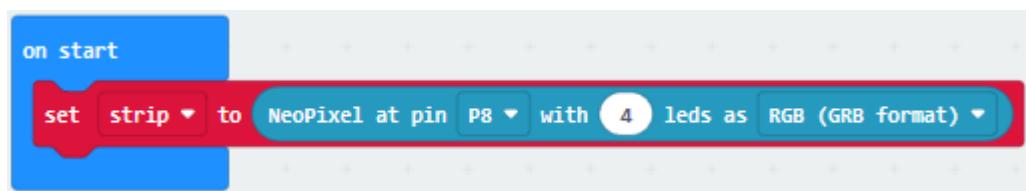
## Code 2:

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 14: WS2812 RGB LEDs-2.hex	Project 14: WS2812 RGB LEDs-2.hex

a. Enter "Neopixel" → "set strip to Neopixel at pin P0 with 24 leds as RGB (GRB format)"



- b. Place it into “on start” block,
- c. Signal end P8 of WS2812 RGB is controlled by P8 of micro:bit . So we set to P8.
- d. Smart car has 4 pcs WS2812 RGB lights, so set to 4 leads



Click “Loops” to drag “for index from 0 to 4...do” into “forever” block

Change 4 into 3



Click “Neopixel” to move block “strip clear” into block “for index from 0 to 3...do”

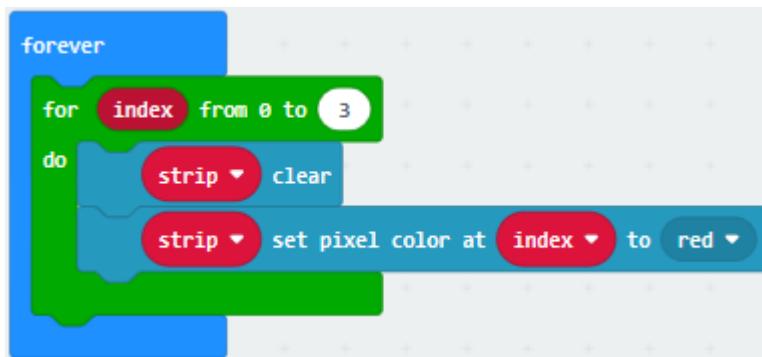


Tap “Neopixel” → “more” → “strip set pixel color at 0 to red”

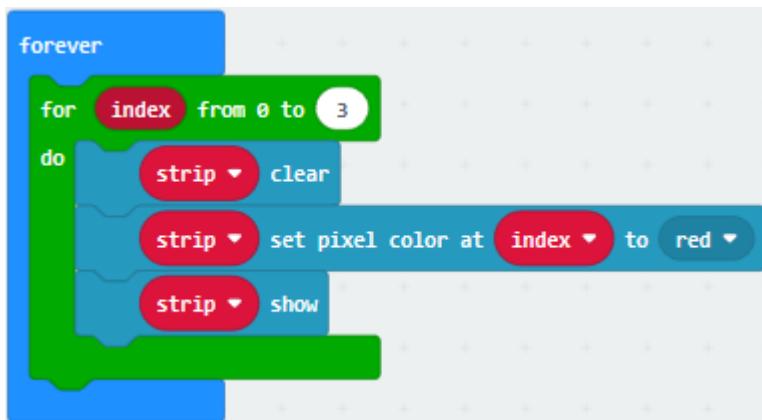


Place it into “for index from 0 to 3...do” block

Click “Variables” to move “index” into 0 box



(5) Click “Neopixel” to move “strip show” into “for index from 0 to 3...do” block



(6) Tap “Basic” to move “pause (ms) 100” block into “index from 0 to 3...do”



```
forever
  for [index] from 0 to [3]
    do
      strip [clear]
      strip [set pixel color at [index] to [red]]
      strip [show]
      pause [ms] [100]
```

```
for [index] from 0 to [3]
do
  strip [clear]
  strip [set pixel color at [index] to [red]]
  strip [show]
  pause [ms] [100]
```

Replicate code string for eight times and place them into “forever” block

Click red to respectively choose orange, yellow, green, blue, indigo, violet, purple and white



## Complete Code:

```
on start
  set strip ▾ to NeoPixel at pin P8 ▾ with 4 leds as RGB (GRB format) ▾
forever
  for index from 0 to 3
    do
      strip ▾ clear
      strip ▾ set pixel color at index ▾ to red ▾
      strip ▾ show
      pause (ms) 100 ▾
  for index from 0 to 3
    do
      strip ▾ clear
      strip ▾ set pixel color at index ▾ to orange ▾
      strip ▾ show
      pause (ms) 100 ▾
  for index from 0 to 3
    do
      strip ▾ clear
      strip ▾ set pixel color at index ▾ to yellow ▾
      strip ▾ show
      pause (ms) 100 ▾
```



"on start" : command block runs once to start program.

Set strip to Neopixel at pin p8 with 4 leads as RGB

The program under the block "forever" runs cyclically.

For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

Set index of WS2812 RGB lights to red color

Strip shows

Delay in 100ms

For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

Set index of WS2812 RGB lights to orange color

Strip shows

Delay in 100ms

For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

Set index of WS2812 RGB lights to yellow color

Strip shows

Delay in 100ms



```
for [index] from 0 to 3
do
    strip clear
    strip set pixel color at [index] to green
    strip show
    pause (ms) 100

for [index] from 0 to 3
do
    strip clear
    strip set pixel color at [index] to blue
    strip show
    pause (ms) 100

for [index] from 0 to 3
do
    strip clear
    strip set pixel color at [index] to indigo
    strip show
    pause (ms) 100
```



.For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

Set the index of WS2812 RGB lights to green color

Strip shows

Delay in 100ms

For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

Set the index of WS2812 RGB lights to blue color

strip shows

Delay in 100ms

For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

Set the index of WS2812 RGB lights to indigo color

Strip shows

Delay in 100ms

For index from 0 to 17, execute the program under do block

Turn off all RGB on strip

Set the index of WS2812 RGB lights to violet color

Set all RGB lights to show violet color

Strip displays all changes

Delay in 100ms

For index from 0 to 17, execute the program under do block

Turn off all RGB on strip

Set the index of WS2812 RGB lights to purple color

Strip displays all changes

Delay in 100ms

For index from 0 to 17, execute the program under do block

Turn off all RGB on strip

Set the index of WS2812 RGB lights to white color

Strip displays all changes

Delay in 100ms



```
for [index from 0 to 3]
do
    strip ▾ clear
    strip ▾ set pixel color at [index ▾] to [violet ▾]
    strip ▾ show
    pause (ms) 100 ▾

for [index from 0 to 3]
do
    strip ▾ clear
    strip ▾ set pixel color at [index ▾] to [purple ▾]
    strip ▾ show
    pause (ms) 100 ▾

for [index from 0 to 3]
do
    strip ▾ clear
    strip ▾ set pixel color at [index ▾] to [white ▾]
    strip ▾ show
    pause (ms) 100 ▾
```

For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

Set the index of WS2812 RGB lights to violet color

Strip shows

Delay in 100ms

For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

Set the index of WS2812 RGB lights to purple color

Strip shows

Delay in 100ms

For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB lights

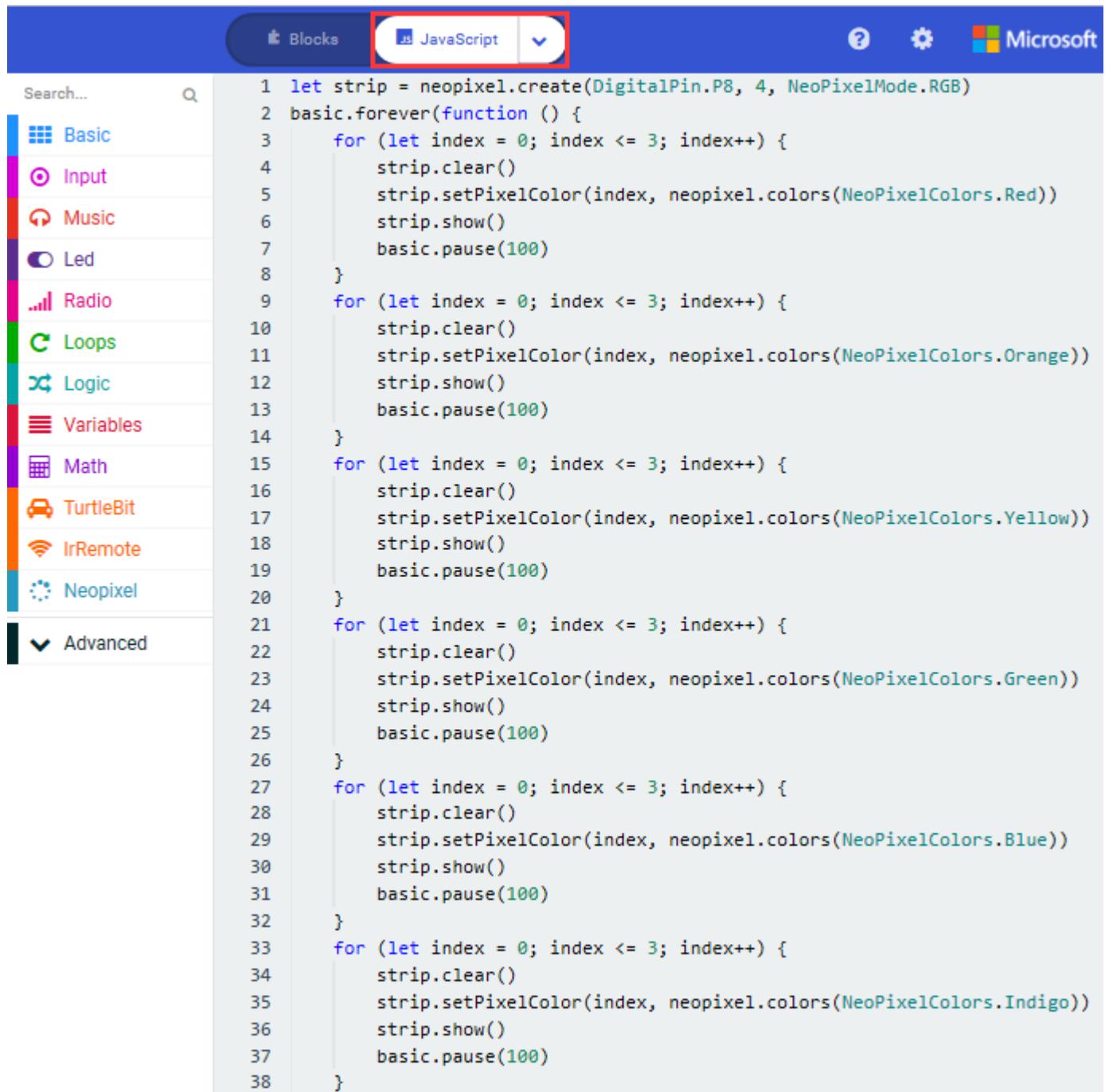
Set the index of WS2812 RGB lights to white color

Strip shows

Delay in 100ms



Click "JavaScript" to switch into the corresponding JavaScript code:



The screenshot shows the Scratch interface with the "JavaScript" tab selected. The code area contains the following JavaScript code:

```
1 let strip = neopixel.create(DigitalPin.P8, 4, NeoPixelMode.RGB)
2 basic.forever(function () {
3     for (let index = 0; index <= 3; index++) {
4         strip.clear()
5         strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Red))
6         strip.show()
7         basic.pause(100)
8     }
9     for (let index = 0; index <= 3; index++) {
10        strip.clear()
11        strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Orange))
12        strip.show()
13        basic.pause(100)
14    }
15    for (let index = 0; index <= 3; index++) {
16        strip.clear()
17        strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Yellow))
18        strip.show()
19        basic.pause(100)
20    }
21    for (let index = 0; index <= 3; index++) {
22        strip.clear()
23        strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Green))
24        strip.show()
25        basic.pause(100)
26    }
27    for (let index = 0; index <= 3; index++) {
28        strip.clear()
29        strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Blue))
30        strip.show()
31        basic.pause(100)
32    }
33    for (let index = 0; index <= 3; index++) {
34        strip.clear()
35        strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Indigo))
36        strip.show()
37        basic.pause(100)
38    }
```



```
39     for (let index = 0; index <= 3; index++) {
40         strip.clear()
41         strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Violet))
42         strip.show()
43         basic.pause(100)
44     }
45     for (let index = 0; index <= 3; index++) {
46         strip.clear()
47         strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Purple))
48         strip.show()
49         basic.pause(100)
50     }
51     for (let index = 0; index <= 3; index++) {
52         strip.clear()
53         strip.setPixelColor(index, neopixel.colors(NeoPixelColors.White))
54         strip.show()
55         basic.pause(100)
56     }
57 })
58 }
```

### Code 3:

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 14: WS2812 RGB LEDs-3.hex	Project 14: WS2812 RGB LEDs-3.hex

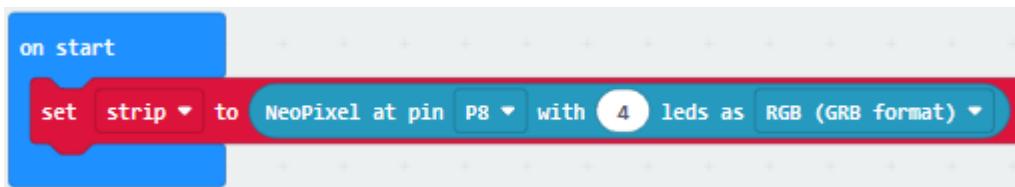
Or you could edit code step by step in the editing area.

- Enter "Neopixel" → "set strip to Neopixel at pin P0 with 24 leds as RGB (GRB format)"
- Place it into "on start" block,



c. Signal end P8 of WS2812 RGB is controlled by P8 of micro:bit . So we set to P8.

d. Smart car has 4 pcs WS2812 RGB lights, set to 4 leads



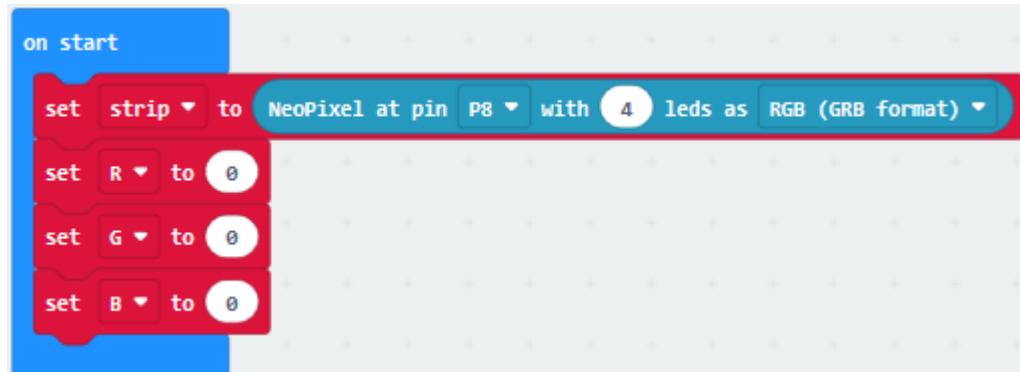
Click "Variables" → "Make a Variable..."

Input R to build up variable R

We create variable "G" and "B" in same way

Drag "set B to 0" into "on start" block

Copy "set B to 0" twice and click triangle button to choose G and B



Click "Loops" to get block "for index from 0 to 4...do"

Leave it into "forever" and change 4 into 3



Move block "set B to 0" into "for index from 0 to 3...do" block,

Click B to choose R

Go to "Math" to drag block "pick random 0 to 10" into 0 box

Change 0 into 10, 10 into 255



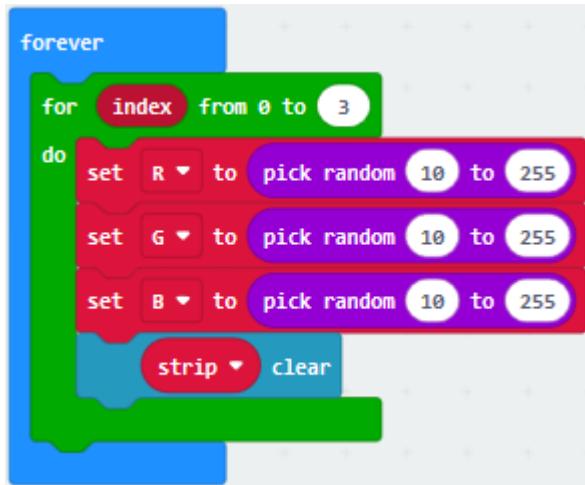
Replicate block  twice and place them into "for index from 0 to 3...do" block.

Click R to select G and B





Tap “Neopixel” and move “strip clear” into “for index from 0 to 3...do” block.



Go to “Neopixel” → “more” → “strip set pixel color at 0 to red”

Leave it in the block “for index from 0 to 3...do” block

Drag block “red 255 green 255 blue 255” into “red” box

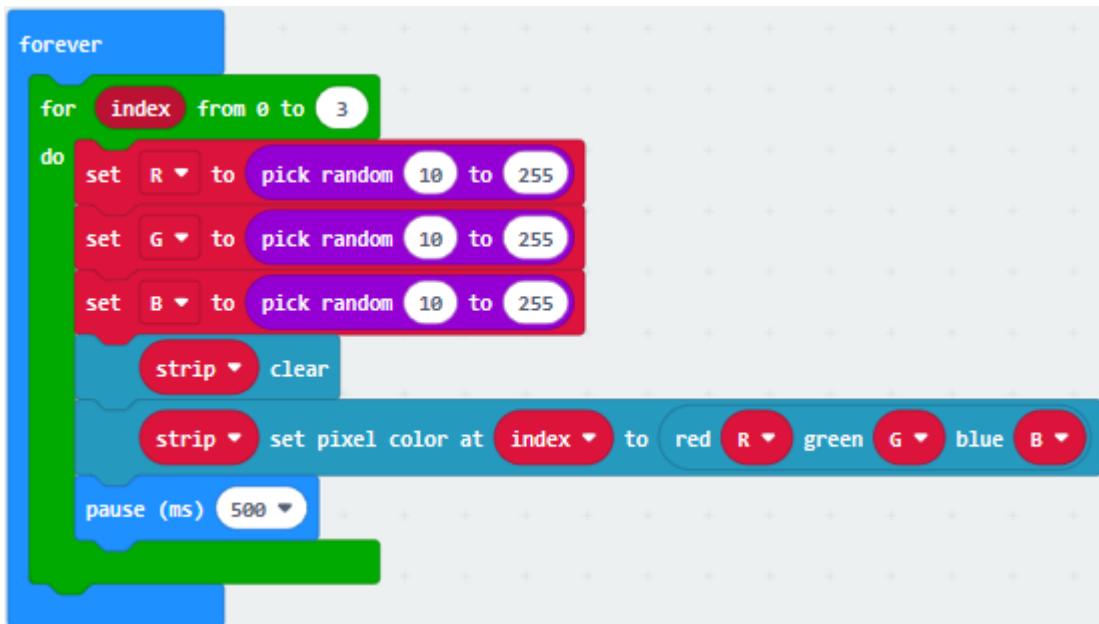
Tap “Variables” to move “index” block into 0 box

Separately drag R , G and B into 255 box, as shown below:



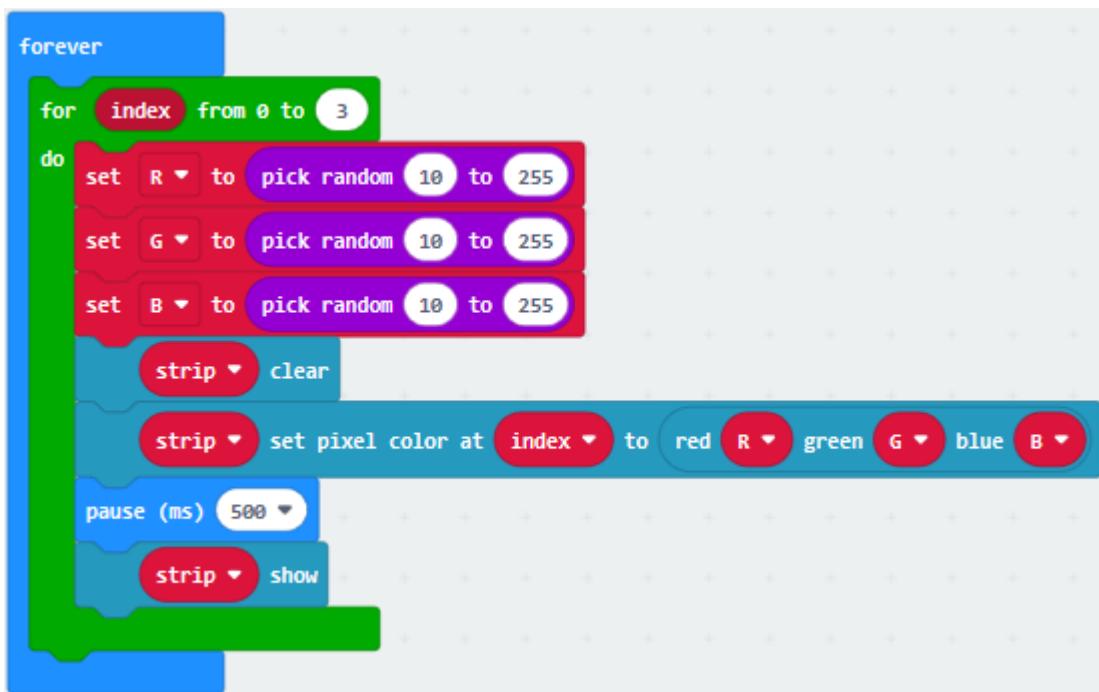
Click “Basic” to drag “pause (ms) 100” under block “strip....B”

Set to 500ms.



```
forever
  for [index v] from 0 to [3]
    do
      set [R v] to [pick random (10) to (255)]
      set [G v] to [pick random (10) to (255)]
      set [B v] to [pick random (10) to (255)]
      strip clear
      strip set pixel color at [index v] to [red R green G blue B]
      pause (ms) [500 v]
```

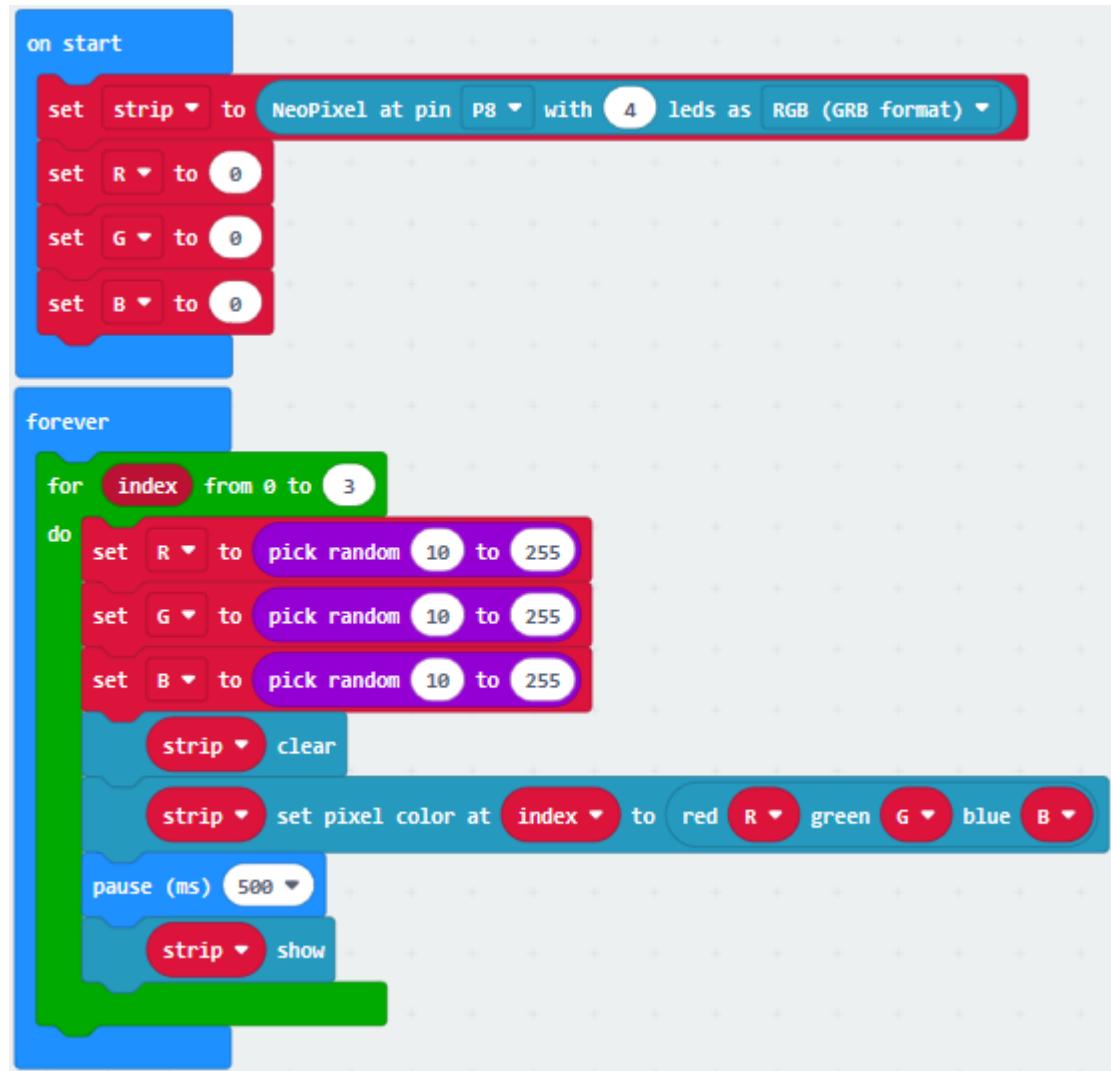
Click “Neopixel” to move “strip show” block under “pause(as) 500”



```
forever
  for [index v] from 0 to [3]
    do
      set [R v] to [pick random (10) to (255)]
      set [G v] to [pick random (10) to (255)]
      set [B v] to [pick random (10) to (255)]
      strip clear
      strip set pixel color at [index v] to [red R green G blue B]
      pause (ms) [500 v]
      strip show
```



## Complete Code:



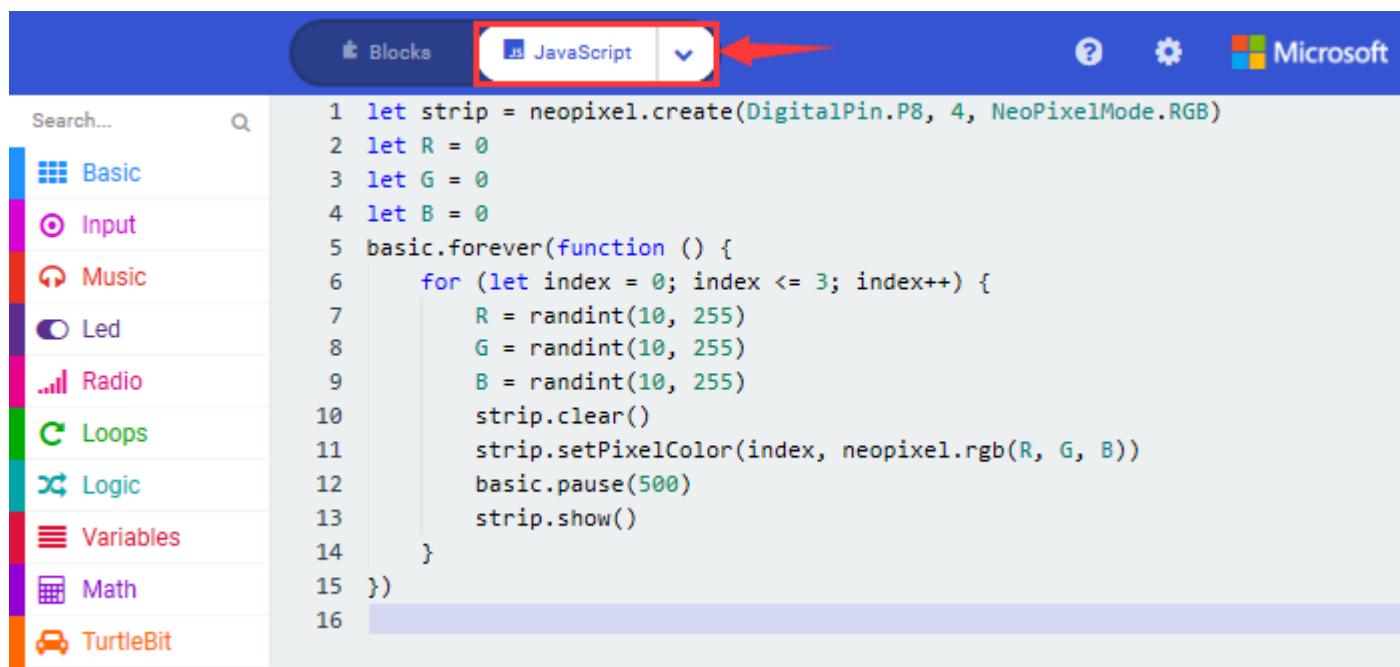
```
on start
  set [strip v] to [NeoPixel at pin P8 with 4 leds as RGB (GRB format)]
  set [R v] to [0]
  set [G v] to [0]
  set [B v] to [0]

forever
  for [index v] from [0] to [3]
    do
      set [R v] to [pick random (10) to (255)]
      set [G v] to [pick random (10) to (255)]
      set [B v] to [pick random (10) to (255)]
      strip clear
      strip set pixel color at [index v] to [red R v green G v blue B v]
      pause (ms) [500 v]
      strip show
```



"on start" : command block runs once to start program.  
Set strip to Neopixel at pin p8 with 4 leads as RGB(GRB format)  
Set variable R to 0  
Set variable G to 0  
Set variable B to 0  
The program under the block "forever" runs cyclically.  
When the value of index is in 0-3, execute the program under do block  
Set variable R to random number in 10-255  
Set variable G to random number in 10-255  
Set variable B to random number in 10-255  
Turn off all RGB on strip  
Set index of 4 pcs WS2812 RGB lights to RGB(red, green, blue)  
Delay in 500ms  
Strip shows

Click "JavaScript" to switch into the corresponding JavaScript code:



The screenshot shows the Scratch interface with the code switcher set to "JavaScript". A red arrow points to the "JavaScript" button in the top menu bar. The left sidebar shows categories like Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, and TurtleBit. The main area contains the following JavaScript code:

```
1 let strip = neopixel.create(DigitalPin.P8, 4, NeoPixelMode.RGB)
2 let R = 0
3 let G = 0
4 let B = 0
5 basic.forever(function () {
6     for (let index = 0; index <= 3; index++) {
7         R = randint(10, 255)
8         G = randint(10, 255)
9         B = randint(10, 255)
10        strip.clear()
11        strip.setPixelColor(index, neopixel.rgb(R, G, B))
12        basic.pause(500)
13        strip.show()
14    }
15 })
16
```

#### (4) Test Results:

Download code 1 to micro: bit, and dial POWER to ON end. All four



---

WS2812RGB LEDs light up a different color a time cyclically.

Download code 2 to micro: bit, WS2812RGB LEDs display like flow light.

Download code 3 to micro: bit, every WS2812RGB light shows random color one by one.

([How to download?](#) [How to quick download?](#))

## Project 15: Servo



### (1) Project Description

For those DIY smart cars, they often have the function of automatic obstacle avoidance. In the DIY process, we need a servo to control the ultrasonic module to rotate left and right, and then detect the distance between the car and the obstacle, so as to control the car to avoid the obstacle. If other microcontrollers are used to control the rotation of the servo, we need to set a certain frequency and a certain width of pulse to control the servo angle. But if the micro:bit main board is used to control the servo angle, we only need to set the control angle in

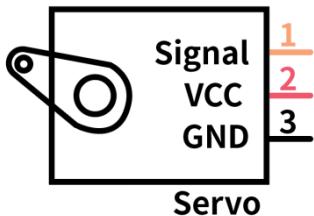


---

the development environment where the corresponding pulse will be automatically set to control the servo rotation. In this project, you will learn how to control the servo to rotate back and forth between 0° and 90°.

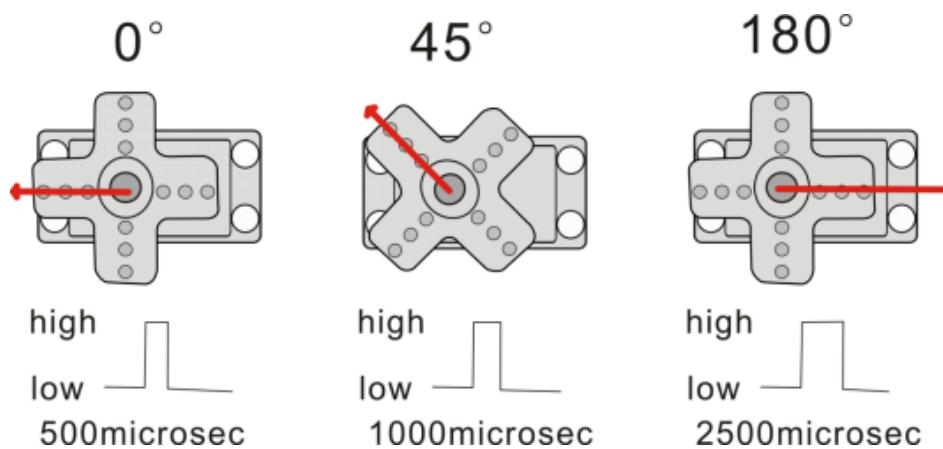
## (2)Background Information of the Servo

Servo motor is a position control rotary actuator. It mainly consists of housing, circuit board, core-less motor, gear and position sensor. Its working principle is that the servo receives the signal sent by MCU or receiver, and produces a reference signal with a period of 20ms and width of 1.5ms, then compares the acquired DC bias voltage to the voltage of the potentiometer and obtains the voltage difference output.



For the servo used in this project, the brown wire is 1 the ground, the red one is the positive wire, and the orange one is the signal wire.

The rotation angle of servo motor is controlled by regulating the duty cycle of PWM (Pulse-Width Modulation) signal. The standard cycle of PWM signal is 20ms (50Hz). Theoretically, the width is distributed between 1ms-2ms, but in fact, it's between 0.5ms-2.5ms. The width corresponds to the rotation angle from 0° to 180°. But note that for different brand motor, the same signal may have different rotation angle.



### More details:

High level time	Servo angle
0.5ms	0 degree
1ms	45 degree
1.5ms	90 degree
2ms	135 degree
2.5ms	180 degree

### (3) Parameters:

- ◆ Working voltage: DC 4.8V ~ 6V
- ◆ Operating angle range: about 180 ° (at 500 → 2500 µsec)
- ◆ Pulse width range: 500 → 2500 µsec
- ◆ No-load speed: 0.12 ± 0.01 sec / 60 (DC 4.8V) 0.1 ± 0.01 sec / 60 (DC 6V)
- ◆ No-load current: 200 ± 20mA (DC 4.8V) 220 ± 20mA (DC 6V)



- ◆ Stopping torque:  $1.3 \pm 0.01\text{kg} \cdot \text{cm}$  (DC 4.8V)  $1.5 \pm 0.1\text{kg} \cdot \text{cm}$  (DC 6V)
- ◆ Stop current:  $\leq 850\text{mA}$  (DC 4.8V)  $\leq 1000\text{mA}$  (DC 6V)
- ◆ Standby current:  $3 \pm 1\text{mA}$  (DC 4.8V)  $4 \pm 1\text{mA}$  (DC 6V)

#### (4) Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- Open online Makecode editor

Import Hex profile [\(How to import?\)](#), or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

[\(How to add Mecanum Robot extension?\)](#)

#### (5) Test Code:

Code path:

File Type	Path	File Name

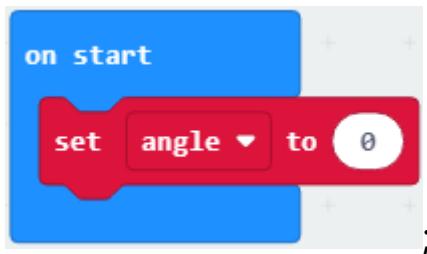


Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 15: Servo.hex	Project 15: Servo.hex
----------	---	-----------------------

Or you could edit code step by step in the editing area.

(1) Click "Variables"; motor "Make a Variable name" create a variable named

"angle" ; set the value to 0: ; and then put it into

"on start" ;



(2) Click "Loops" to find and drag  to "forever" and change the number to "180" ; click "MecanumRobot" to find and drag

 and put variable "angle" into

 ; put into 



```
for index from 0 to 180
do set servo to angle angle ▾
;
```

Click  of "Variable" and  of

"Math" ;put variable" angle" on the left and change the number on the

right to 1:  ;put it into  :

```
set angle to angle + 1
;
```

Put behind  and add delay in 10ms

```
for index from 0 to 180
do set servo to angle angle ▾
    set angle to angle + 1
    pause (ms) 10 ▾
;

```



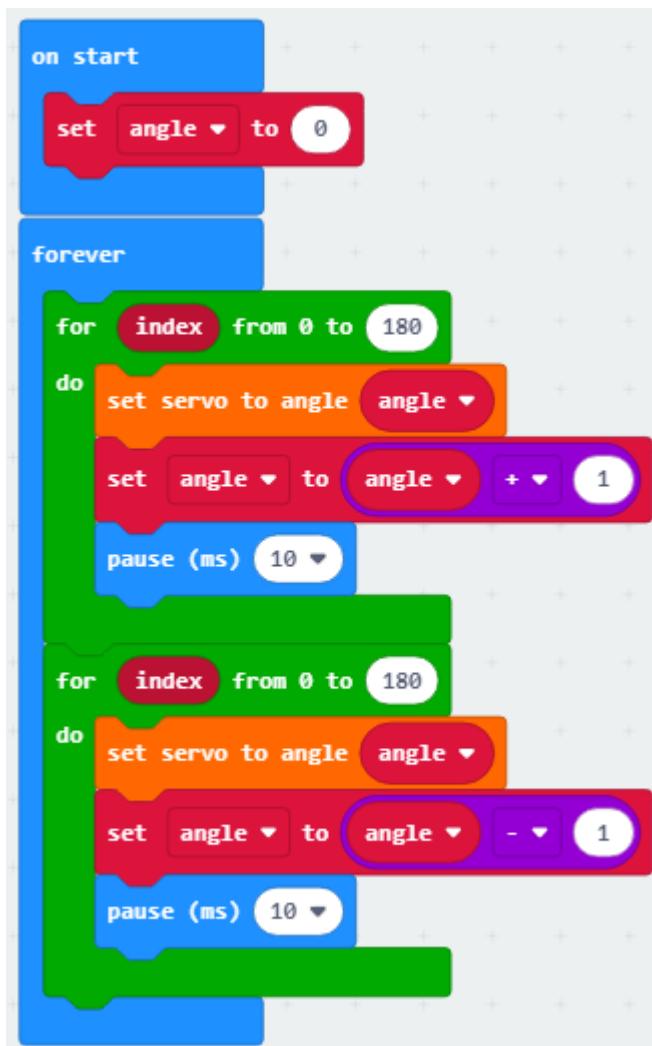
```
for [index v] from 0 to [180 v]
  do
    [set servo to angle <angle> v]
    [set angle v to (angle v) + (1 v)]
    [pause (10 ms) v]
```

Copy once and change the "+"

```
for [index v] from 0 to [180 v]
  do
    [set servo to angle <angle> v]
    [set angle v to (angle v) - (1 v)]
    [pause (10 ms) v]
```

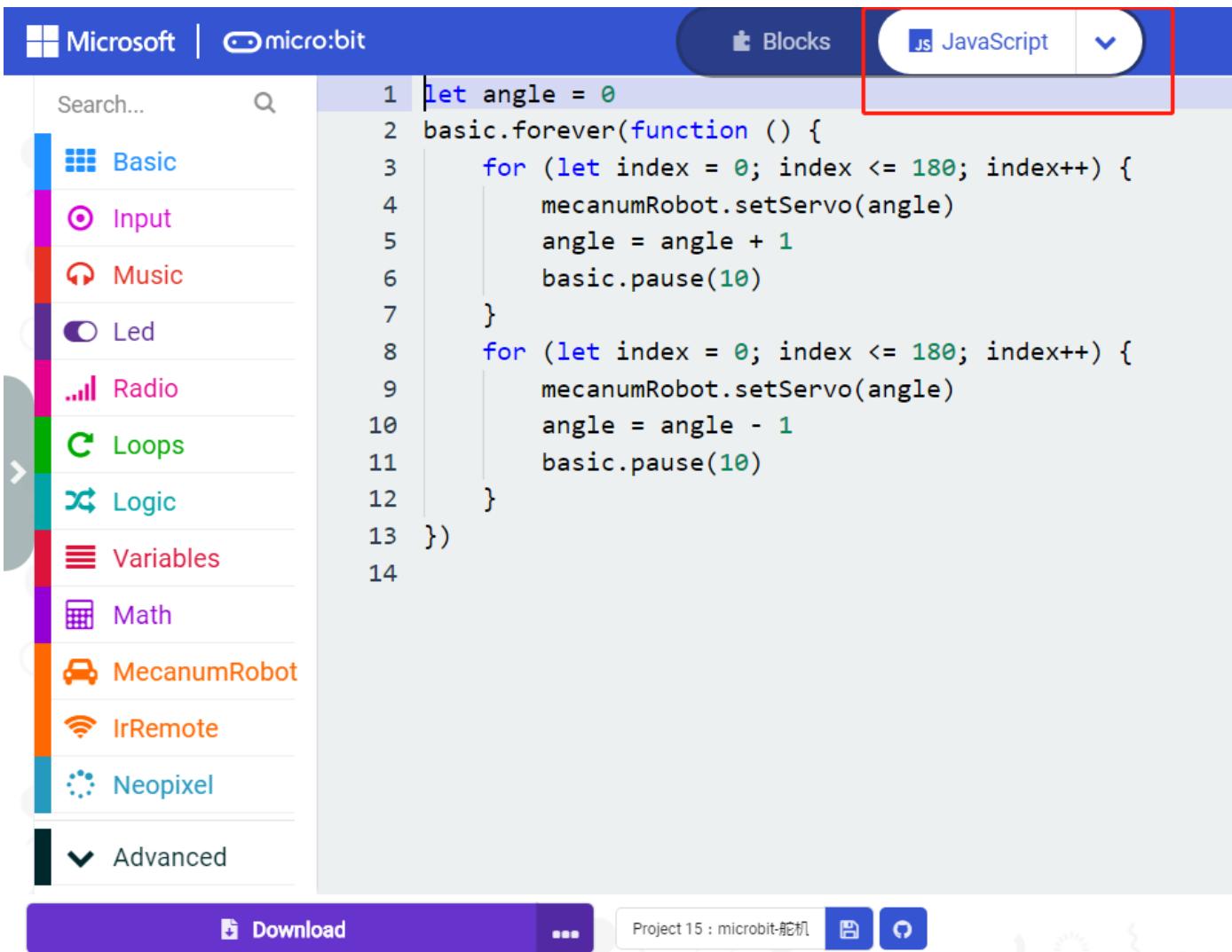
of to "-" :

Complete Program:



- ① The "on start" command block runs only once to start the program.
- ② Set the initial value of the angle variable to 0.
- ③ In the "forever" command box, the program runs cyclically
- ④ Cycle 180 times
- ⑤ Rotate the servo to angle
- ⑥ Angle variable increases 1
- ⑦ Delay in 10ms
- ⑧ Cycle 180 times
- ⑨ The servo rotates to angle
- ⑩ Angle angle variable minus 1
- ⑪ Delay in 10ms

Click "JavaScript" to view the corresponding JavaScript code: :



The screenshot shows the Microsoft micro:bit code editor interface. At the top, there are tabs for 'Blocks' and 'JavaScript', with 'JavaScript' being the active tab and highlighted by a red box. On the left, a sidebar lists categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, MecanumRobot, IrRemote, Neopixel, and Advanced. Below the sidebar is a search bar. The main area contains the following JavaScript code:

```
1 let angle = 0
2 basic.forever(function () {
3     for (let index = 0; index <= 180; index++) {
4         mecanumRobot.setServo(angle)
5         angle = angle + 1
6         basic.pause(10)
7     }
8     for (let index = 0; index <= 180; index++) {
9         mecanumRobot.setServo(angle)
10        angle = angle - 1
11        basic.pause(10)
12    }
13 })
14 
```

At the bottom, there are buttons for 'Download' and three small icons. To the right, it says 'Project 15 : microbit-舵机'.

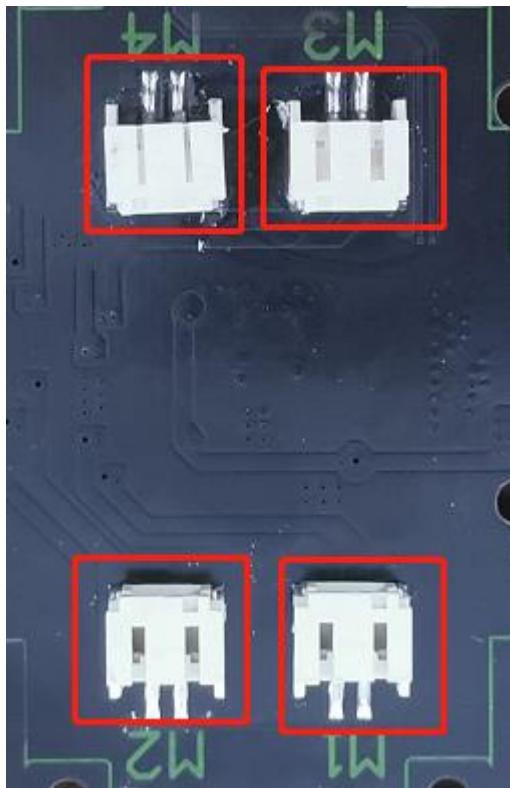
## (6) Test Results:

After uploading the test code and dial POWER switch to ON end, the servo rotates from 0 degree to 180 degrees.

[\(How to download?\)](#) [\(How to quick download?\)](#)



## Project 16: Motor



### (1) Project Description

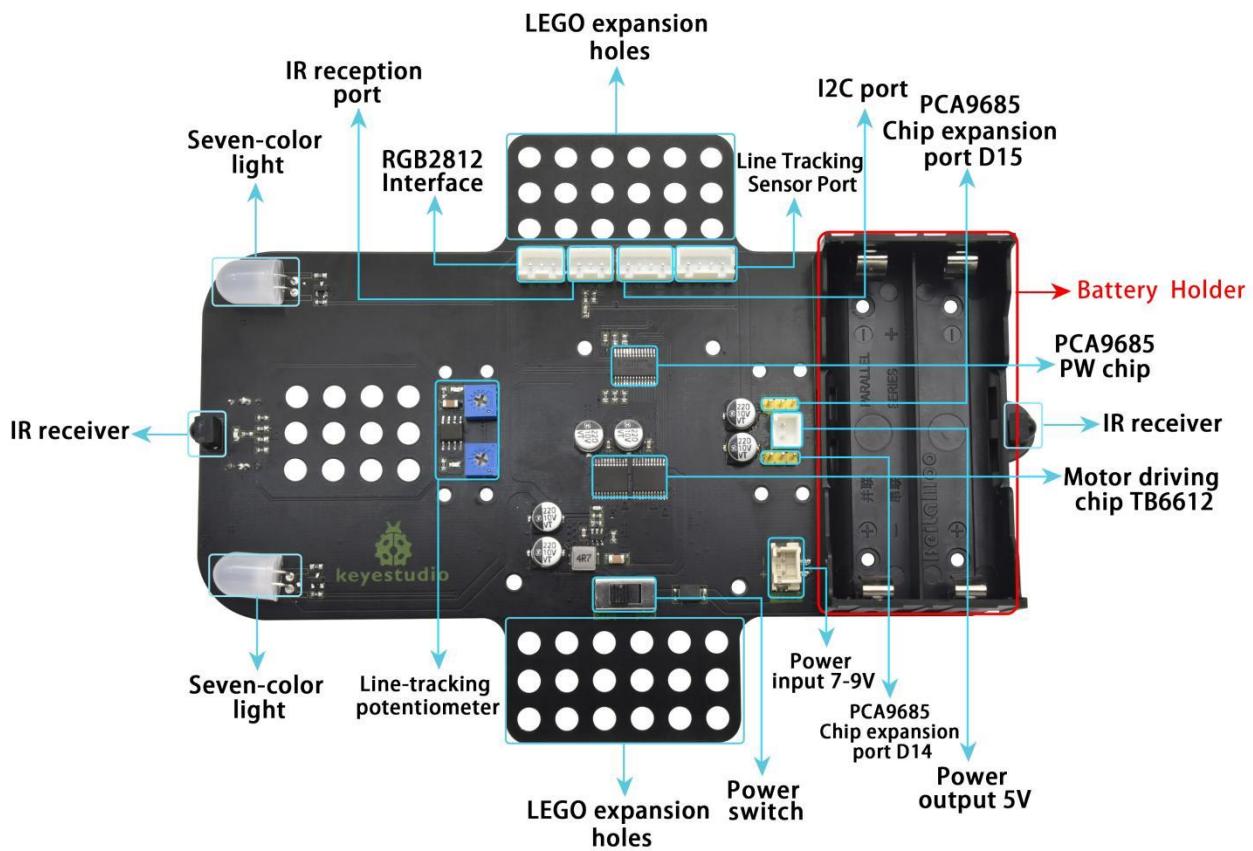
The Keyestudio 4WD Mecanum Robot Car is equipped with 4 DC reduction motors, also called gear reduction motor, which is developed on the ordinary DC motor. It has a matching gear reduction box which provides a lower speed but a larger torque. Furthermore, different reduction ratios of the box can provide different speeds and torques.

Gear motor is the integration of servomotor and motor, which is applied widely in steel and machine industry

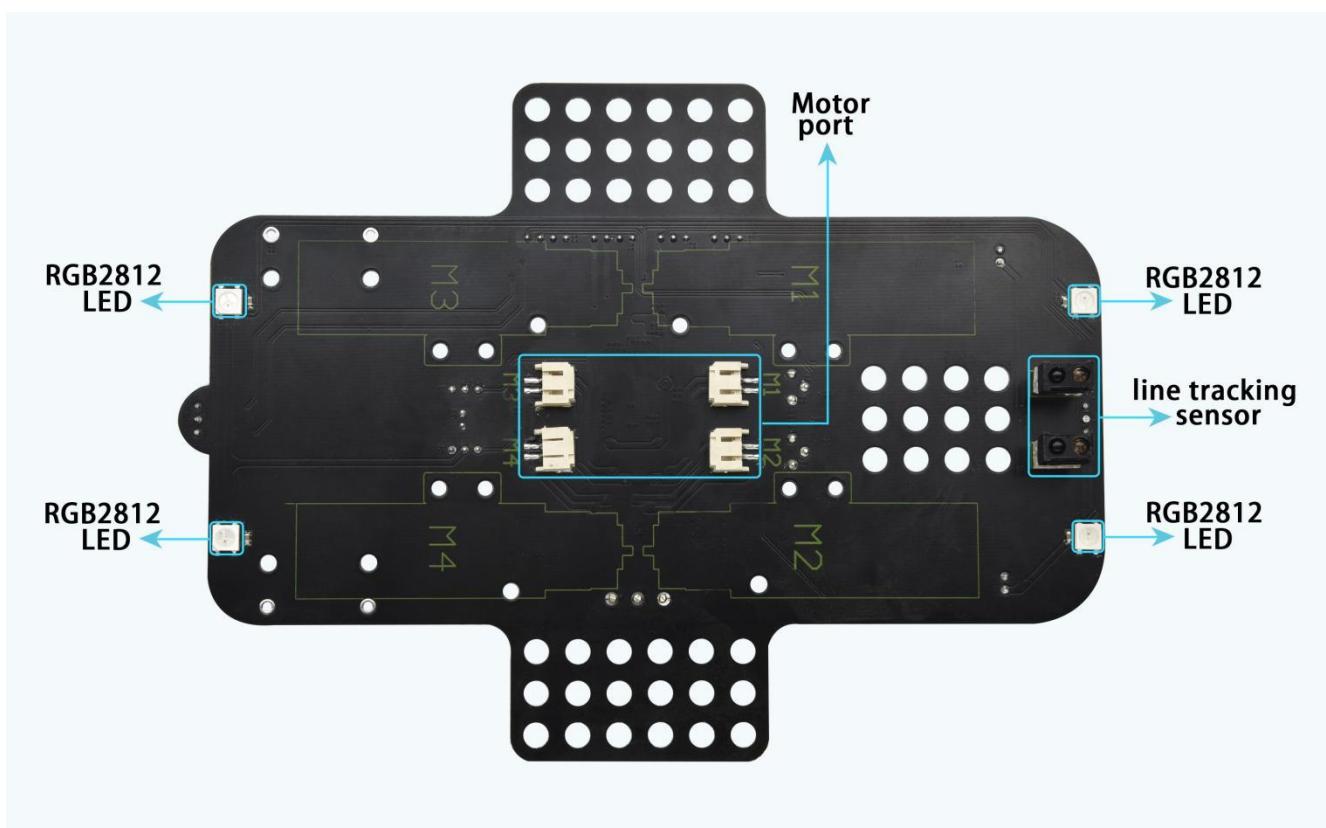


Micro:bit motor driver shield comes with PCA9685PW and TB6612FNG chip. In order to save the IO port resource, we control the rotation direction and speed of two DC gear motors with TB6612FNG chip.

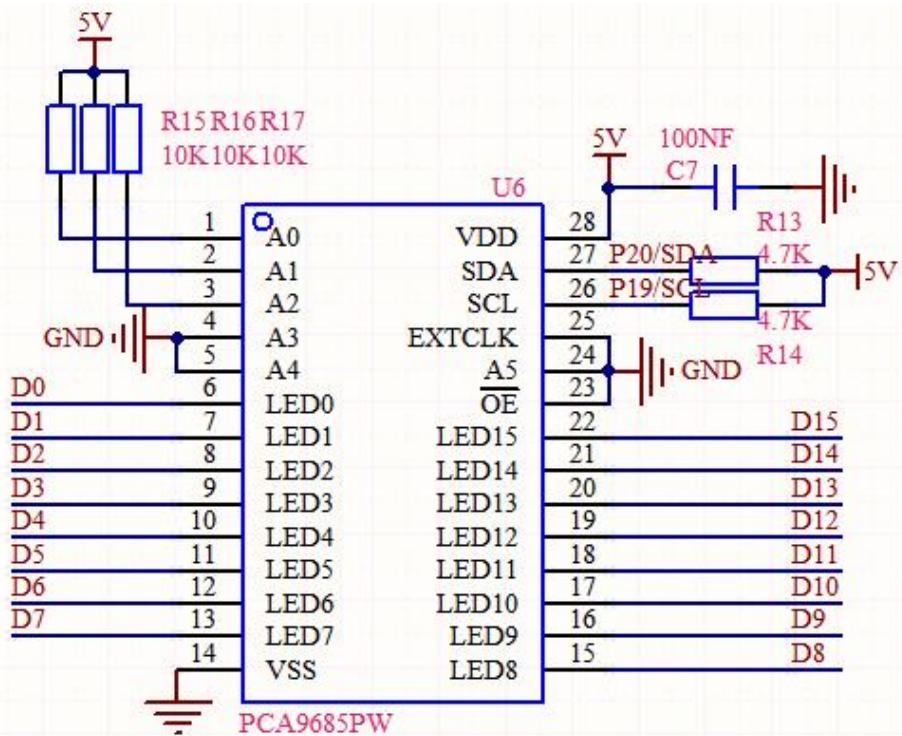
## Details about chips:



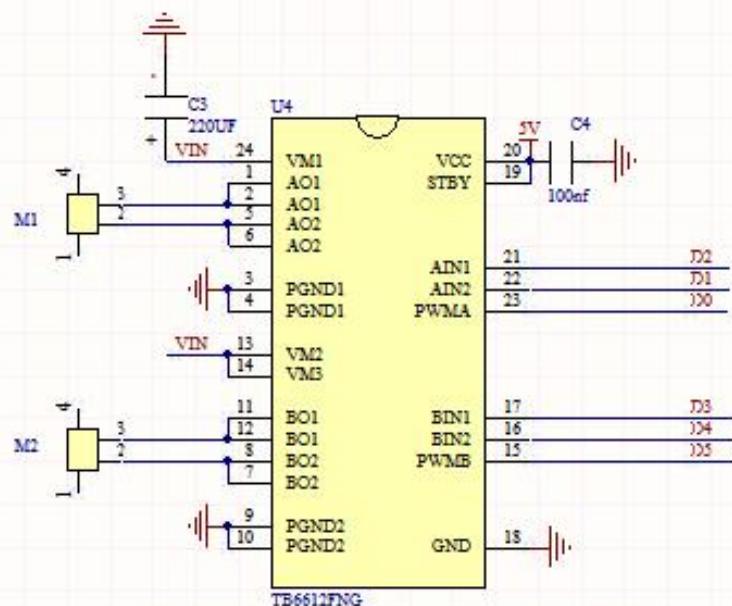
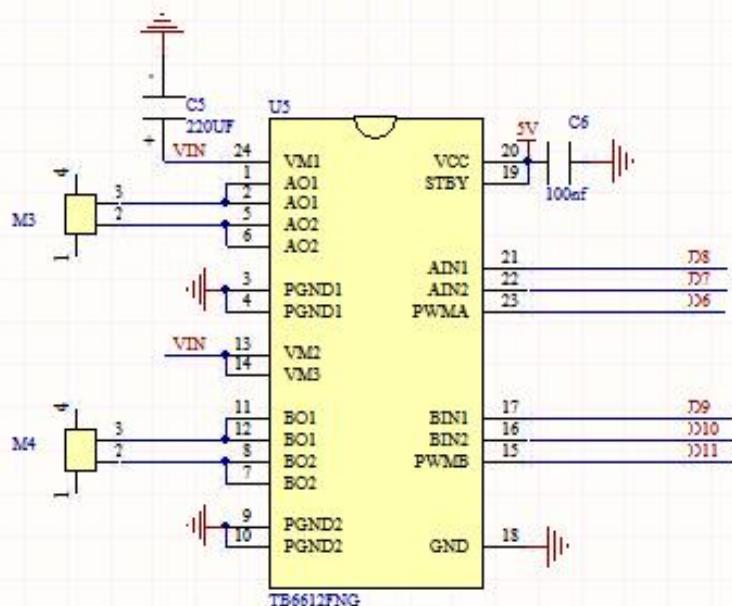
**Front**



Back



## PCA9685PW Module



## Motor Control Chip

### (2) Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end



- Connect micro:bit to computer by USB cable
- Open online Makecode editor

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

### **(How to add MecanumRobot extension?)**

### **(3)Test Code:**

#### **Code 1:**

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 16: Motor-1.hex	Project 16: Motor-1.hex

Or you could edit code step by step in the editing area.

(1)Click “ MecanumRobot ” to find and drag



into “forever”;click the number



behind speed to choose 75



(2)Copy



four times; click the little triangle behind "Motor" to choose Lower\_left, Upper\_right, Lower\_right respectively; and put them all in forever



(2)Click "Basic" to find and drag "pause (ms) 100" to "forever" ;set delay

in 2000ms



(3)Click "MecanumRobot" to find and drag



to

pause (ms) 2000 ;

copy pause (ms) 2000

once and put it behind





## Complete Program:



- ① In the "forever" instruction block, the program runs cyclically.
- ② Set the front left motor speed to 75, and rotate clockwise.
- ③ Set the speed of the rear left motor to 75 and the direction to rotate clockwise.
- ④ Set the front right motor speed to 75 and the direction to rotate clockwise.
- ⑤ Set the right rear motor speed to 75 and the direction to rotate clockwise
- ⑥ The delay time is 2000 milliseconds
- ⑦ 4 motors stop rotating
- ⑧ Delay time 2000 milliseconds

Click "JavaScript" to view the corresponding JavaScript code: :



Microsoft | micro:bit

Blocks JavaScript ▾

Search... Q

Basic

Input

Music

Led

Radio

Loops

Logic

Variables

Math

MecanumRobot

IrRemote

Neopixel

Advanced

```
1 basic.forever(function () {  
2     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)  
3     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)  
4     mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)  
5     mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)  
6     basic.pause(2000)  
7     mecanumRobot.state(MotorState.stop)  
8     basic.pause(2000)  
9 })  
10 }
```

Download ... Project 16 : microbit-电机

## Code2:

Code path:

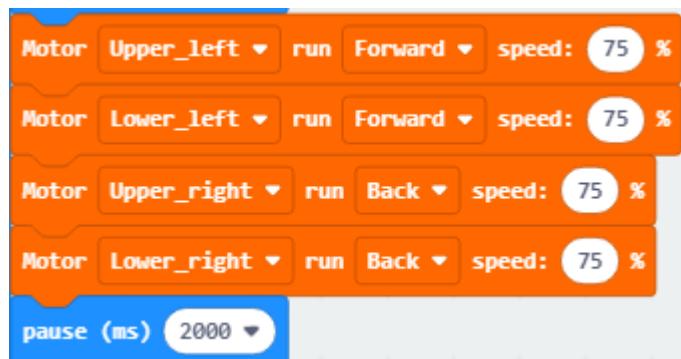
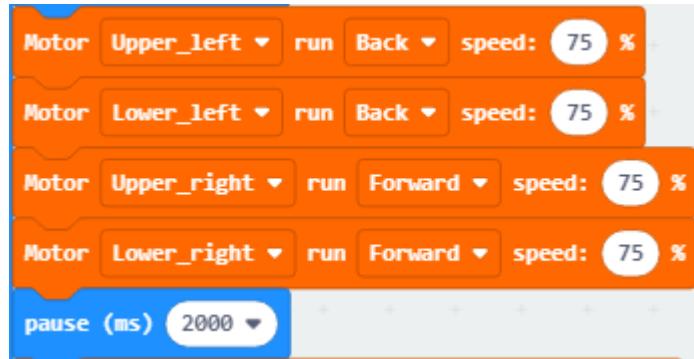
File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 16: Motor-2.hex	Project 16: Motor-2.hex

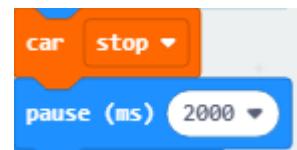


Or you could edit code step by step in the editing area.



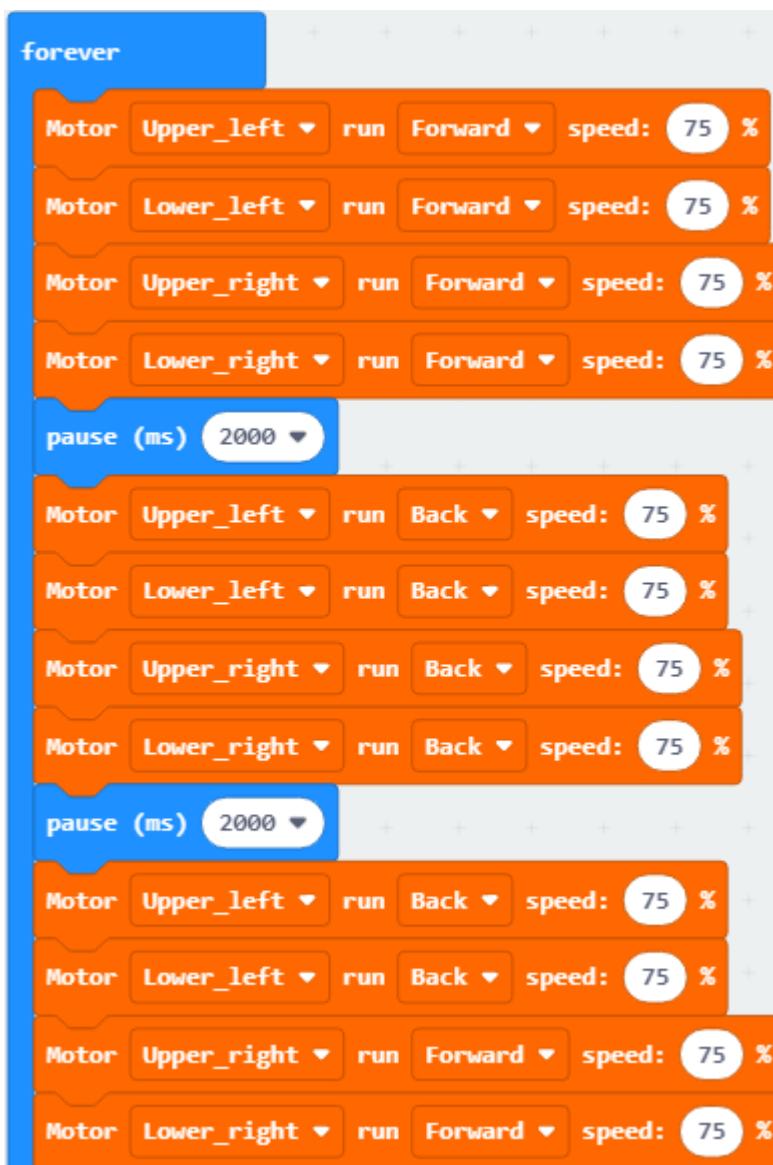
(1) Drag and copy three times; click the little triangle behind "run" to choose as shown



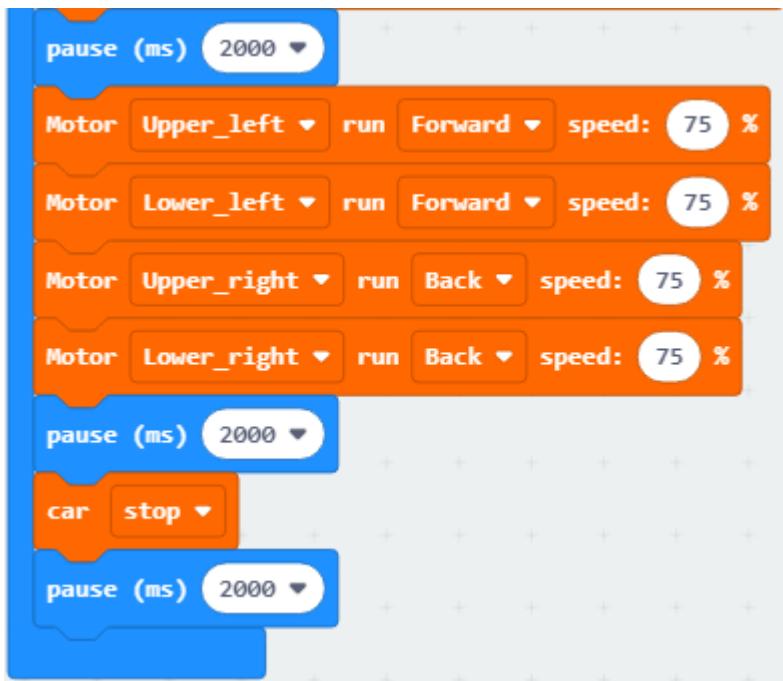


And then put they all in forever and add .

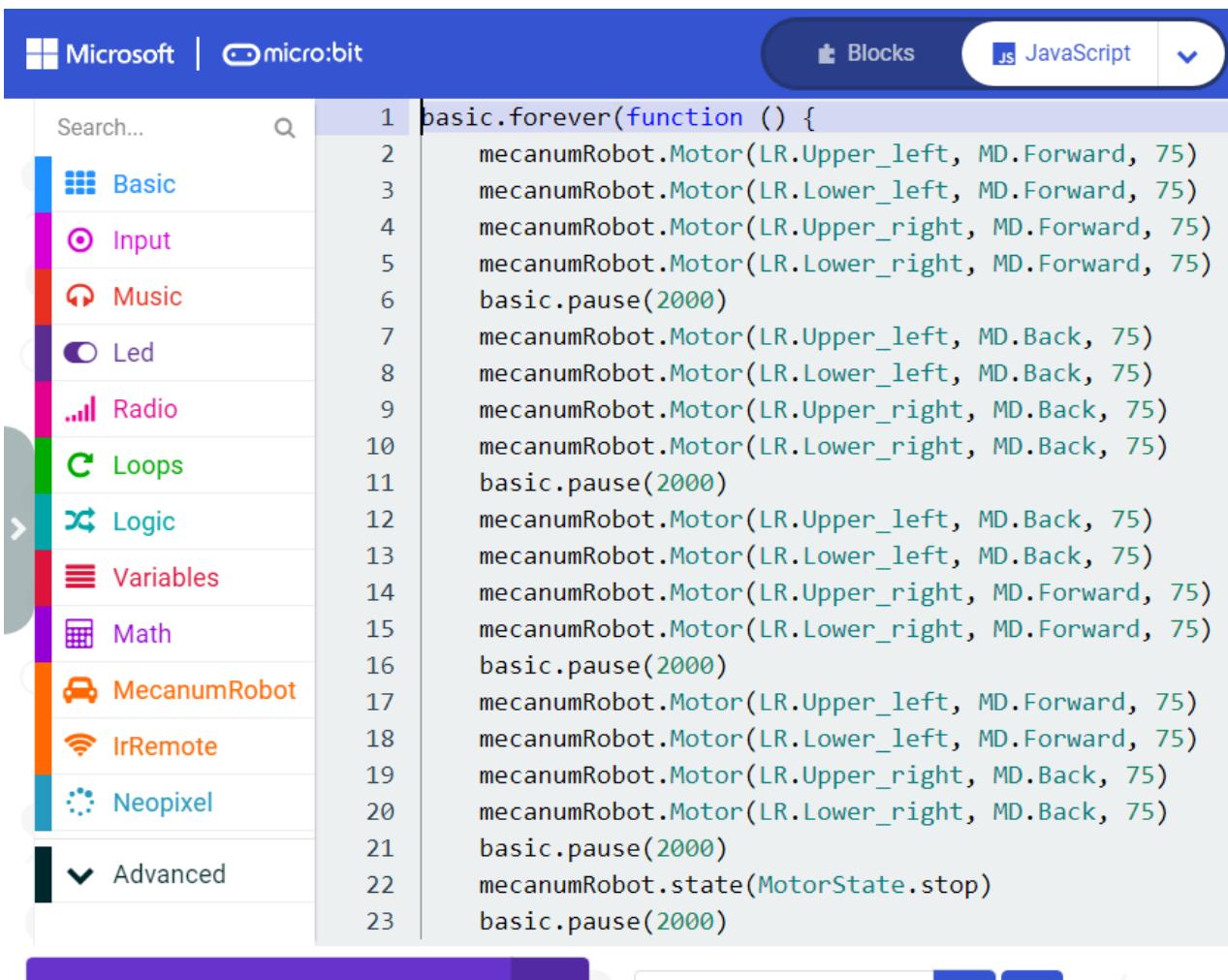
## Complete Program:



- ① In the "forever" instruction block, the program runs cyclically.
- ② Set the front left motor speed to 75 and the direction to rotate forward.
- ③ Set the speed of the rear left motor to 75 and the direction to rotate forward.
- ④ Set the front right motor speed to 75 and the direction to rotate forward.
- ⑤ Set the right rear motor speed to 75 and the direction to rotate forward.
- ⑥ Wait for 2 seconds
- ⑦ Set the front left motor speed to 75, and the direction is reversed.
- ⑧ Set the motor speed at the rear left to 75, and the direction is reversed.
- ⑨ Set the front right motor speed to 75, and the direction is reversed.
- ⑩ Set the right rear motor speed to 75, and the direction is reversed.
- ⑪ Wait for 2 seconds
- ⑫ Set the front left motor speed to 75, and the direction is reversed.
- ⑬ Set the motor speed at the rear left to 75, and the direction is reversed.
- ⑭ Set the front right motor speed to 75 and the direction to rotate forward.
- ⑮ Set the right rear motor speed to 75, the direction is forward.
- ⑯ Wait for 2 seconds
- ⑰ Set the front left motor speed to 75 and the direction to rotate forward.
- ⑱ Set the speed of the rear left motor to 75 and the direction to rotate forward.
- ⑲ Set the front right motor speed to 75, and the direction is reversed.
- ⑳ Set the right rear motor speed to 75, and the direction is reversed.
- ㉑ Wait for 2 seconds
- ㉒ The car stops
- ㉓ Wait for 2 seconds



Click "JavaScript" to view the corresponding JavaScript code: :



The screenshot shows the Microsoft MakeCode interface for the micro:bit board. The top navigation bar includes the Microsoft logo, a micro:bit icon, and tabs for 'Blocks' (selected) and 'JavaScript'. A search bar and a sidebar with category icons are also present. The main area displays the following JavaScript code:

```
1 basic.forever(function () {
2     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)
3     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)
4     mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)
5     mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)
6     basic.pause(2000)
7     mecanumRobot.Motor(LR.Upper_left, MD.Back, 75)
8     mecanumRobot.Motor(LR.Lower_left, MD.Back, 75)
9     mecanumRobot.Motor(LR.Upper_right, MD.Back, 75)
10    mecanumRobot.Motor(LR.Lower_right, MD.Back, 75)
11    basic.pause(2000)
12    mecanumRobot.Motor(LR.Upper_left, MD.Back, 75)
13    mecanumRobot.Motor(LR.Lower_left, MD.Back, 75)
14    mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)
15    mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)
16    basic.pause(2000)
17    mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)
18    mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)
19    mecanumRobot.Motor(LR.Upper_right, MD.Back, 75)
20    mecanumRobot.Motor(LR.Lower_right, MD.Back, 75)
21    basic.pause(2000)
22    mecanumRobot.state(MotorState.stop)
23    basic.pause(2000)
```

## (4) Test Results:

Download code 1 to micro:bit board, dial POWER switch to ON end. Smart car goes forward for 2s and stops for 2s.

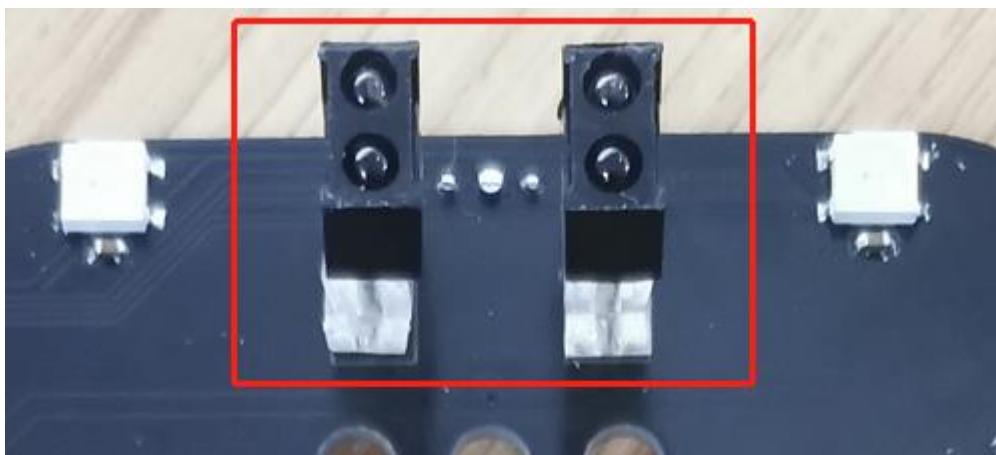
Download code 2 to micro:bit board, the car goes forward for 2s, turns back for 2s, turn left for 2s, turn right for 2s and stops for 2s and repeats this pattern.

[\(How to download?\)](#) [\(How to quick download?\)](#)



## Project 17: Line Tracking Sensor

### 17.1: Detect Line Tracking Sensor



#### (1) Project Description

The motor driving board of the Keyestudio 4WD Mecanum Robot Car comes with a dual-channel line tracking sensors which adopt TCRT5000 IR tubes and 2 potentiometers.

TCRT5000 IR tube has an IR emitting tube and a receiving tube.

Low level(0) is output when IR transmitting tube emits IR signals to receiving tube; high level(1) will be output when smart car runs along black line.

When smart car drives on the white ground, TCRT5000 IR tube will emit IR signals which will be reflected by white ground and received by receiving tube, consequently output low level(0); on the contrary, when driving on the black surface, the high level is output.



## (2)Working Principle:

When the car runs above a white road, the infrared transmitter tube installed under the car emits infrared signals to detect the road and the receiver tube receives signals sending back. Then the output end outputs low level(0); when it detects black lines, it outputs high level(1).

The 2-way tracking sensor integrated port on the 4WD Mecanum Robot Car is connected to the collection port of G ,5V ,P1 and P2 on the micro:bit expansion board, which is controlled by the P1 and P2 of the micro:bit. The left TCRT5000 infrared pair tube on the sensor is controlled by P1, and the right one by P2.

After putting a white paper on the bottom of the 4WD Mecanum Robot Car, we rotate the two potentiometers on the 2-way tracking sensor. When the indicator light on the sensor module is on, pick up the car to make the two wheels on the 4WD Mecanum Robot Car separate. The height of the white paper is about 1.5cm, the indicator light on the sensor module is off, and then the sensitivity is adjusted.

## (3)Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder



- Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- Open online Makecode editor

Import Hex profile (How to import? , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

### **(How to add MecanumRobot extension?)**

#### **(4)Test Code:**

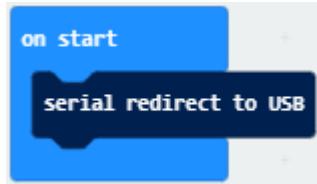
##### **Code1:**

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 17.1: Detect Line Tracking Sensor-1	Project 17.1: Detect Line Tracking Sensor-1

Or you could edit code step by step in the editing area.

Click “Advanced” → “Serial” → “serial redirect to USB”

Place it into “on start”



Enter "Advanced" → "Serial" →

Leave it into "forever" block.

Go to "Pins" → "digital read pin P0 "

Move "digital read pin P0" into 0 box

The right tracking sensor is controlled by P14. Then change P0 into P14 and "x" into "digital signal".

(2) Click "Advanced" → "Serial" to find and drag to

"forever" ;

input" left:" and drag it again;

Click " MecanumRobot " to find and drag to

;

Copy once and change " left:" to " right:"

;

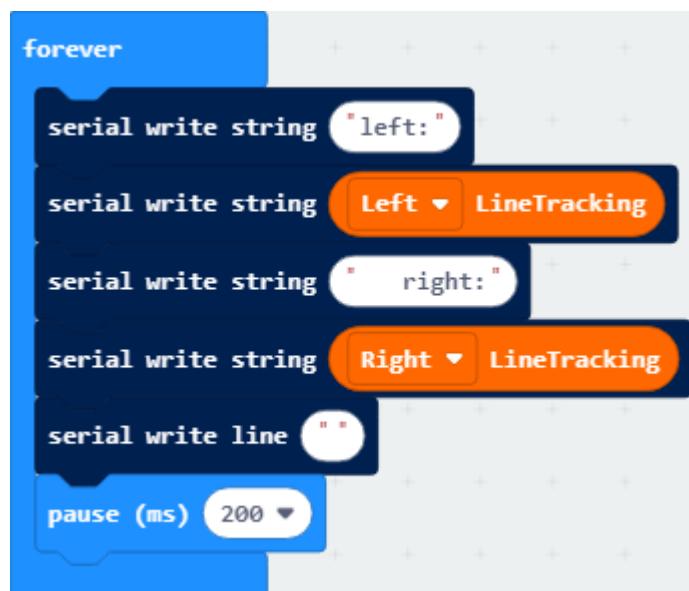


Copy  once and change Leftto Right



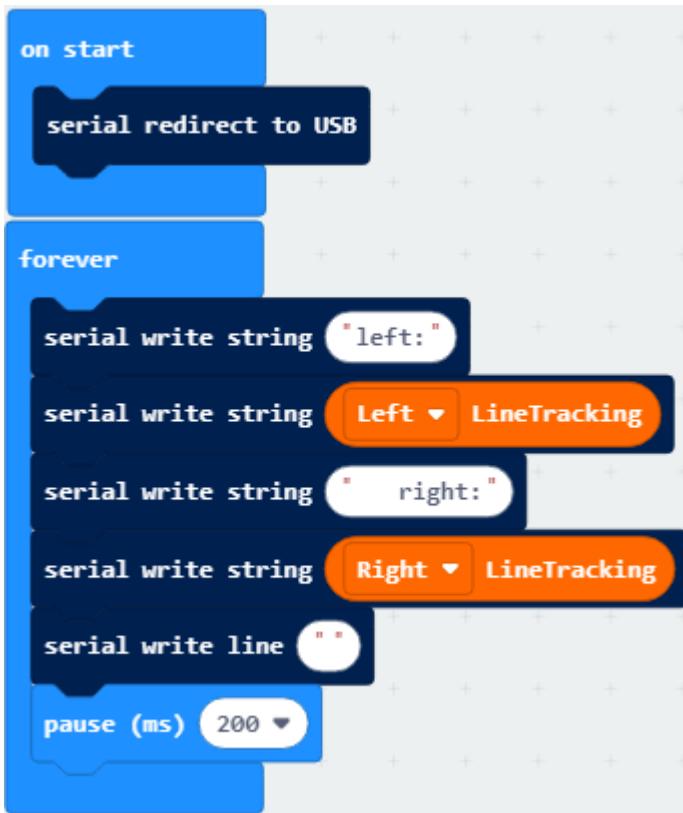
Drag 

(3) Click "Basic" to find and drag "pause (ms) 100" to "forever" and set delay



in 200ms:

Complete Program:



- ① The "on start" command block runs only once to start program.
- ② Serial redirection USB.
- ③ In the "forever" instruction block, the program runs cyclically.
- ④ Write string "left:" serially
- ⑤ Serially write the output value of the tracking sensor on the left
- ⑥ Serial write string "right:"
- ⑦ Serially write the output value of the tracking sensor on the right
- ⑧ Writing in line break
- ⑨ Delay time 200 milliseconds

Click "JavaScript" to view the corresponding JavaScript code:

```
1 serial.redirectToUSB()
2 basic.forever(function () {
3     serial.writeString("left:")
4     serial.writeString("") + (mecanumRobot.LineTracking(LT.Left)))
5     serial.writeString("right:")
6     serial.writeString("") + (mecanumRobot.LineTracking(LT.Right)))
7     serial.writeLine("")
8     basic.pause(200)
9 })
10
```

Open CoolTerm, click Options to select SerialPort. Set COM port and 115200 baud rate. Click "OK" and "Connect".

The CoolTerm serial monitor displays the digital signals read by right line tracking sensors.



## Code 2:

## Code path:

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 17.1: Detect Line Tracking	Project 17.1: Detect Line Tracking Sensor-2



	Sensor-2	
--	----------	--

Or you could edit code step by step in the editing area.

(1) Click “Variables” and then click “Make a Variable...” ;

The dialog box “New variable name: ” pops up and fill it with “LL” ;

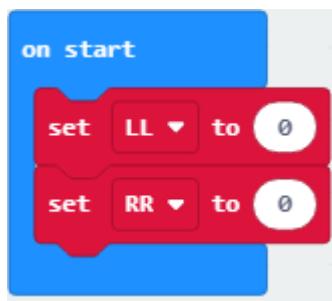
Click “OK” to establish variable “LL” ;

To establish variable “RR” in the same way;

Find and drag “set RR to 0” to “on start” ;

Copy “set RR to 0” once and place it to “on start” ;

Click the little triangle behind “RR” to choose “LL” :



(3) Click “Variables” to find and drag “set RR to 0” to “forever” ;

Click the little triangle behind “RR” to choose LL;

Click “MecanumRobot” to find and drag  to the “0” behind “to” ;

Copy  once and place it to “forever” 指



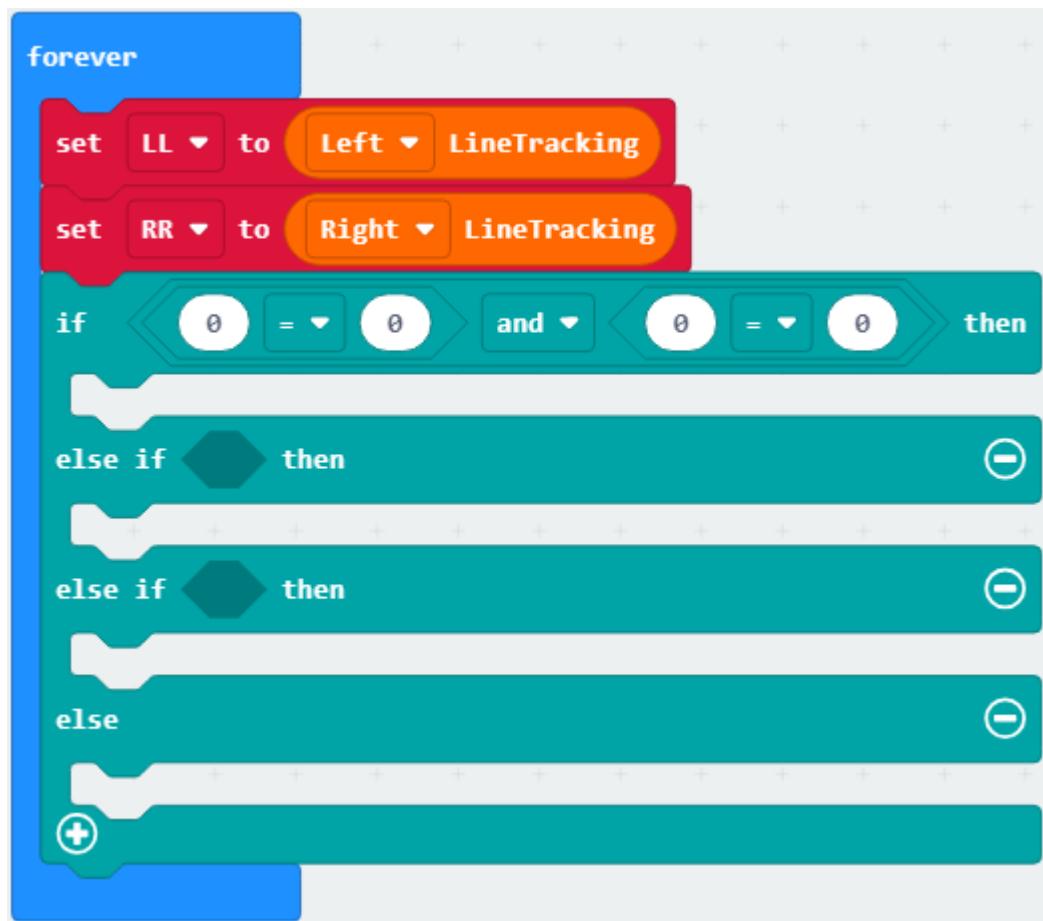
and change the second "LL" to "RR" , and " Left" to " Right" :



(4) Click "Logic" to find and drag "if true then...else" to "forever" ;

Click "⊕" twice and find and drag an "and" to "true" ;

Drag a "=" to "and" :

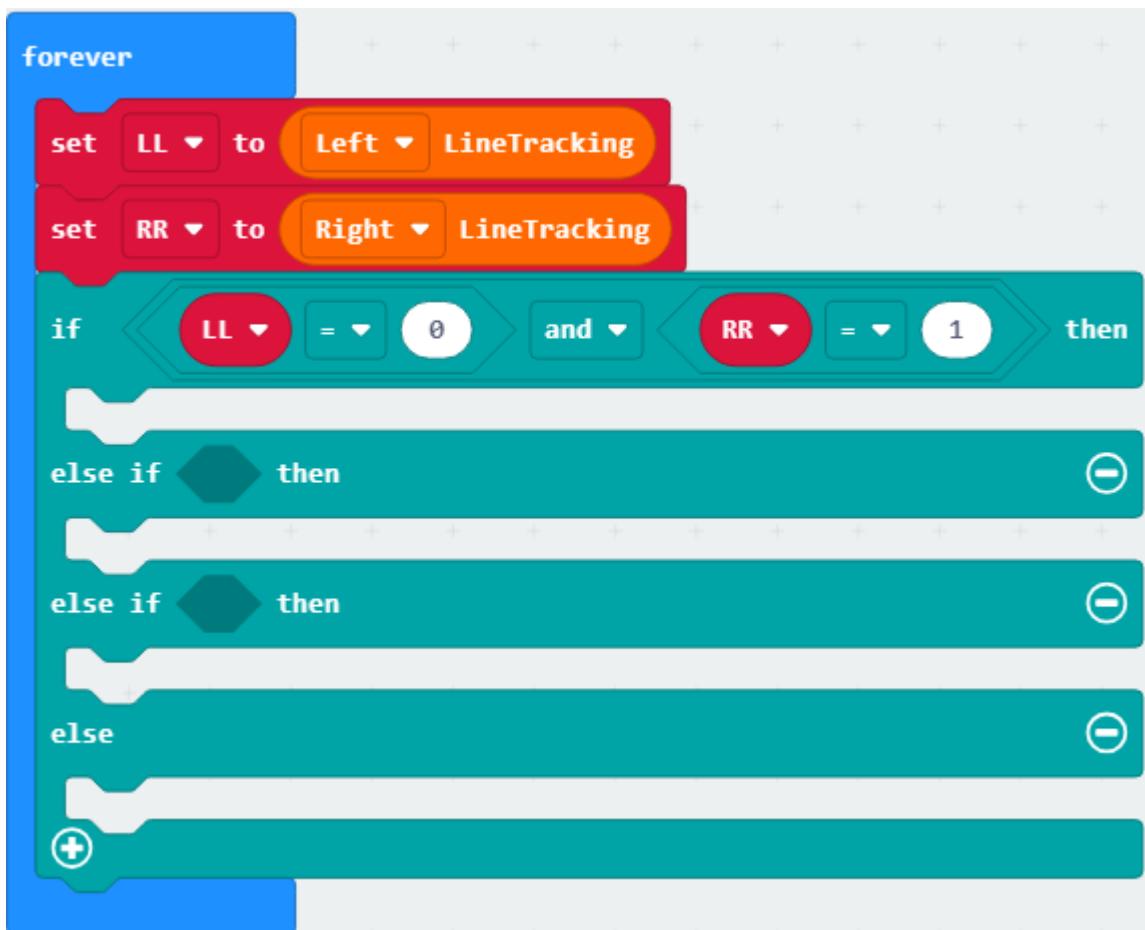


(5) Click "Variables" to find and drag "LL" to the left side of "=" ;the 0 on the right of "=" remains unchanged;

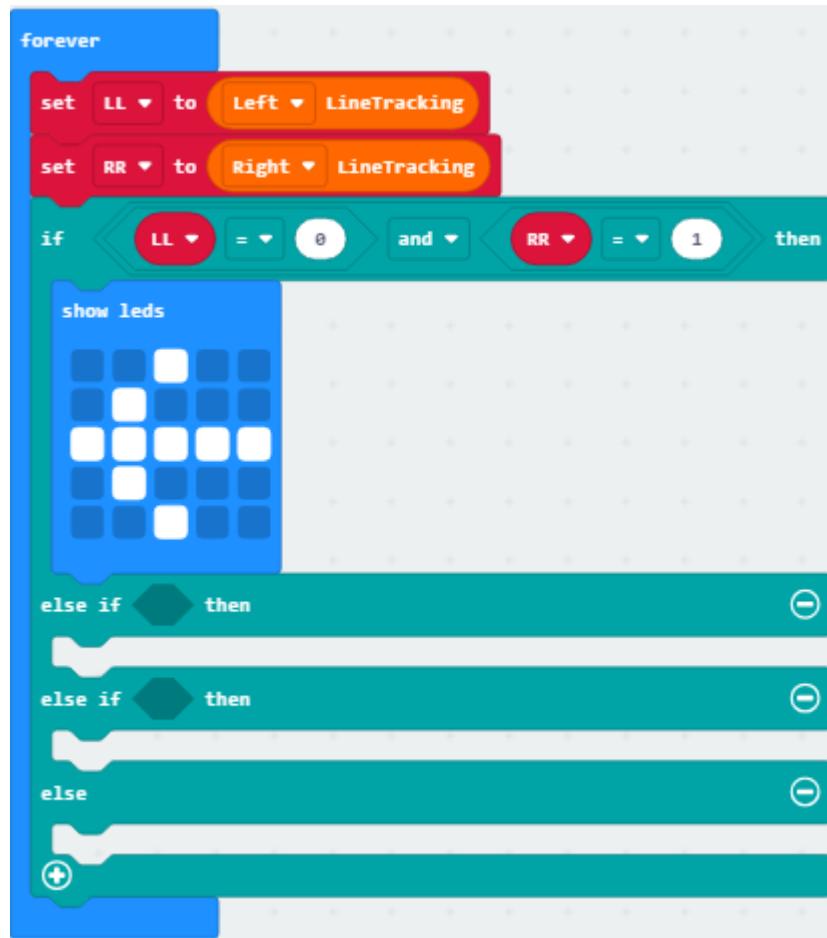


Copy "LL" 1 once and place it to the right of "and" ;

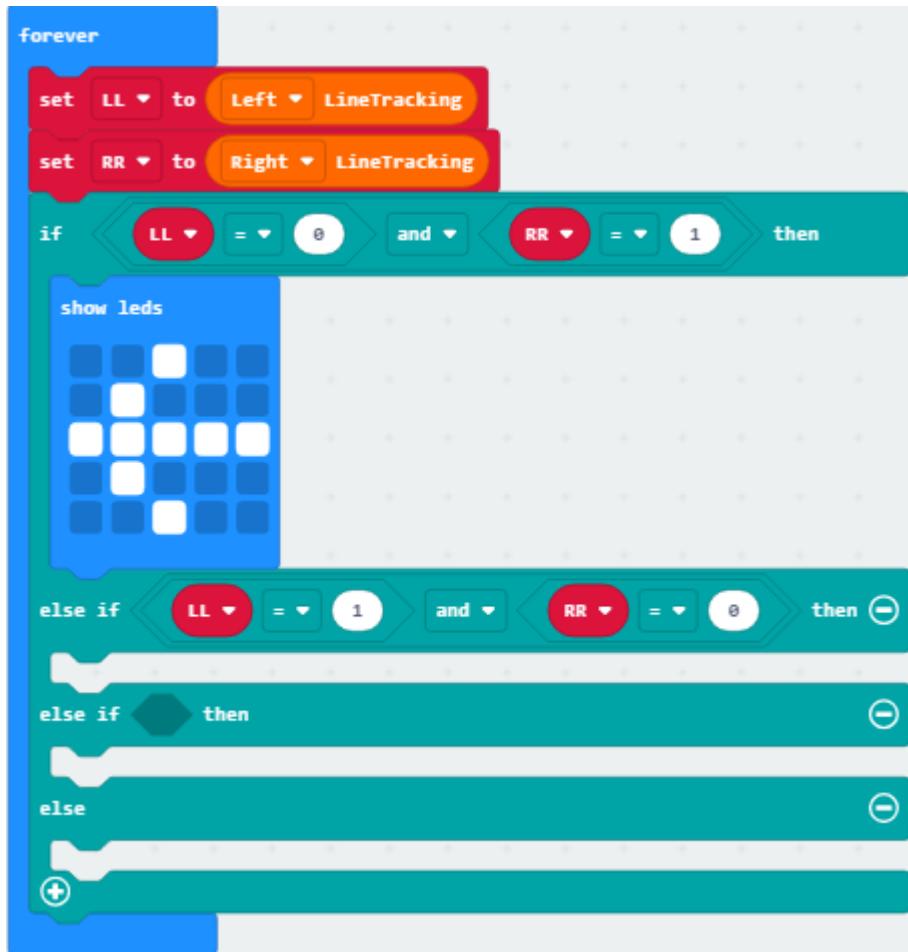
Click the little triangle behind "LL" to choose "RR" and change the "0" to "1" :



(6) Click "Basic" to find and drag "show leds" to the first "then" ; Click the blocks to form pattern " $\leftarrow$ " :



(7)copy “LL=0 and RR=1” once and place it behind the first “else if” , change the first 0 to 1, and the first 0 behind LL to 1; others remain unchanged:

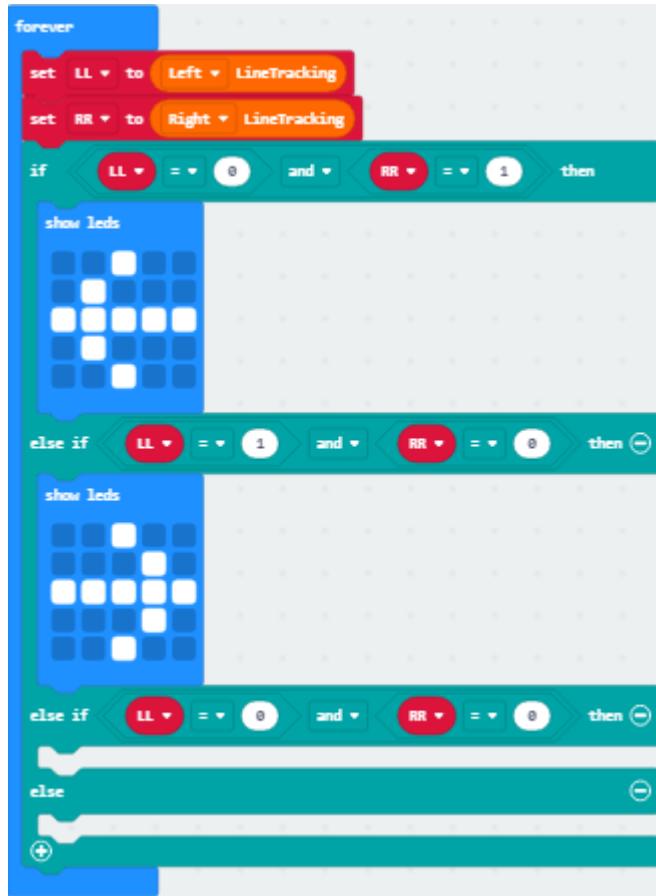


(8) Click "Basic" to find and drag "show leds" to the second "then" ;

Click the blocks to form pattern → :



(9)Copy "LL=1 and RR=0" once and place it to "else if" and change the first number 1 behind LL to 0:



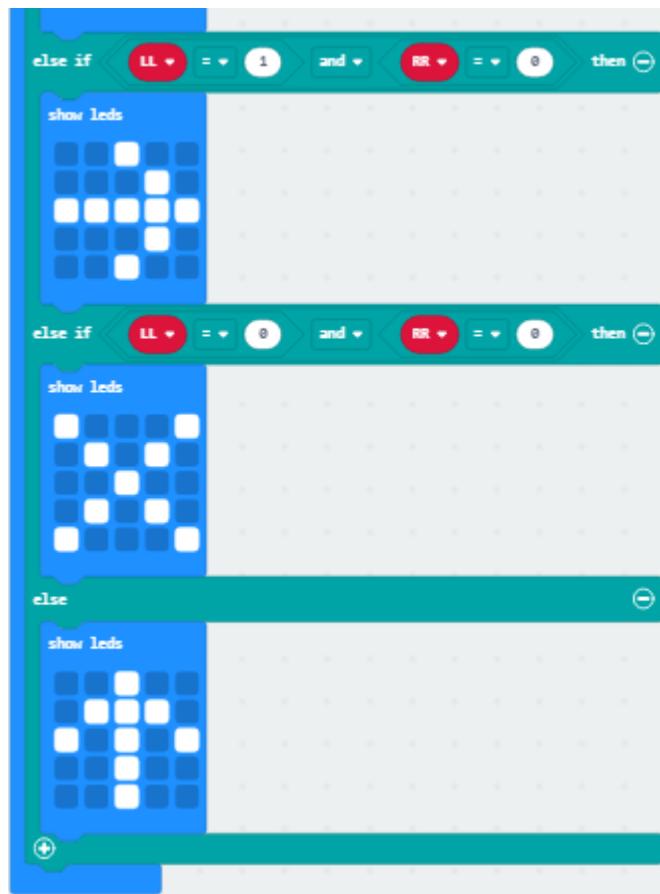
(10) Click “Basic” to find and drag “show leds” to the third else;

Click these blocks to form the pattern “x” :

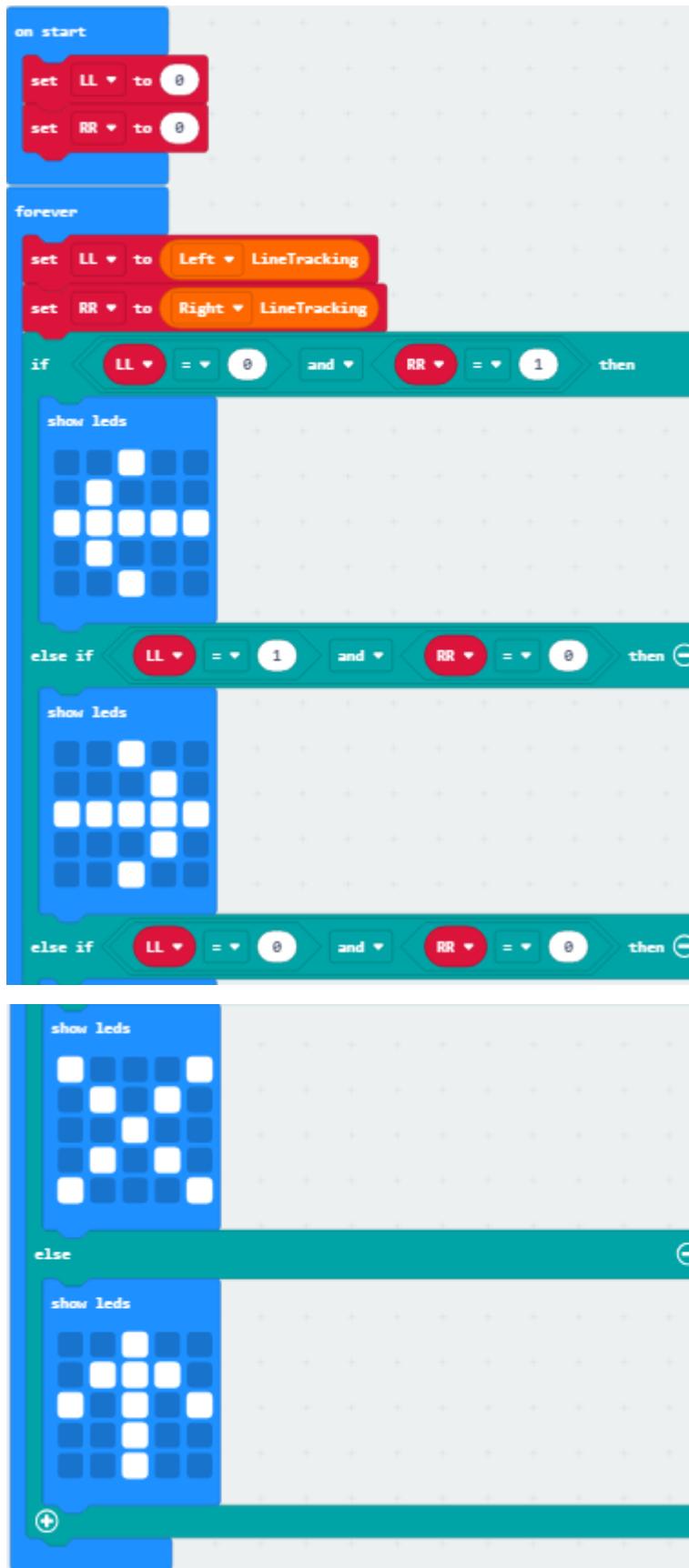


(11)Click “Basic” to find and drag “show leds” to else;

Click these blocks to form the pattern “×” :



Complete Program:



- ① The "on start" command block runs only once to set the initial values.
- ② Set the variable LL to 0
- ③ Set variable RR to 0
- ④ In the "forever" instruction block, the program reads the digital signals from the line tracking sensors and sets the variables LL and RR cyclically.
- ⑤ Set the variable LL to the digital signal read on pin 1 (1/0)
- ⑥ Set the variable RR to the digital signal read on pin 0 (1/0)
- ⑦ When the variables LL=0 and RR=1 are established, the program under the "then" condition of the first if-block is executed.
- ⑧ The left side of the LED dot matrix displays the " $\leftarrow$ " pattern.
- ⑨ When the variables LL=1 and RR=0 are established, the program under the "then" condition of the second if-block is executed.
- ⑩ The " $\rightarrow$ " pattern is displayed on the left of the matrix.
- ⑪ When the variables LL=0 and RR=0 are established, the program under the "then" condition of the third if-block is executed.
- ⑫ The "x" pattern is displayed on the left side of the matrix.
- ⑬ When the above conditions are not met, the program under the "else" block is executed.

Click "JavaScript" to view the corresponding JavaScript code: :



Microsoft | micro:bit

Blocks JavaScript ▾

Search... 

**Basic**

Input

Music

Led

Radio

Loops

Logic

Variables

Math

MecanumRobot

IrRemote

Neopixel

Advanced

```
1 let LL = 0
2 let RR = 0
3 basic.forever(function () {
4     LL = mecanumRobot.LineTracking(LT.Left)
5     RR = mecanumRobot.LineTracking(LT.Right)
6     if (LL == 0 && RR == 1) {
7         basic.showLeds(`.
8             . # .
9             . # ...
10            # # # #
11            . # ...
12            . . # ...
13        `)
14     } else if (LL == 1 && RR == 0) {
15         basic.showLeds(`.
16             . # .
17             . . . #
18             # # # #
19             . . . #
20             . . # ...
21        `)
22     } else if (LL == 0 && RR == 0) {
23         basic.showLeds(`.
24             # . . . #
25             . # . #
26             . . # .
27             . # . #
28             # . . . #
29        `)
30     } else {
31         basic.showLeds(`.
32             . . # ...
```

## (5)Test Results:

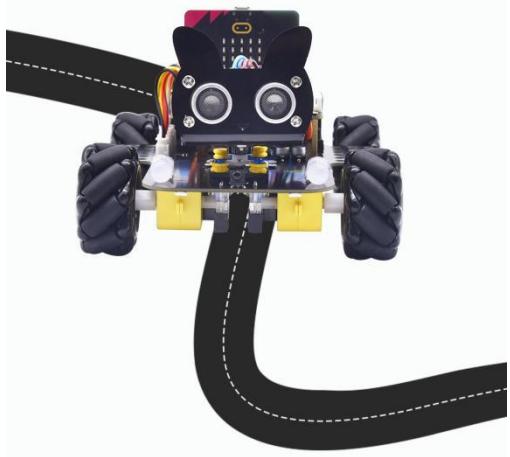
Download code 2 to the micro:bit, when only the left TCRT5000 infrared pair tube on the line tracking sensor detects a white object, the micro bit LED dot matrix displays a " $\leftarrow$ " pattern, and the indicator light on the left side of the tracking sensor lights up;

When only the right TCRT5000 infrared pair tube on the sensor detects a white object, the micro bit LED dot matrix displays a " $\rightarrow$ " pattern, and the indicator light on the right side of the tracking sensor lights up;

[\(How to download?\)](#) [\(How to quick download?\)](#)



## 17.2: Line Tracking Smart Car



### (1) Project Description

In this lesson we will combine line tracking sensors with a motor to make a line tracking smart car.

The micro:bit board will analyze the signals and control smart car to show line tracking function.

### (2) The Working Principle

The smart car will make different moves according to the value received by the 3 channel line tracking sensor.

Left/Right TCRT5000 IR Tunes (Level)		4WD Mecanum Robot Car
LOW (0)	HIGH (1)	Turn Right
HIGH (1)	LOW (0)	Turn Left
HIGH (1)	HIGH (1)	Go forward



LOW (0)	LOW (0)	Stop
---------	---------	------

## (2) Experimental Preparation:

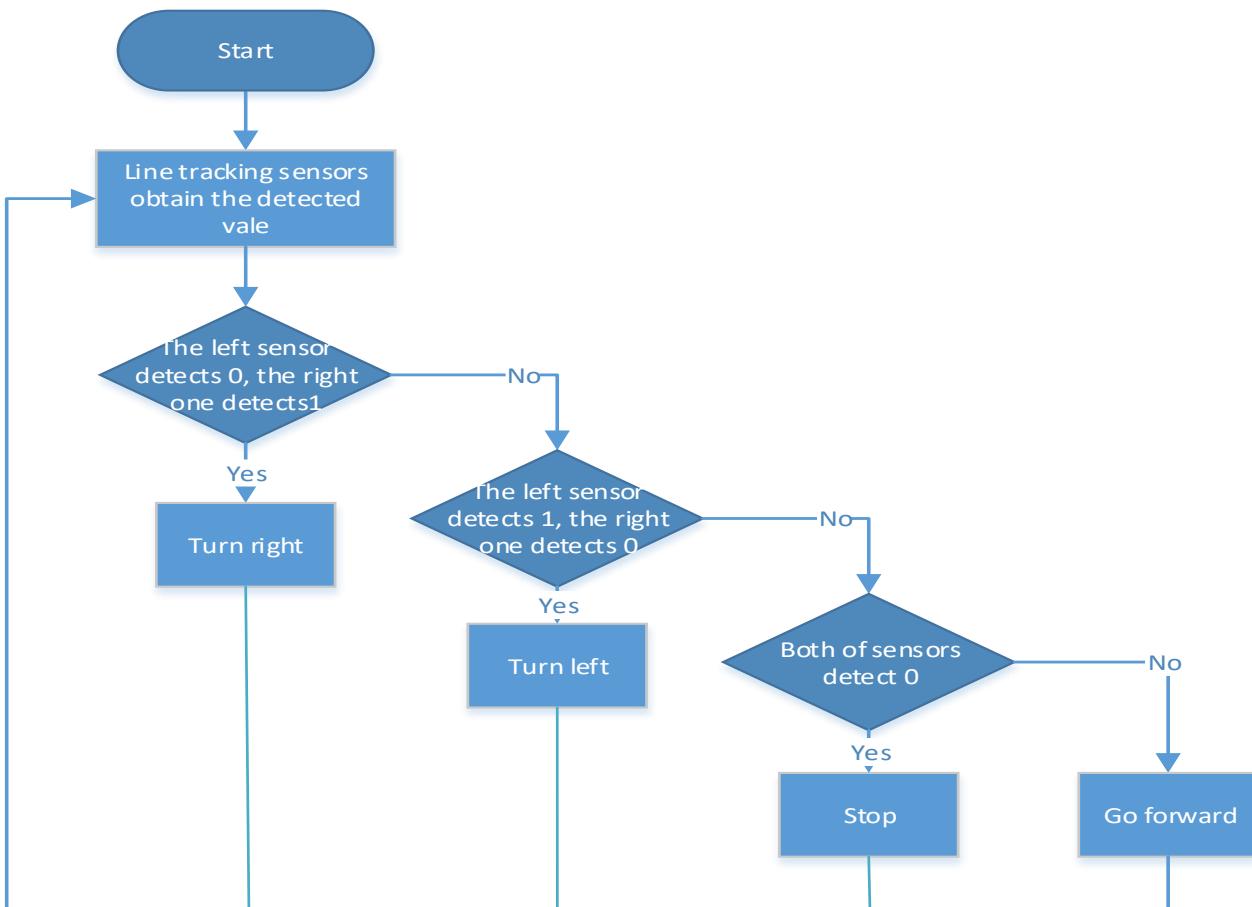
- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- Open online Makecode editor

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

### (How to add MecanumRobot extension?)

Warning: The 2-way tracking sensor should be used in environments without infrared interference such as sunlight. Sunlight contains a lot of invisible light, such as infrared and ultraviolet. In an environment with strong sunlight, the 2-way tracking sensor cannot work properly.

## (3) Flow Chart:



#### (4) Test Code:

Code path:

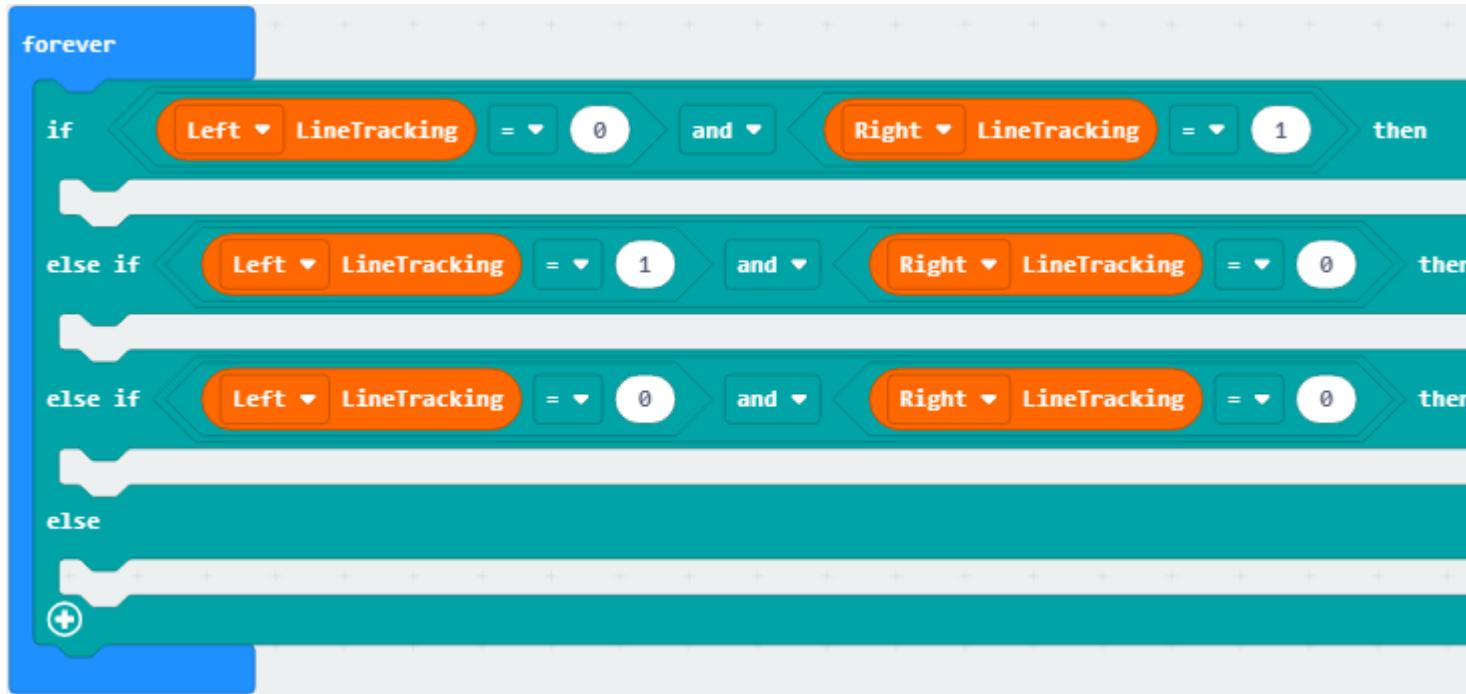
File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 17.2: Line Tracking Smart Car.hex	Project 17.2: Line Tracking Smart Car.hex

Or you could edit code step by step in the editing area.

No need to create variable LL and RR but use



**Left ▾ LineTracking** and **Right ▾ LineTracking** to decide:



Click Functions" of " Advance" and then tap "Make a Function" :



;change "doSomething" to "car\_forward" ,

"car\_back" , "car\_left" , "car\_right" respectively:



```
function car_forward
  Motor Upper_left run Forward speed: 40 %
  Motor Lower_left run Forward speed: 40 %
  Motor Upper_right run Forward speed: 40 %
  Motor Lower_right run Forward speed: 40 %
```

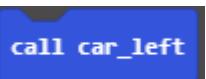
```
function car_back
  Motor Upper_left run Back speed: 40 %
  Motor Lower_left run Back speed: 40 %
  Motor Upper_right run Back speed: 40 %
  Motor Lower_right run Back speed: 40 %
```

```
function car_left
  Motor Upper_left run Back speed: 60 %
  Motor Lower_left run Back speed: 60 %
  Motor Upper_right run Forward speed: 85 %
  Motor Lower_right run Forward speed: 85 %
```

```
function car_right
  Motor Upper_left run Forward speed: 85 %
  Motor Lower_left run Forward speed: 85 %
  Motor Upper_right run Back speed: 60 %
  Motor Lower_right run Back speed: 60 %
```



Click "Functions" of " Advance" to find and drag  to the first

if and drag  to the first else if;

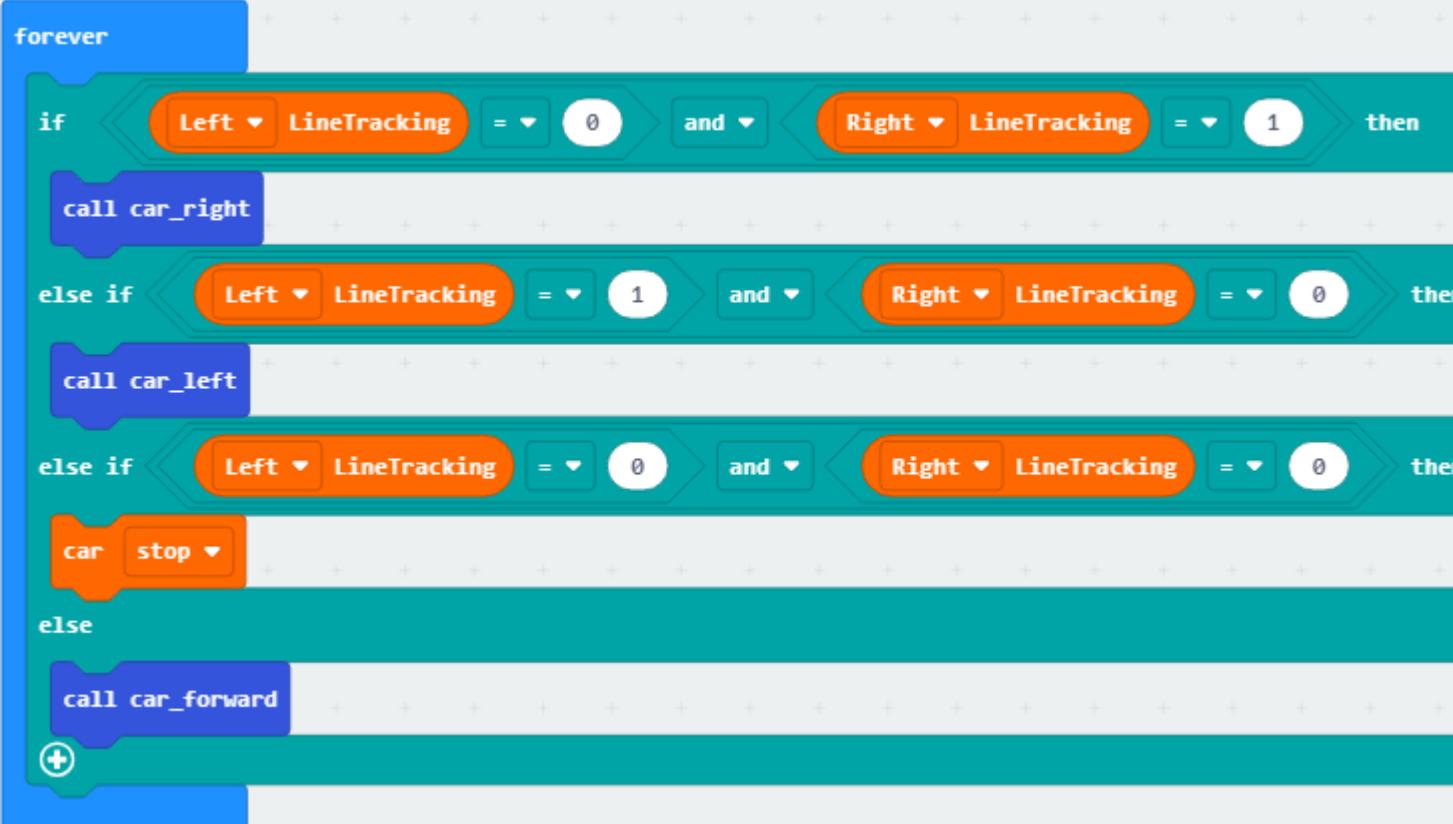
Find and drag  to the last else;

Click "MecanumRobot" to find and drag  to the second else if:





## Complete Program:



The Scratch script is a **forever** loop that continuously checks the line tracking sensors. It follows this logic:

- If Left LineTracking = 0 and Right LineTracking = 1, it calls **car\_right**.
- If Left LineTracking = 1 and Right LineTracking = 0, it calls **car\_left**.
- If both Left LineTracking = 0 and Right LineTracking = 0, it stops the car.
- If none of the above conditions are met, it calls **car\_forward**.



- ① forward function
- ② The front left motor rotates forward at a speed of 40
- ③ The motor at the rear left rotates forward at a speed of 40
- ④ The front right motor rotates forward at a speed of 40
- ⑤ The rear right motor rotates forward at a speed of 40
- ⑥ Backward function
- ⑦ The front left motor reverses, the speed is 40
- ⑧ The rear left motor reverses, the speed is 40
- ⑨ The front right motor reverses, the speed is 40
- ⑩ The rear right motor reverses, the speed is 40
- ⑪ Left turn function
- ⑫ The front left motor reverses, the speed is 60
- ⑬ The rear left motor reverses at a speed of 60
- ⑭ The front right motor rotates forward, the speed is 85
- ⑮ The right rear motor rotates forward, the speed is 85
- ⑯ right turn function
- ⑰ The front left motor rotates forward, the speed is 85
- ⑱ The rear left motor rotates forward, the speed is 85
- ⑲ The front right motor reverses, the speed is 60
- ⑳ The rear right motor reverses, the speed is 60

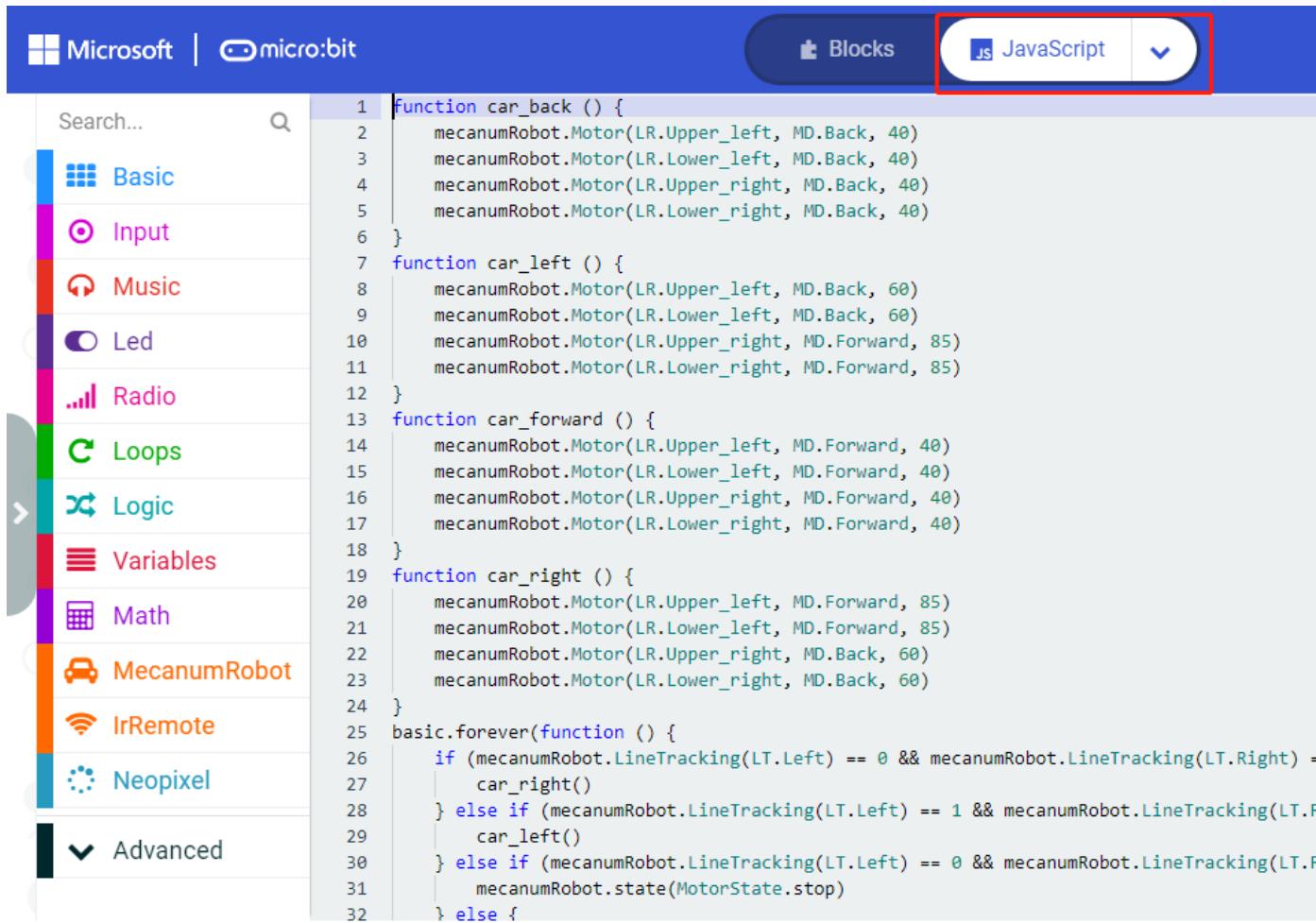


```
function car_left
  Motor Upper_left run Back speed: 60 %
  Motor Lower_left run Back speed: 60 %
  Motor Upper_right run Forward speed: 85 %
  Motor Lower_right run Forward speed: 85 %
end

function car_right
  Motor Upper_left run Forward speed: 85 %
  Motor Lower_left run Forward speed: 85 %
  Motor Upper_right run Back speed: 60 %
  Motor Lower_right run Back speed: 60 %
end
```



Click "JavaScript" to view the corresponding JavaScript code:



The screenshot shows the Microsoft MakeCode interface for micro:bit. The top navigation bar has tabs for "Blocks" and "JavaScript", with "JavaScript" being the active tab and highlighted with a red box. On the left, there's a sidebar with categories like Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, MecanumRobot, IrRemote, Neopixel, and Advanced. The main workspace displays the following JavaScript code:

```
1 function car_back () {
2     mecanumRobot.Motor(LR.Upper_left, MD.Back, 40)
3     mecanumRobot.Motor(LR.Lower_left, MD.Back, 40)
4     mecanumRobot.Motor(LR.Upper_right, MD.Back, 40)
5     mecanumRobot.Motor(LR.Lower_right, MD.Back, 40)
6 }
7 function car_left () {
8     mecanumRobot.Motor(LR.Upper_left, MD.Back, 60)
9     mecanumRobot.Motor(LR.Lower_left, MD.Back, 60)
10    mecanumRobot.Motor(LR.Upper_right, MD.Forward, 85)
11    mecanumRobot.Motor(LR.Lower_right, MD.Forward, 85)
12 }
13 function car_forward () {
14     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 40)
15     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 40)
16     mecanumRobot.Motor(LR.Upper_right, MD.Forward, 40)
17     mecanumRobot.Motor(LR.Lower_right, MD.Forward, 40)
18 }
19 function car_right () {
20     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 85)
21     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 85)
22     mecanumRobot.Motor(LR.Upper_right, MD.Back, 60)
23     mecanumRobot.Motor(LR.Lower_right, MD.Back, 60)
24 }
25 basic.forever(function () {
26     if (mecanumRobot.LineTracking(LT.Left) == 0 && mecanumRobot.LineTracking(LT.Right) == 1) {
27         car_right()
28     } else if (mecanumRobot.LineTracking(LT.Left) == 1 && mecanumRobot.LineTracking(LT.Right) == 0) {
29         car_left()
30     } else if (mecanumRobot.LineTracking(LT.Left) == 0 && mecanumRobot.LineTracking(LT.Right) == 0) {
31         mecanumRobot.state(MotorState.stop)
32     } else {
33     }
34 })
```

## (5)Test Results:

Download code to micro:bit and dial POWER to ON end, line tacking car goes forward along black line .

Note: turn on the switch at the back of micro:bit car.

the width of black line should be larger than the width of line tracking sensor.

Avoid to test smart car under the strong light.



---

## Project 18: Ultrasonic Follow Smart Car

### 18.1: Ultrasonic Ranging

#### (1) Project Description

The ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. It comes complete with ultrasonic transmitter and receiver modules.

The ultrasonic sensor is being used in a wide range of electronics projects for creating obstacle detection and distance measuring application as well as various other applications.

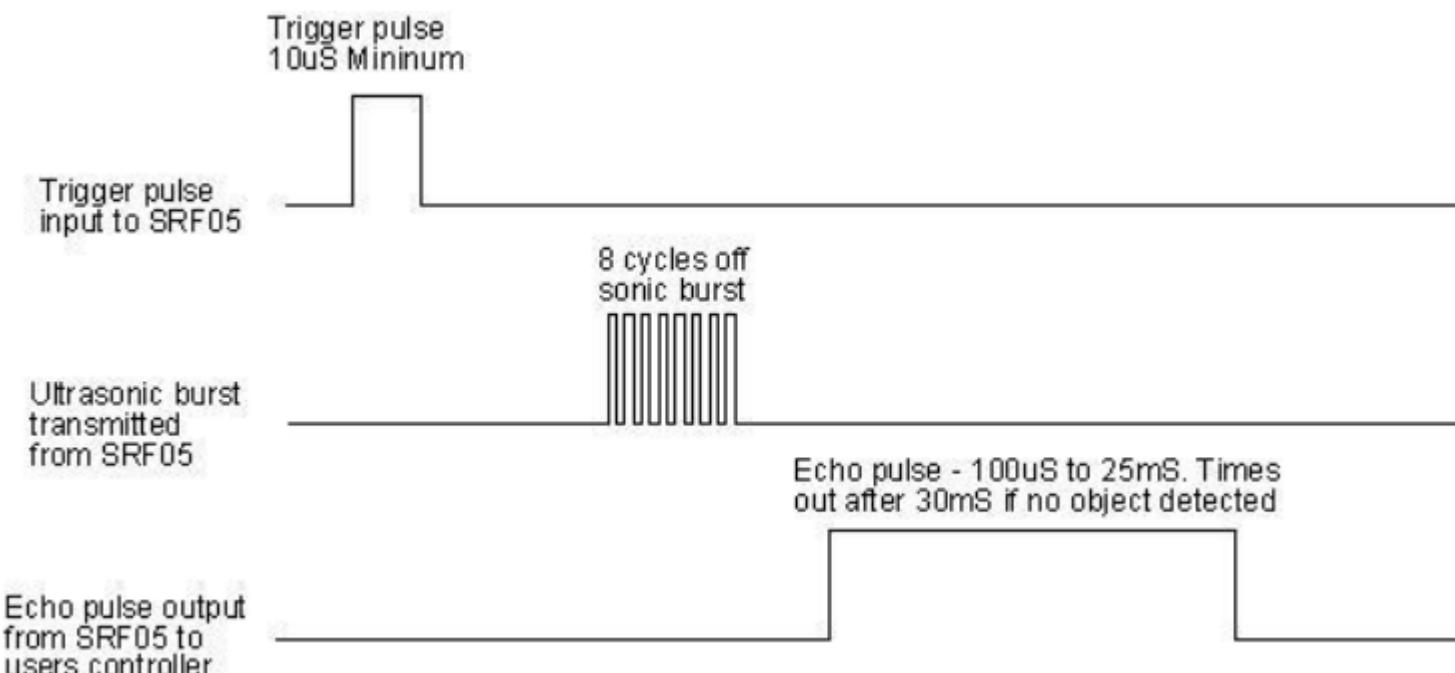
As the above picture shown, it is like two eyes. One is transmitting end, the other is receiving end.

The ultrasonic module will emit the ultrasonic waves after trigger signal. When the ultrasonic waves encounter the object and are reflected back, the module outputs an echo signal, so it can determine the distance of object from the time difference between trigger signal (TRIG)and echo signal(ECHO).



According to the above wiring diagram, the integrated port of the ultrasonic sensor module is connected to the 5V G P15 P16 port on the micro:bit motor drive backplane. The Trig (T) pin is controlled by P15 of the micro:bit and the pin of Echo (E) the P16.

## (2) Working Principle:



Pull down TRIG then trigger high level signals with least 10us

After triggering, the module will automatically send eight 40KHz ultrasonic pulses and detect whether there is a signal return.



---

The propagation speed of sound in the air is about 340m/s, therefore, distance = speed \* time, because the ultrasonic wave emits and comes back, which is 2 times of distance, so it needs to be divided by 2, the distance measured by ultrasonic wave = (speed \* time)/2

### (3)Parameters:

- ◆ Working voltage: 3-5.5V (DC)
- ◆ Working current: 15mA
- ◆ Working frequency: 40khz
- ◆ Maximum detection distance: about 3m
- ◆ Minimum detection distance: 2-3cm
- ◆ Precision: up to 0.2cm
- ◆ Sensing angle: less than 15 degrees
- ◆ Input trigger pulse: 10us TTL level
- ◆ Output echo signal: output TTL level signal (high), proportional to range

### (4)Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- Open online Makecode editor



Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

### **(How to add MecanumRobot extension?)**

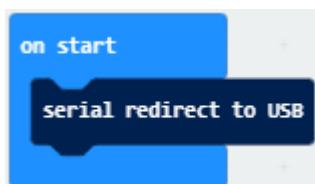
#### **(5)Test Code:**

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 18.1: Ultrasonic Ranging.hex	Project 18.1: Ultrasonic Ranging.hex

Or you could edit code step by step in the editing area.

(1)Tap “Advanced” → “Serial” → “serial redirect to USB”

Combine it with “on start” block



(2)Click “Advanced” → “Serial” to find and drag “serial write value x=0” into “forever” ; Click “MecanumRobot” to find and drag “Ultrasonic” to the



0 on the right side of “serial write value x=0” and change the x on the left side of “=” to distance:



(2) find and drag  of “Basic”, and change 100 to 200 and place it behind



Complete Program:



① The "on start" command block runs on power up.

② Serial redirection USB

③ In the "forever" instruction block, the program

④ Serial write value distance=Ultrasonic

⑤ Delay time 200 milliseconds

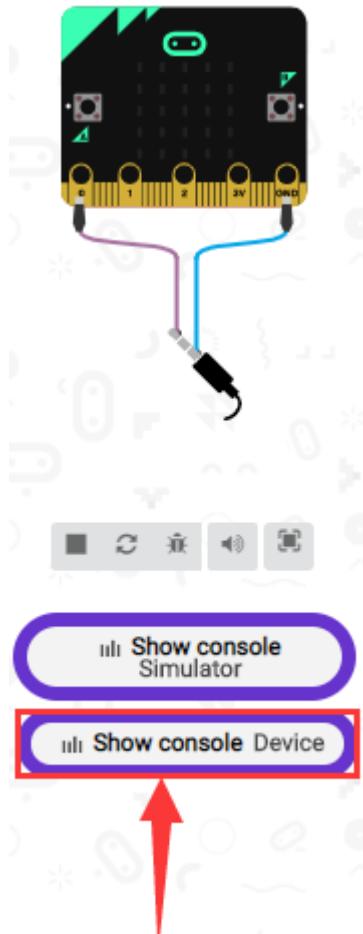
Click "JavaScript" to view the corresponding JavaScript code:

```
serial.redirectToUSB()
basic.forever(function () {
    serial.writeValue("distance", mecanumRobot.ultra())
    basic.pause(200)
})
```

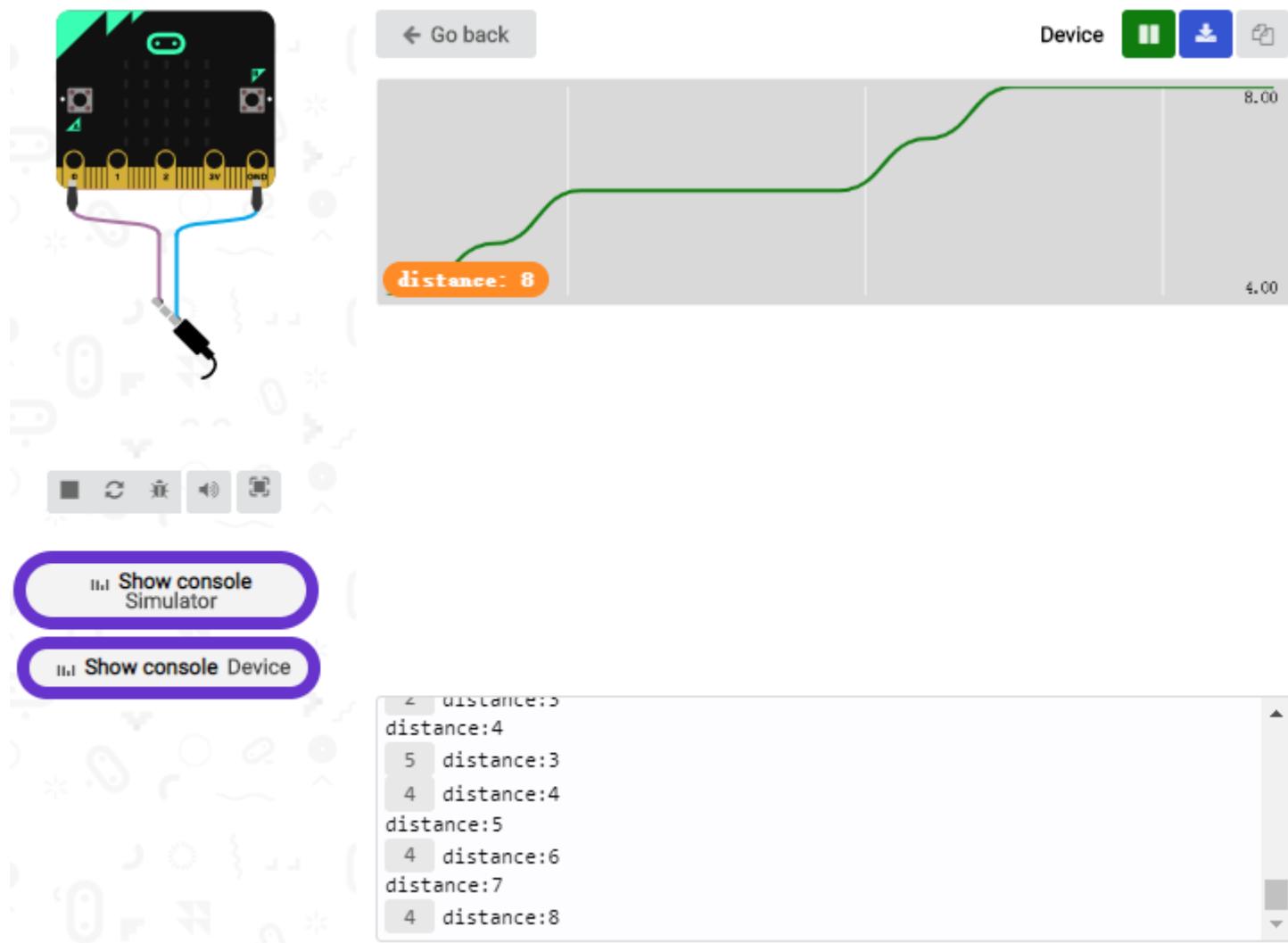
## (6) Test Results:

Download code to micro:bit, keep USB cable connected, dial POWER switch to ON end. The distance value will be displayed on monitor.

[\(How to quick download?\)](#)



The monitor shows the distance between the obstacle and ultrasonic sensor(as shown below). When the distance is less than 10cm, the passive buzzer of smart car emits sound.



Open CoolTerm, click Options to select SerialPort. Set COM port and 115200 baud rate(the baud rate of USB serial communication of Micro:bit is 115200 through the test). Click “OK” and “Connect” .  
CoolTerm serial monitor displays the distance value as follows:



Untitled\_0 \*

File Edit Connection Macros View Remote Window Help

New Open Save Connect Disconnect Clear Data Options View Hex Help

```
distance:3
distance:3
distance:3
distance:4
distance:5
distance:6
distance:8
distance:8
distance:10
distance:10
distance:11
distance:12
distance:14
distance:15
distance:18
distance:17
distance:20
distance:22
distance:25
distance:26
distance:28
distance:31
distance:33
```

COM49 / 115200 8-N-1  
Connected 00:00:29, 4164 / 0 bytes

TX RX RTS CTS DTR DSR DCD RI



## 18.2: Ultrasonic Avoidance Car



### (1) Project Description

We've learned the knowledge of obstacle avoidance sensor. In this project, we will integrate ultrasonic sensor, and car expansion board to make an ultrasonic avoidance car.

Its principle is to detect the distance between the car and obstacle by ultrasonic sensor and control the motion of smart car.

### (2) Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end
- Connect micro:bit to computer by USB cable

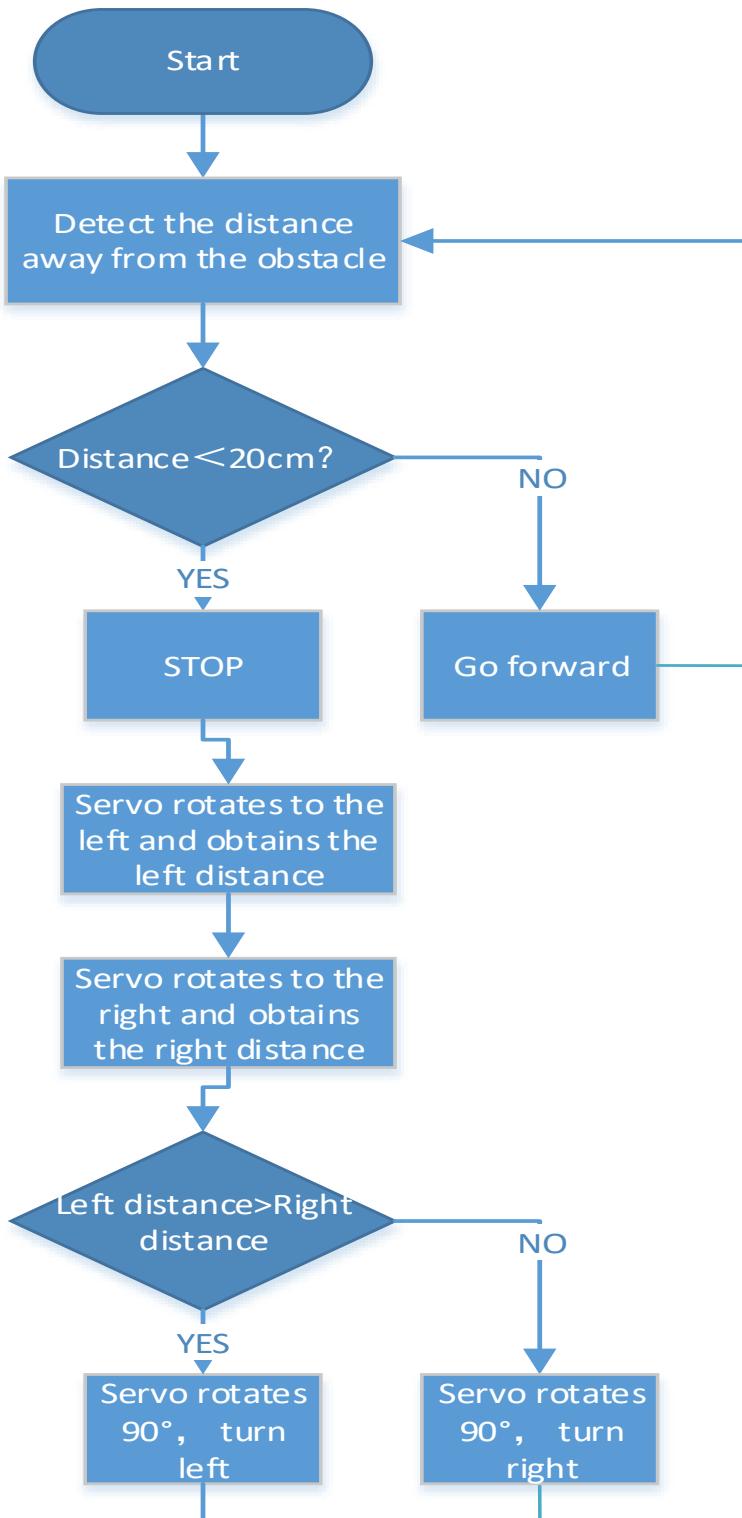


- 
- Open online Makecode editor

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

### **(How to add MecanumRobot extension?)**

### **(3)Flow Chart:**



#### (4) Test Code:

Code path:

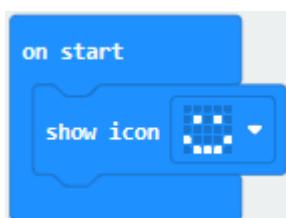


File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 18.2: Ultrasonic Avoidance Car.hex	Project 18.2: Ultrasonic Avoidance Car.hex

Or you could edit code step by step in the editing area.

(1) Enter "Basic" → "show icon ❤"

Place it into "on start" and click the triangle button to select "████" pattern

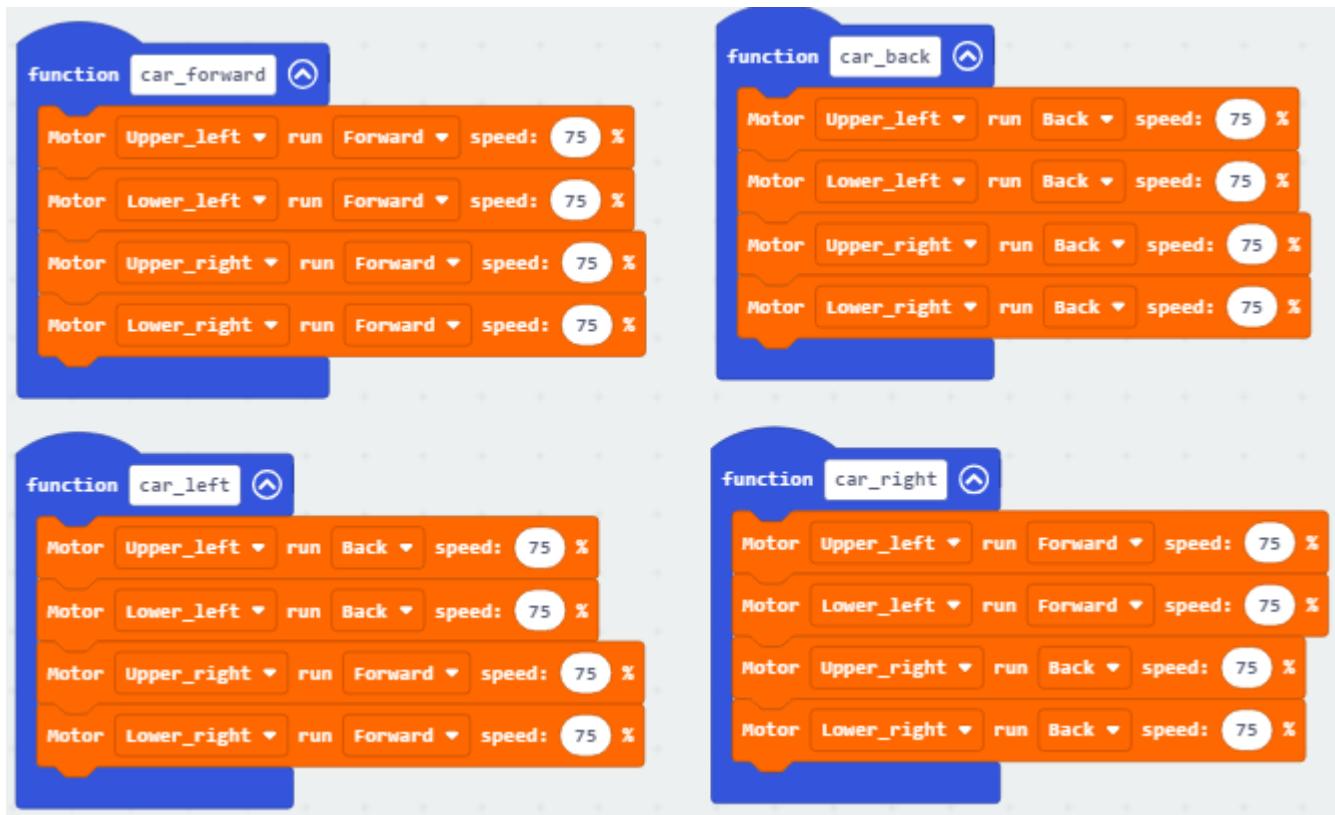


(4) Click "Variables" and then click "Make a Variable...", dialog box "New variable name: " pops up;

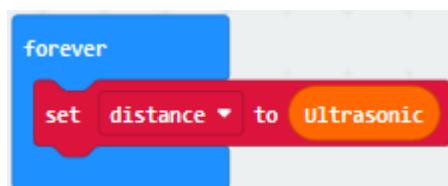
Fill it with "distance" ;

Click "OK" to establish variable "distance";

Set the functions of servo:



- (5) Click “Variables” to find and drag “set distance to 0” to “forever” ;  
Click “MecanumRobot” to find and drag “Ultrasonic” to the 0 behind  
the “to” :



- (6) Click “Logic” to find and drag “if true then...else” to “forever” ;  
Find and drag “=” to “true” ;  
Click “Variables” to find and drag “distance” on the left of “=” ;  
Click the little triangle behind “=” to choose “<” ;  
Change the 2 behind “>” to 20:



```
forever
  set [distance v] to [Ultrasonic]
  if [distance < v] [20] then
    call [car_back v]
    car [stop v]
    pause (100 ms)
  else
    [+1 v]
```



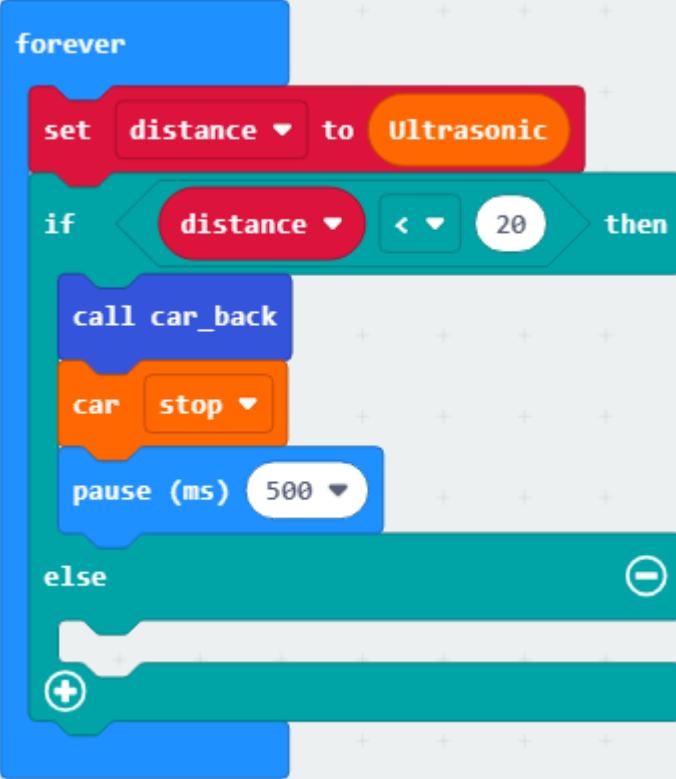
(7) Click Functionsto of “Advance” to find and drag ;



Click “MecanumRobot” to find and drag  to then;

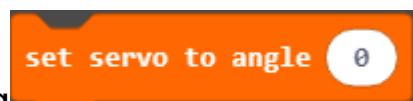


Click “Basic” to find  and change the 100 to 500:



```
forever
  set [distance v] to [Ultrasonic]
  if [distance < v] [20] then
    call [car_back v]
    car [stop v]
    pause (500 ms)
  else
    [+1 v]
```



(8) Click “MecanumRobot” to find and drag  and change the 0 to 180;

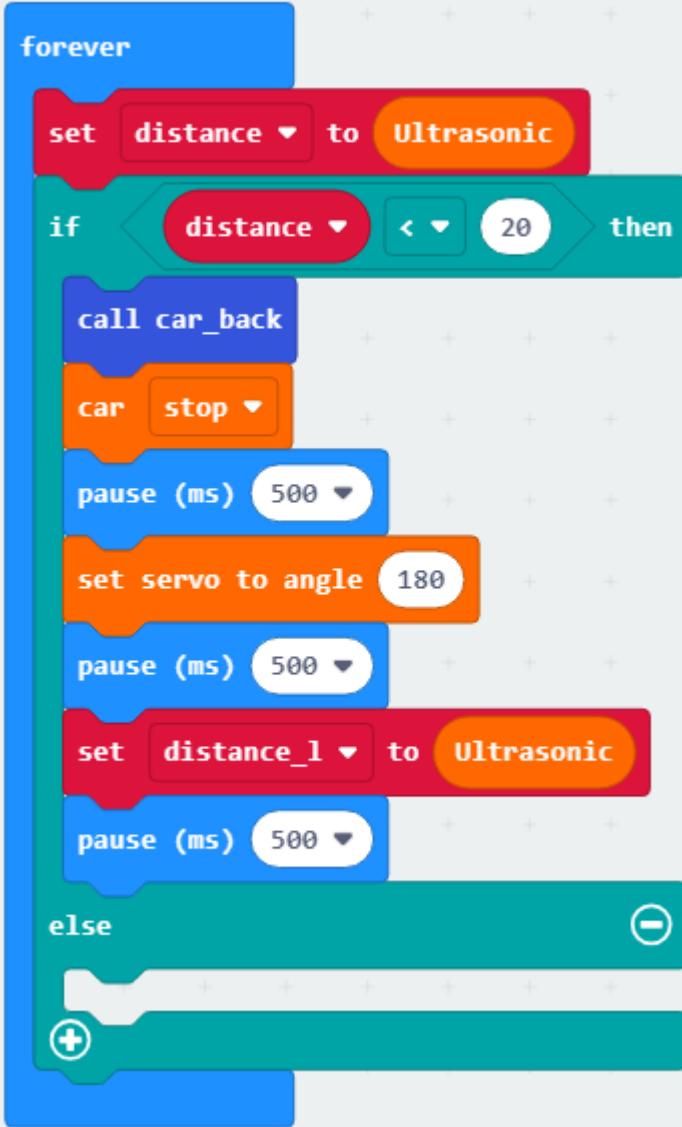
Copy  once;

Click “Variables” to find and drag “set distance\_l to 0” ;

Click “MecanumRobot” to find and drag “Ultrasonic” to 0 behind “to”

;

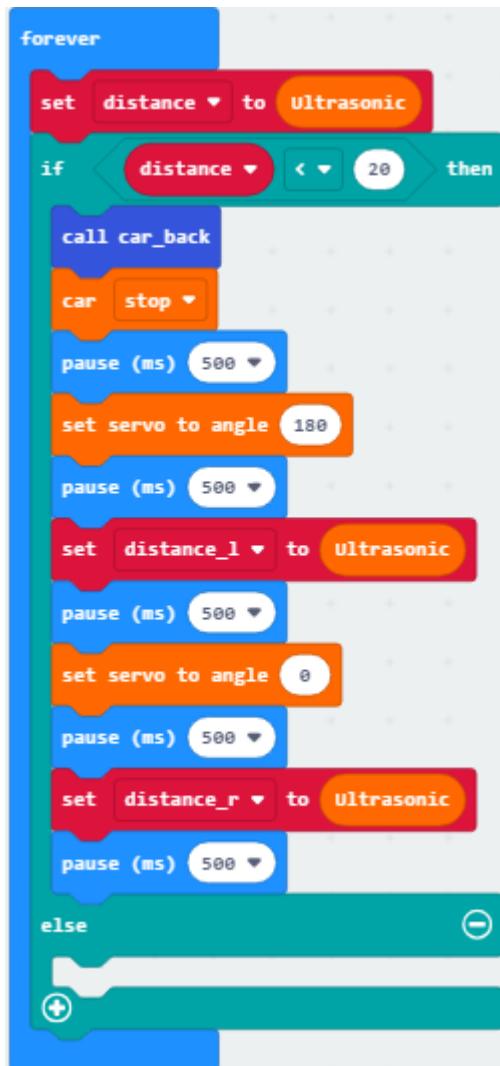
Copy  once;



```
forever
  set [distance v] to [Ultrasonic]
  if [distance < v] [20] then
    call [car_back v]
    [car stop v]
    pause (500 ms)
    set servo to angle [180]
    pause (500 ms)
    set [distance_l v] to [Ultrasonic]
    pause (500 ms)
  else
    + [ ]
  end
  set servo to angle [180]
  pause (500 ms)
  set [distance_l v] to [Ultrasonic]
  pause (500 ms)
```

(9) Copy once;

Change the 180 to 0, distance\_l to distance\_r and others remain unchanged:

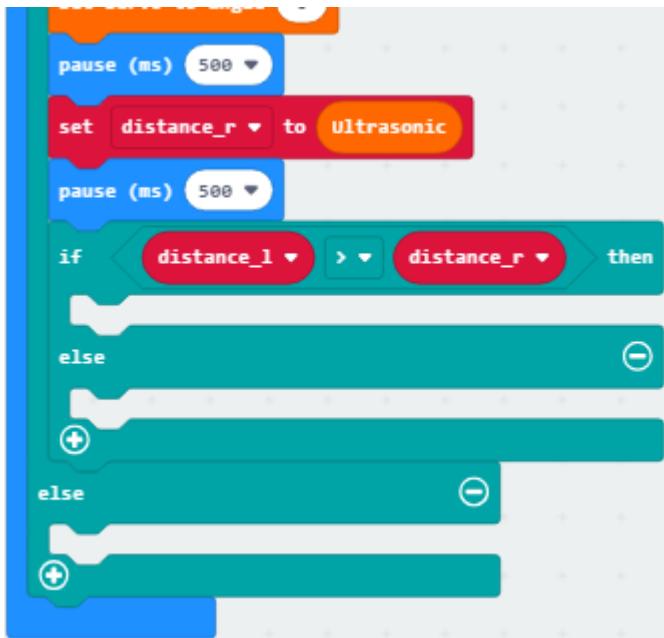


(10) Click “Logic” to find and drag “if true then...else” ;

Find and drag “=” to true;

Click “Variables” to find and drag “distance\_l” to the left of “=” ; Click the little triangle behind “=” to choose “>” ;

Change the 0 behind “>” to “distance\_r” :



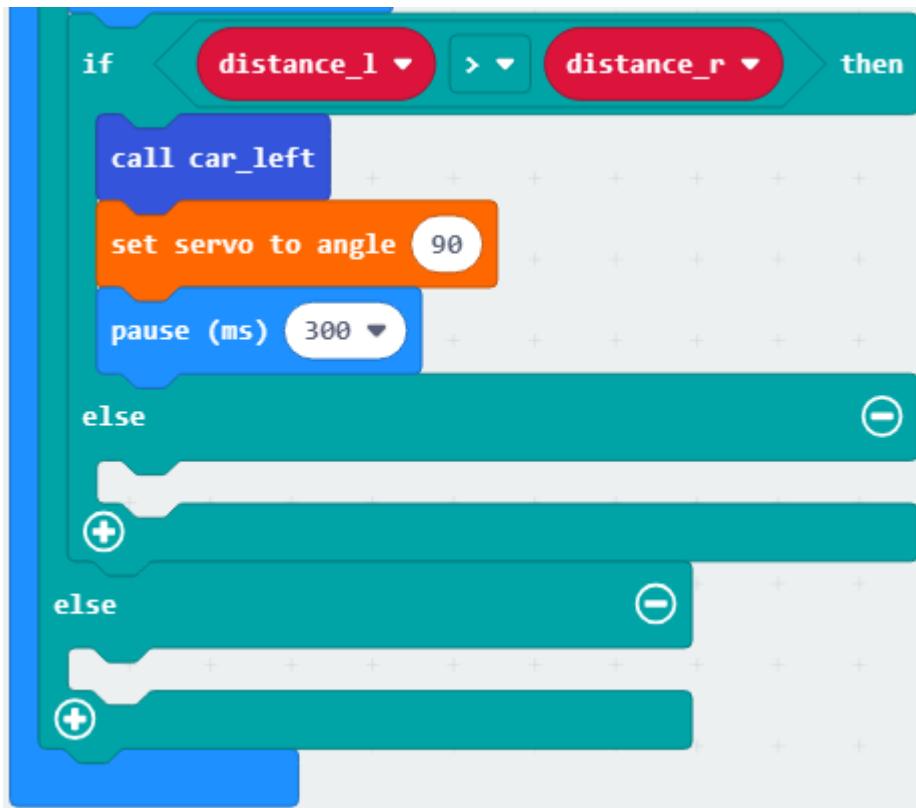
(11) Click Functions of "Advance" to find and drag ;

Click "MecanumRobot" to find and drag ;

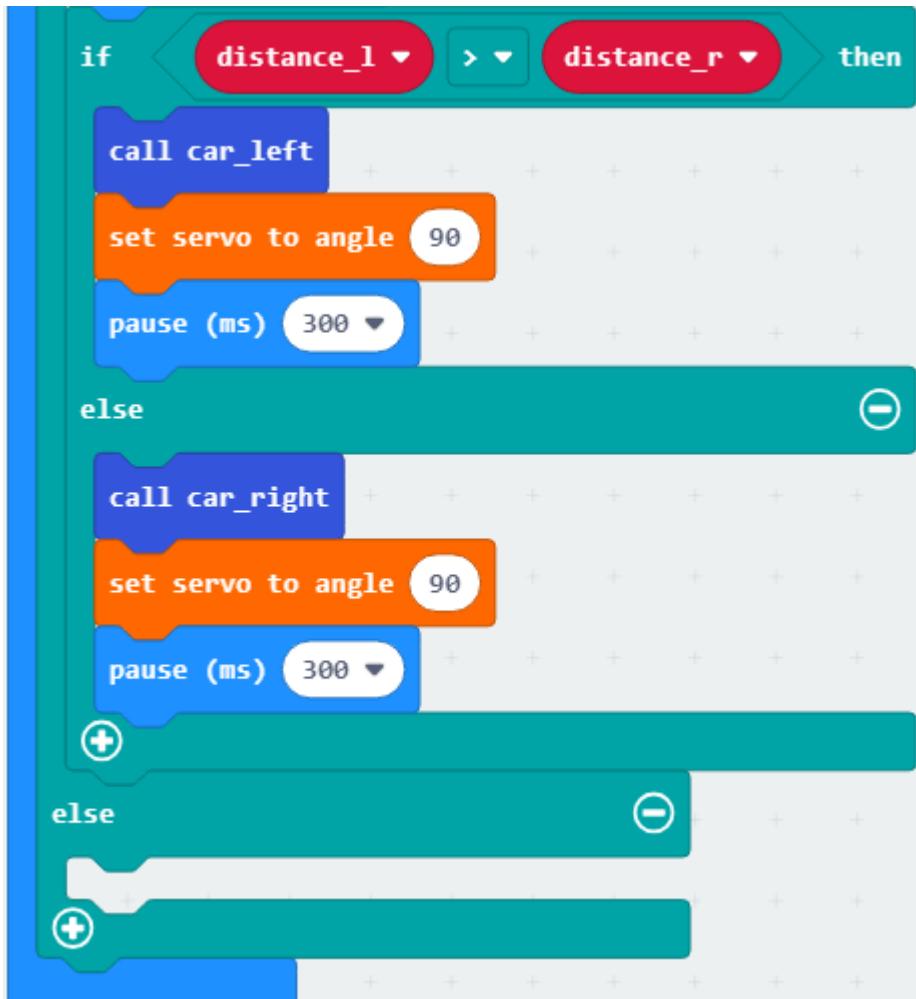
Change the 0 to 90;

Click "Basic" to find and drag  and change the 100 to

300:



(12) Change **call car\_left** to **call car\_right** and place it in the first "else" :



(11) Click “Functionsto” of “Advance” to find and drag **call car\_forward**,

and place it to the second “else” :



```
when green flag clicked
    [loop v]
        [call car_right v]
        [set servo to angle 90 v]
        [loop v]
            [call car_forward v]
        end
        [pause (300 ms) v]
    end
end
```

Complete Program:



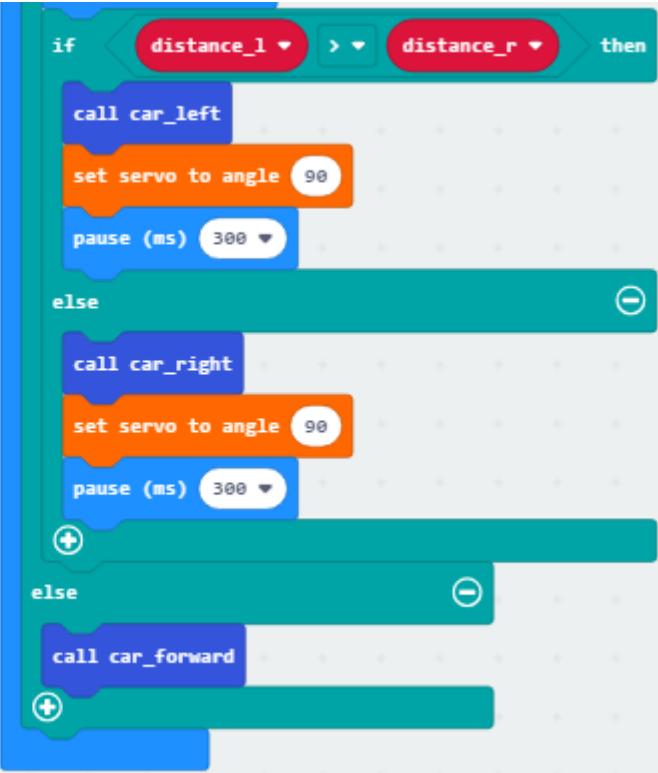
The Scratch script consists of the following components:

- on start:** A blue hat block followed by a "show icon" block.
- forever:** A blue control loop containing:
  - A red "set distance to Ultrasonic" block.
  - An if-then loop:
    - Condition: "distance < 20".
    - Body:
      - A green "call car\_back" block.
      - An orange "car stop" block.
      - A blue "pause (ms) 500" block.
      - An orange "set servo to angle 180" block.
      - A blue "pause (ms) 500" block.
      - A red "set distance\_l to Ultrasonic" block.
      - A blue "pause (ms) 500" block.
      - An orange "set servo to angle 0" block.
      - A blue "pause (ms) 500" block.
      - A red "set distance\_r to Ultrasonic" block.
      - A blue "pause (ms) 500" block.
      - An if-then loop:
        - Condition: "distance\_l > distance\_r".
        - Body:
          - An orange "stop" block.



The Scratch script consists of the following blocks:

- A single Motor block for the Lower\_right motor.
- A function definition named "car\_right" containing four Motor blocks:
  - Upper\_left motor run Forward at 75%
  - Lower\_left motor run Forward at 75%
  - Upper\_right motor run Back at 75%
  - Lower\_right motor run Back at 75%



The Scratch script consists of the following blocks:

- An if-then-else-endif loop comparing distance\_l and distance\_r.
- Inside the if-block:
  - call car\_left
  - set servo to angle 90
  - pause (ms) 300
- Inside the else-block:
  - call car\_right
  - set servo to angle 90
  - pause (ms) 300
- Inside the final else-block:
  - call car\_forward

Click "JavaScript" to view the corresponding JavaScript code: :



Microsoft | micro:bit

Blocks JavaScript ▾

Search... Q

- Basic
- Input
- Music
- Led
- Radio
- Loops
- Logic
- Variables
- Math
- MecanumRobot
- IrRemote
- Neopixel
- Advanced

```
1 function car_back () {  
2     mecanumRobot.Motor(LR.Upper_left, MD.Back, 75)  
3     mecanumRobot.Motor(LR.Lower_left, MD.Back, 75)  
4     mecanumRobot.Motor(LR.Upper_right, MD.Back, 75)  
5     mecanumRobot.Motor(LR.Lower_right, MD.Back, 75)  
6 }  
7 function car_left () {  
8     mecanumRobot.Motor(LR.Upper_left, MD.Back, 75)  
9     mecanumRobot.Motor(LR.Lower_left, MD.Back, 75)  
10    mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)  
11    mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)  
12 }  
13 function car_forward () {  
14     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)  
15     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)  
16     mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)  
17     mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)  
18 }  
19 function car_right () {  
20     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)  
21     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)  
22     mecanumRobot.Motor(LR.Upper_right, MD.Back, 75)  
23     mecanumRobot.Motor(LR.Lower_right, MD.Back, 75)  
24 }  
25 let distance_r = 0  
26 let distance_l = 0  
27 let distance = 0  
28 if (distance > 20) {  
29     car_forward()  
30 } else {  
31     car_left()  
32 }
```

## (5)Test Results:

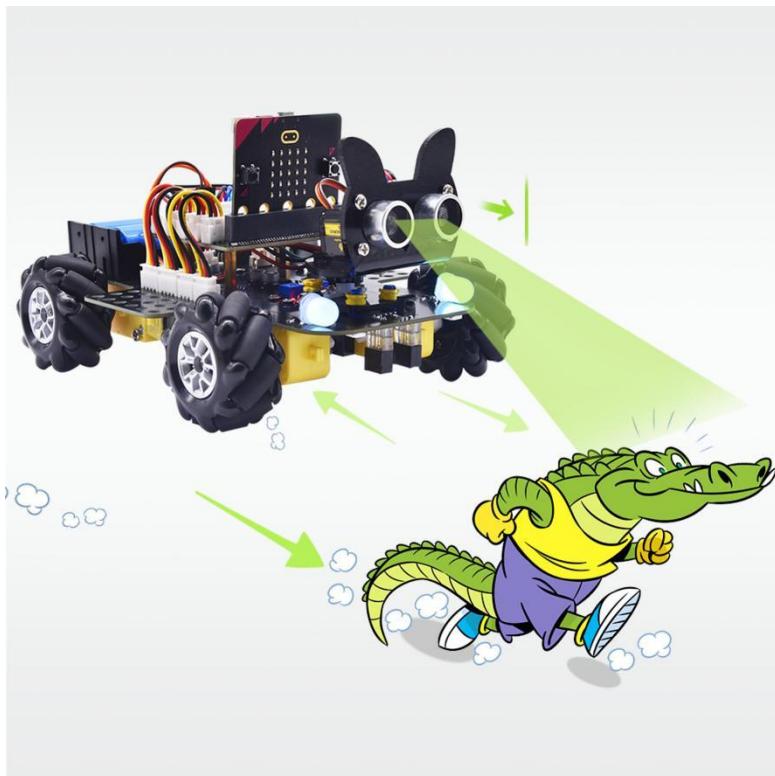
Download code to micro:bit, dial to ON end, and dial POWER to ON end.

When the obstacle distance is greater than 20cm, the car goes forward ; on the contrary, smart car turns left.

[\(How to download?\)](#) [\(How to quick download?\)](#)



## 18.3: Ultrasonic Follow Smart Car



### (1) Project Description

In previous lesson, we've learned the basic principle of line tracking sensor. Next, we will combine ultrasonic sensor with car shield to make an ultrasonic follow car.

The ultrasonic sensor detects the obstacle distance and control the motion status of car.

### (2) Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end

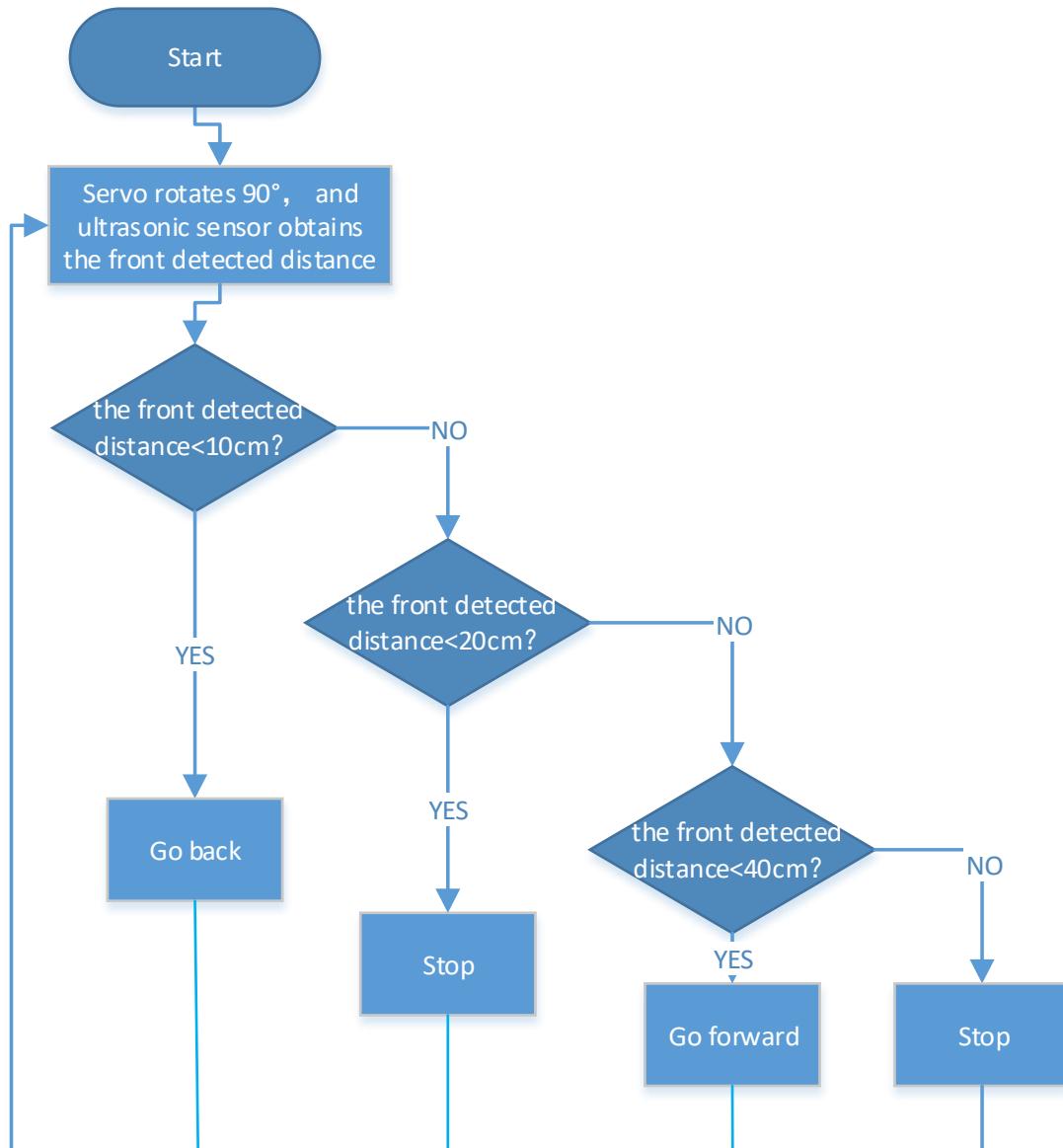


- 
- Connect micro:bit to computer by USB cable
  - Open online Makecode editor

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

**(How to add MecanumRobot extension?)**

**(3)Flow Chat:**



#### (4) Test Code:

Code path:

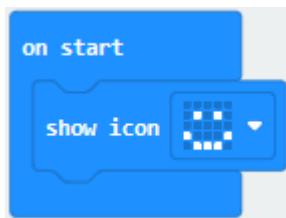
File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 18.3: Ultrasonic Follow Smart Car.hex	Project 18.3: Ultrasonic Follow Smart Car.hex



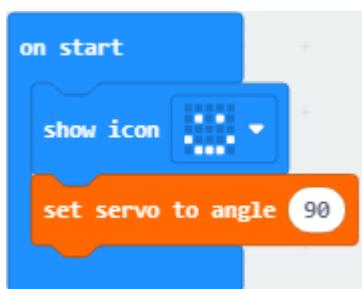
Or you could edit code step by step in the editing area.

(1) Enter "Basic" → "show icon ❤"

Place it into "on start" and click the triangle button to select "grid" pattern.



(2) Click "MecanumRobot" to find and drag  to "on start" and change the angle 0 to 90:

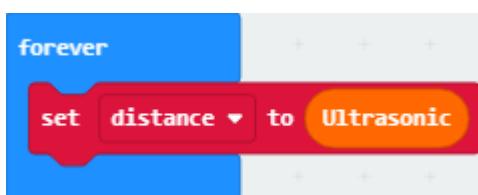


(1) Click "Variables" and then click "Make a Variable..." , the dialog box "New variable name: " pops up; fill it with "distance" ;

Click "OK" to establish variable "distance" ;

Drag "set distance to 0" to "forever" ;

Click "MecanumRobot" to find and drag  to the "0" of "set distance to 0" :





(2) Click "Logic" to find and drag "if true then...else" to "forever" ;

Find and drag "=" to true;

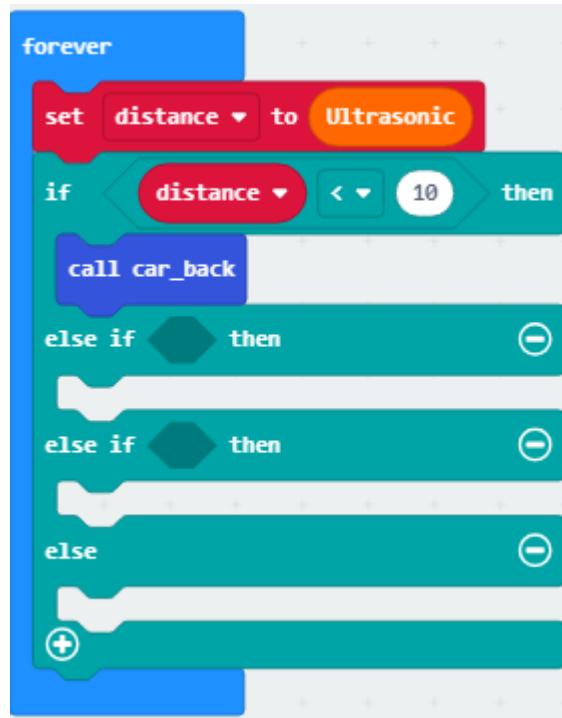
Click "Variables" to find and drag "distance" to the left side of "=" ;

Click the little triangle behind "=" to choose "<" ;

Change the 0 behind "" to 10:

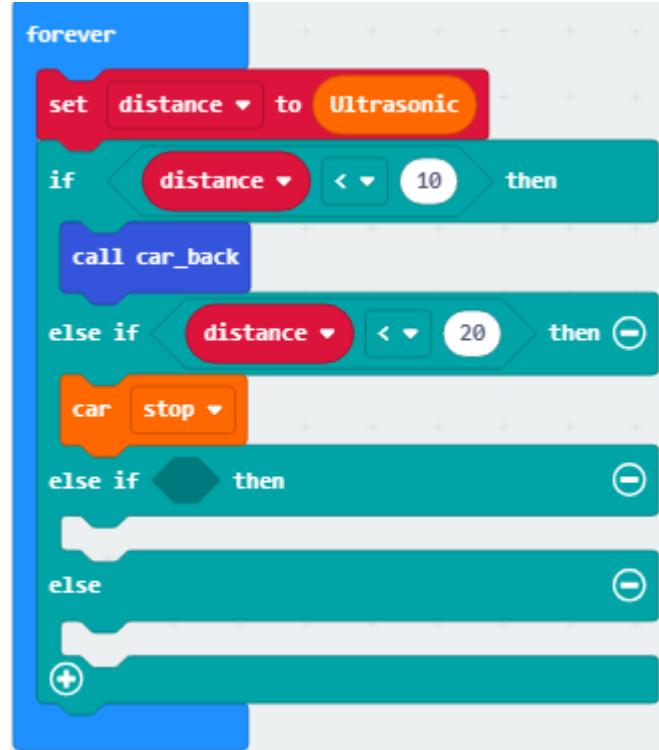


(5) Click "Funcions" of "Advance" to find and drag  to "then" :



```
forever
  set [distance v] to [Ultrasonic]
  if [distance v] < [10] then
    call [car_back v]
  else if [ ] then
  else if [ ] then
  else
  end
```

(6) change the 10 to 20, car\_back to car stop:

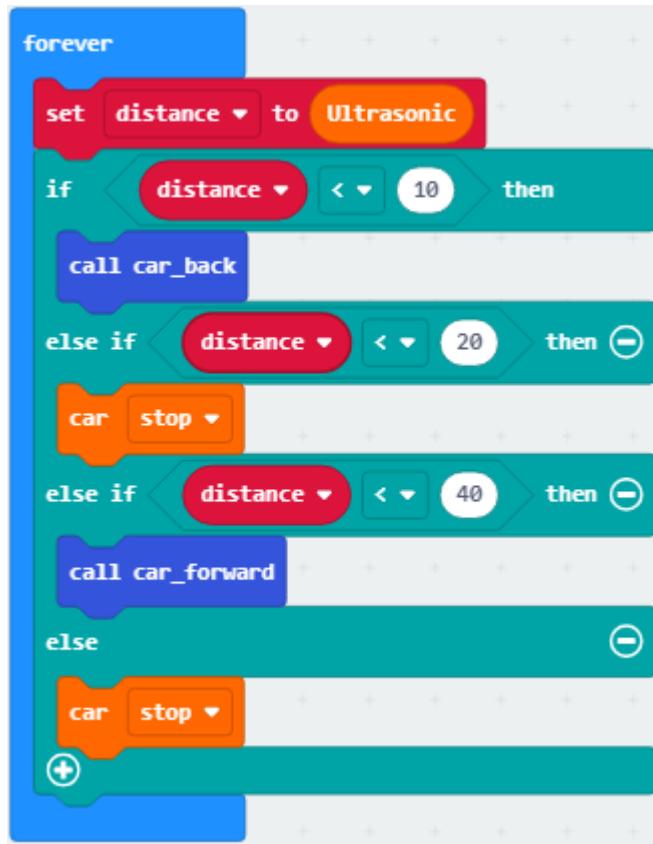


```
forever
  set [distance v] to [Ultrasonic]
  if [distance v] < [10] then
    call [car_back v]
  else if [distance v] < [20] then
    car [stop v]
  else if [ ] then
  else
  end
```

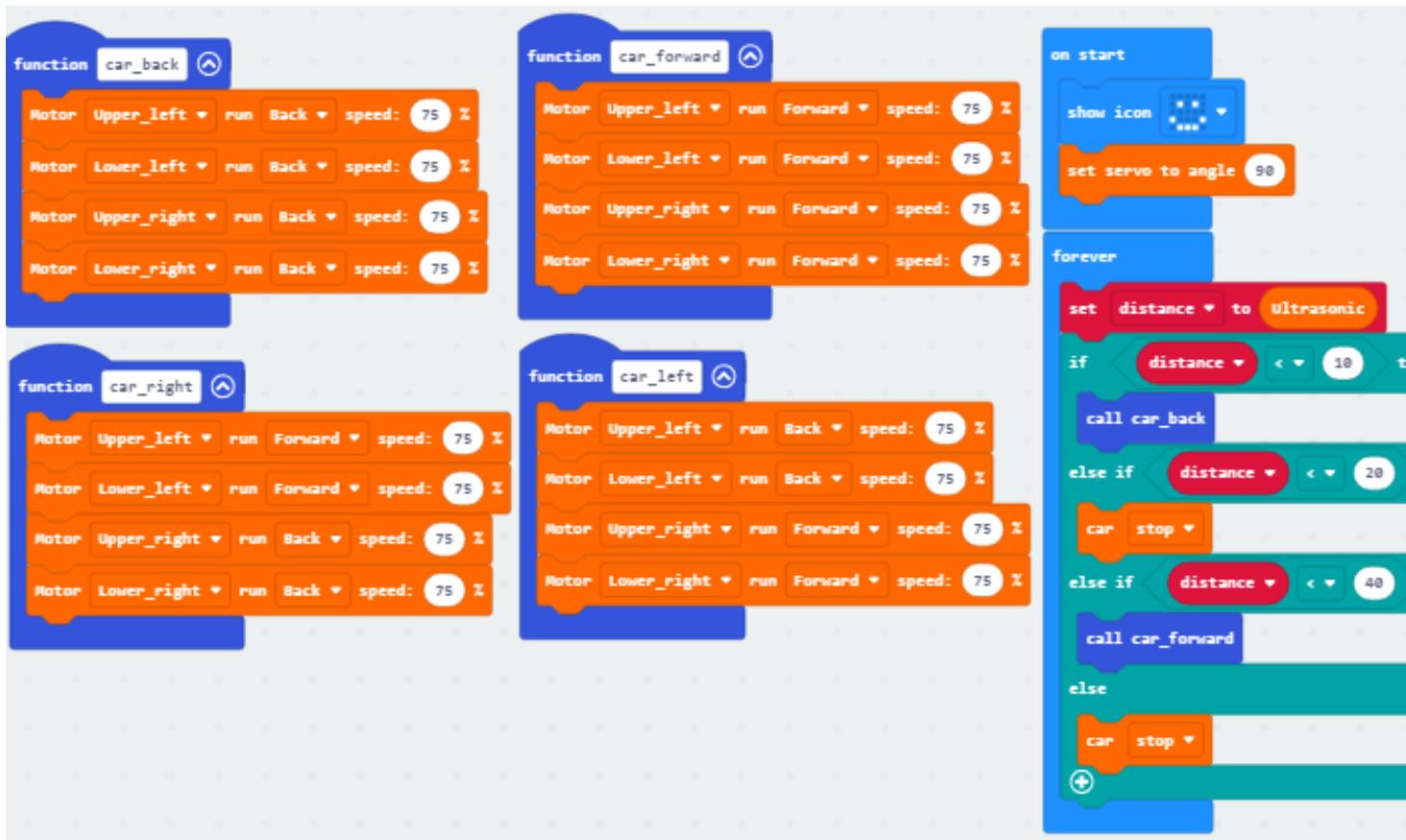
(7) change the 20 to 40, car stop to car forward;



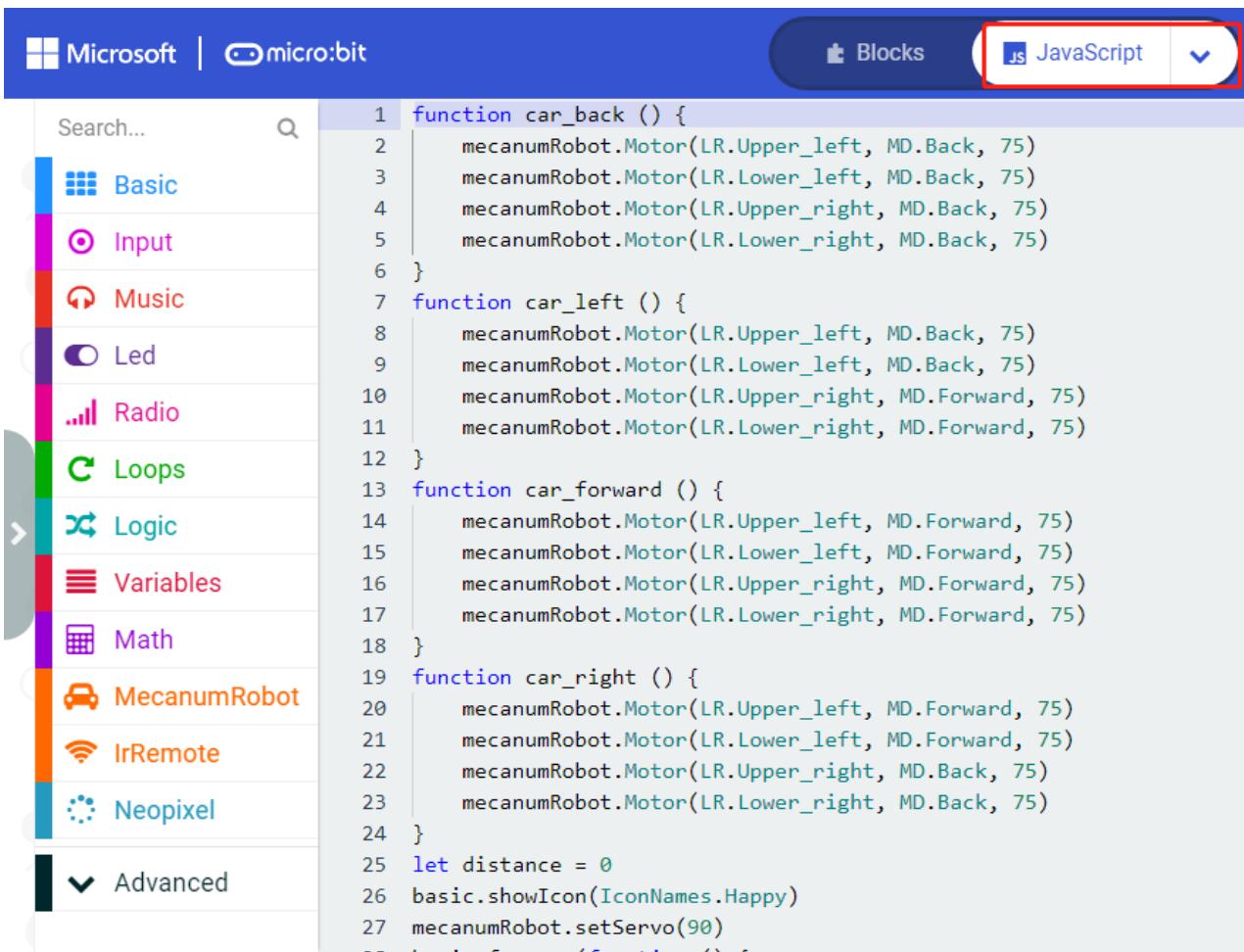
Place car stop to the last "else" :



Complete Program:



Click "JavaScript" to view the corresponding JavaScript code: :



The screenshot shows the Microsoft MakeCode interface for a micro:bit. The top bar includes the Microsoft logo, a micro:bit icon, and tabs for 'Blocks' and 'JavaScript'. The 'JavaScript' tab is selected and highlighted with a red box. On the left, a sidebar lists categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, MecanumRobot (which is selected and highlighted with a green box), IrRemote, Neopixel, and Advanced. The main workspace contains the following JavaScript code:

```
1 function car_back () {
2     mecanumRobot.Motor(LR.Upper_left, MD.Back, 75)
3     mecanumRobot.Motor(LR.Lower_left, MD.Back, 75)
4     mecanumRobot.Motor(LR.Upper_right, MD.Back, 75)
5     mecanumRobot.Motor(LR.Lower_right, MD.Back, 75)
6 }
7 function car_left () {
8     mecanumRobot.Motor(LR.Upper_left, MD.Back, 75)
9     mecanumRobot.Motor(LR.Lower_left, MD.Back, 75)
10    mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)
11    mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)
12 }
13 function car_forward () {
14     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)
15     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)
16     mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)
17     mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)
18 }
19 function car_right () {
20     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)
21     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)
22     mecanumRobot.Motor(LR.Upper_right, MD.Back, 75)
23     mecanumRobot.Motor(LR.Lower_right, MD.Back, 75)
24 }
25 let distance = 0
26 basic.showIcon(IconNames.Happy)
27 mecanumRobot.setServo(90)
```

## (5)Test Results:

Download code to micro:bit, dial POWER switch to ON end on shield, smart car could follow the obstacle to move.

([How to download?](#) [How to quick download?](#))



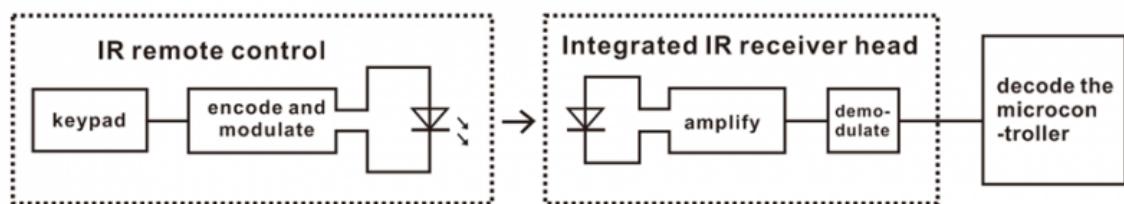
## Project 19: IR Remote Control

### 19.1: Decode IR Remote Control



#### (1) Project Description

There is no doubt that infrared remote control is ubiquitous in daily life. It is used to control various household appliances, such as TVs, stereos, video recorders and satellite signal receivers. Infrared remote control is composed of infrared transmitting and infrared receiving systems, that is, an infrared remote control and infrared receiving module and a single-chip microcomputer capable of decoding.



The 38K infrared carrier signal emitted by remote controller is encoded by the encoding chip in the remote controller. It is composed of a section of pilot code, user code, user inverse code, data code, and data inverse code.



---

The time interval of the pulse is used to distinguish whether it is a 0 or 1 signal and the encoding is made up of these 0, 1 signals.

The user code of the same remote control is unchanged. The data code can distinguish the key.

When the remote control button is pressed, the remote control sends out an infrared carrier signal. When the IR receiver receives the signal, the program will decode the carrier signal and determines which key is pressed.

The MCU decodes the received 01 signal, thereby judging what key is pressed by the remote control.

Infrared receiver we use is an infrared receiver module. Mainly composed of an infrared receiver head, it is a device that integrates reception, amplification, and demodulation. Its internal IC has completed demodulation, and can achieve from infrared reception to output and be compatible with TTL signals. Additionally, it is suitable for infrared remote control and infrared data transmission. The infrared receiving module made by the receiver has only three pins, signal line, VCC and GND.

According to the picture above, the integrated port of the infrared receiver is connected to the G port on the motor driver board and controlled by the P9 of the micro:bit.



## (2)Parameters:

- Operating Voltage: 3.3-5V (DC)
- Interface: 3PIN
- Output Signal: Digital signal
- Receiving Angle: 90 degrees
- Frequency: 38khz
- Receiving Distance: about 5m

## (3)Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- Open online Makecode editor

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

### (How to add MecanumRobot extension?)

## (4)Test Code:

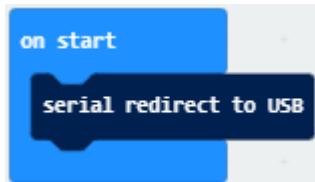
File Type	Path	File Name



Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 19.1: Decode IR Remote Control.hex	Project 19.1: Decode IR Remote Control.hex
----------	--	--

Click “Advanced” → “Serial” → “serial redirect to USB”

Place it into “on start” block.



Enter “IrRemote” → “connect IR receiver at P0”

Put it into “on start” block

IR receiving module is controlled by P9 of micro:bit board, so click P0 to select P9.



Go to “Variables” → “Make a Variable...” → “New variable name: ” dialog box,

Enter “val” and click “OK” to create variable “val”

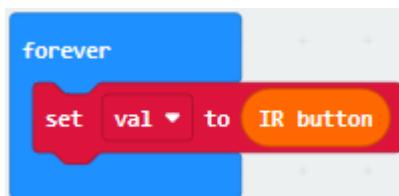


Then drag out “set val to 0” block into “forever” block.



Go to “Ir Remote” → “IR button”

Place it into 0 box



Click “Advanced” → “Serial” → “serial write value “x” =0”

Put it into “forever” block

Change “x” into “IR”

Enter “Variables” to move block “val” into 0 box behind “=”



Drag out block “pause (ms) 100” from “Basic” and delay in 1000ms

Leave it into “forever” block



## Complete Program:



"on start" : command block runs once to start program.

Serial redirect to USB

Connect IR receiver to P9

The program under the block "forever" runs cyclically.

Set val to IR button

Serial port prints IR=val

Delay in 1000ms

Click "JavaScript" to switch into the corresponding JavaScript code:



Microsoft | micro:bit

Blocks    **JavaScript** (selected)

Search...    Q

Basic    Input    Music    Led

```
1 let val = 0
2 serial.redirectToUSB()
3 irRemote.connectInfrared(DigitalPin.P8)
4 basic.forever(function () {
5     val = irRemote.returnIrButton()
6     serial.writeValue("IR", val)
7     basic.pause(1000)
8 })
9
```

Code explanation: when the buttons are not pressed, the serial monitor constantly shows 0; when pressed, the corresponding key values are displayed.

### Notes:

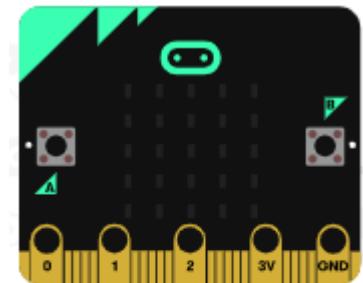
The remote control in this kit is not inclusive of batteries. We recommend you to purchase them online.(battery type:CR2025).

Make sure IR remote is good before test. There is a tip for you to check it. Open the cellphone camera , make IR remote control point at camera and press button. The remote control is good if you see the purple flashing light in the camera.

Download code to micro: bit board and don't plug off USB cable Click

Show console Device

(How to quick download?)

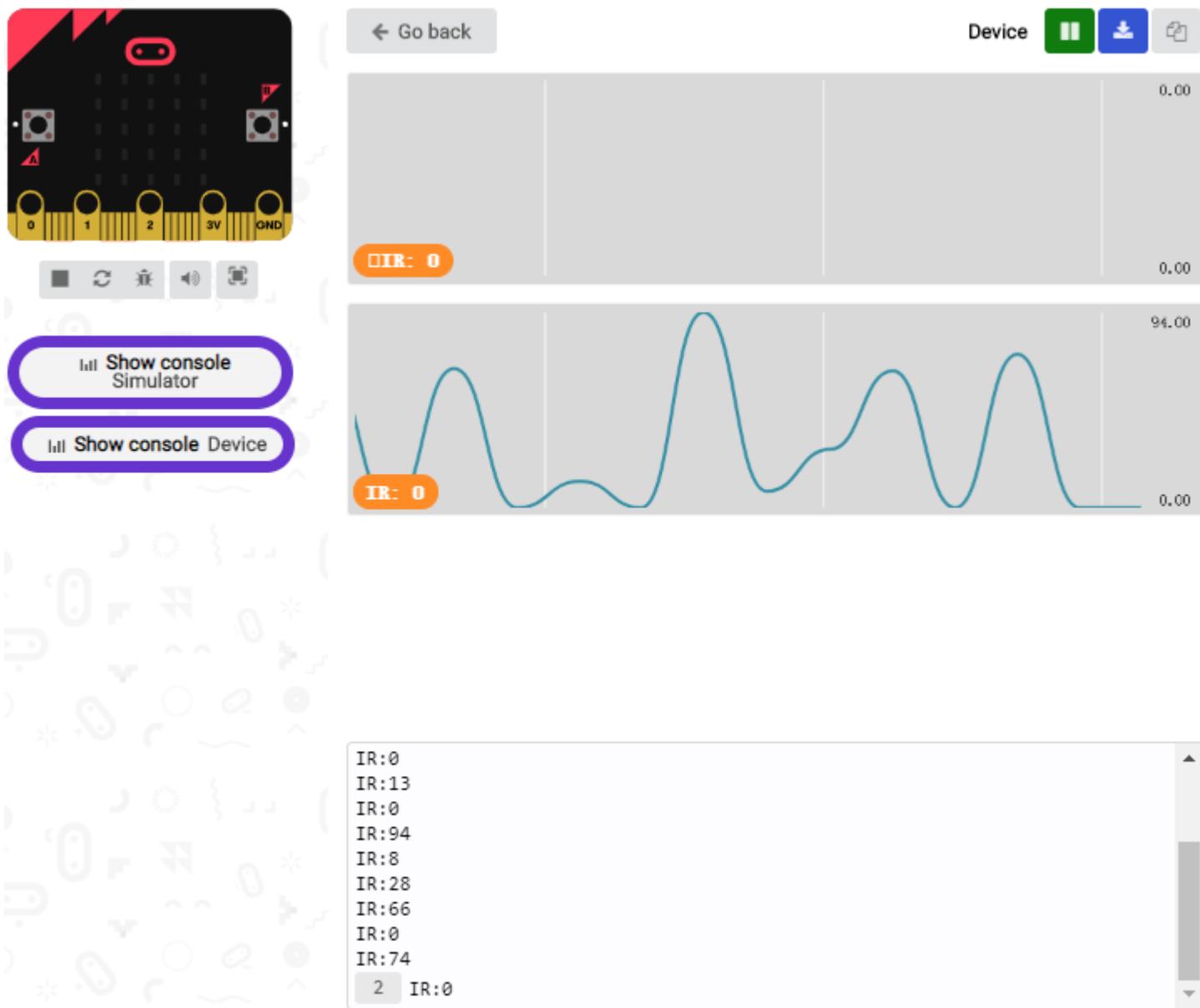


Show console  
Simulator

Show console Device



Make IR remote control point at IR receiver and press the button, the serial monitor will display the corresponding key values, as shown below:



Open CoolTerm, click Options to select SerialPort. Set COM port and 115200 baud rate. Click “OK” and “Connect” .

CoolTerm serial monitor shows the key value as follows:



Untitled\_0 \*

File Edit Connection View Window Help

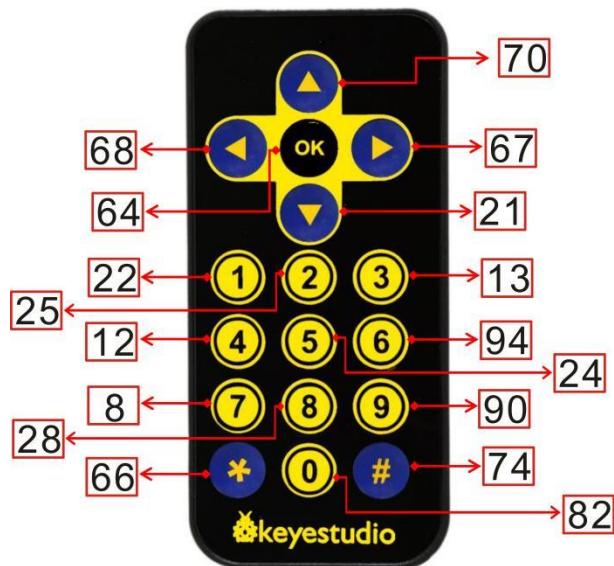
New Open Save Connect Disconnect Clear Data Options HEX View Hex Help

```
IR: 0
IR: 0
IR: 70
IR: 0
IR: 68
IR: 21
IR: 0
IR: 0
IR: 67
IR: 0
IR: 64
IR: 0
IR: 22
IR: 25
IR: 13
IR: 0
IR: 12
IR: 0
IR: 0
IR: 24
IR: 94
IR: 8
IR: 0
IR: 28
IR: 0
IR: 90
IR: 66
IR: 0
IR: 82
IR: 74
IR: 0
IR: 0
```

COM16 / 115200 8-N-1  
Connected 00:04:47

TX	RTS	DTR	DCD
RX	CTS	DSR	RI

The key value is displayed as for your reference:



## 19.2: IR Remote Control





## (1)Project Description

In this project, we combine IR remote control with car shield to make an IR remote smart car. Its principle is to control the motion of car by sending key signals from IR remote control to IR receiving module of car shield.

## (2)Experimental Preparation:

- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- Open online Makecode editor

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

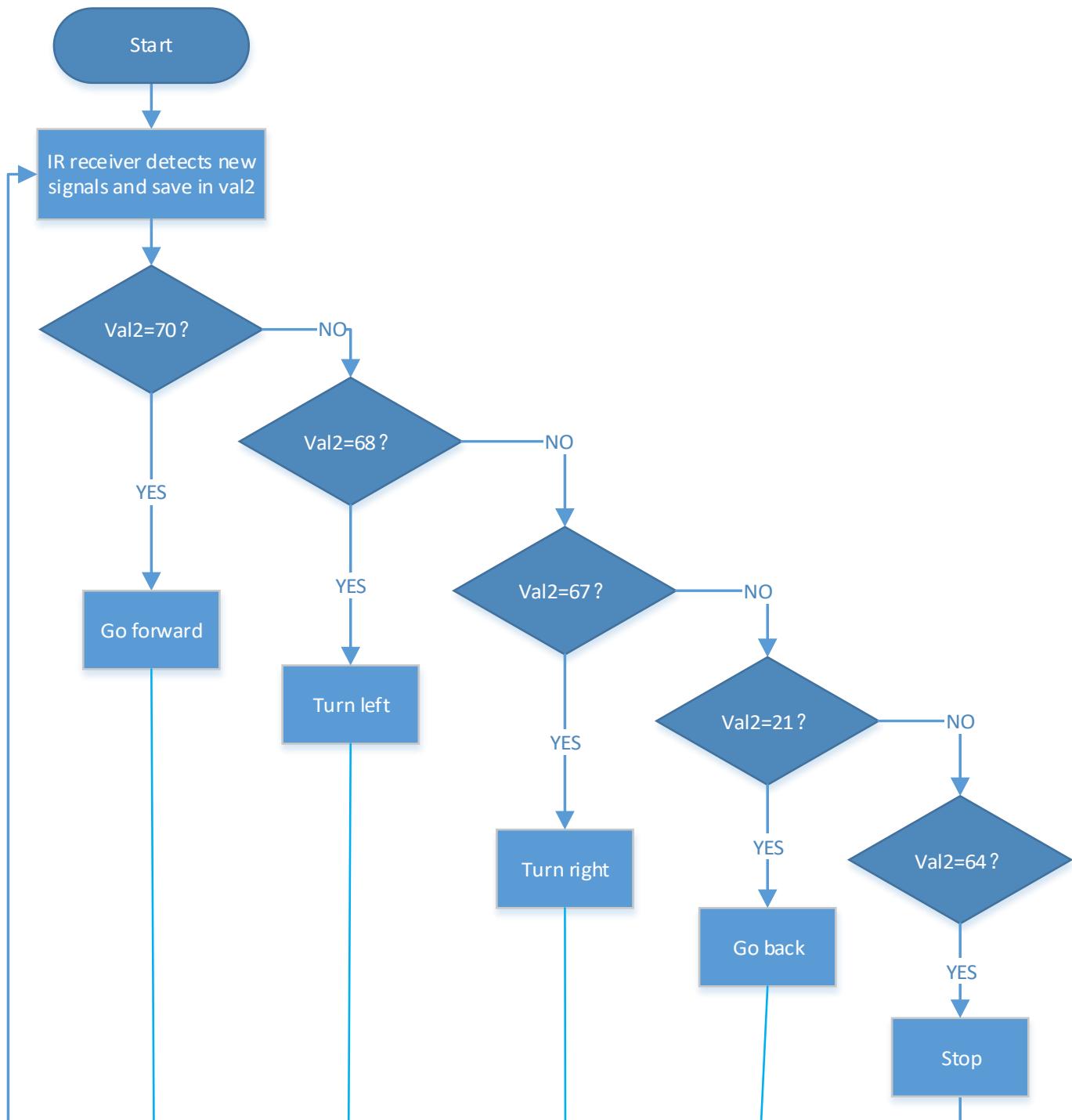
### (How to add MecanumRobot extension?)

**Note: The infrared sensor and infrared remote control should not be used in environments with infrared interference such as sunlight.**

**Because sunlight contains a lot of invisible lights, such as infrared and ultraviolet. In an environment with strong sunlight, they cannot work normally.**



### (3) Flow Chart:



### (4) Test Code

Code path:



File Type	Path	File Name
Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 19.2: IR Remote Control .hex	Project 19.2: IR Remote Control .hex

Or you could edit code step by step in the editing area.

(1) Create four functions controlling the car to move forward and back and turn left and right:



(2) Click "Ir Remote" to find and drag "connect IR receiver at P0" into "on start"; Click the little triangle behind P0 to choose P9;



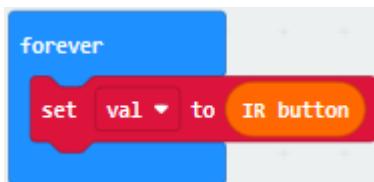
(3) Click “Variables” then click “Make a Variable...” , the dialog box “New variable name:” pops up; fill it with “val” and click “OK” to create variable “val” ;

Create variable “val2” with the same method; find and drag “set val2 to 0” to “on start” and copy it once to put into “on start” too;

Click the little triangle behind the first val2 to choose “val” ;



(4) Click “Variables” to find and drag “set val2 to 0” to “forever” ; Click the little triangle behind val2 to choose val; Click “IrRemote” to find and drag “IR button” to the “0” behind “to” ;

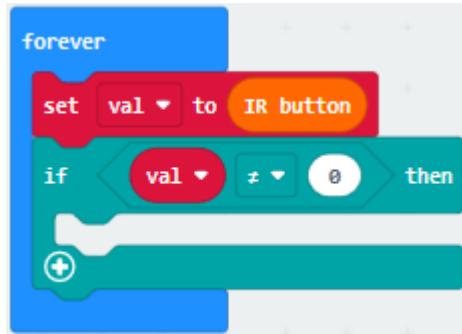


(5) Click “Logic” to find and drag “if true then” into “forever” ; find and drag

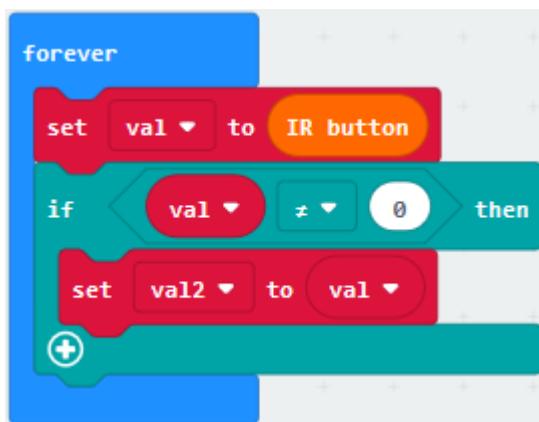


“=” into “true” ;

Click “Variables” to find and drag “val” to the left side of “=”; the 0 on the right side of “=” remain unchanged; click the little triangle behind “=” to choose “≠” ;



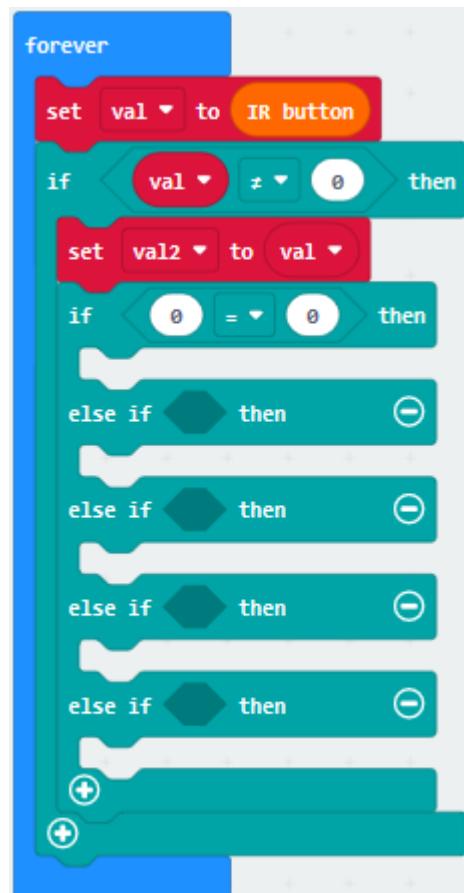
(6) Click “Variables” to find and drag “set val2 to 0” into “then” ; find and drag “val” into the o behind “to” of “set val2 to 0” ;



(7) Click “Logic” to find and drag “if...then...else” to then;

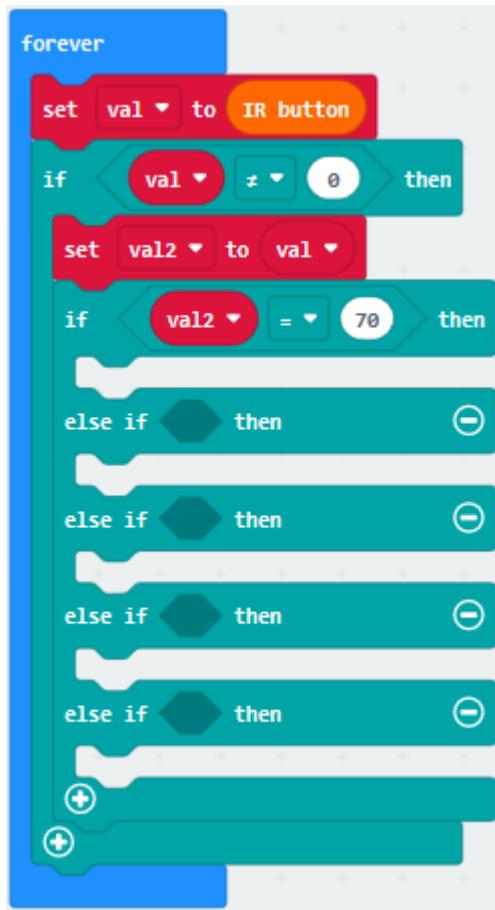
Click “⊕” of “if...then...else” four times;

Click “⊖” behind “else” once to delete else;



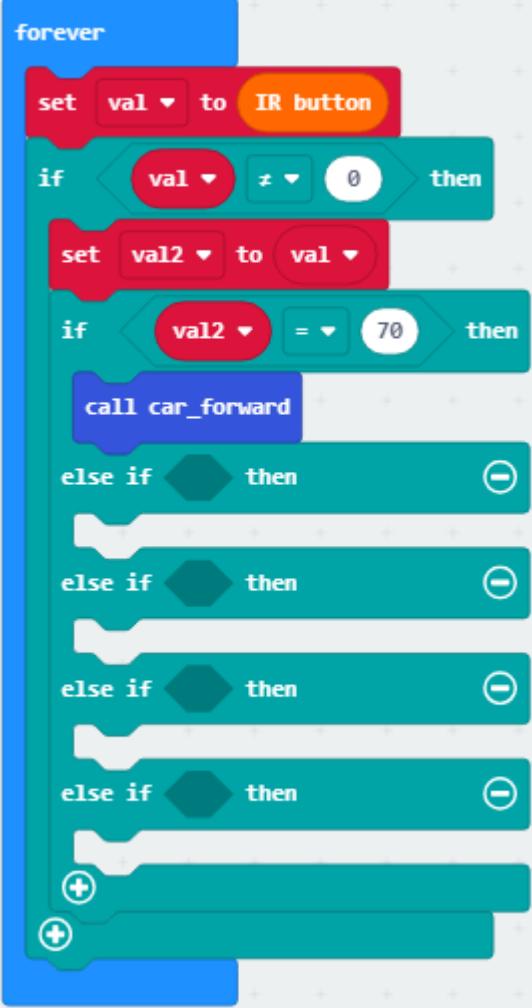
Find and drag “=” to “true” ;

(8) Click “Variables” to find and drag “val2” to the left side of “=” and change the 0 on the right of “=” to 70:



call car\_forward

(9) Click “Functions” of “Advance” to find and drag 指 to the second “then” :



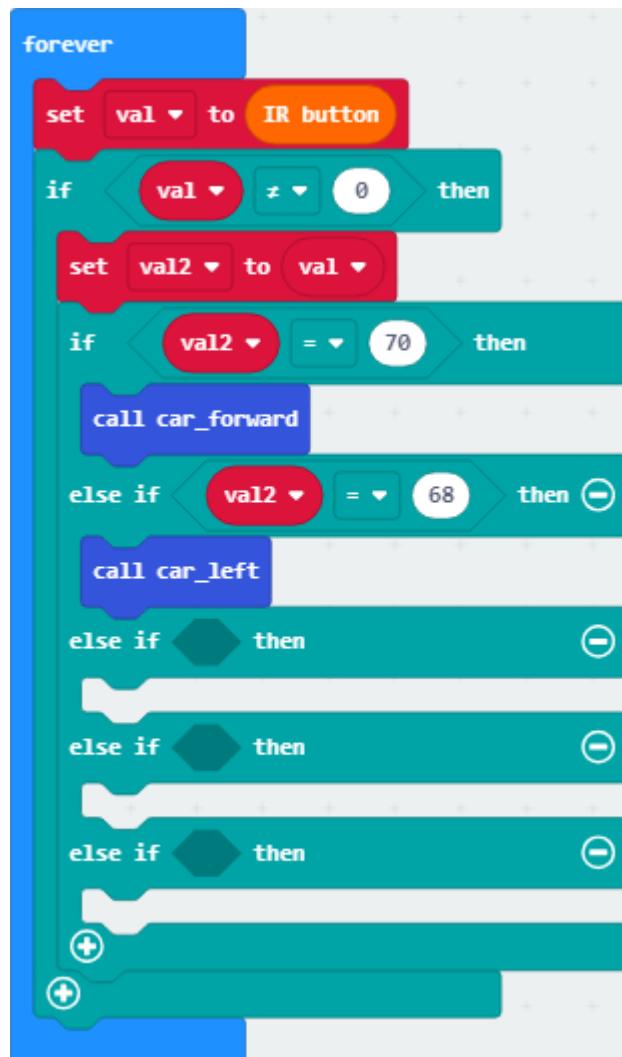
```
forever
  set [val v] to [IR button]
  if [val v] ≠ [0 v] then
    set [val2 v] to [val v]
    if [val2 v] = [70 v] then
      call [car-forward v]
    else if [ ] then
    else
    else
```

(10)Copy "val2=70" once and place it behind the first "if" ;change the 70



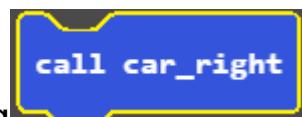
behind “=” to 68;

(11) Click “Functions” of “Advance” to find and drag  to the

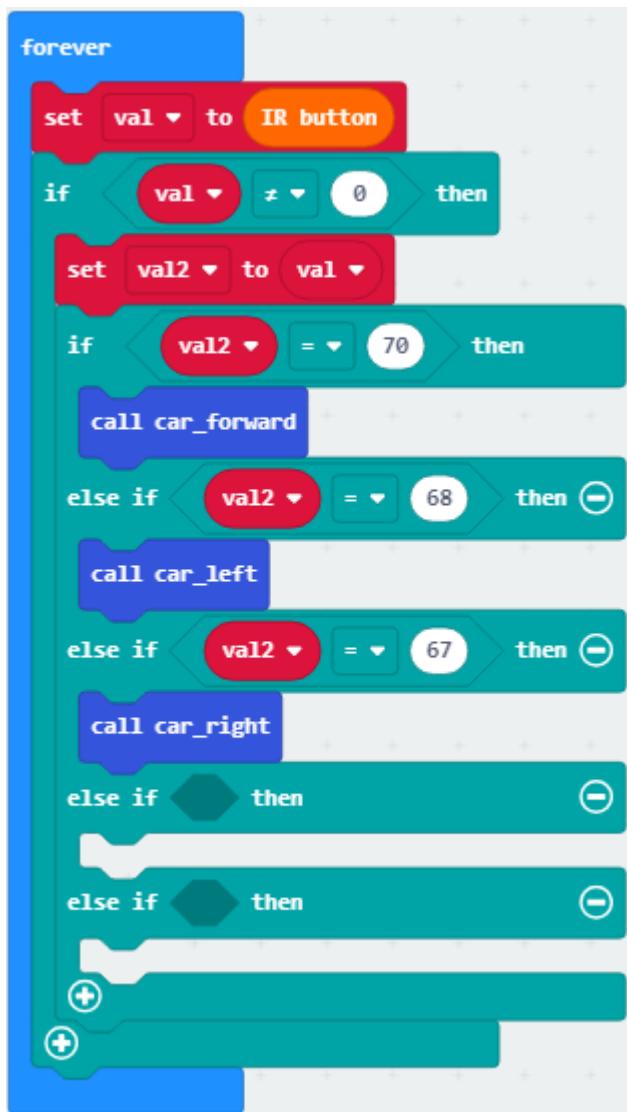


second "then" :

(2) Copy "val2=68" once and place it behind the second "else if"; change the 68 behind "=" to 67; place it in the forth "then"; Click



"Functions" of "Advance" to find and drag to the forth "then" :



(13)Copy “val2=67” once and put it behind “=” of the third “else if ; change the number 67 to 21;click “Functions” of “Advance” to find and

drag  to the fifth “then” :



```
forever
  set [val v] to [IR button]
  if [val v] ≠ [0 v] then
    set [val2 v] to [val v]
    if [val2 v] = [70 v] then
      call [car-forward v]
    else if [val2 v] = [68 v] then
      call [car-left v]
    else if [val2 v] = [67 v] then
      call [car-right v]
    else if [val2 v] = [21 v] then
      call [car-back v]
    else if [ ] then
      stop
```

(14)Copy "val2=21" once and place it behind the fourth "else if ";change the the number 21 behind "=" to 64; Click "MecanumRobot" to find and



drag **car stop** to the sixth "then" :



```
forever
  set [val v] to [IR button]
  if [val v] ≠ [0 v] then
    set [val2 v] to [val v]
    if [val2 v] = [70 v] then
      call [car-forward v]
    else if [val2 v] = [68 v] then
      call [car-left v]
    else if [val2 v] = [67 v] then
      call [car-right v]
    else if [val2 v] = [21 v] then
      call [car-back v]
    else if [val2 v] = [64 v] then
      stop
    end
  end
end
```

Complete Program:



- ① The "on start" command begins the program.
- ② Connect the IR receiver to Port 9.
- ③ Set the variable val to 0.
- ④ Set the variable val2 to 0.
- ⑤ In the "forever" instruction block:
- ⑥ Set val to IR button
- ⑦ When the variable val ≠ 0, the program under then:
- ⑧ Set variable val2 to val.
- ⑨ When val2=70 is established, then:
- ⑩ The car goes forward
- ⑪ When val2=68 is established, then:
- ⑫ Turn left
- ⑬ When val2=67 is established, then:
- ⑭ Car turn right
- ⑮ When val2=21 is established, then:
- ⑯ The car goes back
- ⑰ When val2=64 is established, then:
- ⑱ The car stops

Click "JavaScript" to switch into the corresponding JavaScript code:



Microsoft | micro:bit

Blocks JavaScript ▾

Search... Q

Basic Input Music Led Radio Loops Logic Variables Math MecanumRobot IrRemote Neopixel Advanced

```
1 function car_back () {
2     mecanumRobot.Motor(LR.Upper_left, MD.Back, 75)
3     mecanumRobot.Motor(LR.Lower_left, MD.Back, 75)
4     mecanumRobot.Motor(LR.Upper_right, MD.Back, 75)
5     mecanumRobot.Motor(LR.Lower_right, MD.Back, 75)
6 }
7 function car_left () {
8     mecanumRobot.Motor(LR.Upper_left, MD.Back, 75)
9     mecanumRobot.Motor(LR.Lower_left, MD.Back, 75)
10    mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)
11    mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)
12 }
13 function car_forward () {
14     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)
15     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)
16     mecanumRobot.Motor(LR.Upper_right, MD.Forward, 75)
17     mecanumRobot.Motor(LR.Lower_right, MD.Forward, 75)
18 }
19 function car_right () {
20     mecanumRobot.Motor(LR.Upper_left, MD.Forward, 75)
21     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 75)
22     mecanumRobot.Motor(LR.Upper_right, MD.Back, 75)
23     mecanumRobot.Motor(LR.Lower_right, MD.Back, 75)
24 }
25 irRemote.connectInfrared(DigitalPin.P9)
26 let val = 0
27 let val2 = 0
28 let i = 0
29 let f = 0
30 let r = 0
31 let g = 0
32 let b = 0
33 let c = 0
34 let d = 0
35 let e = 0
36 let f = 0
37 let g = 0
38 let h = 0
39 let i = 0
40 let j = 0
41 let k = 0
42 let l = 0
43 let m = 0
44 let n = 0
45 let o = 0
46 let p = 0
47 let q = 0
48 let r = 0
49 let s = 0
50 let t = 0
51 let u = 0
52 let v = 0
53 let w = 0
54 let x = 0
55 let y = 0
56 let z = 0
57 let aa = 0
58 let bb = 0
59 let cc = 0
60 let dd = 0
61 let ee = 0
62 let ff = 0
63 let gg = 0
64 let hh = 0
65 let ii = 0
66 let jj = 0
67 let kk = 0
68 let ll = 0
69 let mm = 0
70 let nn = 0
71 let oo = 0
72 let pp = 0
73 let rr = 0
74 let tt = 0
75 let uu = 0
76 let vv = 0
77 let ww = 0
78 let xx = 0
79 let yy = 0
80 let zz = 0
81 let aa = 0
82 let bb = 0
83 let cc = 0
84 let dd = 0
85 let ee = 0
86 let ff = 0
87 let gg = 0
88 let hh = 0
89 let ii = 0
90 let jj = 0
91 let kk = 0
92 let ll = 0
93 let mm = 0
94 let nn = 0
95 let oo = 0
96 let pp = 0
97 let rr = 0
98 let tt = 0
99 let uu = 0
100 let vv = 0
101 let ww = 0
102 let xx = 0
103 let yy = 0
104 let zz = 0
105 let aa = 0
106 let bb = 0
107 let cc = 0
108 let dd = 0
109 let ee = 0
110 let ff = 0
111 let gg = 0
112 let hh = 0
113 let ii = 0
114 let jj = 0
115 let kk = 0
116 let ll = 0
117 let mm = 0
118 let nn = 0
119 let oo = 0
120 let pp = 0
121 let rr = 0
122 let tt = 0
123 let uu = 0
124 let vv = 0
125 let ww = 0
126 let xx = 0
127 let yy = 0
128 let zz = 0
129 let aa = 0
130 let bb = 0
131 let cc = 0
132 let dd = 0
133 let ee = 0
134 let ff = 0
135 let gg = 0
136 let hh = 0
137 let ii = 0
138 let jj = 0
139 let kk = 0
140 let ll = 0
141 let mm = 0
142 let nn = 0
143 let oo = 0
144 let pp = 0
145 let rr = 0
146 let tt = 0
147 let uu = 0
148 let vv = 0
149 let ww = 0
150 let xx = 0
151 let yy = 0
152 let zz = 0
153 let aa = 0
154 let bb = 0
155 let cc = 0
156 let dd = 0
157 let ee = 0
158 let ff = 0
159 let gg = 0
160 let hh = 0
161 let ii = 0
162 let jj = 0
163 let kk = 0
164 let ll = 0
165 let mm = 0
166 let nn = 0
167 let oo = 0
168 let pp = 0
169 let rr = 0
170 let tt = 0
171 let uu = 0
172 let vv = 0
173 let ww = 0
174 let xx = 0
175 let yy = 0
176 let zz = 0
177 let aa = 0
178 let bb = 0
179 let cc = 0
180 let dd = 0
181 let ee = 0
182 let ff = 0
183 let gg = 0
184 let hh = 0
185 let ii = 0
186 let jj = 0
187 let kk = 0
188 let ll = 0
189 let mm = 0
190 let nn = 0
191 let oo = 0
192 let pp = 0
193 let rr = 0
194 let tt = 0
195 let uu = 0
196 let vv = 0
197 let ww = 0
198 let xx = 0
199 let yy = 0
200 let zz = 0
201 let aa = 0
202 let bb = 0
203 let cc = 0
204 let dd = 0
205 let ee = 0
206 let ff = 0
207 let gg = 0
208 let hh = 0
209 let ii = 0
210 let jj = 0
211 let kk = 0
212 let ll = 0
213 let mm = 0
214 let nn = 0
215 let oo = 0
216 let pp = 0
217 let rr = 0
218 let tt = 0
219 let uu = 0
220 let vv = 0
221 let ww = 0
222 let xx = 0
223 let yy = 0
224 let zz = 0
225 let aa = 0
226 let bb = 0
227 let cc = 0
228 let dd = 0
229 let ee = 0
230 let ff = 0
231 let gg = 0
232 let hh = 0
233 let ii = 0
234 let jj = 0
235 let kk = 0
236 let ll = 0
237 let mm = 0
238 let nn = 0
239 let oo = 0
240 let pp = 0
241 let rr = 0
242 let tt = 0
243 let uu = 0
244 let vv = 0
245 let ww = 0
246 let xx = 0
247 let yy = 0
248 let zz = 0
249 let aa = 0
250 let bb = 0
251 let cc = 0
252 let dd = 0
253 let ee = 0
254 let ff = 0
255 let gg = 0
256 let hh = 0
257 let ii = 0
258 let jj = 0
259 let kk = 0
260 let ll = 0
261 let mm = 0
262 let nn = 0
263 let oo = 0
264 let pp = 0
265 let rr = 0
266 let tt = 0
267 let uu = 0
268 let vv = 0
269 let ww = 0
270 let xx = 0
271 let yy = 0
272 let zz = 0
273 let aa = 0
274 let bb = 0
275 let cc = 0
276 let dd = 0
277 let ee = 0
278 let ff = 0
279 let gg = 0
280 let hh = 0
281 let ii = 0
282 let jj = 0
283 let kk = 0
284 let ll = 0
285 let mm = 0
286 let nn = 0
287 let oo = 0
288 let pp = 0
289 let rr = 0
290 let tt = 0
291 let uu = 0
292 let vv = 0
293 let ww = 0
294 let xx = 0
295 let yy = 0
296 let zz = 0
297 let aa = 0
298 let bb = 0
299 let cc = 0
300 let dd = 0
301 let ee = 0
302 let ff = 0
303 let gg = 0
304 let hh = 0
305 let ii = 0
306 let jj = 0
307 let kk = 0
308 let ll = 0
309 let mm = 0
310 let nn = 0
311 let oo = 0
312 let pp = 0
313 let rr = 0
314 let tt = 0
315 let uu = 0
316 let vv = 0
317 let ww = 0
318 let xx = 0
319 let yy = 0
320 let zz = 0
321 let aa = 0
322 let bb = 0
323 let cc = 0
324 let dd = 0
325 let ee = 0
326 let ff = 0
327 let gg = 0
328 let hh = 0
329 let ii = 0
330 let jj = 0
331 let kk = 0
332 let ll = 0
333 let mm = 0
334 let nn = 0
335 let oo = 0
336 let pp = 0
337 let rr = 0
338 let tt = 0
339 let uu = 0
340 let vv = 0
341 let ww = 0
342 let xx = 0
343 let yy = 0
344 let zz = 0
345 let aa = 0
346 let bb = 0
347 let cc = 0
348 let dd = 0
349 let ee = 0
350 let ff = 0
351 let gg = 0
352 let hh = 0
353 let ii = 0
354 let jj = 0
355 let kk = 0
356 let ll = 0
357 let mm = 0
358 let nn = 0
359 let oo = 0
360 let pp = 0
361 let rr = 0
362 let tt = 0
363 let uu = 0
364 let vv = 0
365 let ww = 0
366 let xx = 0
367 let yy = 0
368 let zz = 0
369 let aa = 0
370 let bb = 0
371 let cc = 0
372 let dd = 0
373 let ee = 0
374 let ff = 0
375 let gg = 0
376 let hh = 0
377 let ii = 0
378 let jj = 0
379 let kk = 0
380 let ll = 0
381 let mm = 0
382 let nn = 0
383 let oo = 0
384 let pp = 0
385 let rr = 0
386 let tt = 0
387 let uu = 0
388 let vv = 0
389 let ww = 0
390 let xx = 0
391 let yy = 0
392 let zz = 0
393 let aa = 0
394 let bb = 0
395 let cc = 0
396 let dd = 0
397 let ee = 0
398 let ff = 0
399 let gg = 0
400 let hh = 0
401 let ii = 0
402 let jj = 0
403 let kk = 0
404 let ll = 0
405 let mm = 0
406 let nn = 0
407 let oo = 0
408 let pp = 0
409 let rr = 0
410 let tt = 0
411 let uu = 0
412 let vv = 0
413 let ww = 0
414 let xx = 0
415 let yy = 0
416 let zz = 0
417 let aa = 0
418 let bb = 0
419 let cc = 0
420 let dd = 0
421 let ee = 0
422 let ff = 0
423 let gg = 0
424 let hh = 0
425 let ii = 0
426 let jj = 0
427 let kk = 0
428 let ll = 0
429 let mm = 0
430 let nn = 0
431 let oo = 0
432 let pp = 0
433 let rr = 0
434 let tt = 0
435 let uu = 0
436 let vv = 0
437 let ww = 0
438 let xx = 0
439 let yy = 0
440 let zz = 0
441 let aa = 0
442 let bb = 0
443 let cc = 0
444 let dd = 0
445 let ee = 0
446 let ff = 0
447 let gg = 0
448 let hh = 0
449 let ii = 0
450 let jj = 0
451 let kk = 0
452 let ll = 0
453 let mm = 0
454 let nn = 0
455 let oo = 0
456 let pp = 0
457 let rr = 0
458 let tt = 0
459 let uu = 0
460 let vv = 0
461 let ww = 0
462 let xx = 0
463 let yy = 0
464 let zz = 0
465 let aa = 0
466 let bb = 0
467 let cc = 0
468 let dd = 0
469 let ee = 0
470 let ff = 0
471 let gg = 0
472 let hh = 0
473 let ii = 0
474 let jj = 0
475 let kk = 0
476 let ll = 0
477 let mm = 0
478 let nn = 0
479 let oo = 0
480 let pp = 0
481 let rr = 0
482 let tt = 0
483 let uu = 0
484 let vv = 0
485 let ww = 0
486 let xx = 0
487 let yy = 0
488 let zz = 0
489 let aa = 0
490 let bb = 0
491 let cc = 0
492 let dd = 0
493 let ee = 0
494 let ff = 0
495 let gg = 0
496 let hh = 0
497 let ii = 0
498 let jj = 0
499 let kk = 0
500 let ll = 0
501 let mm = 0
502 let nn = 0
503 let oo = 0
504 let pp = 0
505 let rr = 0
506 let tt = 0
507 let uu = 0
508 let vv = 0
509 let ww = 0
510 let xx = 0
511 let yy = 0
512 let zz = 0
513 let aa = 0
514 let bb = 0
515 let cc = 0
516 let dd = 0
517 let ee = 0
518 let ff = 0
519 let gg = 0
520 let hh = 0
521 let ii = 0
522 let jj = 0
523 let kk = 0
524 let ll = 0
525 let mm = 0
526 let nn = 0
527 let oo = 0
528 let pp = 0
529 let rr = 0
530 let tt = 0
531 let uu = 0
532 let vv = 0
533 let ww = 0
534 let xx = 0
535 let yy = 0
536 let zz = 0
537 let aa = 0
538 let bb = 0
539 let cc = 0
540 let dd = 0
541 let ee = 0
542 let ff = 0
543 let gg = 0
544 let hh = 0
545 let ii = 0
546 let jj = 0
547 let kk = 0
548 let ll = 0
549 let mm = 0
550 let nn = 0
551 let oo = 0
552 let pp = 0
553 let rr = 0
554 let tt = 0
555 let uu = 0
556 let vv = 0
557 let ww = 0
558 let xx = 0
559 let yy = 0
560 let zz = 0
561 let aa = 0
562 let bb = 0
563 let cc = 0
564 let dd = 0
565 let ee = 0
566 let ff = 0
567 let gg = 0
568 let hh = 0
569 let ii = 0
570 let jj = 0
571 let kk = 0
572 let ll = 0
573 let mm = 0
574 let nn = 0
575 let oo = 0
576 let pp = 0
577 let rr = 0
578 let tt = 0
579 let uu = 0
580 let vv = 0
581 let ww = 0
582 let xx = 0
583 let yy = 0
584 let zz = 0
585 let aa = 0
586 let bb = 0
587 let cc = 0
588 let dd = 0
589 let ee = 0
590 let ff = 0
591 let gg = 0
592 let hh = 0
593 let ii = 0
594 let jj = 0
595 let kk = 0
596 let ll = 0
597 let mm = 0
598 let nn = 0
599 let oo = 0
600 let pp = 0
601 let rr = 0
602 let tt = 0
603 let uu = 0
604 let vv = 0
605 let ww = 0
606 let xx = 0
607 let yy = 0
608 let zz = 0
609 let aa = 0
610 let bb = 0
611 let cc = 0
612 let dd = 0
613 let ee = 0
614 let ff = 0
615 let gg = 0
616 let hh = 0
617 let ii = 0
618 let jj = 0
619 let kk = 0
620 let ll = 0
621 let mm = 0
622 let nn = 0
623 let oo = 0
624 let pp = 0
625 let rr = 0
626 let tt = 0
627 let uu = 0
628 let vv = 0
629 let ww = 0
630 let xx = 0
631 let yy = 0
632 let zz = 0
633 let aa = 0
634 let bb = 0
635 let cc = 0
636 let dd = 0
637 let ee = 0
638 let ff = 0
639 let gg = 0
640 let hh = 0
641 let ii = 0
642 let jj = 0
643 let kk = 0
644 let ll = 0
645 let mm = 0
646 let nn = 0
647 let oo = 0
648 let pp = 0
649 let rr = 0
650 let tt = 0
651 let uu = 0
652 let vv = 0
653 let ww = 0
654 let xx = 0
655 let yy = 0
656 let zz = 0
657 let aa = 0
658 let bb = 0
659 let cc = 0
660 let dd = 0
661 let ee = 0
662 let ff = 0
663 let gg = 0
664 let hh = 0
665 let ii = 0
666 let jj = 0
667 let kk = 0
668 let ll = 0
669 let mm = 0
670 let nn = 0
671 let oo = 0
672 let pp = 0
673 let rr = 0
674 let tt = 0
675 let uu = 0
676 let vv = 0
677 let ww = 0
678 let xx = 0
679 let yy = 0
680 let zz = 0
681 let aa = 0
682 let bb = 0
683 let cc = 0
684 let dd = 0
685 let ee = 0
686 let ff = 0
687 let gg = 0
688 let hh = 0
689 let ii = 0
690 let jj = 0
691 let kk = 0
692 let ll = 0
693 let mm = 0
694 let nn = 0
695 let oo = 0
696 let pp = 0
697 let rr = 0
698 let tt = 0
699 let uu = 0
700 let vv = 0
701 let ww = 0
702 let xx = 0
703 let yy = 0
704 let zz = 0
705 let aa = 0
706 let bb = 0
707 let cc = 0
708 let dd = 0
709 let ee = 0
710 let ff = 0
711 let gg = 0
712 let hh = 0
713 let ii = 0
714 let jj = 0
715 let kk = 0
716 let ll = 0
717 let mm = 0
718 let nn = 0
719 let oo = 0
720 let pp = 0
721 let rr = 0
722 let tt = 0
723 let uu = 0
724 let vv = 0
725 let ww = 0
726 let xx = 0
727 let yy = 0
728 let zz = 0
729 let aa = 0
730 let bb = 0
731 let cc = 0
732 let dd = 0
733 let ee = 0
734 let ff = 0
735 let gg = 0
736 let hh = 0
737 let ii = 0
738 let jj = 0
739 let kk = 0
740 let ll = 0
741 let mm = 0
742 let nn = 0
743 let oo = 0
744 let pp = 0
745 let rr = 0
746 let tt = 0
747 let uu = 0
748 let vv = 0
749 let ww = 0
750 let xx = 0
751 let yy = 0
752 let zz = 0
753 let aa = 0
754 let bb = 0
755 let cc = 0
756 let dd = 0
757 let ee = 0
758 let ff = 0
759 let gg = 0
760 let hh = 0
761 let ii = 0
762 let jj = 0
763 let kk = 0
764 let ll = 0
765 let mm = 0
766 let nn = 0
767 let oo = 0
768 let pp = 0
769 let rr = 0
770 let tt = 0
771 let uu = 0
772 let vv = 0
773 let ww = 0
774 let xx = 0
775 let yy = 0
776 let zz = 0
777 let aa = 0
778 let bb = 0
779 let cc = 0
780 let dd = 0
781 let ee = 0
782 let ff = 0
783 let gg = 0
784 let hh = 0
785 let ii = 0
786 let jj = 0
787 let kk = 0
788 let ll = 0
789 let mm = 0
790 let nn = 0
791 let oo = 0
792 let pp = 0
793 let rr = 0
794 let tt = 0
795 let uu = 0
796 let vv = 0
797 let ww = 0
798 let xx = 0
799 let yy = 0
800 let zz = 0
801 let aa = 0
802 let bb = 0
803 let cc = 0
804 let dd = 0
805 let ee = 0
806 let ff = 0
807 let gg = 0
808 let hh = 0
809 let ii = 0
810 let jj = 0
811 let kk = 0
812 let ll = 0
813 let mm = 0
814 let nn = 0
815 let oo = 0
816 let pp = 0
817 let rr = 0
818 let tt = 0
819 let uu = 0
820 let vv = 0
821 let ww = 0
822 let xx = 0
823 let yy = 0
824 let zz = 0
825 let aa = 0
826 let bb = 0
827 let cc = 0
828 let dd = 0
829 let ee = 0
830 let ff = 0
831 let gg = 0
832 let hh = 0
833 let ii = 0
834 let jj = 0
835 let kk = 0
836 let ll = 0
837 let mm = 0
838 let nn = 0
839 let oo = 0
840 let pp = 0
841 let rr = 0
842 let tt = 0
843 let uu = 0
844 let vv = 0
845 let ww = 0
846 let xx = 0
847 let yy = 0
848 let zz = 0
849 let aa = 0
850 let bb = 0
851 let cc = 0
852 let dd = 0
853 let ee = 0
854 let ff = 0
855 let gg = 0
856 let hh = 0
857 let ii = 0
858 let jj = 0
859 let kk = 0
860 let ll = 0
861 let mm = 0
862 let nn = 0
863 let oo = 0
864 let pp = 0
865 let rr = 0
866 let tt = 0
867 let uu = 0
868 let vv = 0
869 let ww = 0
870 let xx = 0
871 let yy = 0
872 let zz = 0
873 let aa = 0
874 let bb = 0
875 let cc = 0
876 let dd = 0
877 let ee = 0
878 let ff = 0
879 let gg = 0
880 let hh = 0
881 let ii = 0
882 let jj = 0
883 let kk = 0
884 let ll = 0
885 let mm = 0
886 let nn = 0
887 let oo = 0
888 let pp = 0
889 let rr = 0
890 let tt = 0
891 let uu = 0
892 let vv = 0
893 let ww = 0
894 let xx = 0
895 let yy = 0
896 let zz = 0
897 let aa = 0
898 let bb = 0
899 let cc = 0
900 let dd = 0
901 let ee = 0
902 let ff = 0
903 let gg = 0
904 let hh = 0
905 let ii = 0
906 let jj = 0
907 let kk = 0
908 let ll = 0
909 let mm = 0
910 let nn = 0
911 let oo = 0
912 let pp = 0
913 let rr = 0
914 let tt = 0
915 let uu = 0
916 let vv = 0
917 let ww = 0
918 let xx = 0
919 let yy = 0
920 let zz = 0
921 let aa = 0
922 let bb = 0
923 let cc = 0
924 let dd = 0
925 let ee = 0
926 let ff = 0
927 let gg = 0
928 let hh = 0
929 let ii = 0
930 let jj = 0
931 let kk = 0
932 let ll = 0
933 let mm = 0
934 let nn = 0
935 let oo = 0
936 let pp = 0
937 let rr = 0
938 let tt = 0
939 let uu = 0
940 let vv = 0
941 let ww = 0
942 let xx = 0
943 let yy = 0
944 let zz = 0
945 let aa = 0
946 let bb = 0
947 let cc = 0
948 let dd = 0
949 let ee = 0
950 let ff = 0
951 let gg = 0
952 let hh = 0
953 let ii = 0
954 let jj = 0
955 let kk = 0
956 let ll = 0
957 let mm = 0
958 let nn = 0
959 let oo = 0
960 let pp = 0
961 let rr = 0
962 let tt = 0
963 let uu = 0
964 let vv = 0
965 let ww = 0
966 let xx = 0
967 let yy = 0
968 let zz = 0
969 let aa = 0
970 let bb = 0
971 let cc = 0
972 let dd = 0
973 let ee = 0
974 let ff = 0
975 let gg = 0
976 let hh = 0
977 let ii = 0
978 let jj = 0
979 let kk = 0
980 let ll = 0
981 let mm = 0
982 let nn = 0
983 let oo = 0
984 let pp = 0
985 let rr = 0
986 let tt = 0
987 let uu = 0
988 let vv = 0
989 let ww = 0
990 let xx = 0
991 let yy = 0
992 let zz = 0
993 let aa = 0
994 let bb = 0
995 let cc = 0
996 let dd = 0
997 let ee = 0
998 let ff = 0
999 let gg = 0
1000 let hh = 0
1001 let ii = 0
1002 let jj = 0
1003 let kk = 0
1004 let ll = 0
1005 let mm = 0
1006 let nn = 0
1007 let oo = 0
1008 let pp = 0
1009 let rr = 0
1010 let tt = 0
1011 let uu = 0
1012 let vv = 0
1013 let ww = 0
1014 let xx = 0
1015 let yy = 0
1016 let zz = 0
1017 let aa = 0
1018 let bb = 0
1019 let cc = 0
1020 let dd = 0
1021 let ee = 0
1022 let ff = 0
1023 let gg = 0
1024 let hh = 0
1025 let ii = 0
1026 let jj = 0
1027 let kk = 0
1028 let ll = 0
1029 let mm = 0
1030 let nn = 0
1031 let oo = 0
1032 let pp = 0
1033 let rr = 0
1034 let tt = 0
1035 let uu = 0
1036 let vv = 0
1037 let ww = 0
1038 let xx = 0
1039 let yy = 0
1040 let zz = 0
1041 let aa = 0
1042 let bb = 0
1043 let cc = 0
1044 let dd = 0
1045 let ee = 0
1046 let ff = 0
1047 let gg = 0
1048 let hh = 0
1049 let ii = 0
1050 let jj = 0
1051 let kk = 0
1052 let ll = 0
1053 let mm = 0
1054 let nn = 0
1055 let oo = 0
1056 let pp = 0
1057 let rr = 0
1058 let tt = 0
1059 let uu = 0
1060 let vv = 0
1061 let ww = 0
1062 let xx = 0
1063 let yy = 0
1064 let zz = 0
1065 let aa = 0
1066 let bb = 0
1067 let cc = 0
1068 let dd = 0
1069 let ee = 0
1070 let ff = 0
1071 let gg = 0
1072 let hh = 0
1073 let ii = 0
1074 let jj = 0
1075 let kk = 0
1076 let ll = 0
1077 let mm = 0
1078 let nn = 0
1079 let oo = 0
1080 let pp = 0
1081 let rr = 0
1082 let tt = 0
1083 let uu = 0
1084 let vv = 0
1085 let ww = 0
1086 let xx = 0
1087 let yy = 0
1088 let zz = 0
1089 let aa = 0
1090 let bb = 0
1091 let cc = 0
1092 let dd = 0
1093 let ee = 0
1094 let ff = 0
1095 let gg = 0
1096 let hh = 0
1097 let ii = 0
1098 let jj = 0
1099 let kk = 0
1100 let ll = 0
1101 let mm = 0
1102 let nn = 0
1103 let oo = 0
1104 let pp = 0
1105 let rr = 0
1106 let tt = 0
1107 let uu = 0
1108 let vv = 0
1109 let ww = 0
1110 let xx = 0
1111 let yy = 0
1112 let zz = 0
1113 let aa = 0
1114 let bb = 0
1115 let cc = 0
1116 let dd = 0
1117 let ee = 0
1118 let ff = 0
1119 let gg = 0
1120 let hh = 0
1121 let ii = 0
1122 let jj = 0
1123 let kk = 0
1124 let ll = 0
1125 let mm = 0
1126 let nn = 0
1127 let oo = 0
1128 let pp = 0
1129 let rr = 0
1130 let tt = 0
1131 let uu = 0
1132 let vv = 0
1133 let ww = 0
1134 let xx = 0
1135 let yy = 0
1136 let zz = 0
1137 let aa = 0
1138 let bb = 0
1139 let cc = 0
1140 let dd = 0
1141 let ee = 0
1142 let ff = 0
1143 let gg = 0
1144 let hh = 0
1145 let ii = 0
1146 let jj = 0
1147 let kk = 0
1148 let ll = 0
1149 let mm = 0
1150 let nn = 0
1151 let oo = 0
1152 let pp = 0
1153 let rr = 0
1154 let tt = 0
1155 let uu = 0
1156 let vv = 0
1157 let ww = 0
1158 let xx = 0
1159 let yy = 0
1160 let zz = 0
1161 let aa = 0
1162 let bb = 0
1163 let cc = 0
1164 let dd = 0
1165 let ee = 0
1166 let ff = 0
1167 let gg = 0
1168 let hh = 0
1169 let ii = 0
1170 let jj = 0
1171 let kk = 0
1172 let ll = 0
1173 let mm = 0
1174 let nn = 0
1175 let oo = 0
1176 let pp = 0
1177 let rr = 0
1178 let tt = 0
1179 let uu = 0
1180 let vv = 0
1181 let ww = 0
1182 let xx = 0
1183 let yy = 0
1184 let zz = 0
1185 let aa = 0
1186 let bb = 0
1187 let cc = 0
1188 let dd = 0
11
```



(How to download?    How to quick download?)

Note: the distance between IR remote control and IR receiving head of smart car are supposed less than 5m, during the test.

## 8.20: Bluetooth Multi-purpose Smart Car

### 20.1: Read Bluetooth Data



#### (1) Project Description

Micro:bit main board comes with a built in Bluetooth which can be used to communicate with it. And the Micro:bit can also be controlled by Bluetooth or transmit signals back to smartphone or computer via it. This Bluetooth can communicate with the Bluetooth equipped in other devices or with Bluetooth App to control other equipment. It is compatible with both Android system and IOS system. And we have designed two Bluetooth App for both systems.



---

The connection of the Bluetooth on the board with these two Apps is similar. In this lesson, we will introduce the functions of all keys and patterns on the Apps and control the smart car via Bluetooth App.

## **(2)Experimental Preparation:**

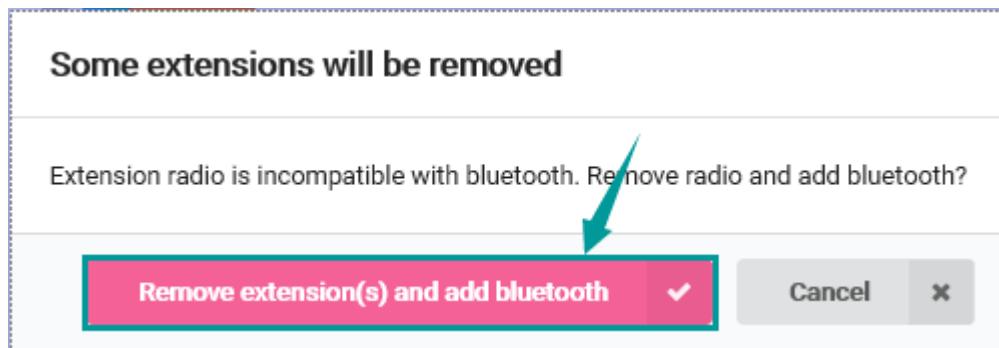
- Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- Open online Makecode editor

Import Hex profile (How to import?) , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

### **(How to add MecanumRobot extension?)**

As the Bluetooth and extension radio can't work together, therefore, their extension libraries are not compatible.

Therefore, remove extension(s) and add Bluetooth please if you see the following prompt box pop up.



### (3) Test Code:

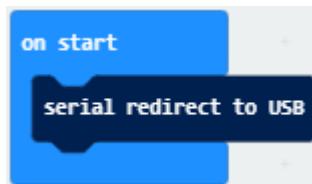
Code Path:

File Type	Path	File Name
Hex file	KS4031(4032) folder/Makemode Tutorial/Makemode Code/Project 20.1: Read Bluetooth Data.hex	Project 20.1: Read Bluetooth Data.hex

Or you could edit code step by step in the editing area.

Enter "Advanced" → "Serial" → "serial redirect to USB"

Place it into "on start"

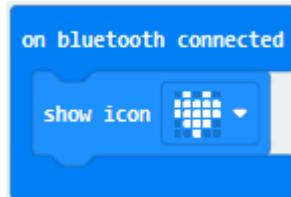


\*\*\*\*\*



Click “Bluetooth” → “on bluetooth connected”

Go to “Basic” to move “show icon” block into “on bluetooth connected” block.

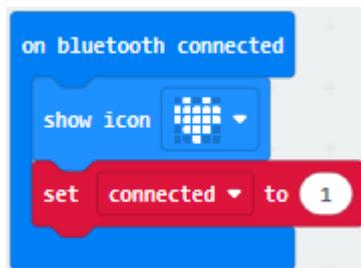


\*\*\*\*\*

Click “Variables” → “Make a Variable...” → “New variable name: ” dialog box.

Input “connected” and click “OK” to create variable “connected” .

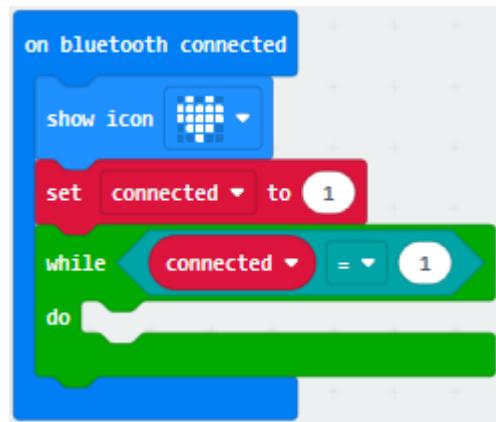
Drag “set connected to 0” under block “show icon” and change 0 into 1.



Go to “Loops” to move block “while true do...” into “on bluetooth connected” block.

Enter “Logic” to drag out “=” block.

Click “Variables” to drag “connected” into left box of “=” block and change 0 into 1.



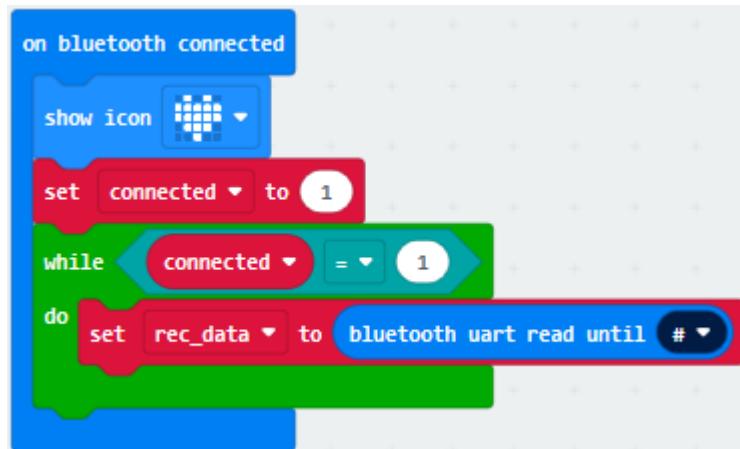
\*\*\*\*\*

Then we generate variable "rec\_data" in same way.

Then drag out "set rec\_data to 0" and place it into block "while connected=1 do..." block.

Click "Bluetooth" → "more" → "bluetooth uart read until new line( )"

Keep it into 0 box and click triangle button to select #.

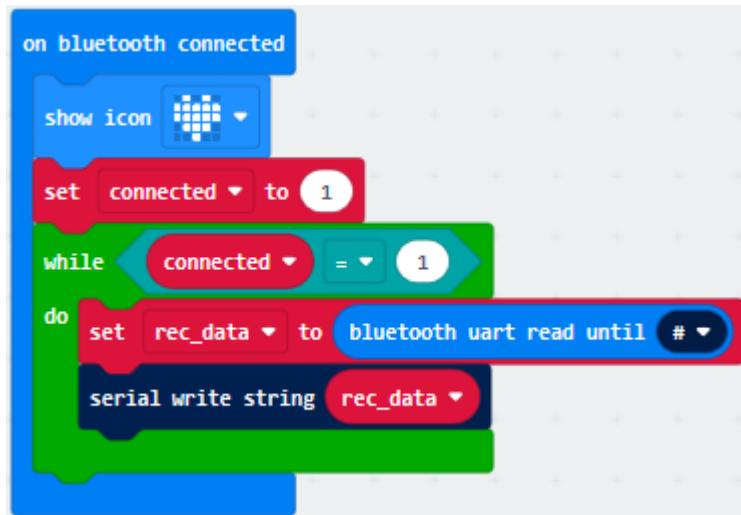


Go to "Advanced" → "Serial" → "serial write string"

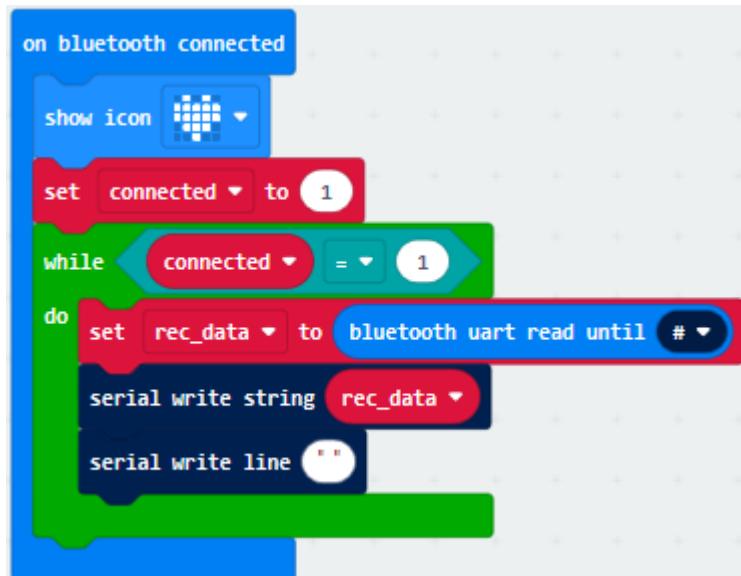
Move it below "set rec\_data...until#" block



And combine variable “rec\_data” with “serial write string” block.



Click “Advanced” → “Serial” → “serial write line” and edit code string as follows:





---

Click “Bluetooth” to drag out “on bluetooth disconnected” .

Go to “Bluetooth” → “on bluetooth disconnected”

Copy “show icon” block and keep it into block “on bluetooth disconnected”

Click triangle button to select “

Complete Program



```
on start
    serial redirect to USB

on bluetooth connected
    show icon [grid icon v]
    set connected to 1
    while connected = 1
        do
            set rec_data to bluetooth uart read until #
            serial write string rec_data
            serial write line [ ]
    end

on bluetooth disconnected
    show icon [grid icon v]
```



"on start" : command block runs once to start program.

Serial redirect to USB

Connect Bluetooth

LED dot matrix shows "❤️" pattern

Set variable connected to 1

When connected=1, the code under do block will be executed.

Set rec\_data to bluetooth uart read until #

Serial port prints rec\_data

Print a blank space

Disconnect Bluetooth

LED dot matrix displays "████" pattern.

Click "JavaScript" to view the corresponding JavaScript code:



Microsoft | micro:bit

Blocks JavaScript ▾

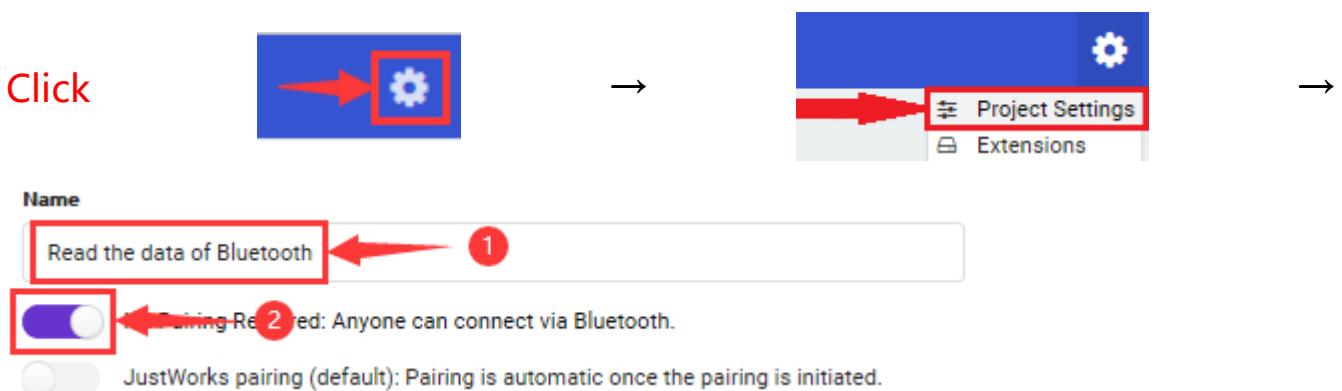
Search... Q

Basic      Input      Music      Led      Bluetooth      Loops      Logic      Variables      Math      MecanumRobot      IrRemote

```
1 bluetooth.onBluetoothConnected(function () {  
2     basic.showIcon(IconNames.Heart)  
3     connected = 1  
4     while (connected == 1) {  
5         rec_data = bluetooth.uartReadUntil(serial.delimiters(Delimiters.Hash))  
6         serial.writeString(rec_data)  
7         serial.writeLine("")  
8     }  
9 })  
10 bluetooth.onBluetoothDisconnected(function () {  
11     basic.showIcon(IconNames.Sad)  
12 }  
13 let rec_data = ""  
14 let connected = 0  
15 serial.redirectToUSB()  
16
```

#### (4) Test Results:

If you drag blocks step by step, you need to set as follows after finishing test code.



However, you could skip this step if you directly import test code.

After setting, download code to micro:bit board, don't plug off USB cable([How to download?](#) [How to quick download?](#))

Next to download App.



## For IOS System:

a.open App Store;



b.search mecanum\_robot and click "cloud with arrow" to download the Bluetooth App of mecanum\_robot;

c. After downloading the APP, click "OPEN" or click the application mecanum\_robot on the phone/iPad desktop to open the APP. A dialog box appears on the APP interface, and click "OK" in the dialog box.

d. First turn on the Bluetooth of the mobile phone/iPad, and then click the connect button (control) in the upper left corner of the APP interface to perform a Bluetooth search. In the search results, click "BCC micro:bit". After a few seconds, the Bluetooth is connected.

## For Android System:

a. Use the scanning function in the browser to scan and identify the QR code or enter the [http://8.210.52.206/mecanum\\_robot.apk](http://8.210.52.206/mecanum_robot.apk) to download. After the identification is successful, click "go to website" to enter the download mecanum\_robot.apk page , Click "Download" to download the mecanum\_robot application.



B.Click “Allow allow” to enter Installation Diagram; click “install” to install

Allow "Downloads" to install a...

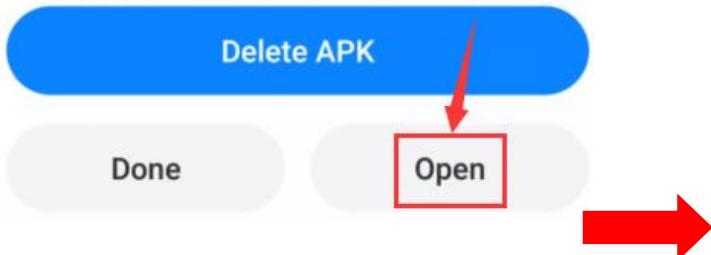
Your device and personal data are vulnerable to attacks by the apps installed from unknown sources. By allowing to install apps from this source, you agree to accept responsibility for any damage or data loss that might occur while you're using these apps.

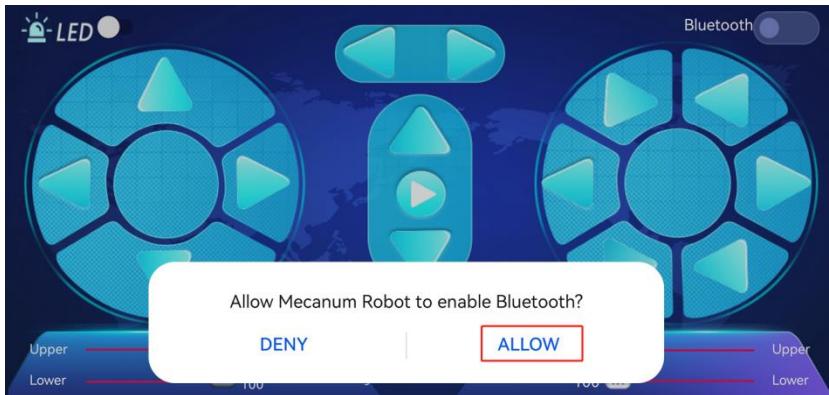


the App                      Restrict

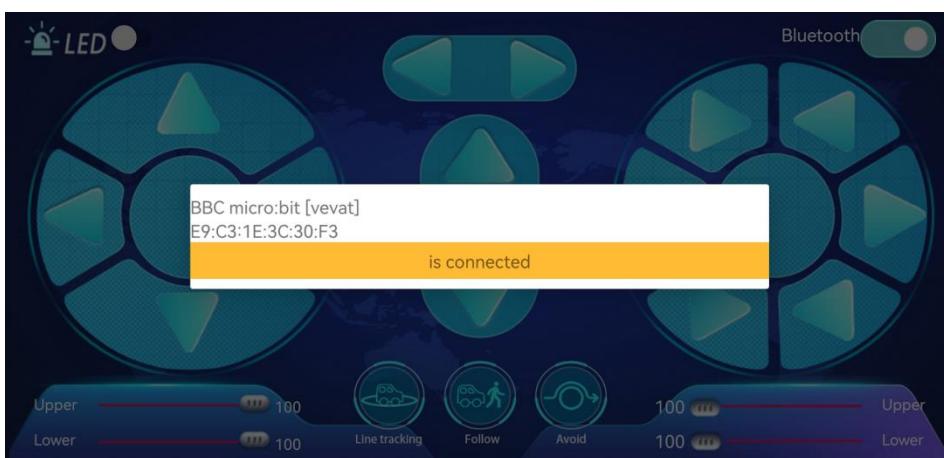
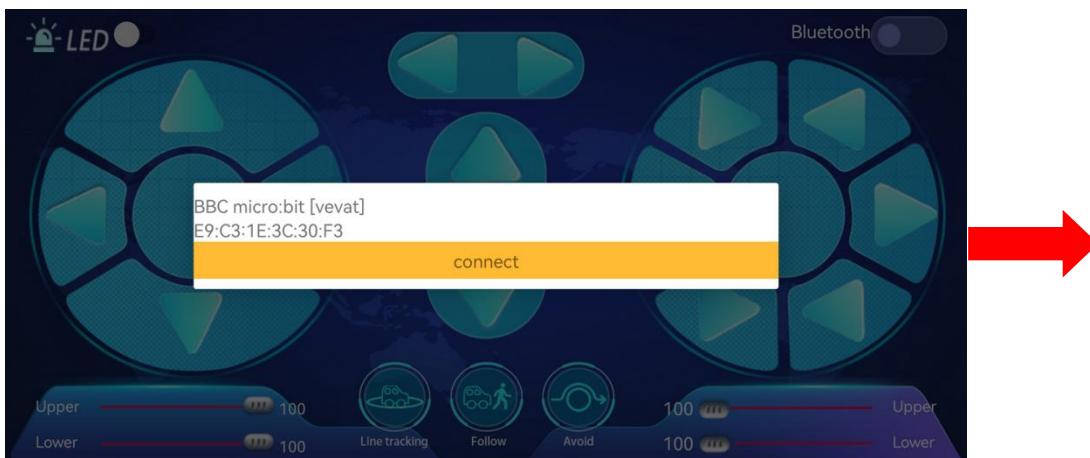
C.Click "Open" or click the application `mecanum_robot` on the mobile phone desktop to open the APP, and a dialog box appears. In the dialog box, click "Allow" to turn on the Bluetooth of the mobile phone. You can also turn on the phone's Bluetooth before opening the APP.

## App details and required permis... ►





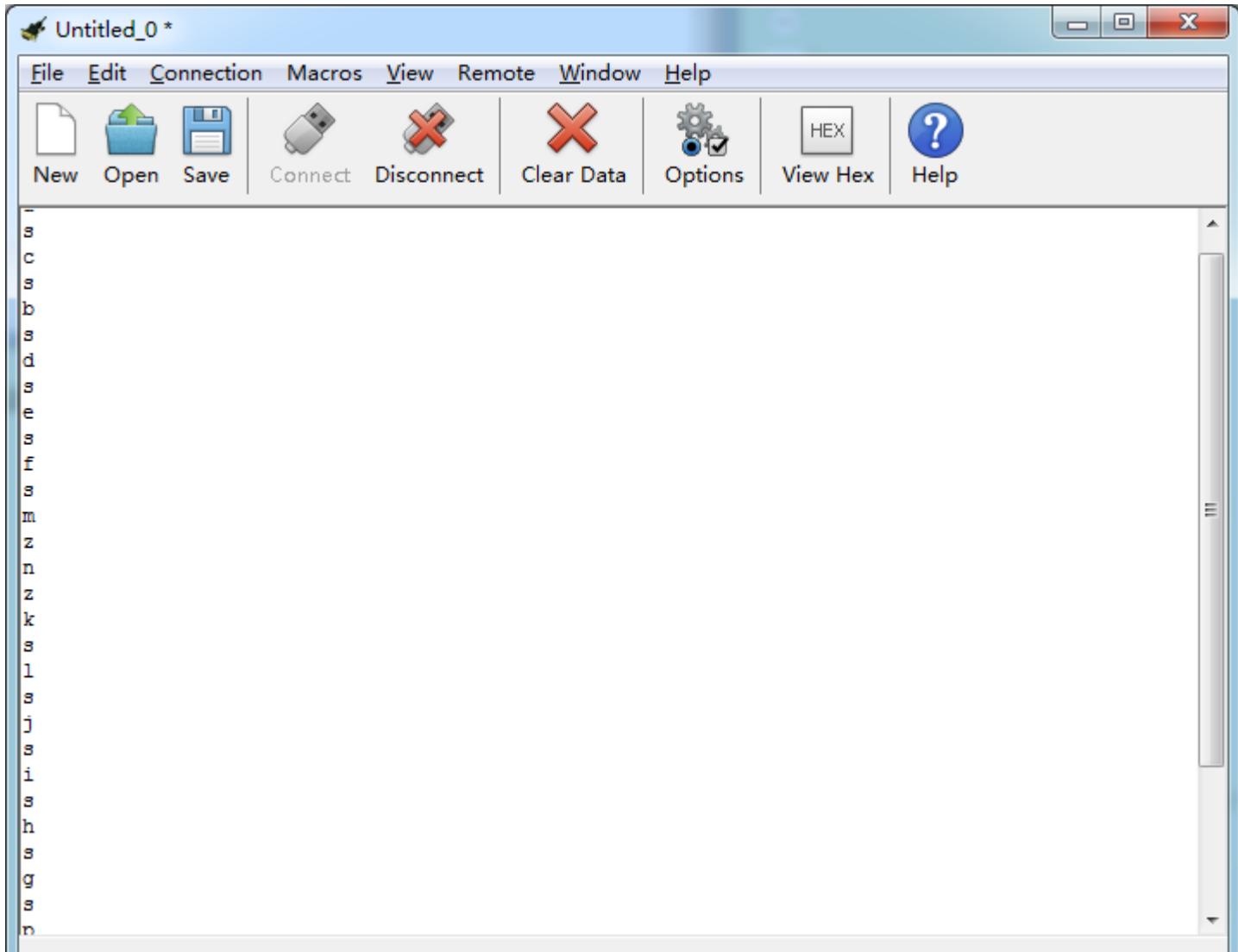
Click  on the upper right corner to search for Bluetooth and click "connect" ; a few seconds later, the Bluetooth is paired.



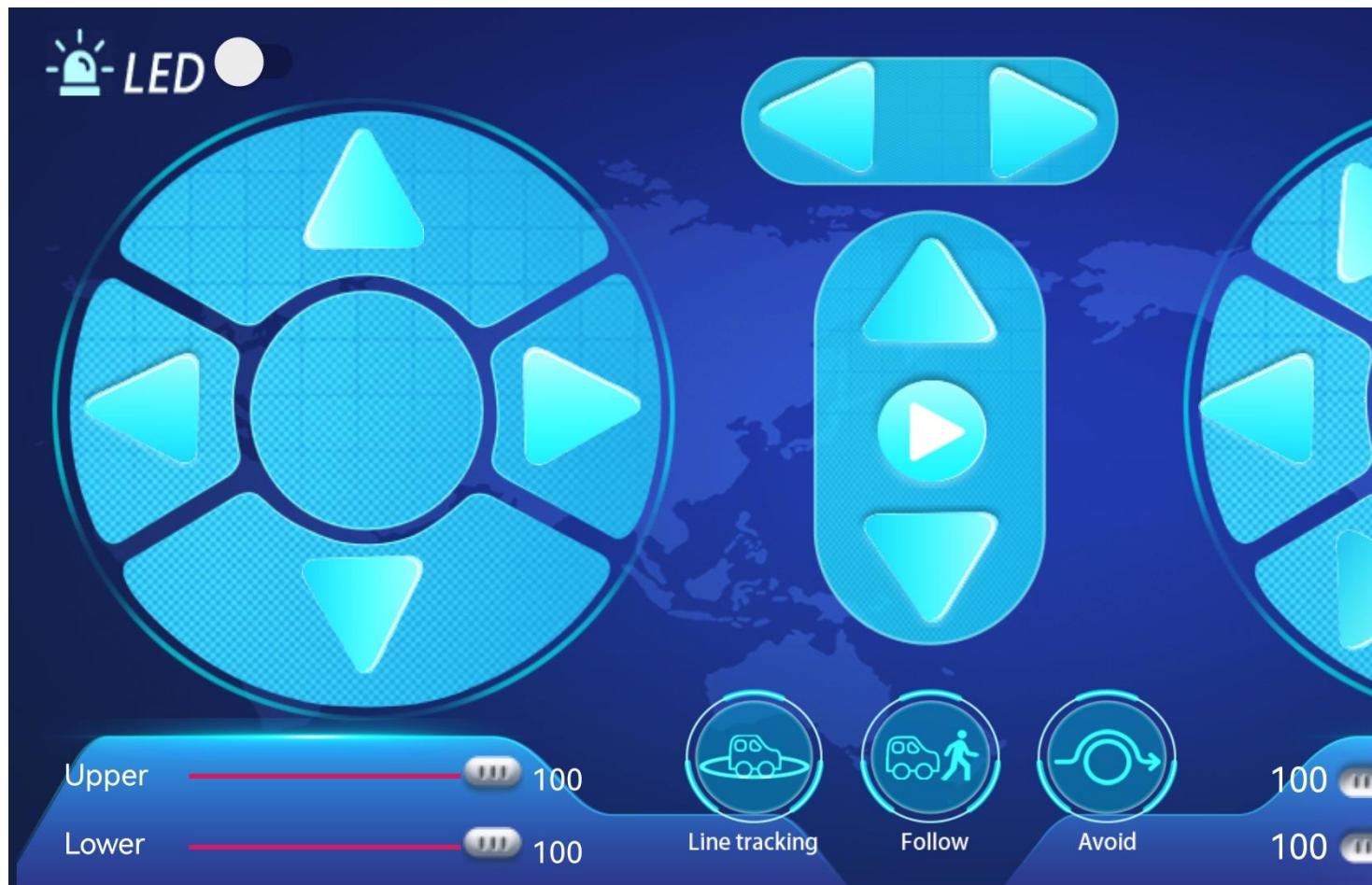
Open CoolTerm, click Options to select SerialPort. Set COM port and 115200 baud rate. Click "OK" and "Connect" .



Point at micro:bit board and press the icons on APP, the corresponding characters are shown on CoolTerm monitor.



Through the test, we get the function of every icon, as shown below:





## 20.2: Multi-purpose Smart Car



### (1) Project Description

In this lesson, we will control the smart car to perform multipurpose function.

### (2) Experimental Preparation:

Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car

Place batteries into battery holder



Dial power switch to ON end

Connect micro:bit to computer by USB cable

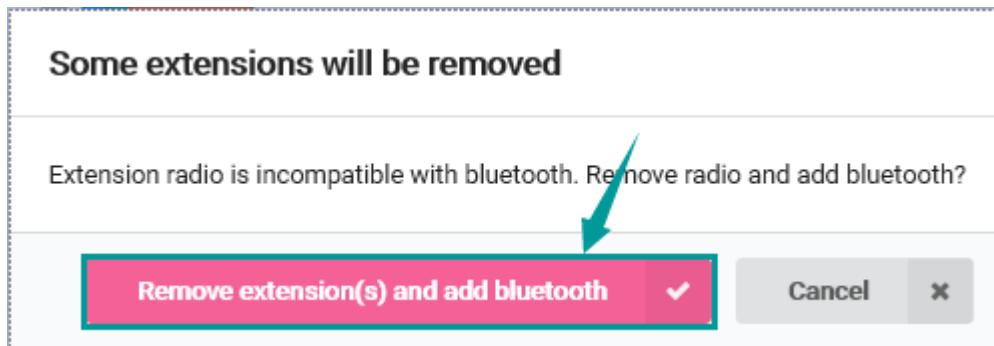
Open online Makecode editor

Import Hex profile (How to import? , or click “New Project” and drag blocks step by step(add MecanumRobot extension library first)

### **(How to add MecanumRobot extension?)**

As the Bluetooth and extension radio can't work together, therefore, their extension libraries are not compatible.

Therefore, remove extension(s) and add Bluetooth please if you see the following prompt box pop up.



### **(3) Test Code:**

#### **Code path:**

File Type	Path	File Name



Hex file	KS4031(4032) folder/Makecode Tutorial/Makecode Code/Project 20.2: Multi-purpose Smart Car.hex	Project 20.2: Multi-purpose Smart Car.hex
----------	---	--

## Complete Code:

The image shows a Scratch script for a multi-purpose smart car. It starts with an "on bluetooth connected" hat block. Inside, it sets a variable `connect_flag` to 1. It then enters a `do` loop that reads a value from the Bluetooth serial port until a specific character is received. Inside this loop, it checks for different characters (a through l) to call specific movement functions: `car_forward`, `car_left`, `car_back`, `car_right`, `car_left_move`, `car_right_move`, `car_move_RF`, `car_move_RB`, and `car_move_LB`. Each function is defined with a blue hat block and contains four motor blocks. The motors are grouped by side (Upper/Lower Left and Upper/Lower Right) and run either forward or backward at a speed percentage. The functions are:

- `function car_forward`: Motor Upper\_left run Forward speed: speed\_LF %, Motor Lower\_left run Forward speed: speed\_LB %, Motor Upper\_right run Forward speed: speed\_RF %, Motor Lower\_right run Forward speed: speed\_RB %
- `function car_left`: Motor Upper\_left run Back speed: speed\_LF %, Motor Lower\_left run Back speed: speed\_LB %, Motor Upper\_right run Back speed: speed\_RF %, Motor Lower\_right run Back speed: speed\_RB %
- `function car_back`: Motor Upper\_left run Back speed: speed\_LF %, Motor Lower\_left run Back speed: speed\_LB %, Motor Upper\_right run Back speed: speed\_RF %, Motor Lower\_right run Back speed: speed\_RB %
- `function car_right`: Motor Upper\_left run Forward speed: speed\_LF %, Motor Lower\_left run Back speed: speed\_LB %, Motor Upper\_right run Back speed: speed\_RF %, Motor Lower\_right run Forward speed: speed\_RB %
- `function car_left_move`: Motor Upper\_left run Back speed: speed\_LF %, Motor Lower\_left run Back speed: speed\_LB %, Motor Upper\_right run Forward speed: speed\_RF %, Motor Lower\_right run Back speed: speed\_RB %
- `function car_right_move`: Motor Upper\_left run Forward speed: speed\_LF %, Motor Lower\_left run Back speed: speed\_LB %, Motor Upper\_right run Back speed: speed\_RF %, Motor Lower\_right run Forward speed: speed\_RB %
- `function car_move_RF`: Motor Upper\_left run Forward speed: 0 %, Motor Lower\_left run Forward speed: speed\_LB %, Motor Upper\_right run Forward speed: speed\_RF %, Motor Lower\_right run Forward speed: 0 %
- `function car_move_RB`: Motor Upper\_left run Forward speed: speed\_LF %, Motor Lower\_left run Forward speed: 0 %, Motor Upper\_right run Back speed: speed\_RF %, Motor Lower\_right run Forward speed: speed\_RB %
- `function car_move_LB`: Motor Upper\_left run Forward speed: speed\_LF %, Motor Lower\_left run Forward speed: 0 %, Motor Upper\_right run Forward speed: 0 %, Motor Lower\_right run Forward speed: speed\_RB %



```
else if [ble_val v] is [-1] then
    call car_move_left
else if [ble_val v] is [-5] then
    car stop
else if [ble_val v] is [-1] then
    Left v Colorful LED turn ON v
    Right v Colorful LED turn ON v
else if [ble_val v] is [-u] then
    Left v Colorful LED turn OFF v
    Right v Colorful LED turn OFF v
else if [ble_val v] is [-e] then
    call drift_left
else if [ble_val v] is [-f] then
    call drift_right
else if [ble_val v] is [-n] then
    if [color_num v] < [9] then
        set [color_num v] to [color_num v] + [1]
    end
    call showcolor
else if [ble_val v] is [-n] then
    if [color_num v] > [6] then
        set [color_num v] to [color_num v] - [1]
    end
    call showcolor
else if [ble_val v] is [-o] then
```

```
Function [showcolor v]
if [color_num v] = [8] then
    strip show color red v
else if [color_num v] = [1] then
    strip show color orange v
else if [color_num v] = [2] then
    strip show color yellow v
else if [color_num v] = [3] then
    strip show color green v
else if [color_num v] = [4] then
    strip show color blue v
else if [color_num v] = [5] then
    strip show color indigo v
else if [color_num v] = [6] then
    strip show color violet v
else if [color_num v] = [7] then
    strip show color purple v
else if [color_num v] = [8] then
    strip show color white v
end
strip show
end
```

Forever

```
if [ble_val v] = [p] then
    call tracking
```

```
Function [tracking v]
if [Left v LineTracking v] = [1] and
    [Left v LineTracking v] = [8] then
    call car_left
else if [Left v LineTracking v] = [8] then
    call car_right
else if [Left v LineTracking v] = [1] then
    call car_forward
else
    car stop
end
```

Function [follow v]

```
set servo to angle [90]
pause [500 ms]
if Ultrasonic < [18] then
    call car_back
else if Ultrasonic < [26] then
    car stop
else if Ultrasonic < [46] then
    call car_forward
else
    car stop
end
```



The Scratch script consists of several main sections:

- On start:** Sets initial speeds (speed\_LB, speed\_RF) to 75, initializes color\_num to 0, and configures a NeoPixel strip at pin PB with 4 RGB LEDs.
- On bluetooth disconnected:** Shows a specific icon.
- Loop (controlled by `ble_val`):** This loop handles four cases based on the character received:
  - q:** Calls the `follow` procedure.
  - r:** Calls the `avoid` procedure.
  - s:** Stops the car and sets the servo angle to 90.
  - x:** Sets speed\_LB to 75.
  - y:** Sets speed\_RF to 75.
- Serial communication:** The script reads from Bluetooth until a specific character is received (e.g., 'q', 'r', 's', 'x', 'y') and then prints the received value to the serial port.

Click "JavaScript" to view the corresponding JavaScript code: :



Microsoft | micro:bit

Blocks JavaScript ▾

Search... Q

Basic Input Music Led Bluetooth Loops Logic Variables Math MecanumRobot IrRemote Neopixel Advanced

```
1 function car_back () {
2     mecanumRobot.Motor(LR.Upper_left, MD.Back, speed_LF)
3     mecanumRobot.Motor(LR.Lower_left, MD.Back, speed_LB)
4     mecanumRobot.Motor(LR.Upper_right, MD.Back, speed_RF)
5     mecanumRobot.Motor(LR.Lower_right, MD.Back, speed_RB)
6 }
7 function car_move_RF () {
8     mecanumRobot.Motor(LR.Upper_left, MD.Forward, speed_LF)
9     mecanumRobot.Motor(LR.Lower_left, MD.Forward, 0)
10    mecanumRobot.Motor(LR.Upper_right, MD.Forward, 0)
11    mecanumRobot.Motor(LR.Lower_right, MD.Forward, speed_RB)
12 }
13 function drift_left () {
14     mecanumRobot.Motor(LR.Upper_left, MD.Back, 0)
15     mecanumRobot.Motor(LR.Lower_left, MD.Back, speed_LB)
16     mecanumRobot.Motor(LR.Upper_right, MD.Back, 0)
17     mecanumRobot.Motor(LR.Lower_right, MD.Forward, speed_RB)
18 }
19 function car_left () {
20     mecanumRobot.Motor(LR.Upper_left, MD.Back, speed_LF)
21     mecanumRobot.Motor(LR.Lower_left, MD.Back, speed_LB)
22     mecanumRobot.Motor(LR.Upper_right, MD.Forward, speed_RF)
23     mecanumRobot.Motor(LR.Lower_right, MD.Forward, speed_RB)
24 }
25 bluetooth.onBluetoothConnected(function () {
26     basic.showIcon(IconNames.Heart)
27     connect_flag = 1
28 }
```

## (4) Test Results:

This experiment combines the previous projects to make the car to perform actions by Bluetooth.

Enter Makecode online editor → Projecting Settings →

Name

, enable “No Pairing....” (you could skip  
 No Pairing Required: Anyone can connect via Bluetooth)

this step if you import test code directly)



---

Download code to micro:bit board, dial POWER to ON end, and connect the Bluetooth, then you can control the car via the Bluetooth App of mecanum\_robot.

([How to download?](#)    [How to quick download?](#))

## 9. Resources:

Download PDF files: <https://fs.keyestudio.com/KS4031-4032>

BBC microbit MicroPython:

<https://microbit-micropython.readthedocs.io/en/latest/tutorials/introduction.html>

MicroPython:

<https://docs.openmv.io/reference/index.html>

ustruct library:

<https://docs.openmv.io/library/ustruct.html>

math library:

<https://docs.openmv.io/library/math.html>

utime(sleep\_us,tick\_us) library:

<https://docs.openmv.io/library/utime.html#>