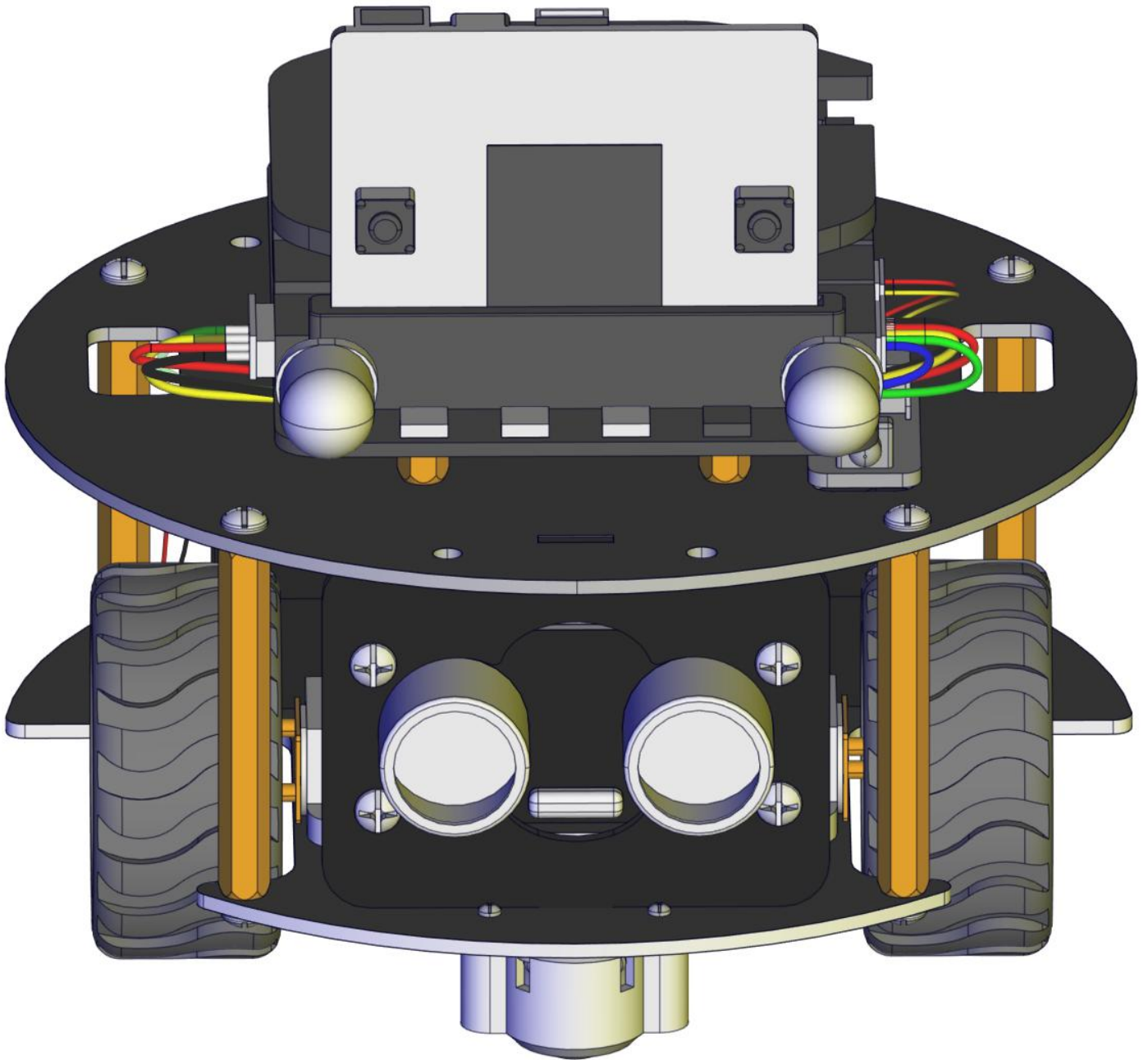




# Keyestudio Micro: bit Mini Smart Turtle

## Car





# Content

- 1. Description..... 4
- 2. Specification..... 5
- 3. Kit..... 6
- 4. Micro:bit:..... 11
  - 4.1.micro:bit..... 11
  - 4.2. Install the Driver of Micro:bit..... 14
- 5. Keyestudio Micro:bit Mini Smart Turtle Car..... 15
  - 5.1. Install Micro:bit Mini Turtle Smart Car..... 16
- 6. Code and Programming..... 45
  - 6.1 Makecode..... 53
  - 6.2 Quick Download..... 55
  - 6.3 How to Import Makecode Extension Library..... 59
  - 6.4 Resources and Code..... 65
  - 6.5 Import Code..... 66
  - 6.6 Install CoolTerm..... 70
- 7. Projects..... 74
  - 7.1: Heart beat..... 74
  - 7.2: Light Up A Single LED..... 77



7.3: 5 x 5 LED Dot Matrix.....	84
7.4: Programmable Buttons.....	95
7.5: Temperature Measurement.....	105
7.6: Micro:bit' s Compass.....	118
7.7: Accelerometer.....	133
7.8: Detect Light Intensity by Micro:bit.....	147
7.9: Bluetooth Wireless Communication.....	154
7.10: Passive Sensor.....	170
7.11: RGB Experiments.....	183
7.12: WS2812 RGB.....	212
7.13: Motor Driving.....	235
7.14: Line Tracking Smart Car.....	269
7.14.1: Detect Line Tracking Sensor.....	269
7.14.2: Line Tracking Smart Car.....	310
7.15: Ultrasonic Follow Smart Car.....	324
7.15.1: Ultrasonic Ranging.....	324
7.15.2: Ultrasonic Avoidance Car.....	337
7.15.3: Ultrasonic Follow Smart Car.....	346
7.16: IR Remote Control Smart Car.....	357
7.16.1: Decode IR Remote Control.....	357
7.16.2: IR Remote Control.....	368
7.17: Bluetooth Multi-purpose Smart Car.....	388



7.17.1: Read Bluetooth Data.....	388
7.17.2: Multi-purpose Smart Car.....	403
8. Resource.....	416

## 1. Description

Micro:bit is significantly applied to STEM education for teenagers, as a small microcontroller, which features small in size, easy to carry, and powerful function. At present, innovative technology products, like robots, wearable devices and interactive electronic games can be produced by programming and code.

In this kit, we will guide you how to control and generate a Micro:bit turtle smart car through programming in Makecode.

MakeCode is a framework for creating interactive and engaging programming experiences for those new to the world of programming. The platform provides the foundation for a tailored coding experience to create and run user programs on actual hardware or in a simulated target. What' s more, we also provide test code and projects so as to make smart



car display different effects.

Launched by KEYES group, Keyestudio micro:bit smart car integrates obstacle avoidance, line tracking and IR and Bluetooth control. It is made up of DC geared motors, wheels , sensors and acrylic boards. In addition, it cooperates a passive buzzer with music play function, 4 pcs WS2812RGB LEDs and 2 pcs RGB lights.

We believe that your imagination and creativity can be stimulated through DIY smart car and acquire how to code through Makecode, a new method to program.

**Note: we adopt V1.5 micro:bit in the whole tutorial, but our tutorial is also compatible with the latest version micro:bit.**

**When doing experiment with latest micro:bit, you need to transfer code into Makecode online editor first, save code again then download it to micro:bit.**

## **2. Specification**

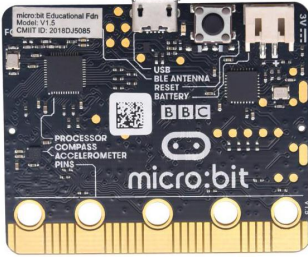
- Connector port input: DC 6V---9V
- Operating voltage of drive board system: 5V
- Standard operating power consumption: about 2.2W



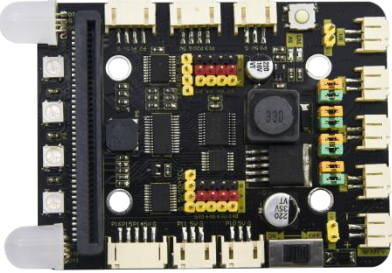
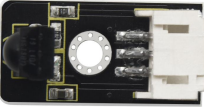




- Maximum power: Maximum output power is 12W
- Motor speed: 100RPM/1min
- Working temperature range: 0-50°C
- Size: 120\*120\*120mm
- Environmental protection attributes: ROHS

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

### 3. Kit

Components			
#	Model	QTY	Picture
0	Micro:bit main board <b>is Not Included</b> in KS4014 Kit		
0	Micro:bit main board is <b>Included</b> in KS4024 Kit	1	











1	Keyestudio Micro:bit Driver Shield	1	
2	Keyestudio Quick Connectors IR Receiver	1	
3	Keyestudio Quick Connectors Line Tracking Sensor	1	
4	Keyestudio Quick Connectors Ultrasonic Sensor	1	
5	Micro USB Cable AM/MK5P(micro)	1	
6	Keyestudio JMFP-4 17 Key	1	











1	Keyestudio Baseplate for Smart Small Turtle Robot V2.0	1	
2	Keyestudio Round Board	1	
3	Keyestudio Acrylic Top Board	1	
4	Micro:bit Fixed Mount	1	
5	Keyestudio Quick Connectors 12FN20 Motor A	1	
6	Keyestudio Quick Connectors 12FN20 Motor B	1	
7	N20 Motor White Mount	2	
8	Car Wheels	2	








9	Universal Wheel	2	
10	Dual head JST-PH2.0MM-5P Dupont Line	1	
11	Dual head JST-PH2.0MM-4P Dupont Line	1	
12	Dual head JST-PH2.0MM-3P Dupont Line	1	
13	Dual head JST-PH2.0MM-2P Dupont Line	2	
14	18650 2-Slot Battery Holder with 15cm Lead	1	
<b>Nuts/Screws</b>			
1	M2*12MM Round Head Screws	6	
2	M2 Nickel Plated	6	



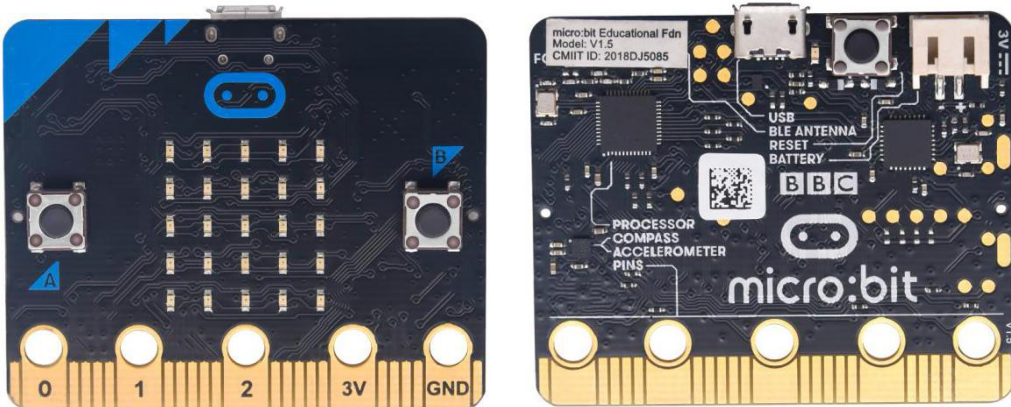
	Nuts		
3	M3*6MM Round Head Screws	20	
4	M3*8MM Round Head Screws	8	
5	M3*10MM Flat Head Screws	4	
6	M3 Nickel Plated Nuts	12	
7	M3*15+6MM Hex Copper Bush	4	
8	Dual-pass M3*12MM Hex Copper Bush	4	
9	Dual-pass M3*40MM Hex Copper Bush	4	
<b>Tools</b>			
1	3*40MM Screwdriver	1	



2	Map	1	
3	Black Ties 3*100MM	5	
4	18650 Battery (not included)	2	

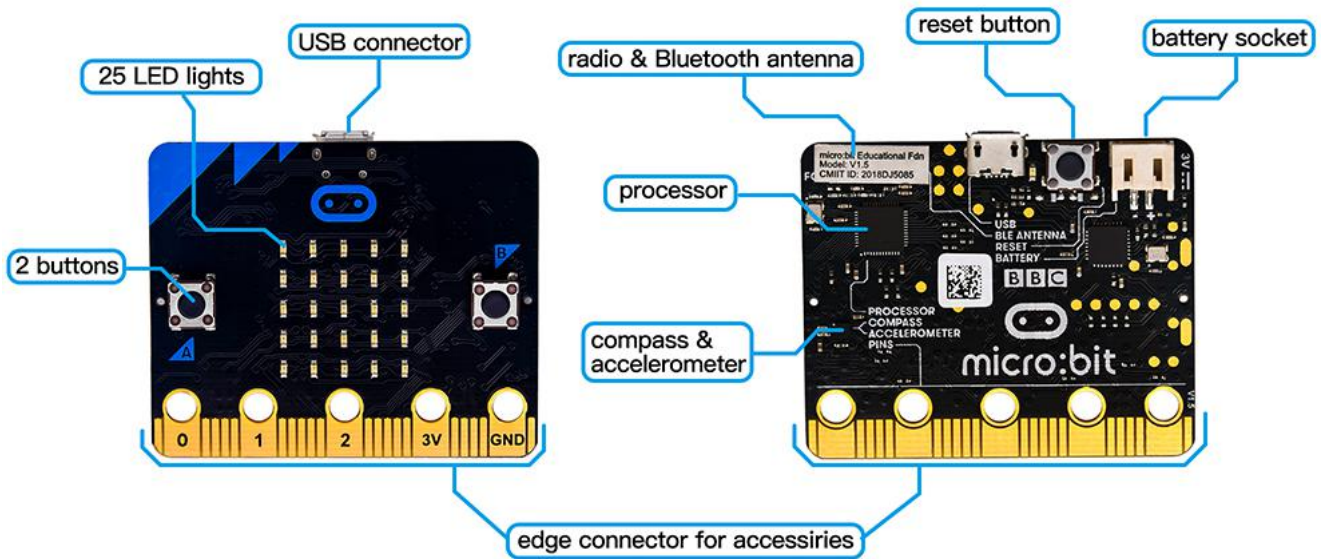
## 4. Micro:bit:

### 4.1. Micro:bit



For more details, enter the website please: <https://microbit.org/guide/>

**Function:**



## Features:

- NRF51822 processor (16 MHz 32bit, ARM Cortex-M0, Bluetooth 4.0 low consumption/2.4GHz RF wireless, 16kB RAM and 256kB Flash)
- KL26Z micro controller (48 MHz ARM Cortex-M0+ core, 128 KB Flash)
- 25 pcs programmable LEDs
- 2 programmable buttons
- Physical pins
- Light and temperature sensors
- Accelerometer (MMA8652 and I2C get the data from accelerator sensor)
- Geomagnetic sensor/compass (MAG3110, I2C obtain three-axis geomagnetic data)
- Wireless communication, by radio and Bluetooth.

## Micro USB port



More details, please enter website: <https://microbit.org/guide/features/>

## Hardware:

The details of micro:bit board: <https://tech.microbit.org/hardware/>

## Micro:bit Pins:

Before getting started with the following projects, firstly need to figure out each pin of micro:bit main board.

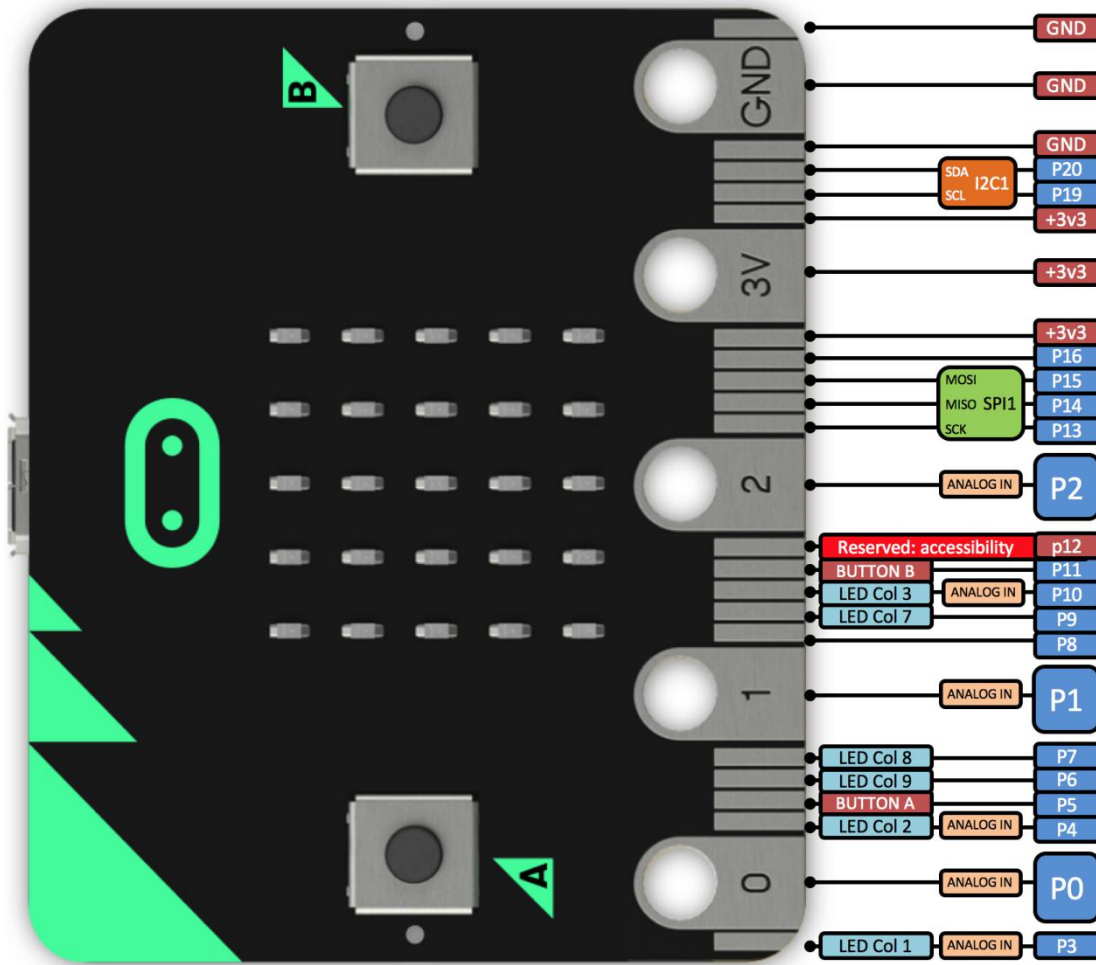
The BBC micro:bit has 25 external connections on the edge connector of the board, which we refer to as "pins" .

The edge connector is the gray area on the right side of the figure below.

There are five large pins, that are also connected to holes in the board labeled: 0, 1, 2, 3V, and GND.

And along the same edge, there are 20 small pins that you can use when plugging the BBC micro:bit into an edge connector.

Please refer to the following diagram shown below.




More details you could refer to the official website:

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

## 4.2. Install the Driver of Micro:bit

You have to install the driver of micro:bit if it's your first time to use micro:bit.

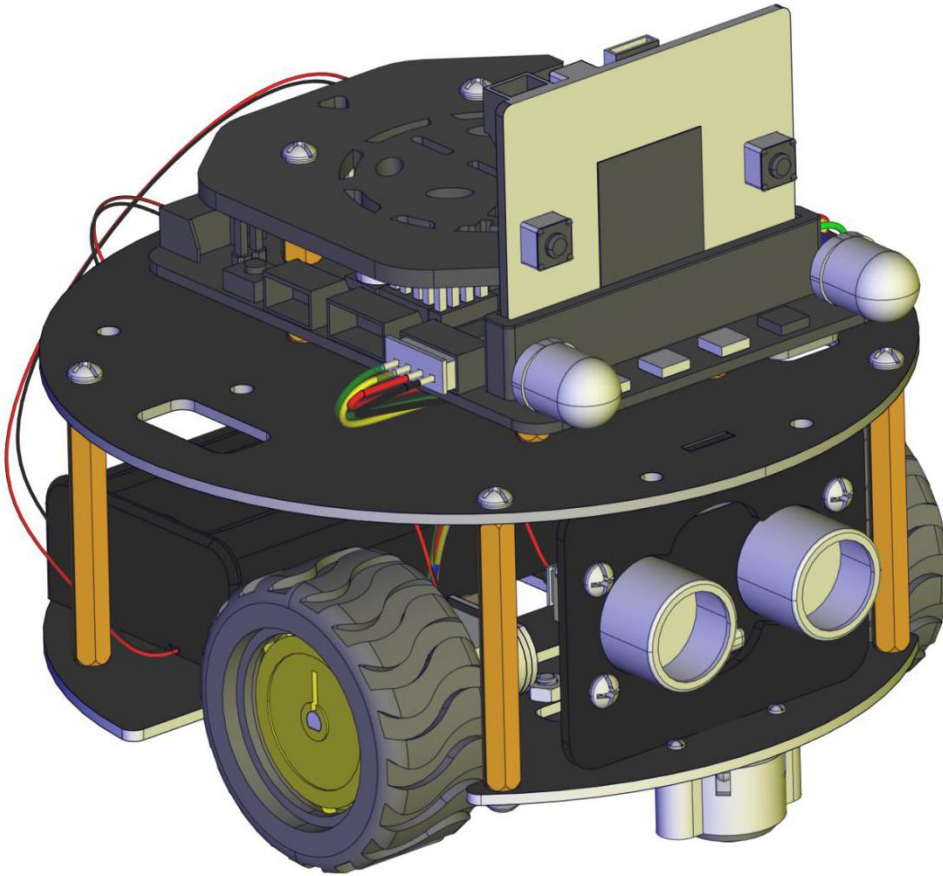
Download link: <https://fs.keyestudio.com/KS4014Driver>

You could download driver file (  mbed\_usb\_2020\_x64\_1212.exe ) in the

- 2. Install Microbit Driver folder.



## 5. Keyestudio Micro:bit Mini Smart Turtle Car



Connection of Micro:bit and Turtle Smart Car

Pins of Micro:bit	Components of Keyestudio micro:bit Robot Car
P0	Passive Buzzer
P1	Trig (T) of ultrasonic sensor



P2	Echo (E) of ultrasonic sensor
P8	4 pcs WS2812RGB
P11	IR Receiver
P14	Left TCRT5000 IR tubes of line tracking sensor
P15	Middle TCRT5000 IR tubes of line tracking sensor
P16	Right TCRT5000 IR tubes of line tracking sensor

## Supply Power

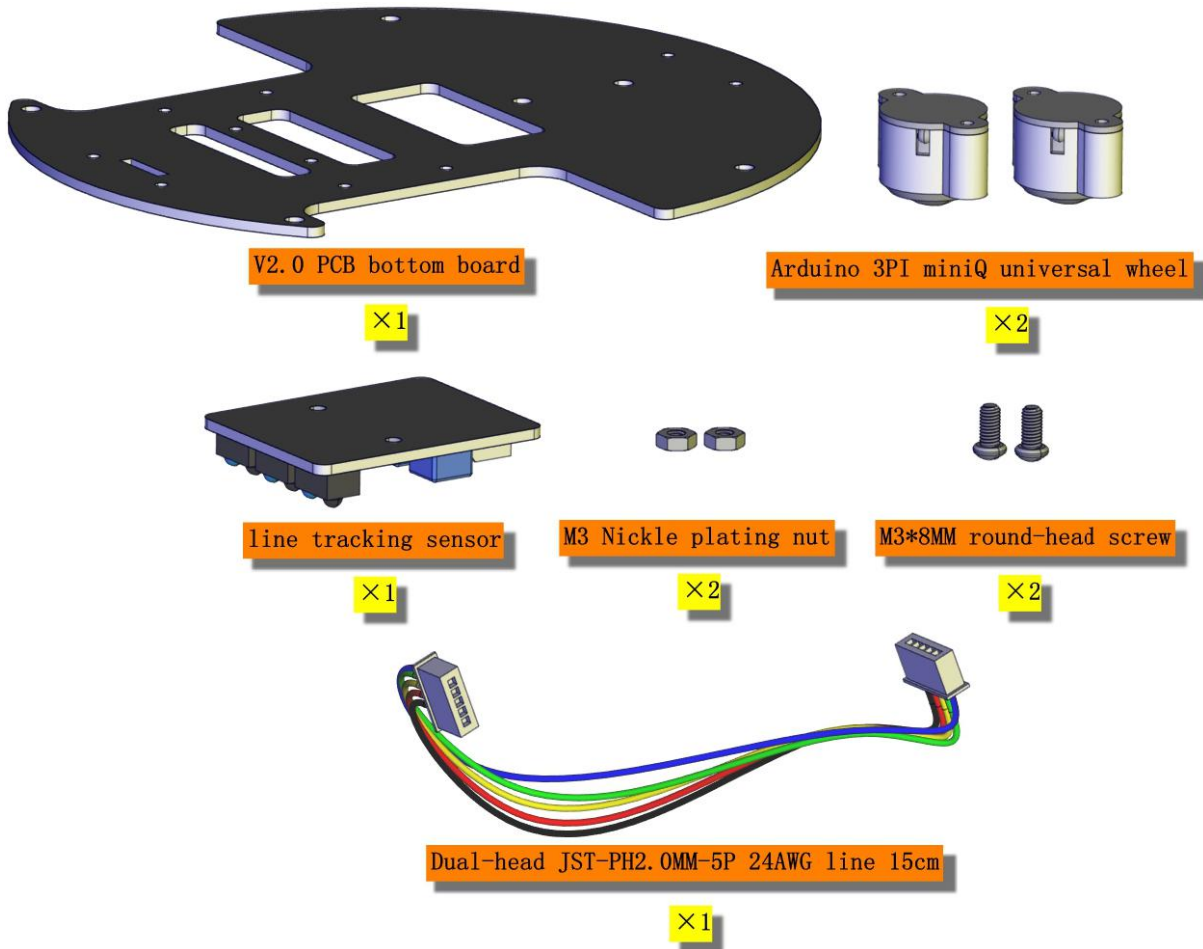
This smart car is powered by 2pcs 18650 batteries. And battery holder is chargeable.

Note: battery is not included.

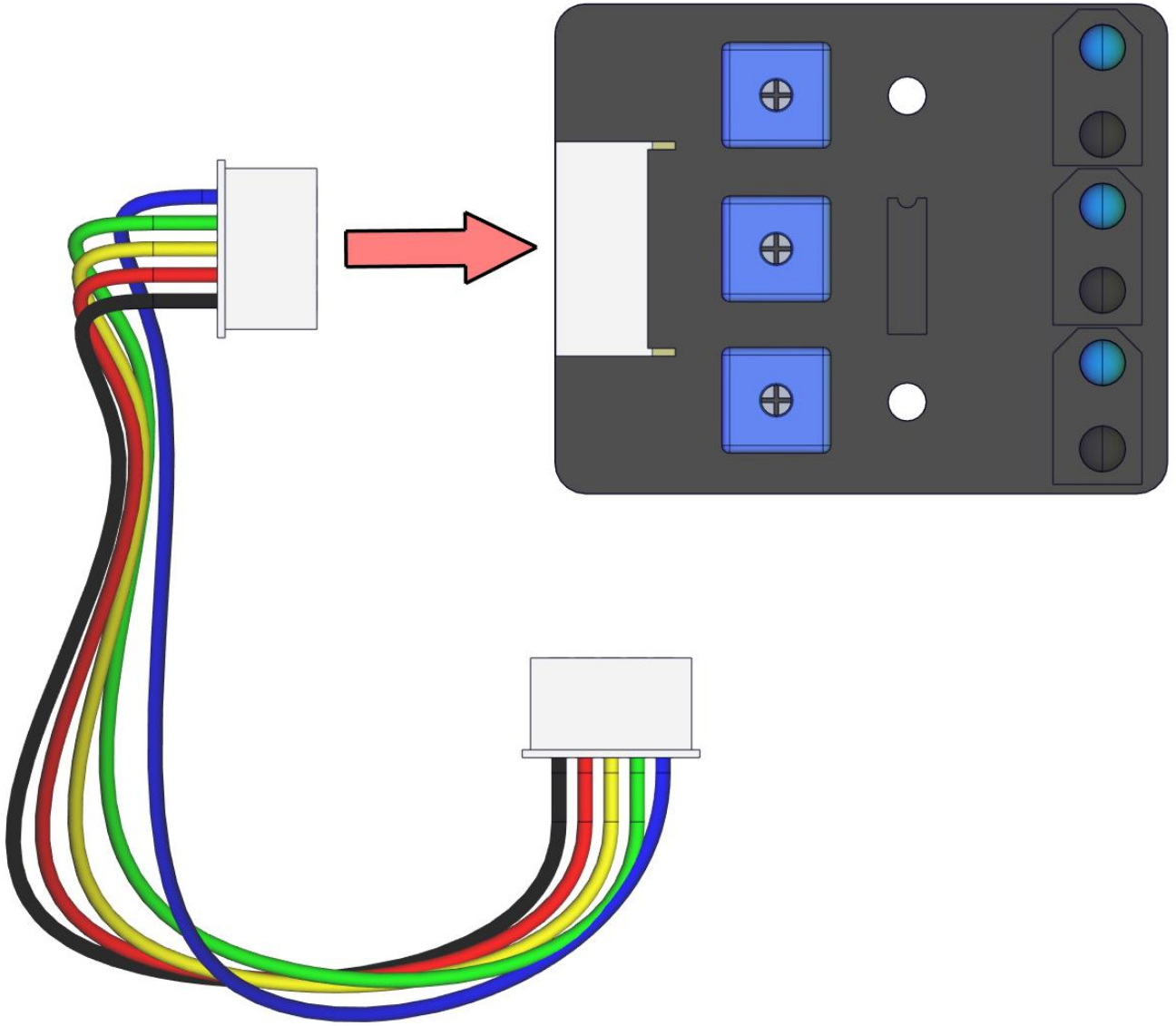
## 5.1. Install Micro:bit Mini Turtle Smart Car

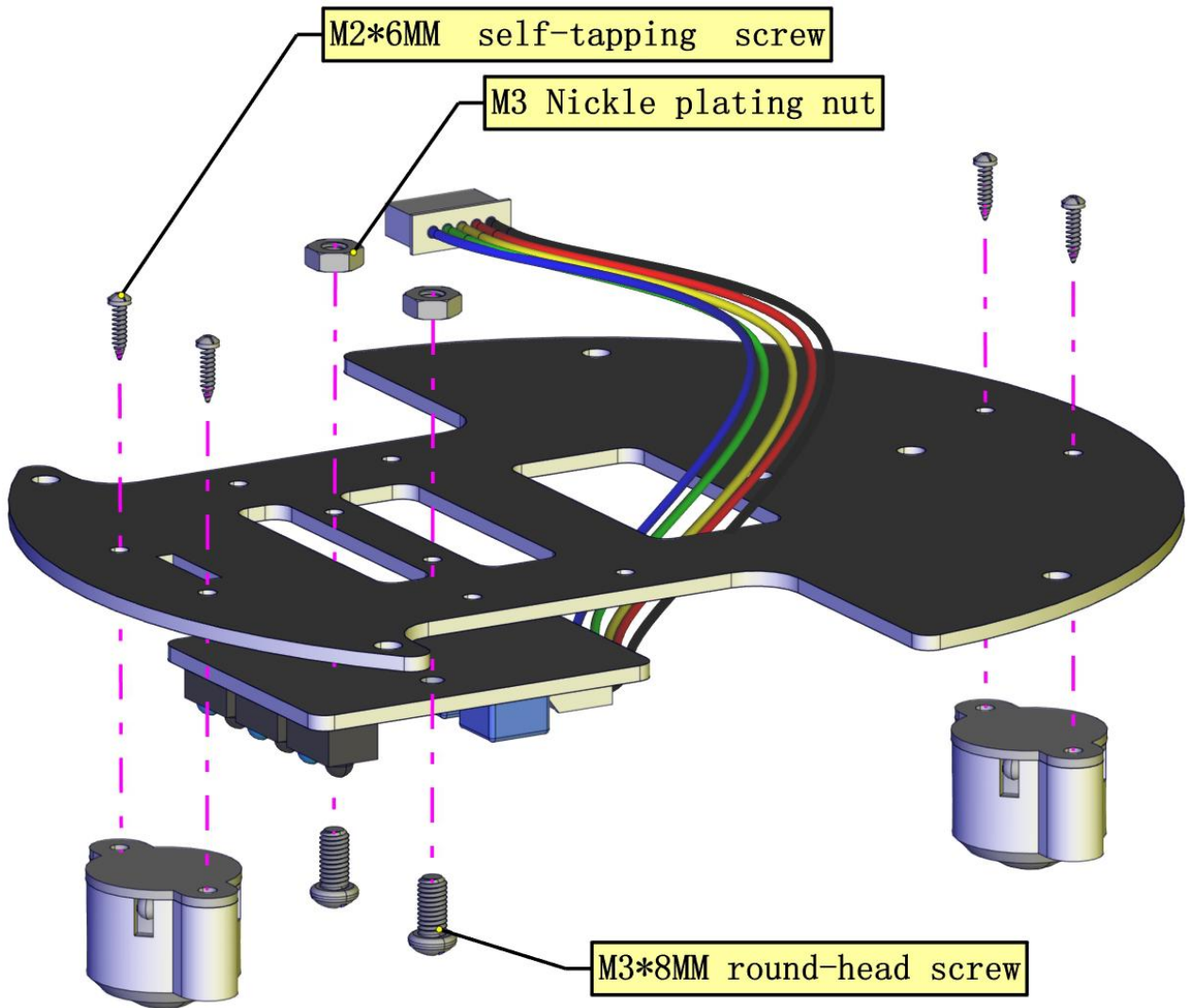
### 1. Install baseplate and line tracking sensor

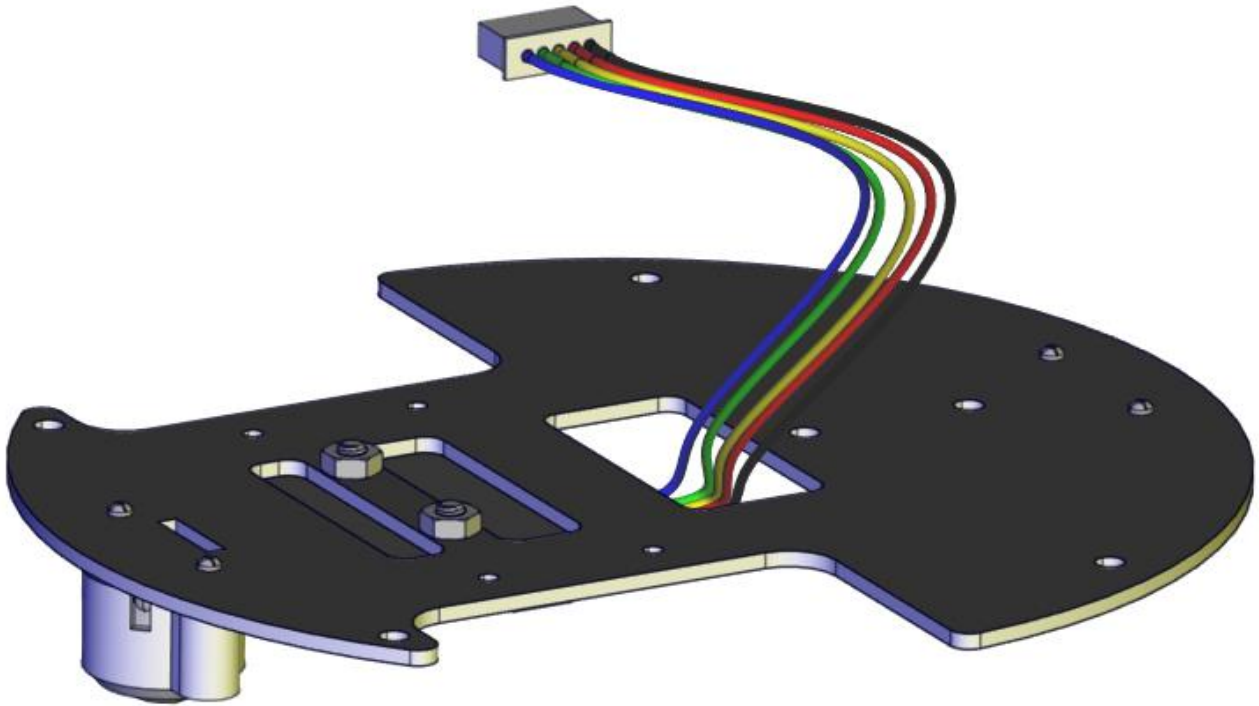




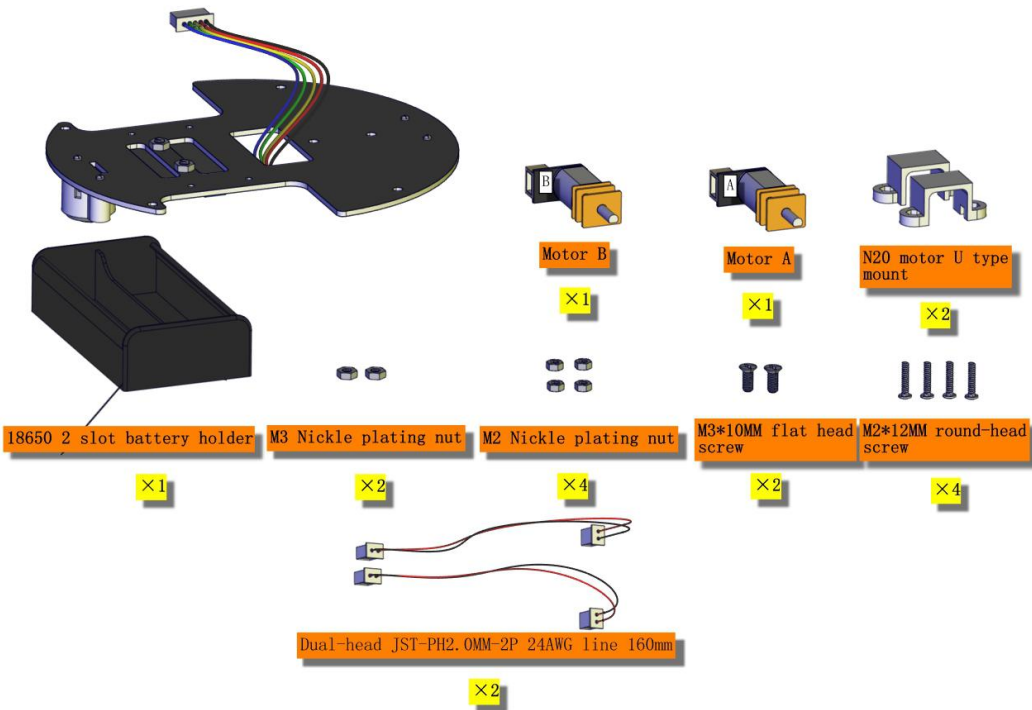
Note: we screw out the two self-locking screws first, and install universal wheel on V2.0 baseplate with these screws(don' t screw too tightly)  
Make the side printed "Keyes" downward.

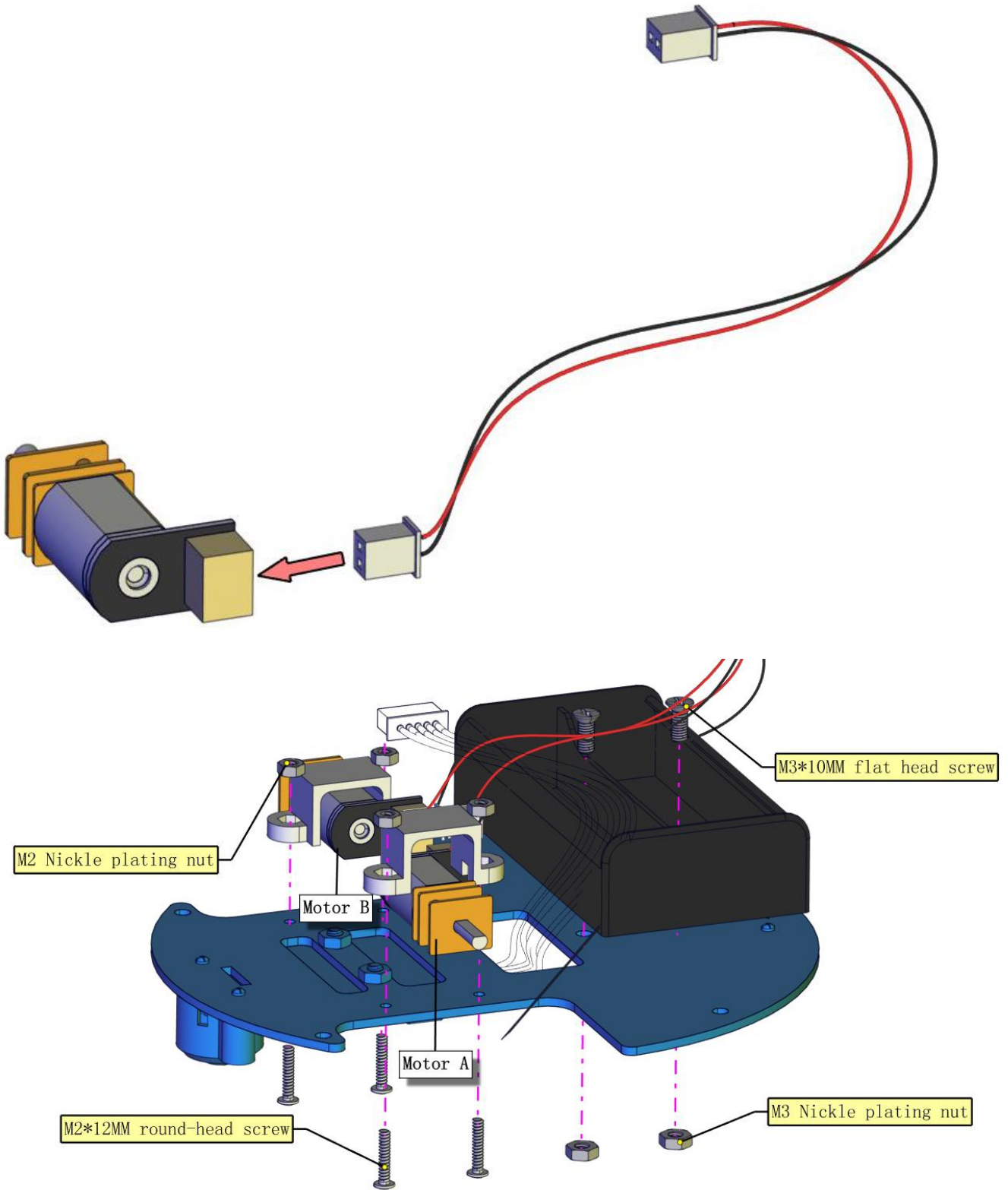


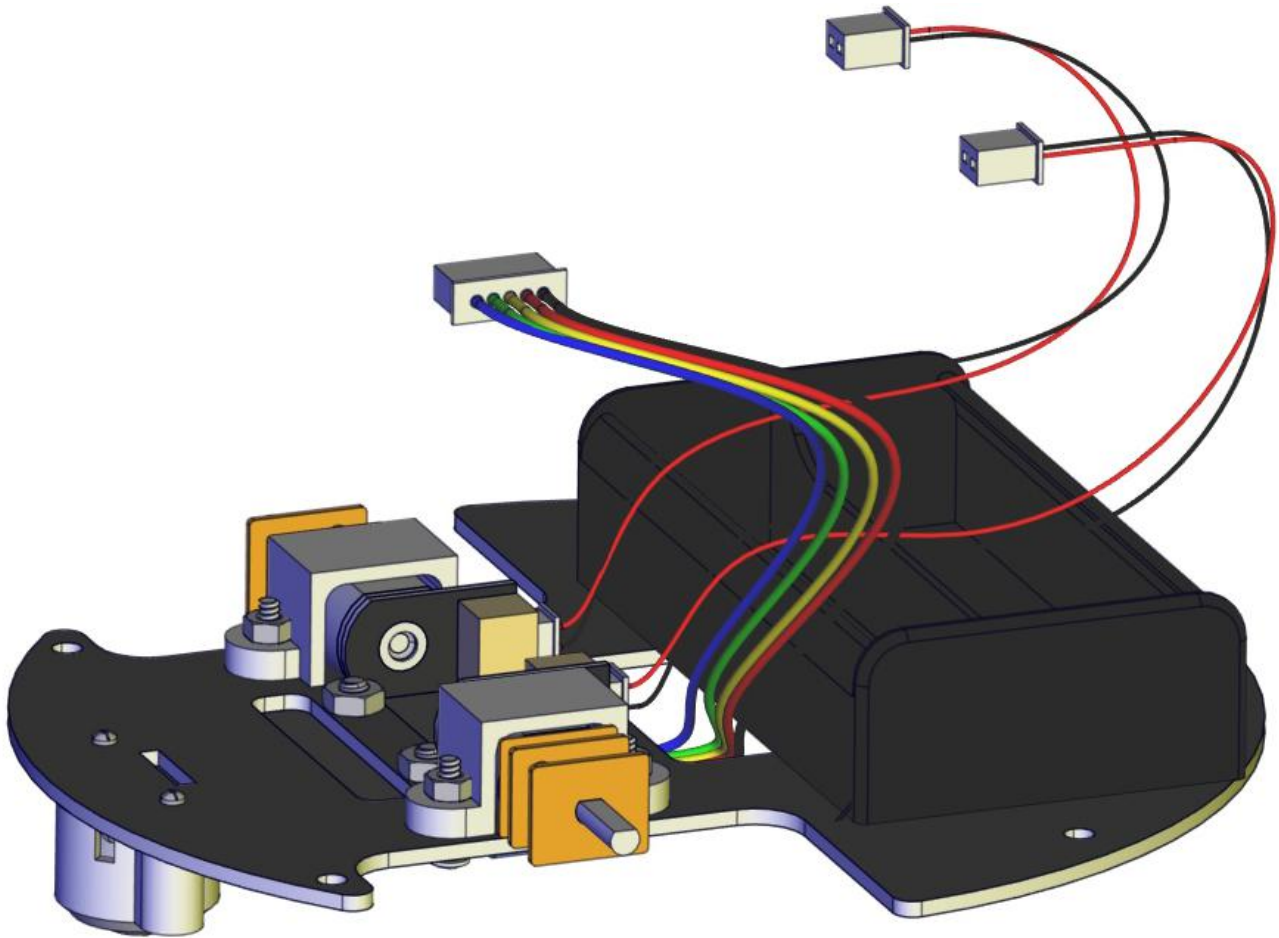




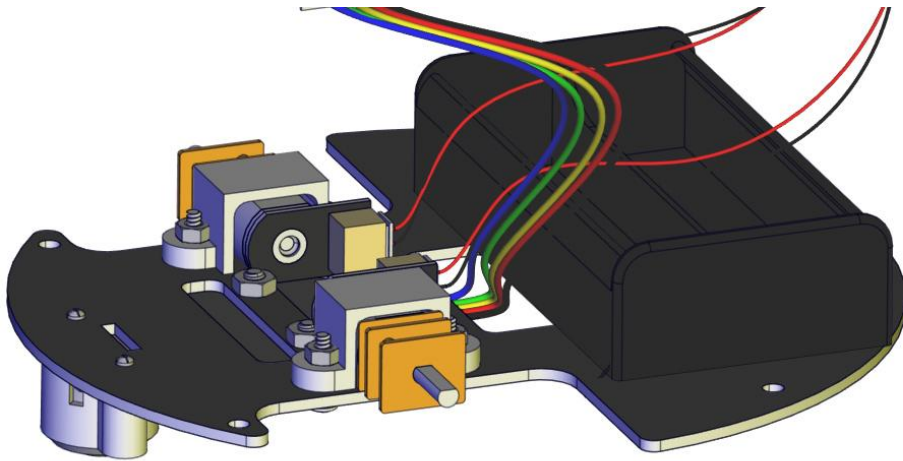
## 2. Mount Motor and Battery Holder





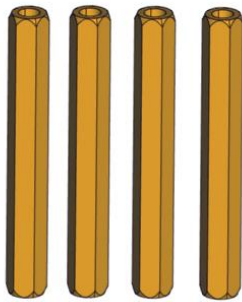


### 3. Assemble Car Wheels and Dual-pass Hex Copper Bushes



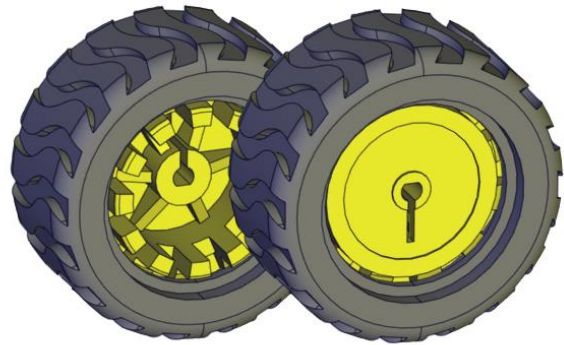
M3\*6MM round-head screw

×4



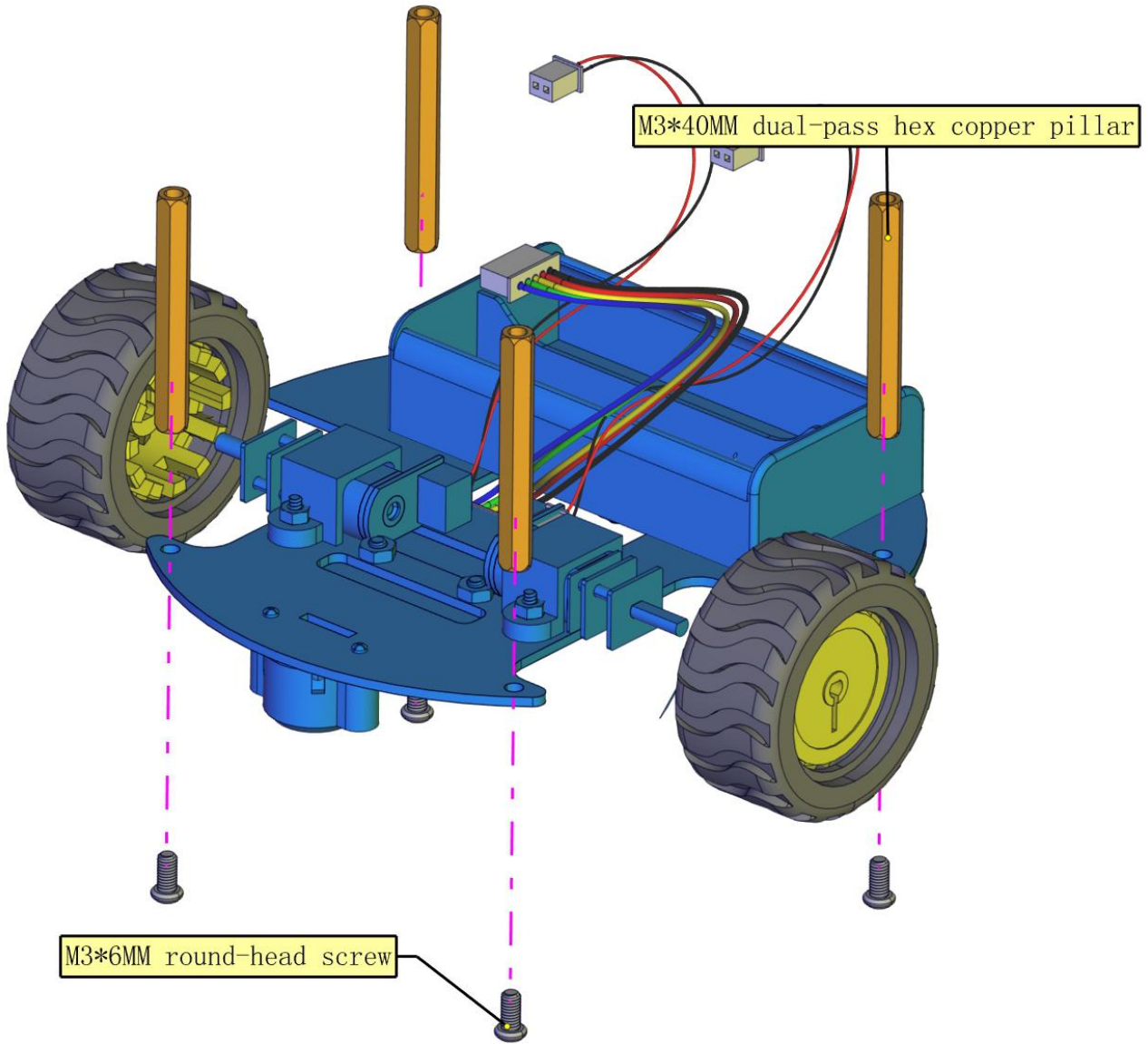
M3\*40MM dual-pass hex copper pillar

×4

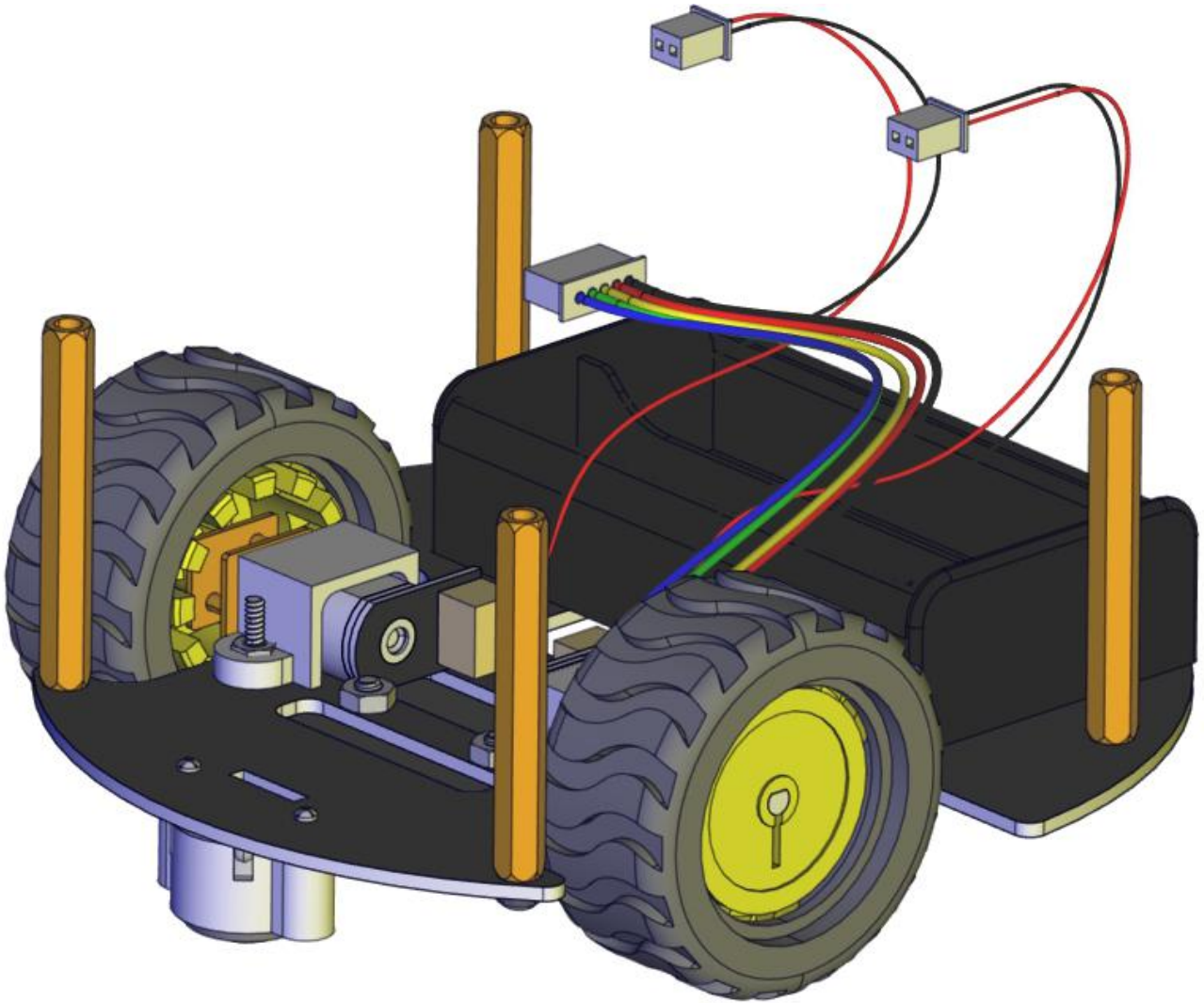


N20 motor special wheel

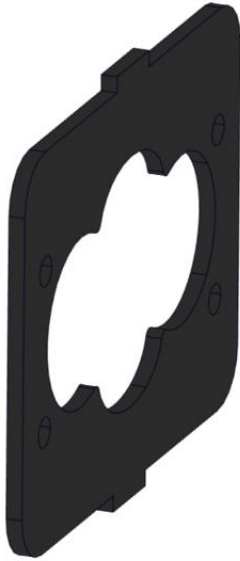
×2





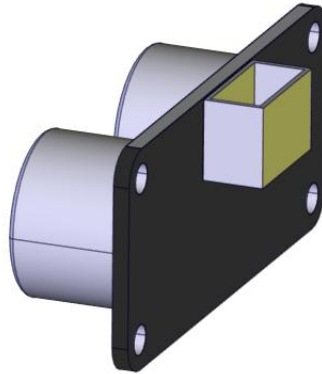


#### 4. Install Ultrasonic Sensor



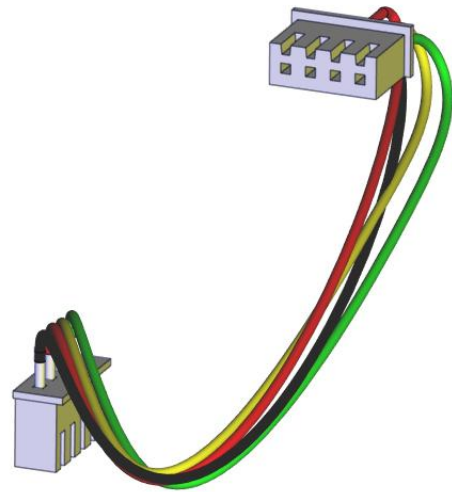
Ultrasonic fixed mount

×1



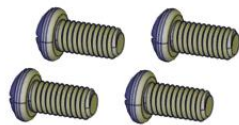
ultrasonic module

×1



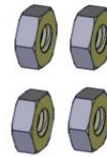
Dual-head JST-PH2.0MM-4P 24AWG line 8cm

×1



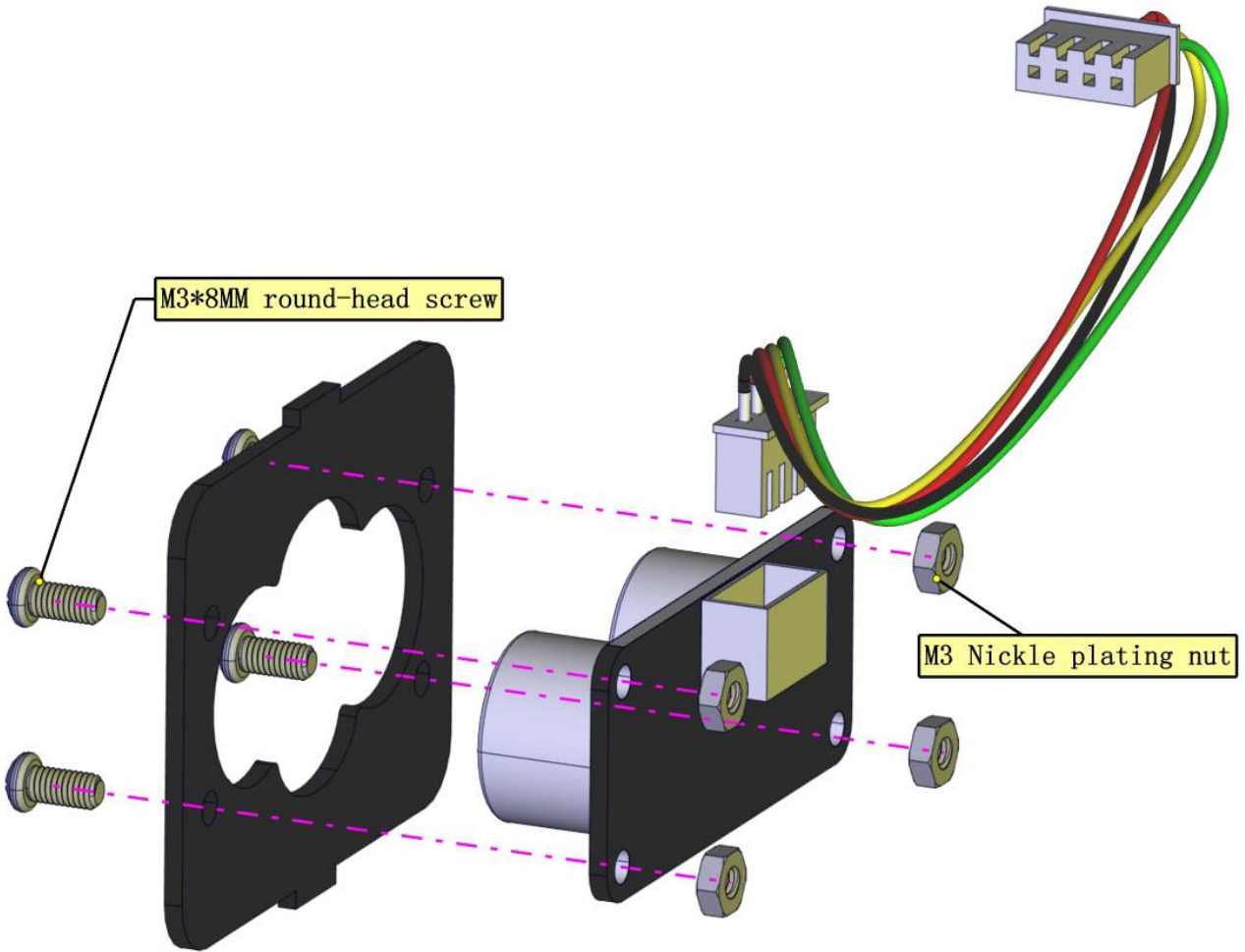
M3\*8MM round-head screw

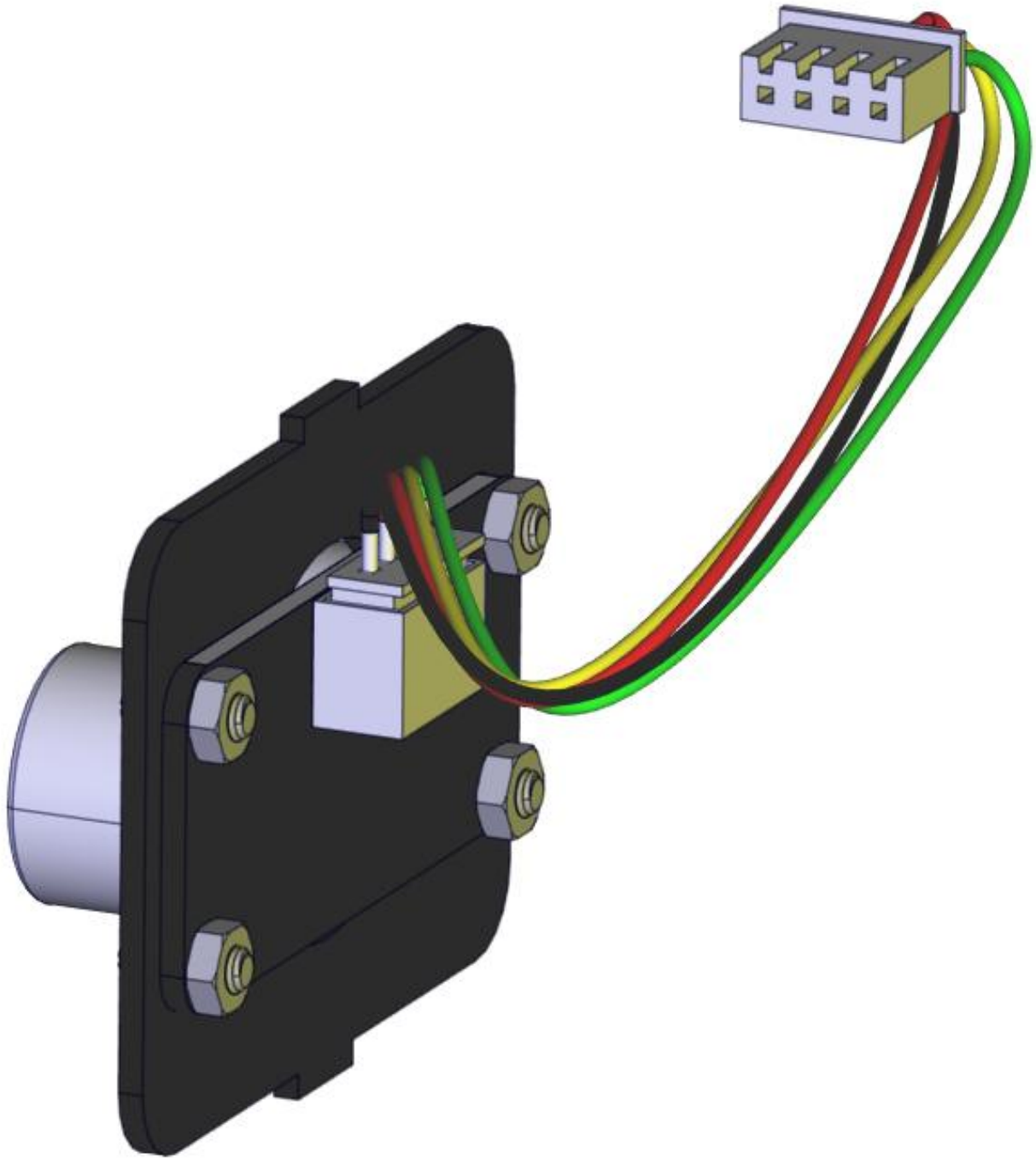
×4



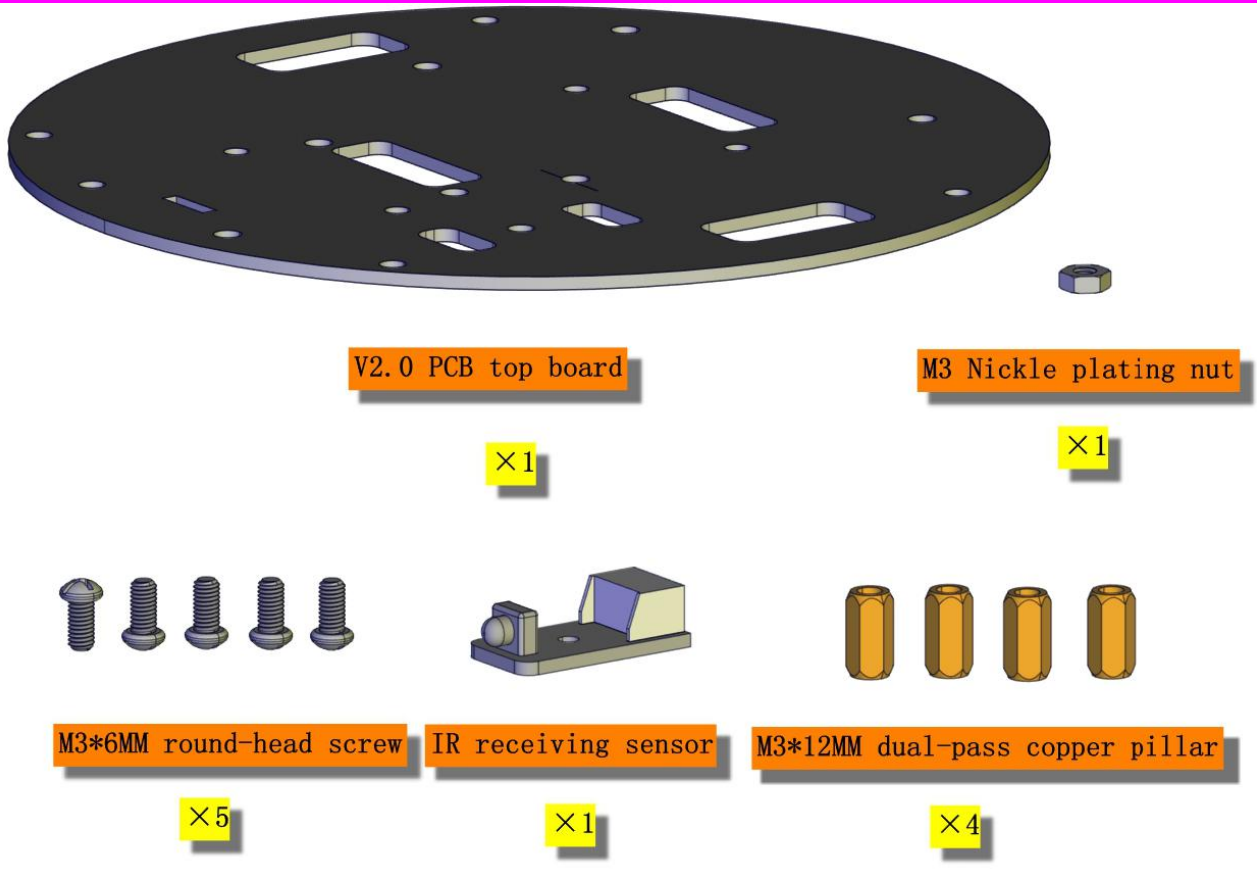
M3 Nickle plating nut

×4

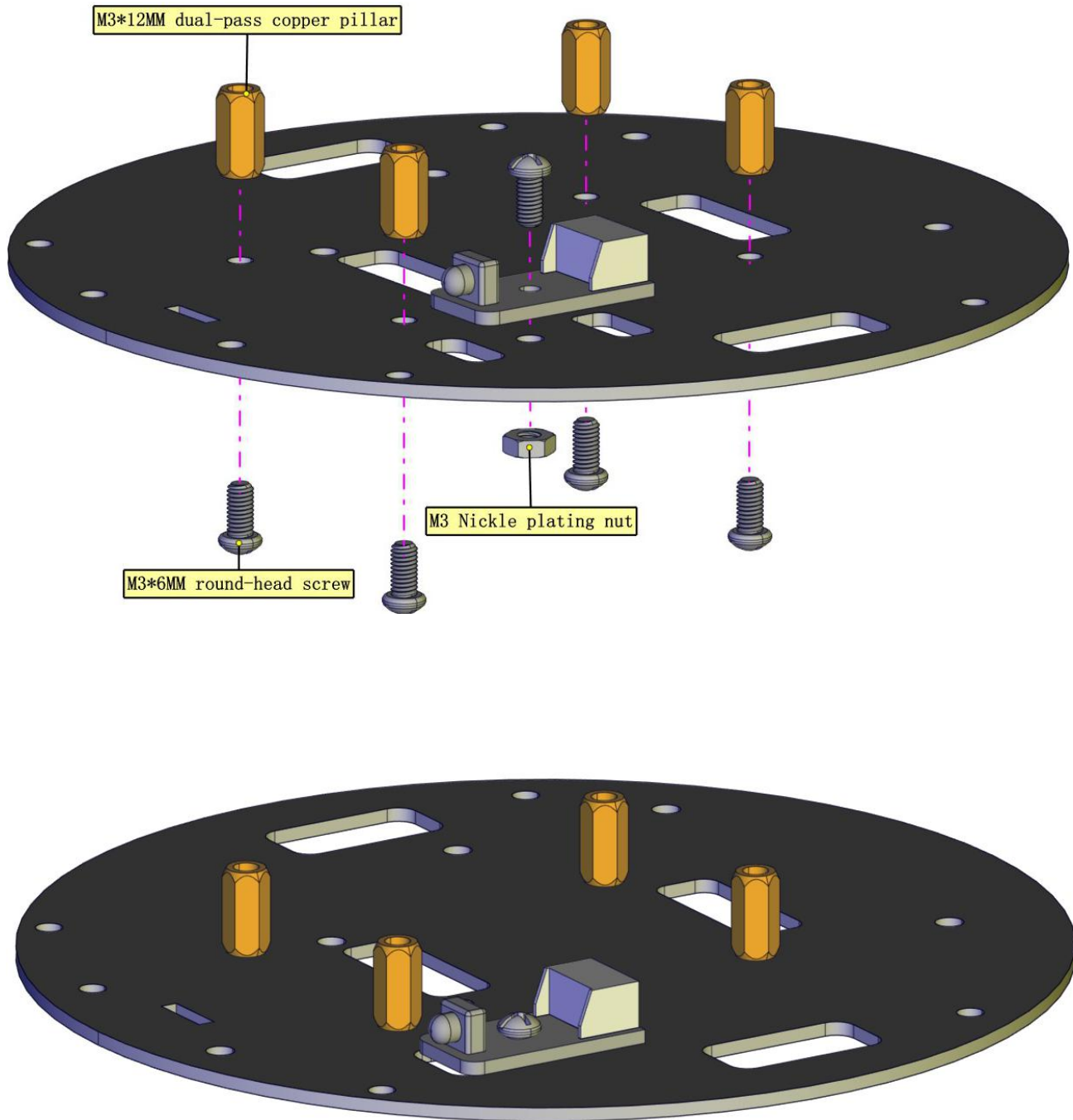


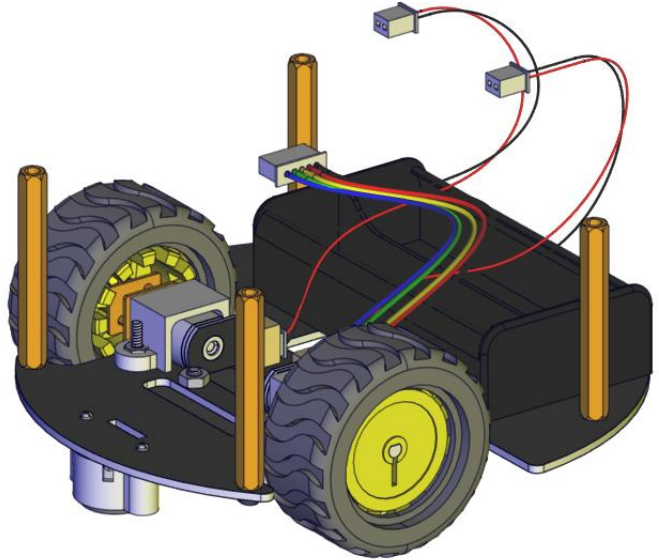
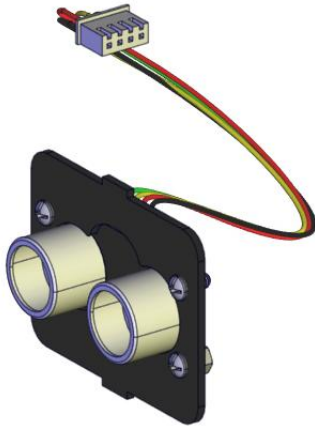
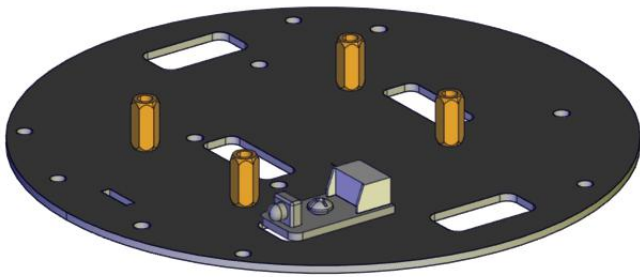


5. Mount the Middle Board and IR Receiver



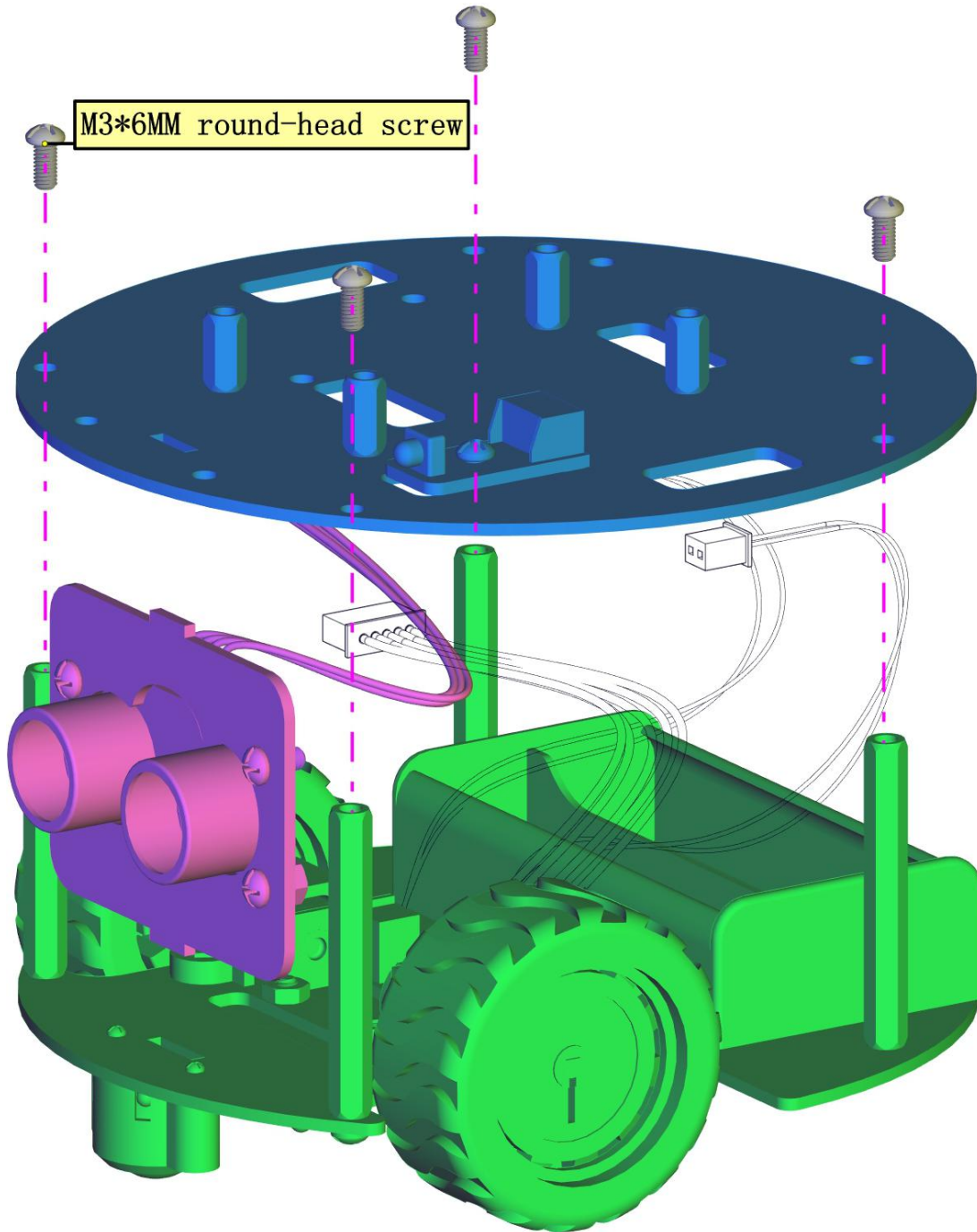
Note: make the side printed "Keys" upward when installing the middle board.



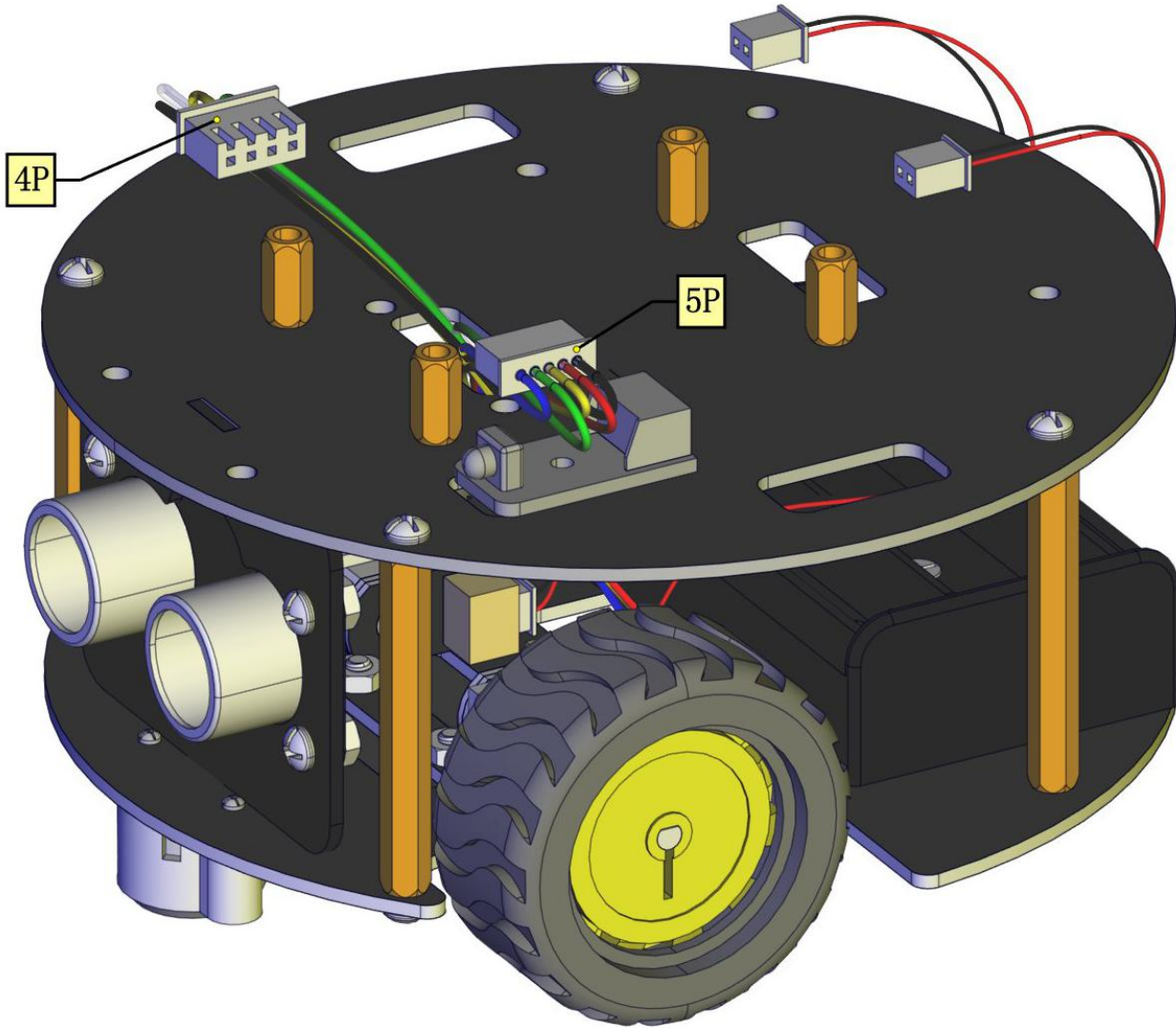


M3\*6MM round-head screw

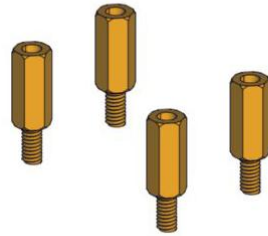
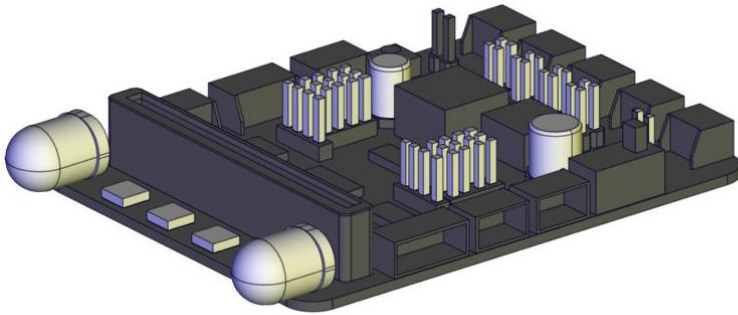
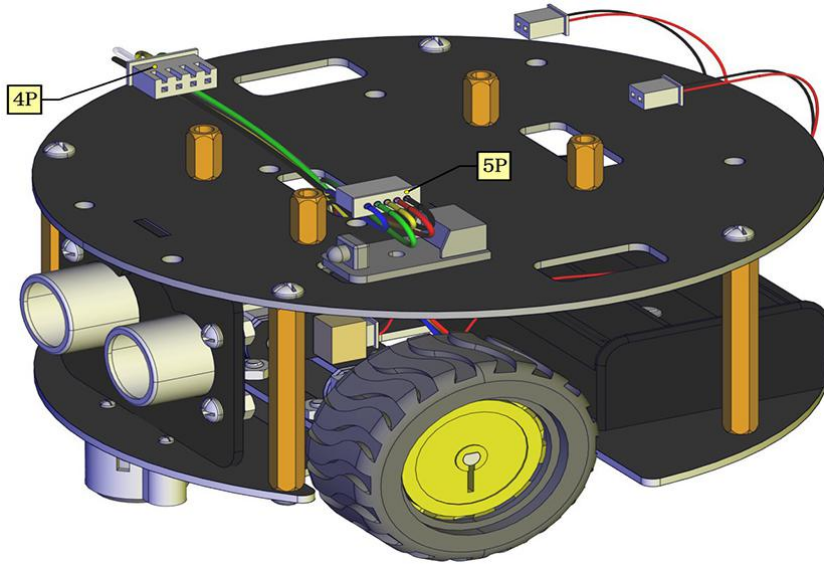
×4







6. Install micro:bit TB6612 Motor Driver Shield

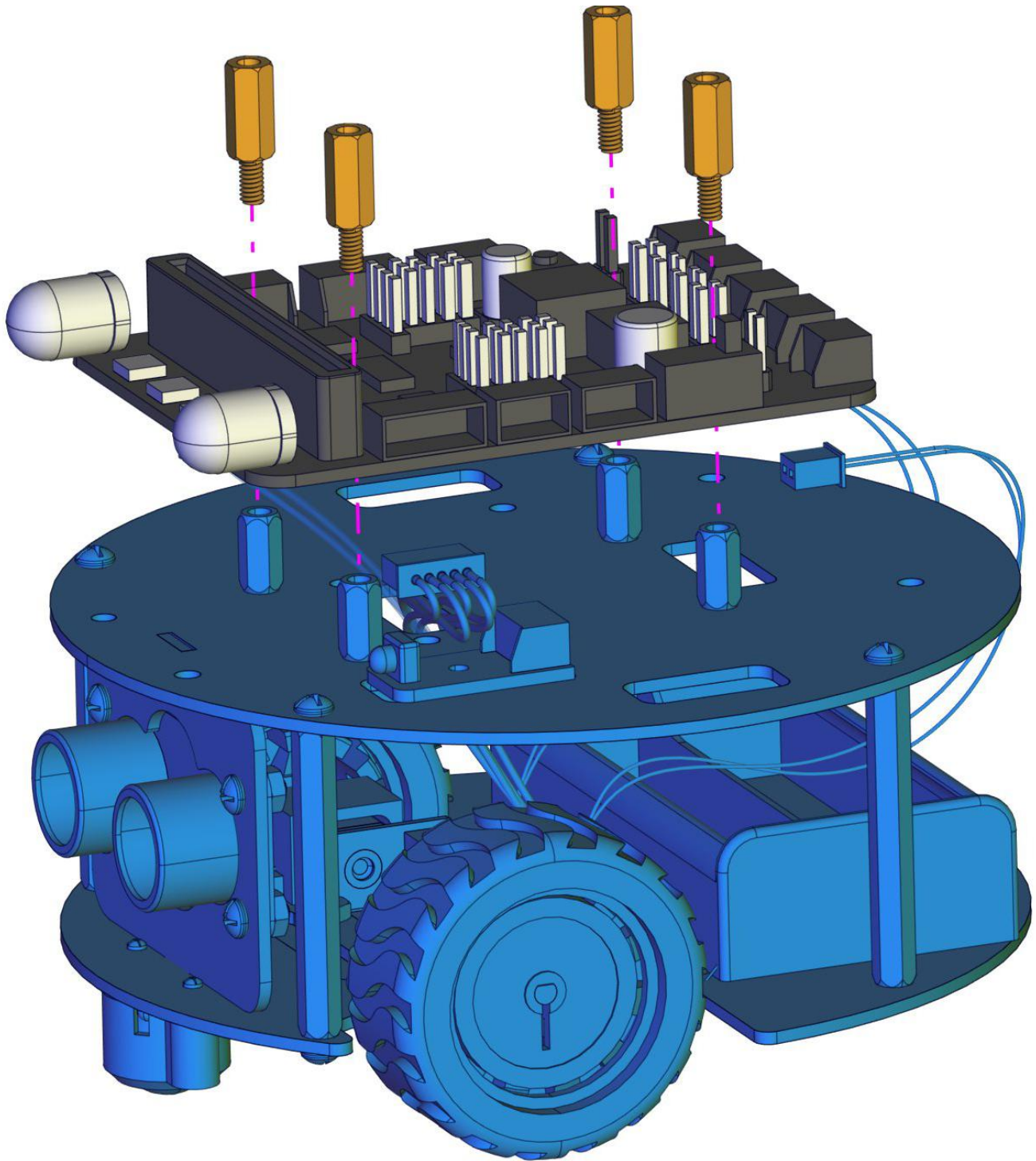


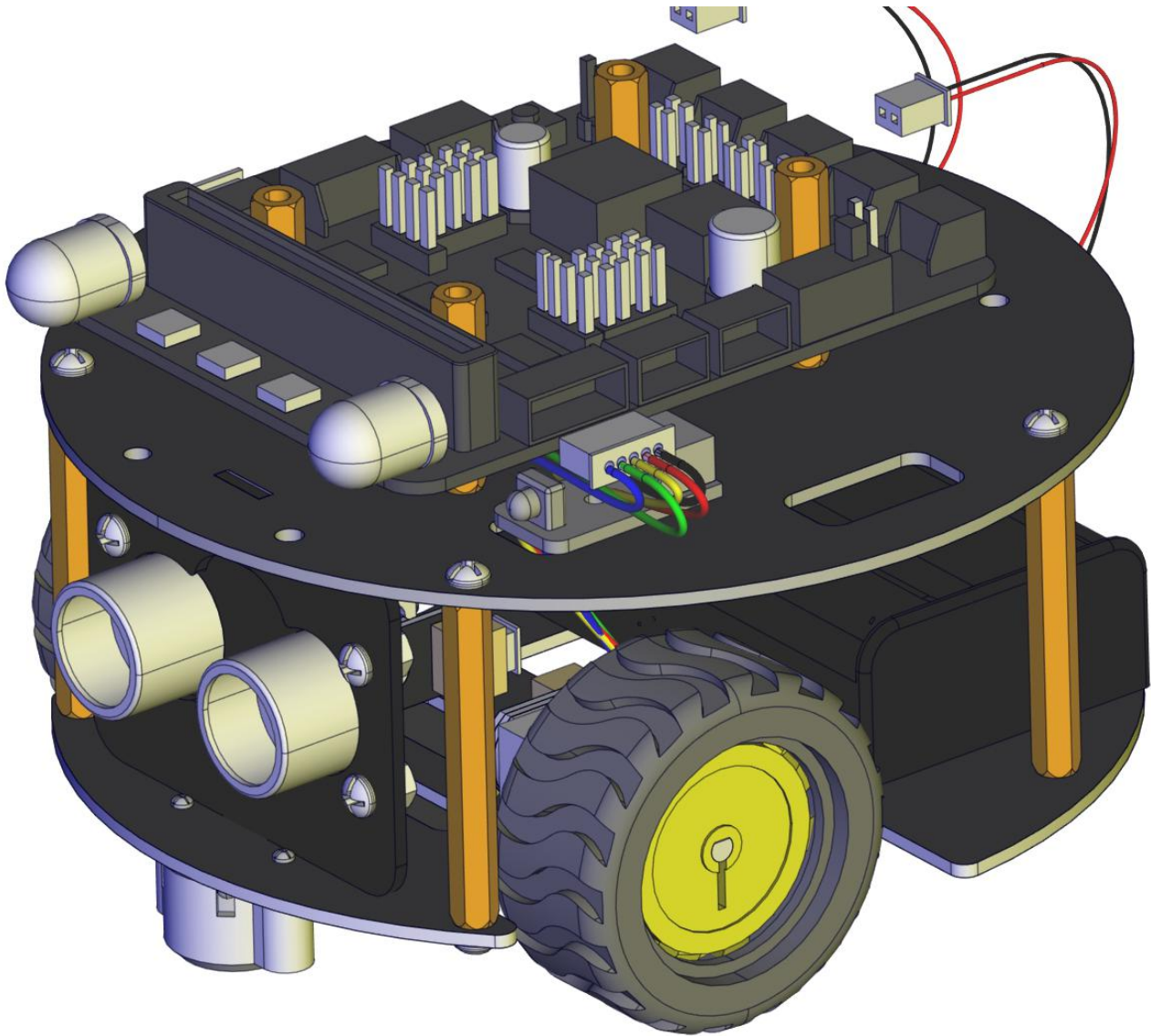
Micro:bit TB6612 Motor  
Driving Shield

M3\*15+6MM hex copper pillar

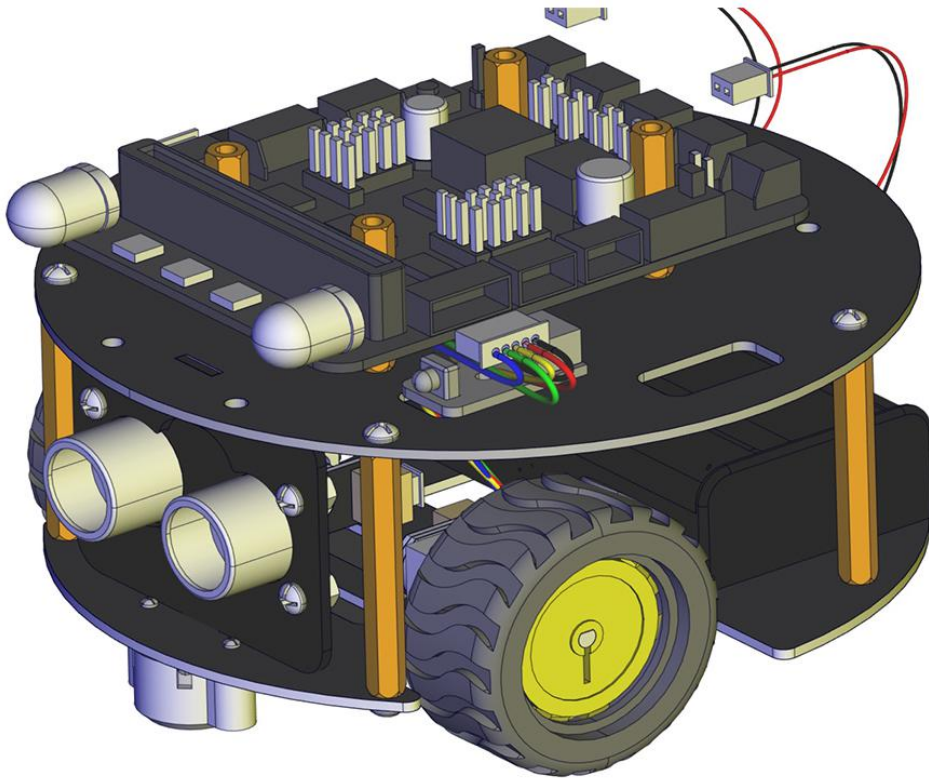
×1

×4



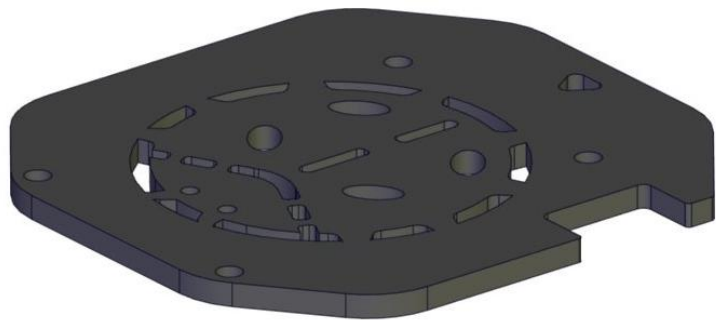


## 7. Install Top Board



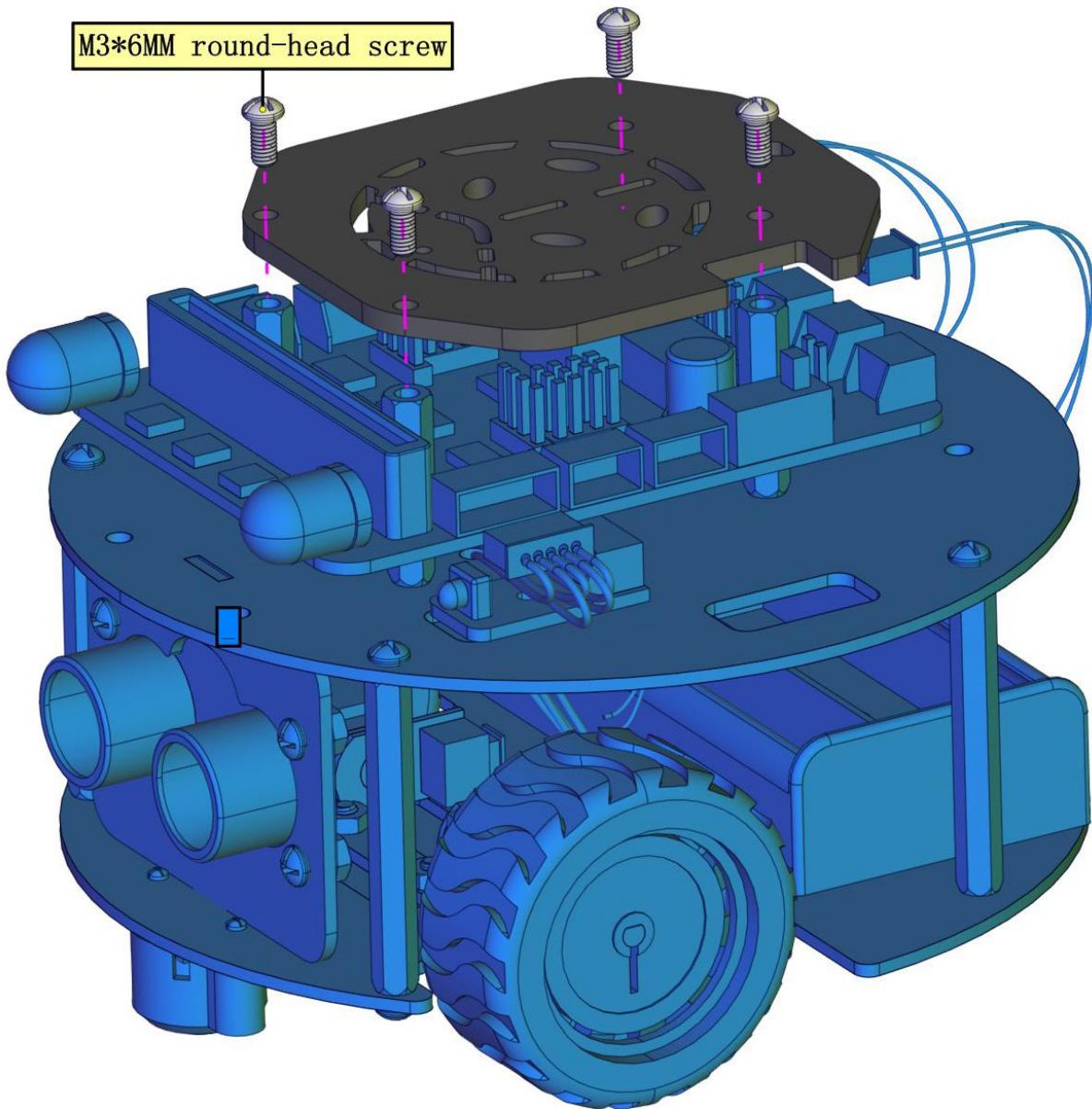
M3\*6MM round-head screw

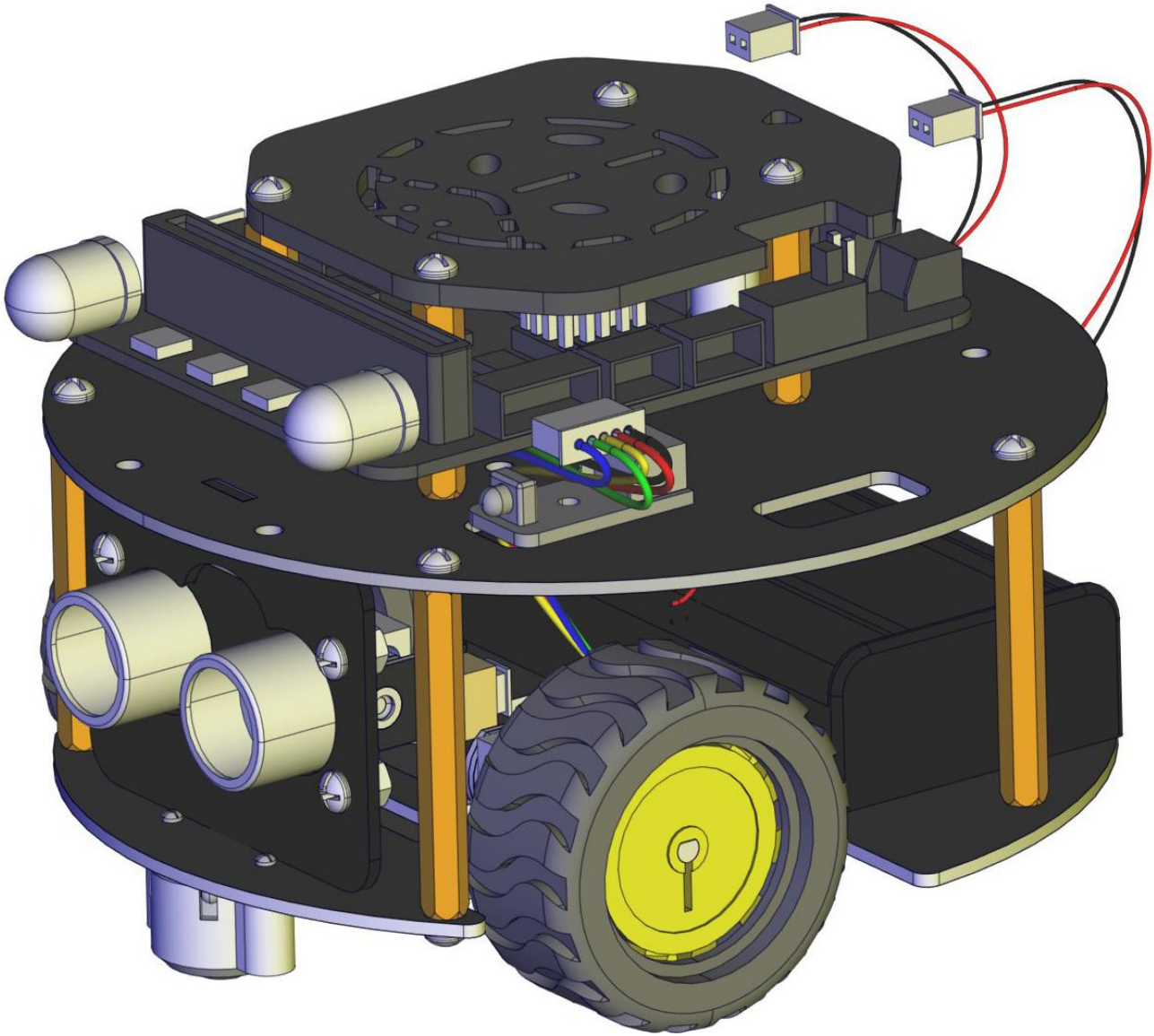
×4



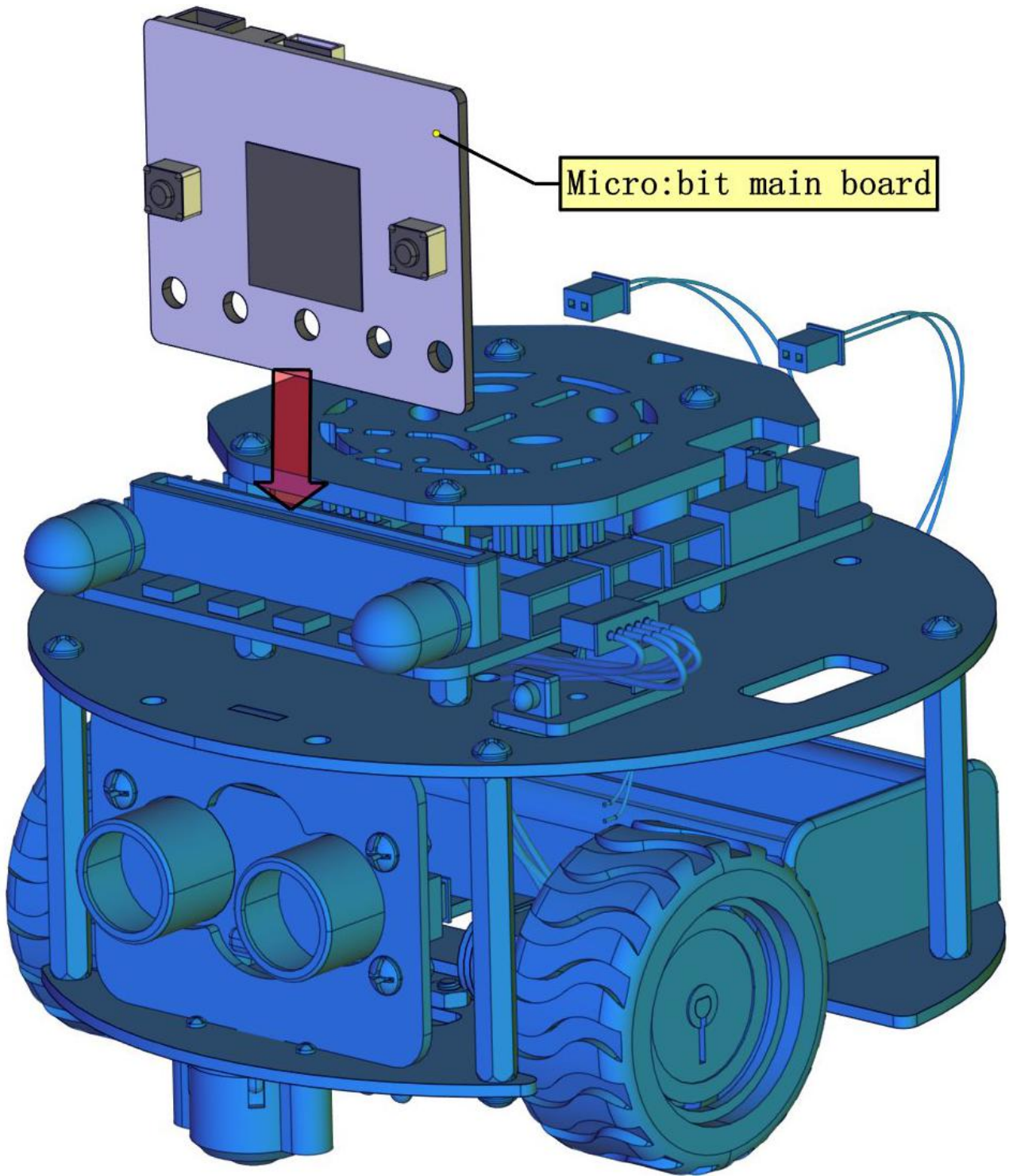
Microbit turtle Top acrylic board

×1

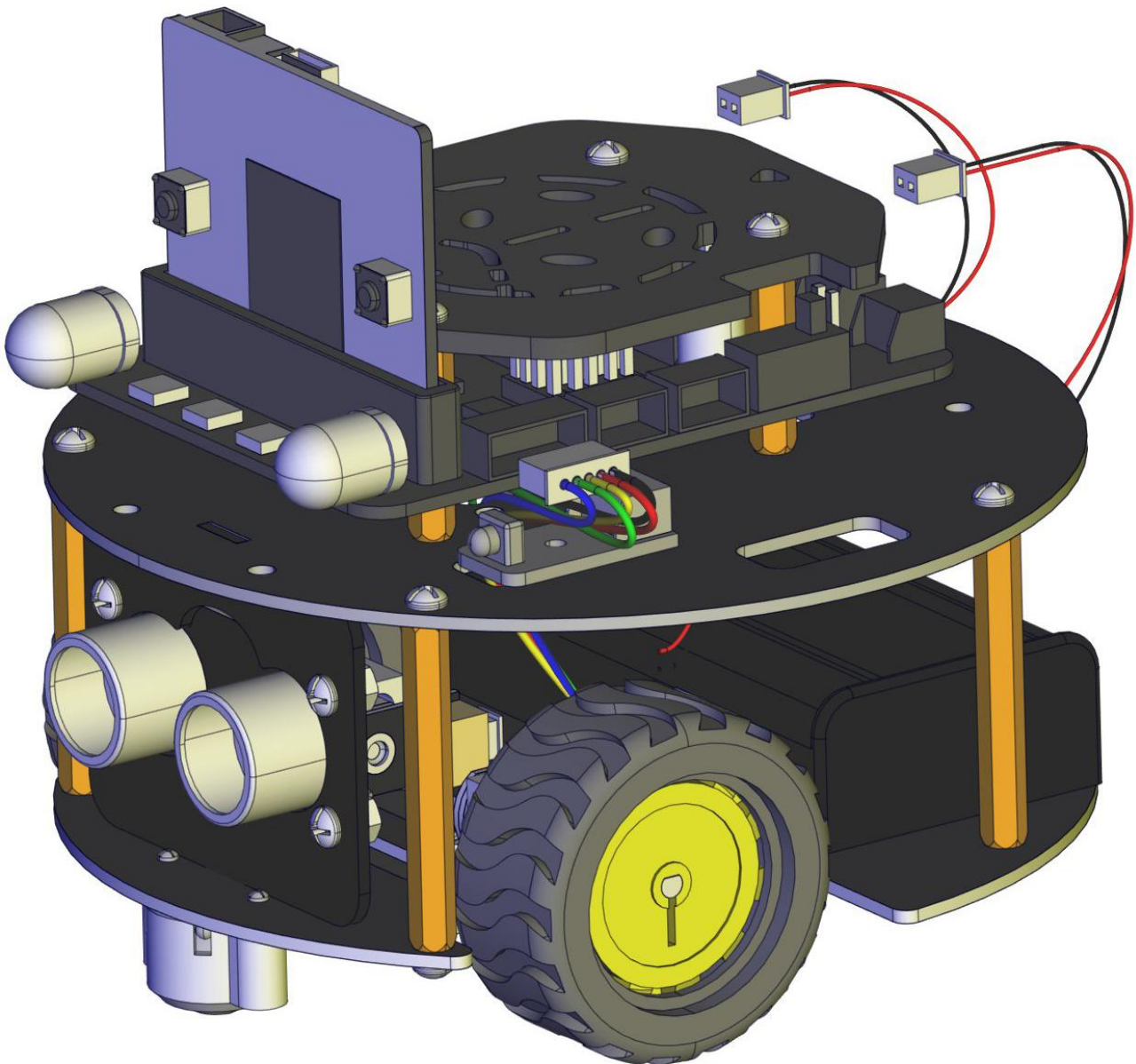




## 8. Install Micro:bit Board







## Wiring Up

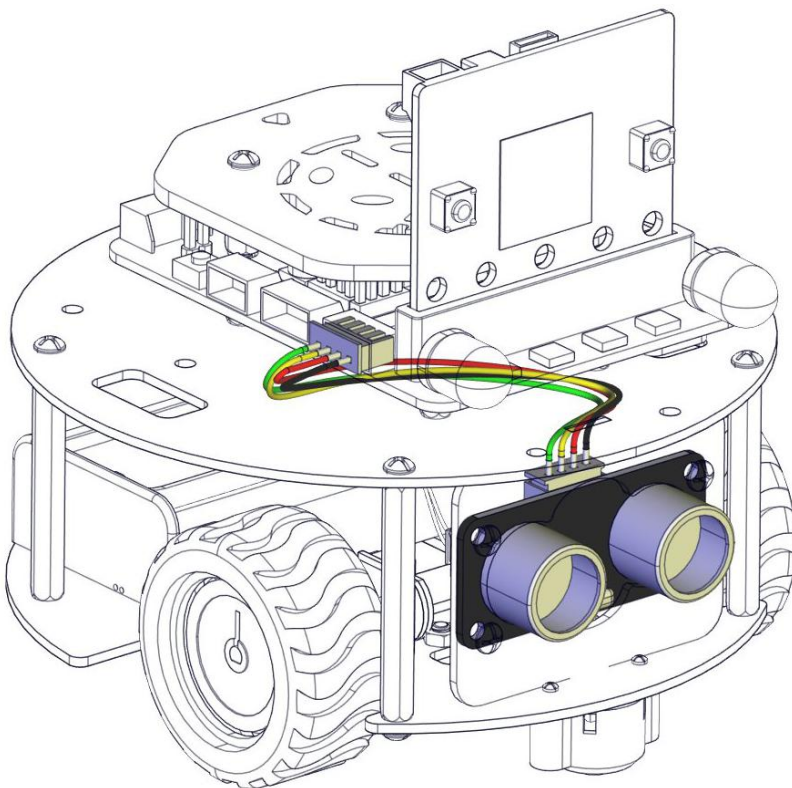
We need an additional 3P wire to connect IR receiver.

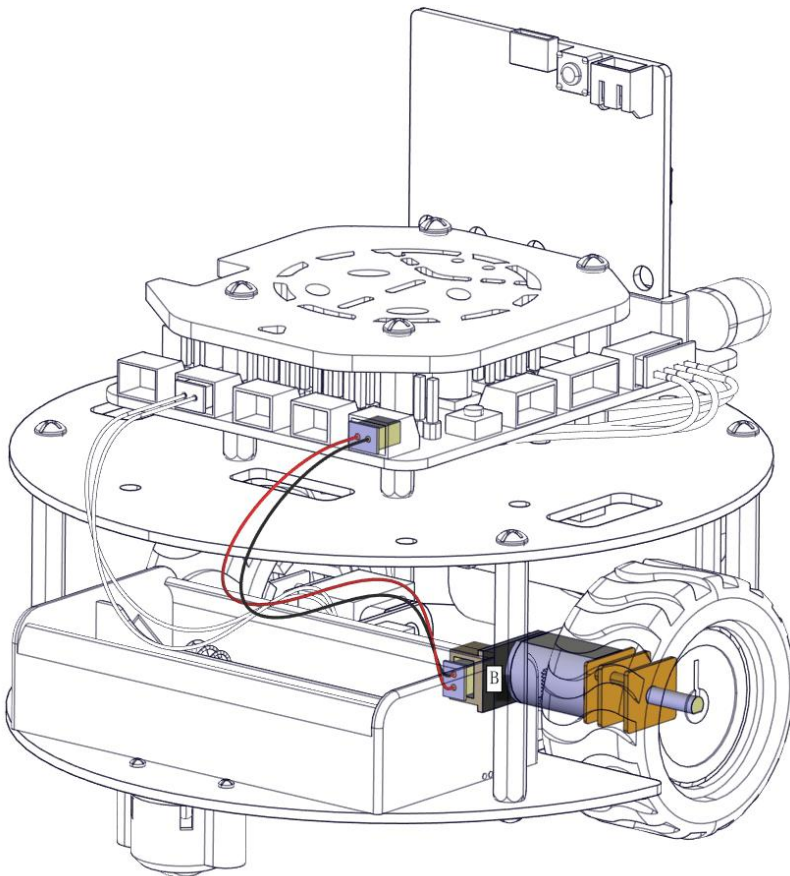
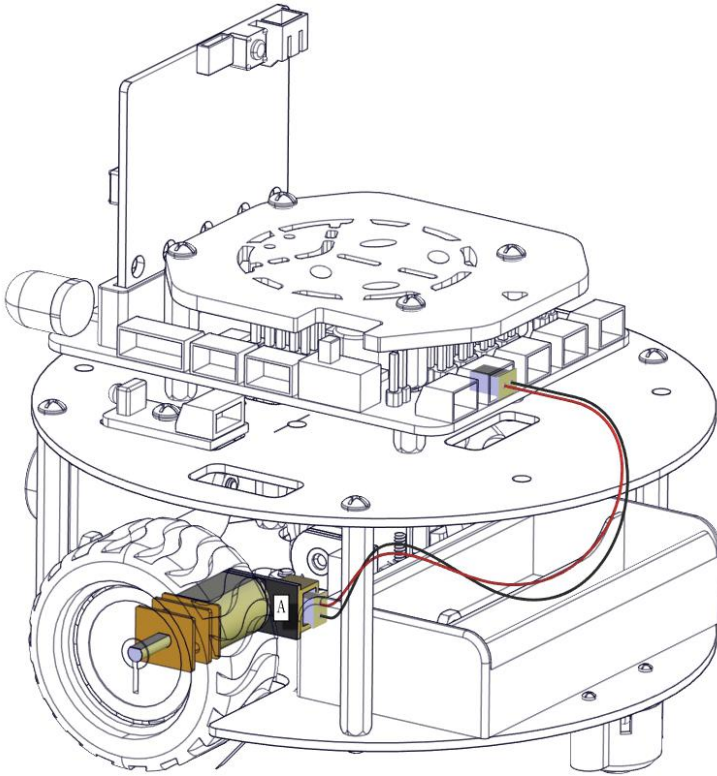
Black line is connected to G (-) , red wire is connected to 5V (+)

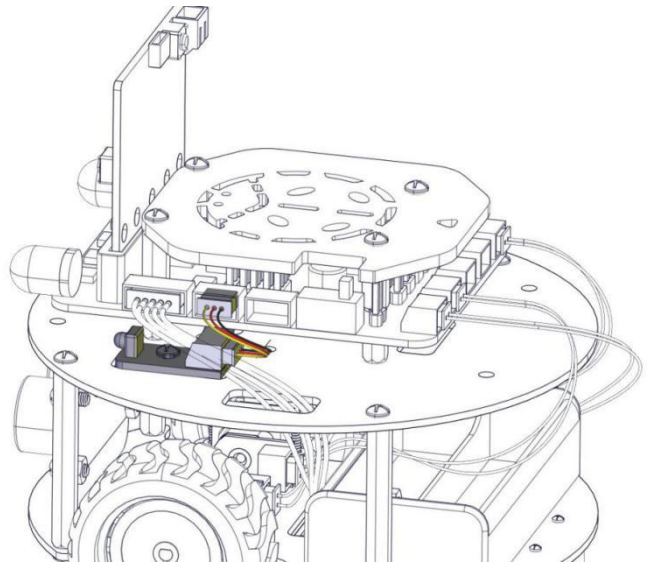
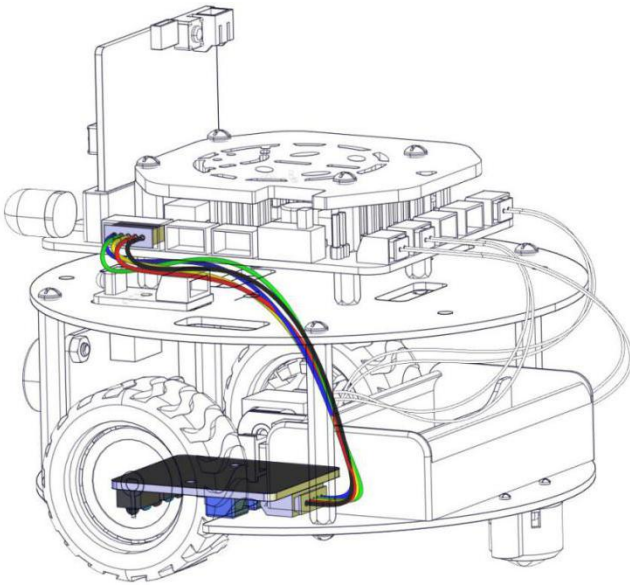
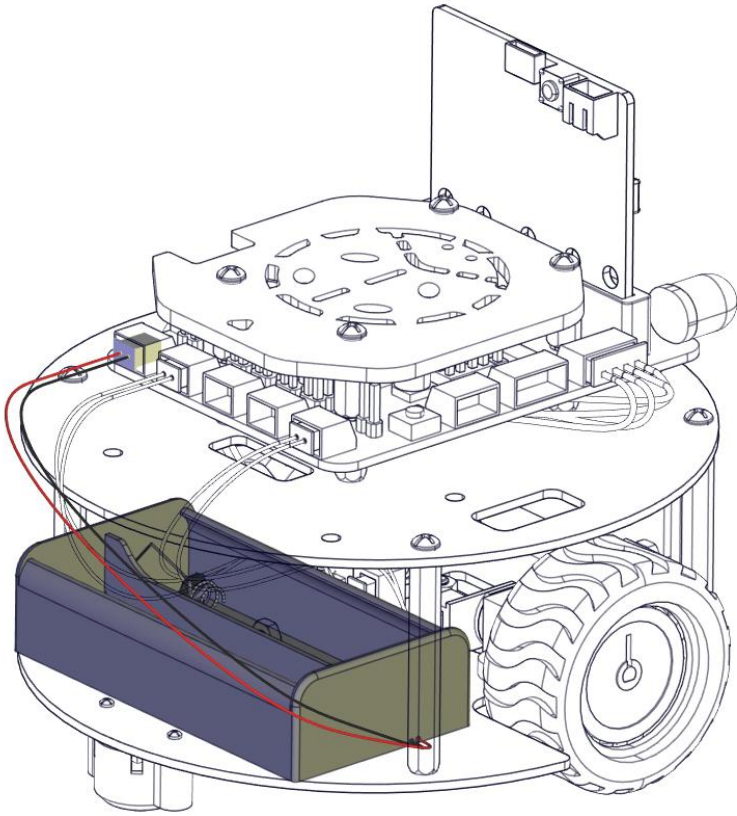


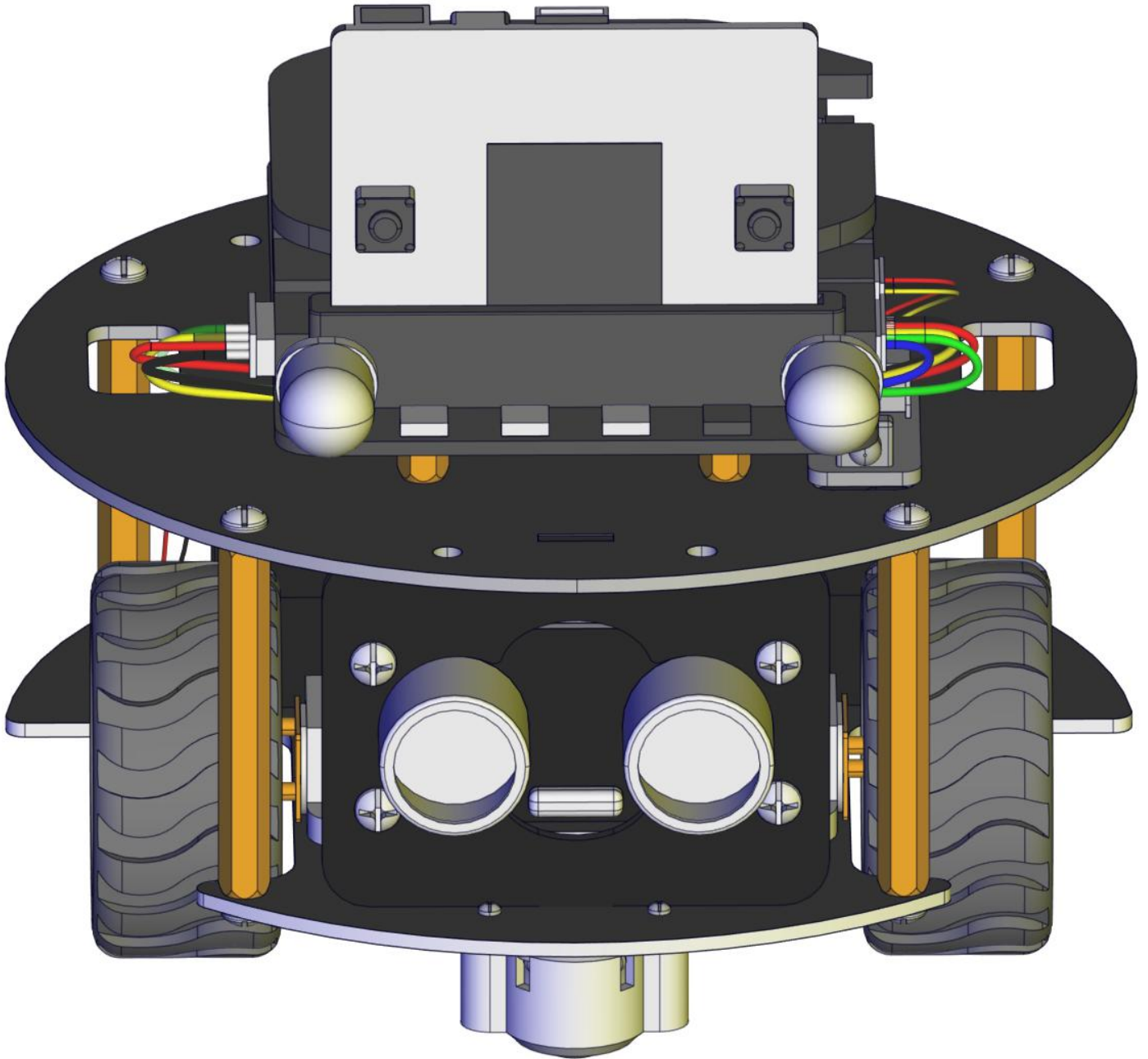
	Battery Holder	Motor A(Left)	Motor B(Right)	Line Tracking Sensor	IR Receiver	Ultrasonic Module
Driver	7-12V	A1	B1	P16 P15	P11 5V G	P2 P1 5V G
Shield	(+ -)			P14 5V G		

Wire up ultrasonic sensor, motors, IR receiver, line tracking sensor and battery holder









## 6. Code and Programming

We take Windows system as example to show you.

Get started with Micro:bit: <https://microbit.org/guide/quick/>



## Step 1: Connect Micro: bit Board

Link micro:bit board to computer with USB cable. (Guide to mobile apps:

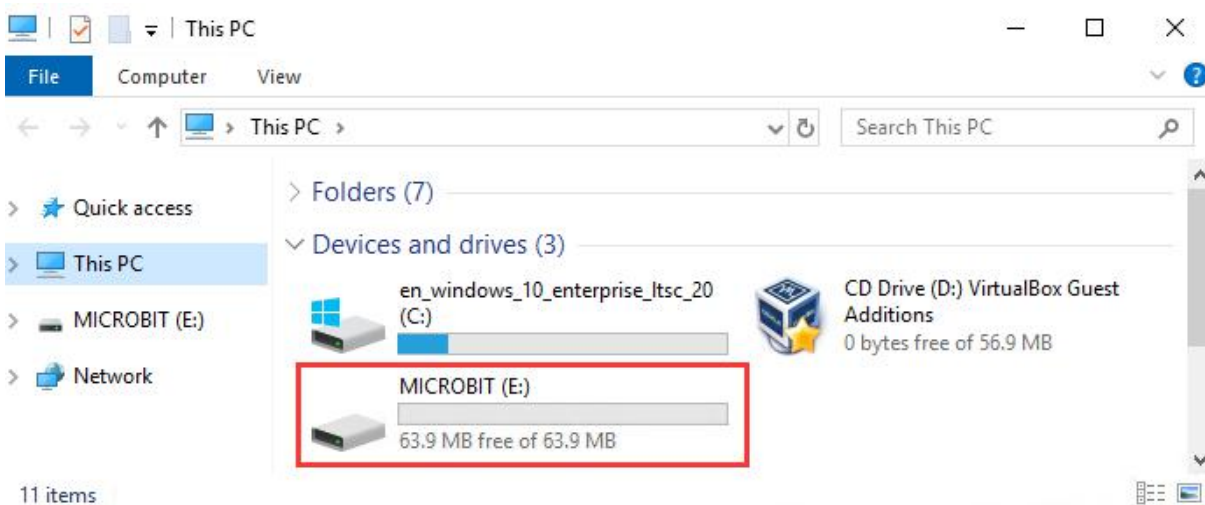
<https://microbit.org/get-started/user-guide/mobile/>)

Macs ,PCs, Chromebooks and Linux system (including Raspberry Pi)

support micro: bit.



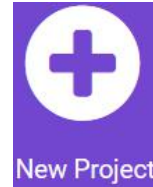
There will be a MICROBIT drive in your computer, as shown below:





## Step 2: Programming

Enter <https://makecode.microbit.org/> then click

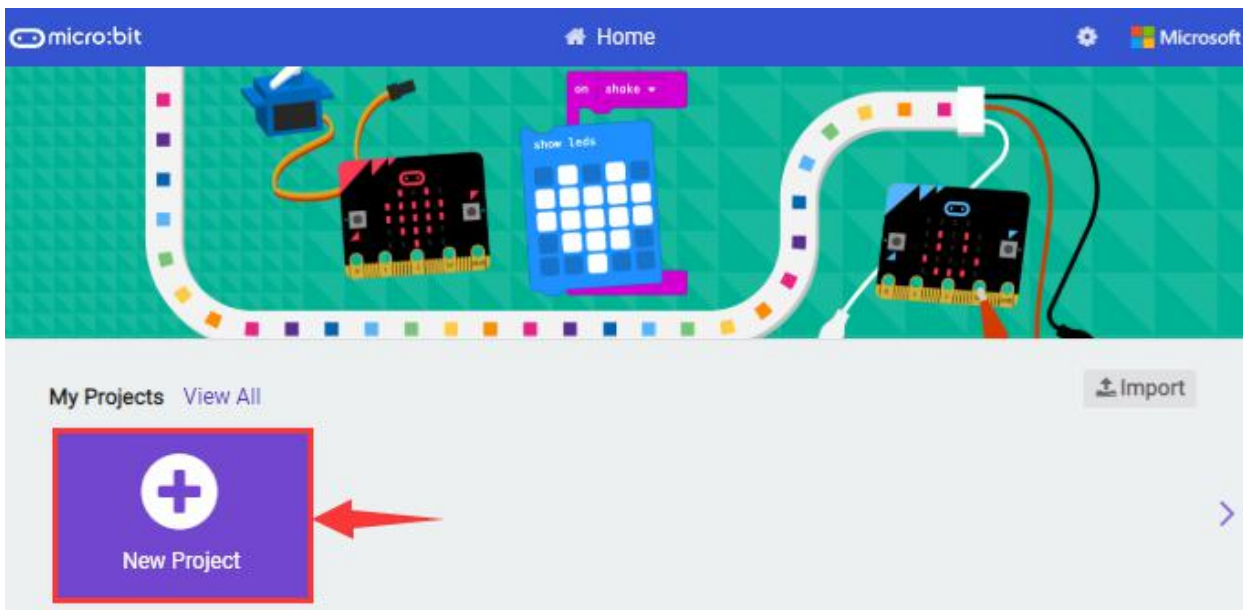


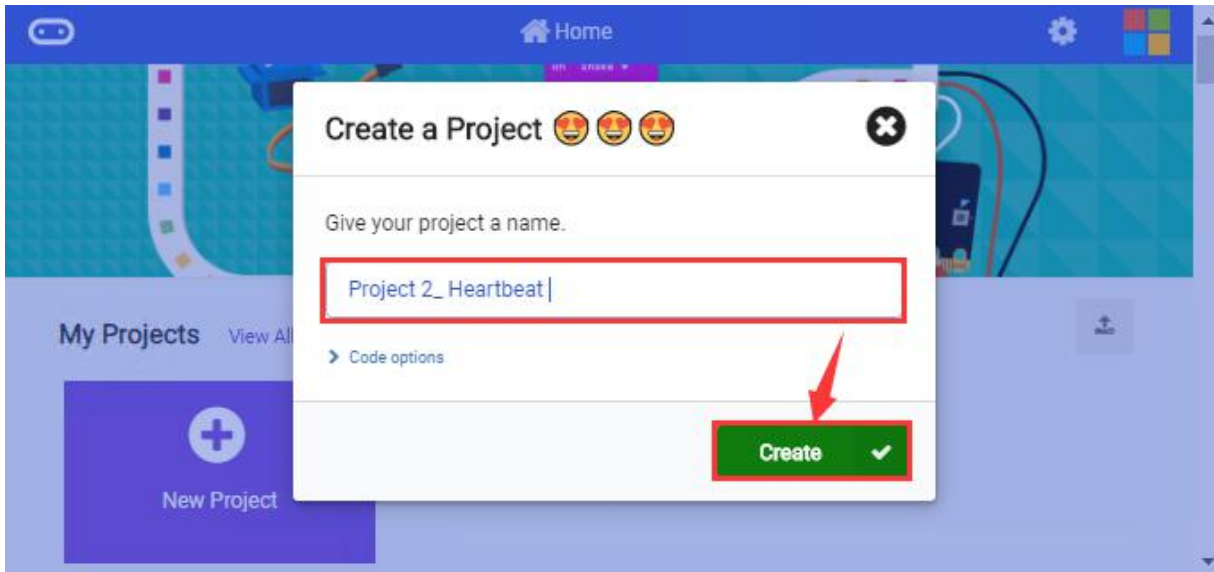
and you will

view a dialog box.

We recommend you to use Google Chrome

Input **“Project 2\_ Heartbeat”** to name your project and click **“Create”**



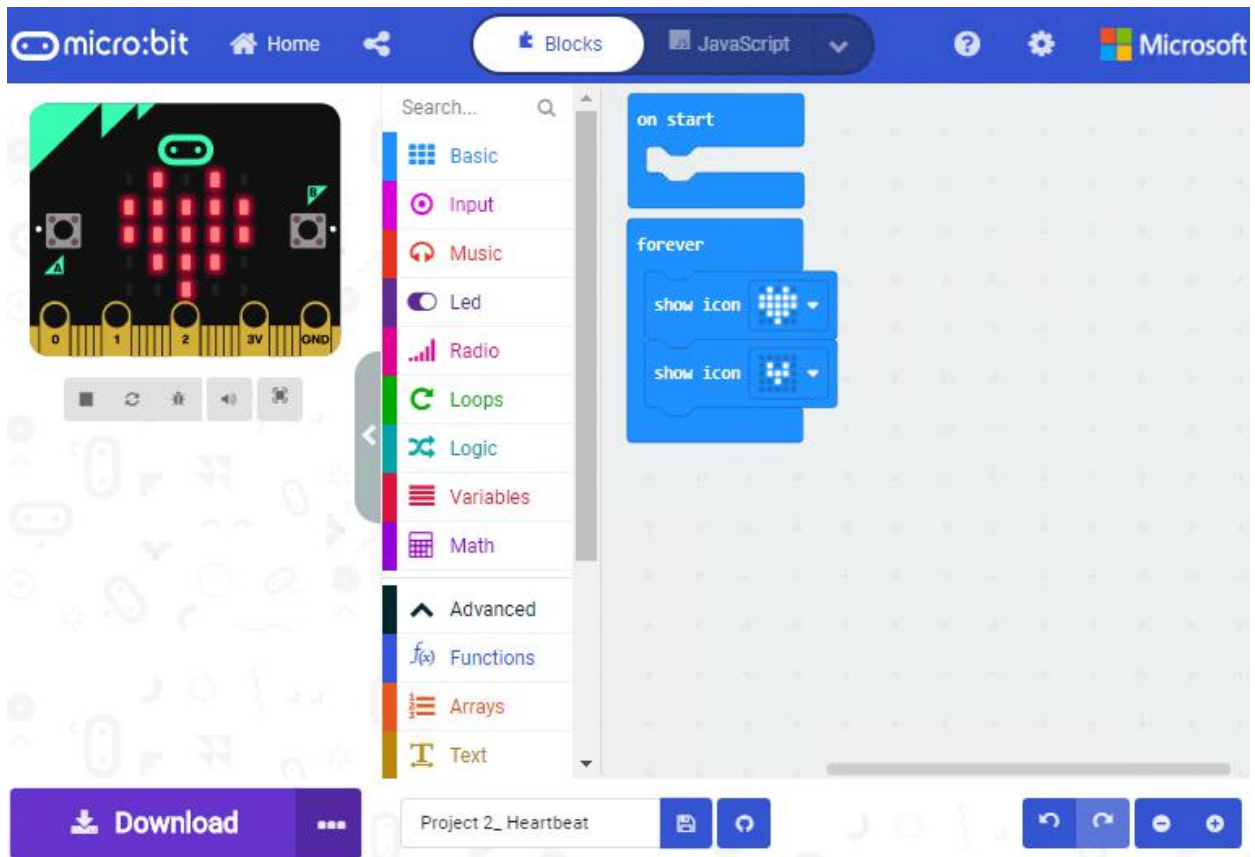


Through MakeCode editor, you just need to drag blocks into code editing area to design program. There is a video to show you how to finish “heartbeat” pattern.

[microbit-heartbeat.mp4](#)

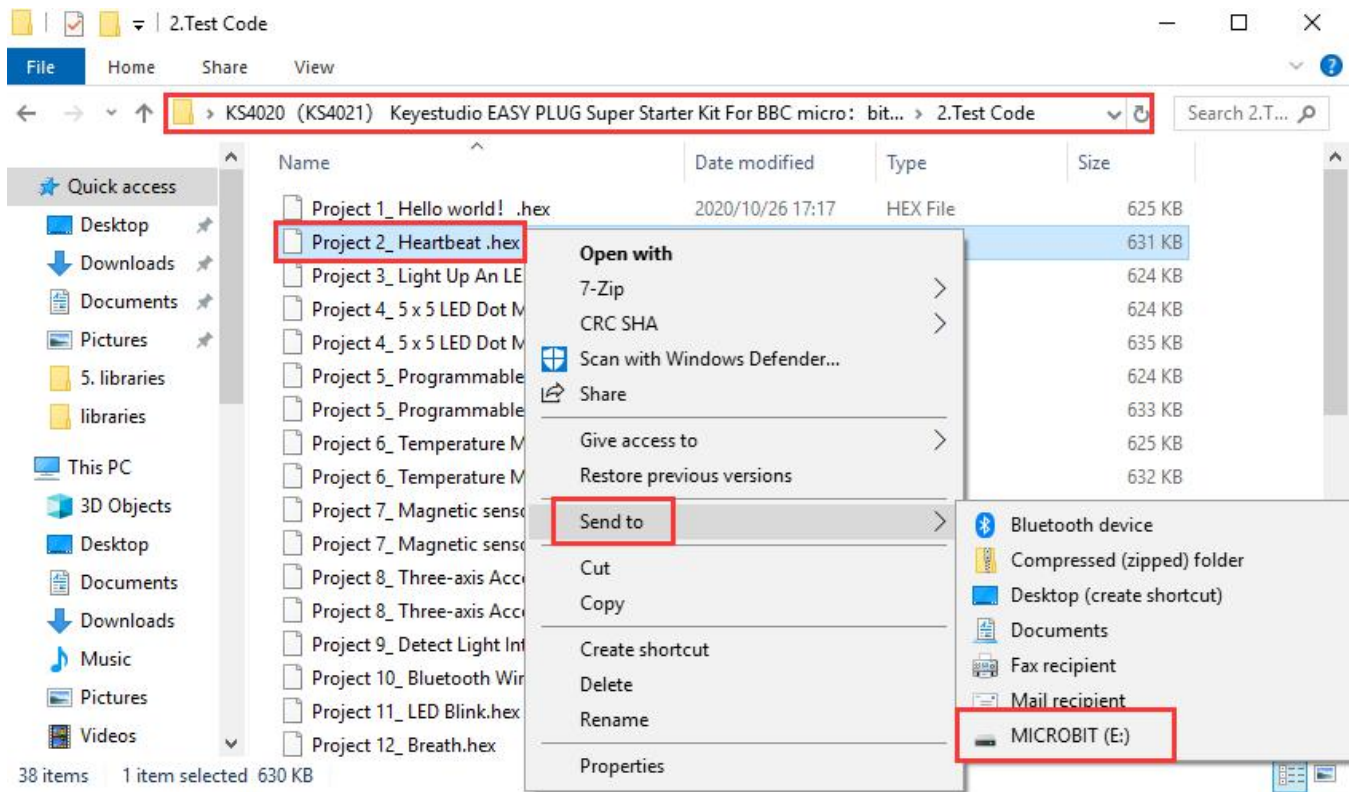
Next, we will introduce Makecode.





### Step 3: Download Code

Enter Makecode editor, tap “Download” to get a “hex” file. Then copy it into MICROBIT drive.



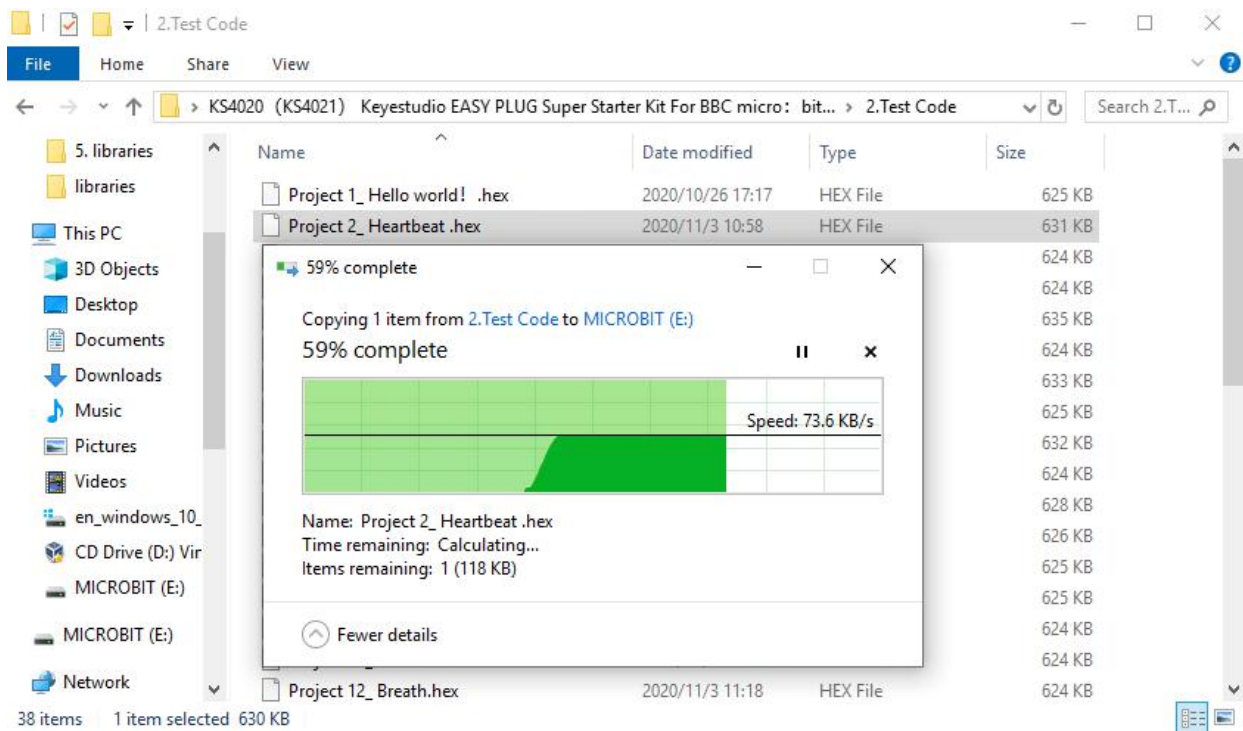
Or you could find out “hex” file firstly and right-click to select “Send to MICROBIT (E) ” .

Then hex file will be copied on MICROBIT drive.



File Explorer window showing a directory of files. The selected file is **Project 2\_Heartbeat.hex**. A red arrow points from this file to the MICROBIT (E:) drive in the left sidebar. A red diagonal watermark reads "drag it to here". A context menu is open over the MICROBIT (E:) drive with "Copy to MICROBIT (E:)" selected.

Name	Date modified	Type	Size
Project 1_ Hello world! .hex	2020/10/26 17:17	HEX File	625 KB
<b>Project 2_Heartbeat .hex</b>	2020/11/3 10:58	HEX File	631 KB
Project 3_ Light Up An LED.hex	2020/11/3 10:58	HEX File	624 KB
Project 4_ 5 x 5 LED Dot Matrix-1.hex	2020/11/3 10:59	HEX File	624 KB
Project 4_ 5 x 5 LED Dot Matrix-2.hex	2020/11/3 10:59	HEX File	635 KB
Project 5_ Programmable Buttons-1.hex	2020/11/3 11:00	HEX File	624 KB
Project 5_ Programmable Buttons-2.hex	2020/11/3 11:00	HEX File	633 KB
Project 6_ Temperature Measurement-1....	2020/11/3 11:00	HEX File	625 KB
Project 6_ Temperature Measurement-2....	2020/11/3 11:01	HEX File	632 KB
Project 7_ Magnetic sensor-1.hex	2020/11/3 11:01	HEX File	624 KB
Project 7_ Magnetic sensor-2.hex	2020/11/3 11:02	HEX File	628 KB
Project 8_ Three-axis Acceleration Sensor...	2020/11/3 11:02	HEX File	626 KB
Project 8_ Three-axis Acceleration Sensor...	2020/11/3 11:03	HEX File	625 KB
Project 10_ Bluetooth Wireless Communi...	2020/11/3 11:03	HEX File	625 KB
Project 10_ Bluetooth Wireless Communi...	2020/11/3 11:15	HEX File	624 KB
Project 11_ LED Blink.hex	2020/11/3 11:18	HEX File	624 KB
Project 12_ Breath.hex	2020/11/3 11:18	HEX File	624 KB

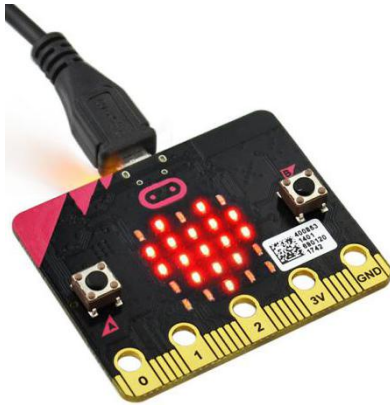


When the hex file is being copied, indicator on micro:bit will flash; after it is copied, the indicator will keep on.

#### Step 4: Run program

Upload code on micro:bit board and plug in power with USB cable.

LED dot matrix will display heartbeat pattern.



micro USB for power supply

Programming each time, MICROBIT drive will automatically eject and return. Micro:bit only receives hex files rather than save any file.

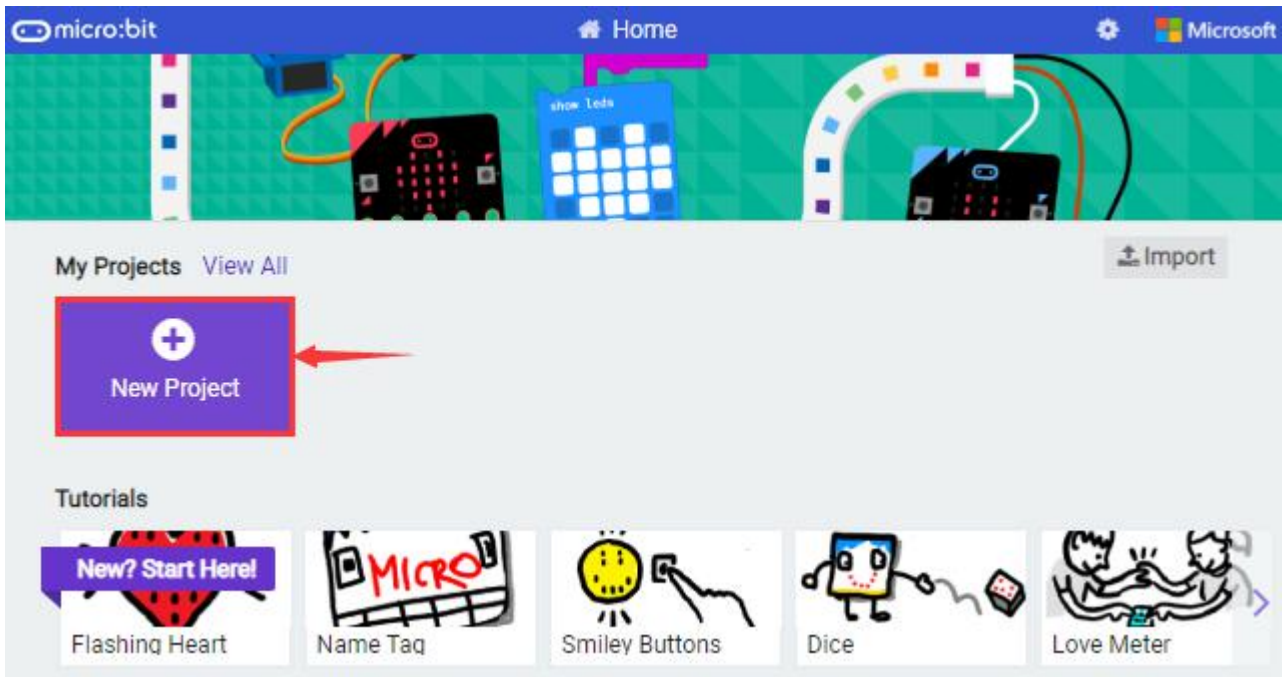
This chapter only shows you how to get started with micro:bit board. Python and JavaScript also support micro:bit board with the exception of Makecode.

<https://microbit.org/code/>

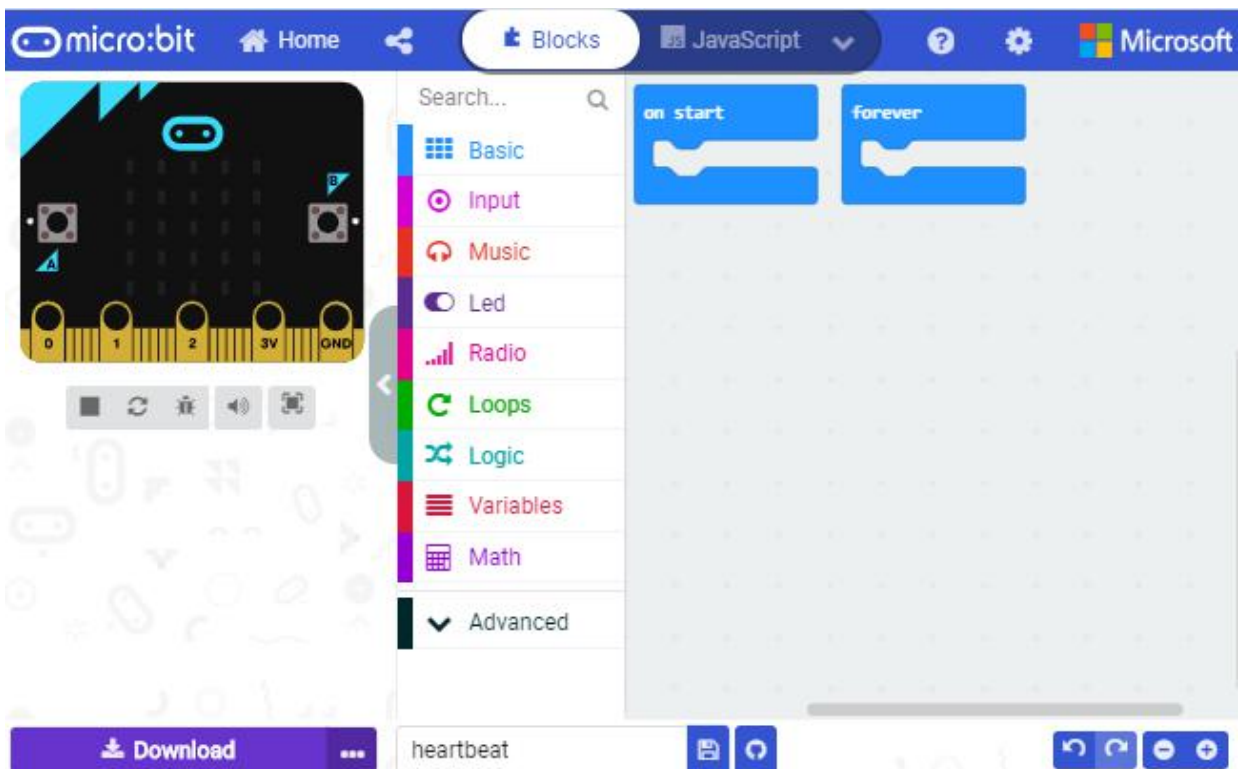
<https://microbit.org/projects/>

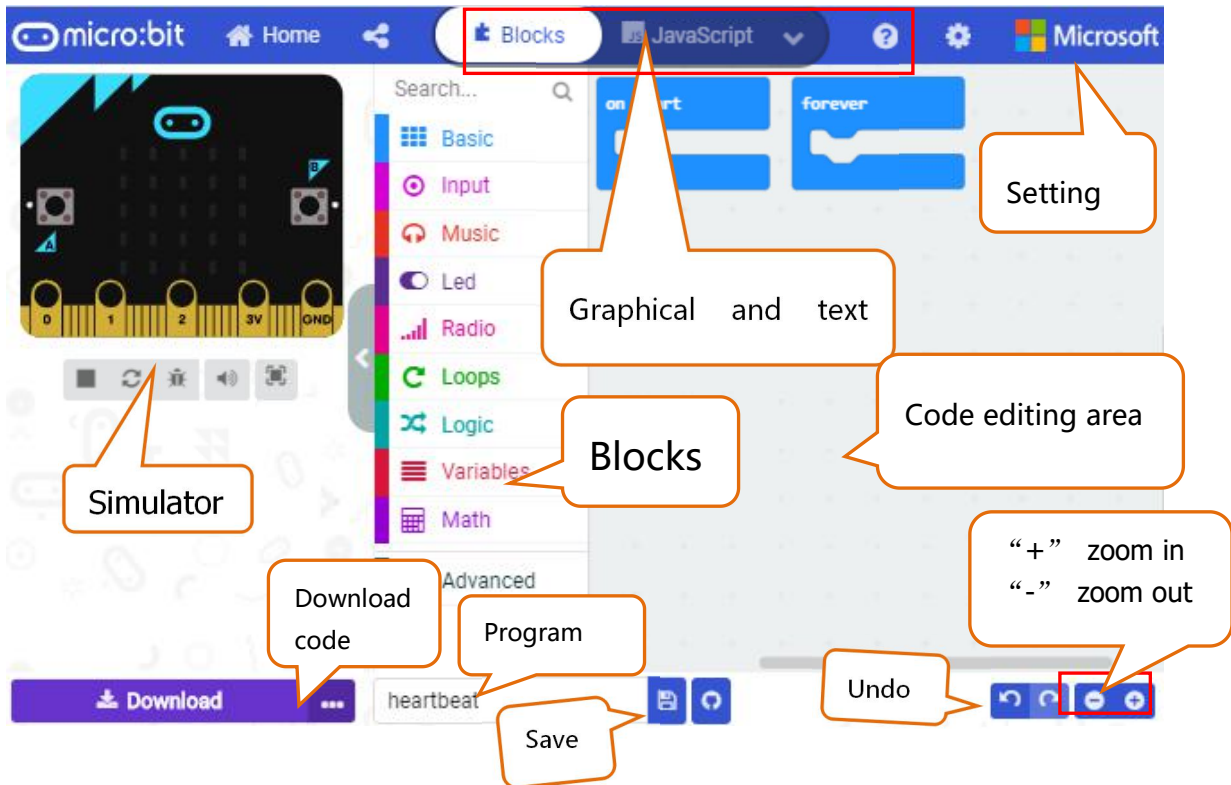
## 6.1 Makecode

Browse <https://makecode.microbit.org/> and enter Makecode online editor.



Click "New Project" , and input "heartbeat" , then enter Makecode editor, as shown below:





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

After power on or reset, “on start” means that command blocks in the code are only executed once, however, “forever” implies that code runs cyclically.

## 6.2 Quick Download

If you use Google Chrome on Android, ChromeOS, Linux, macOS and Windows 10 system, you will achieve the quick download.

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

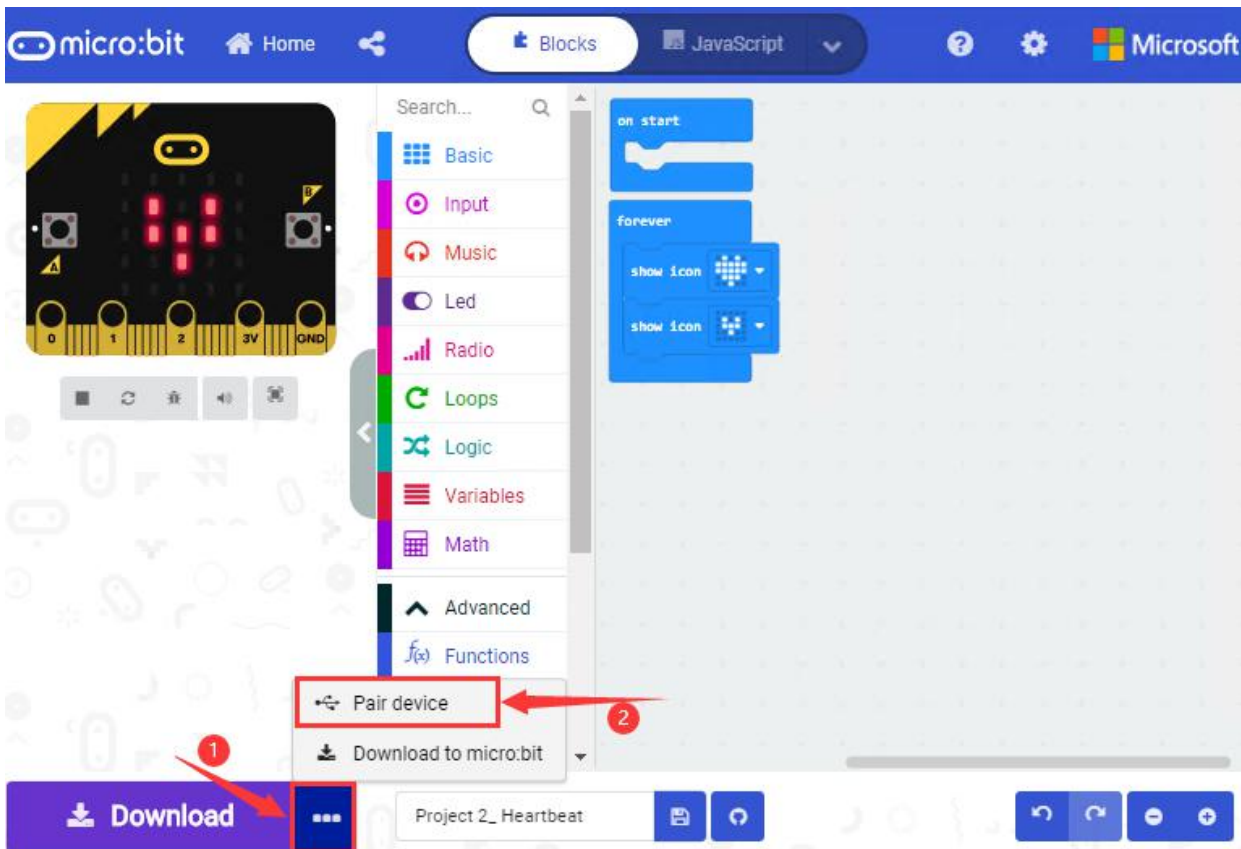


You can refer to the following steps to connect and pair device.

## Pairing device

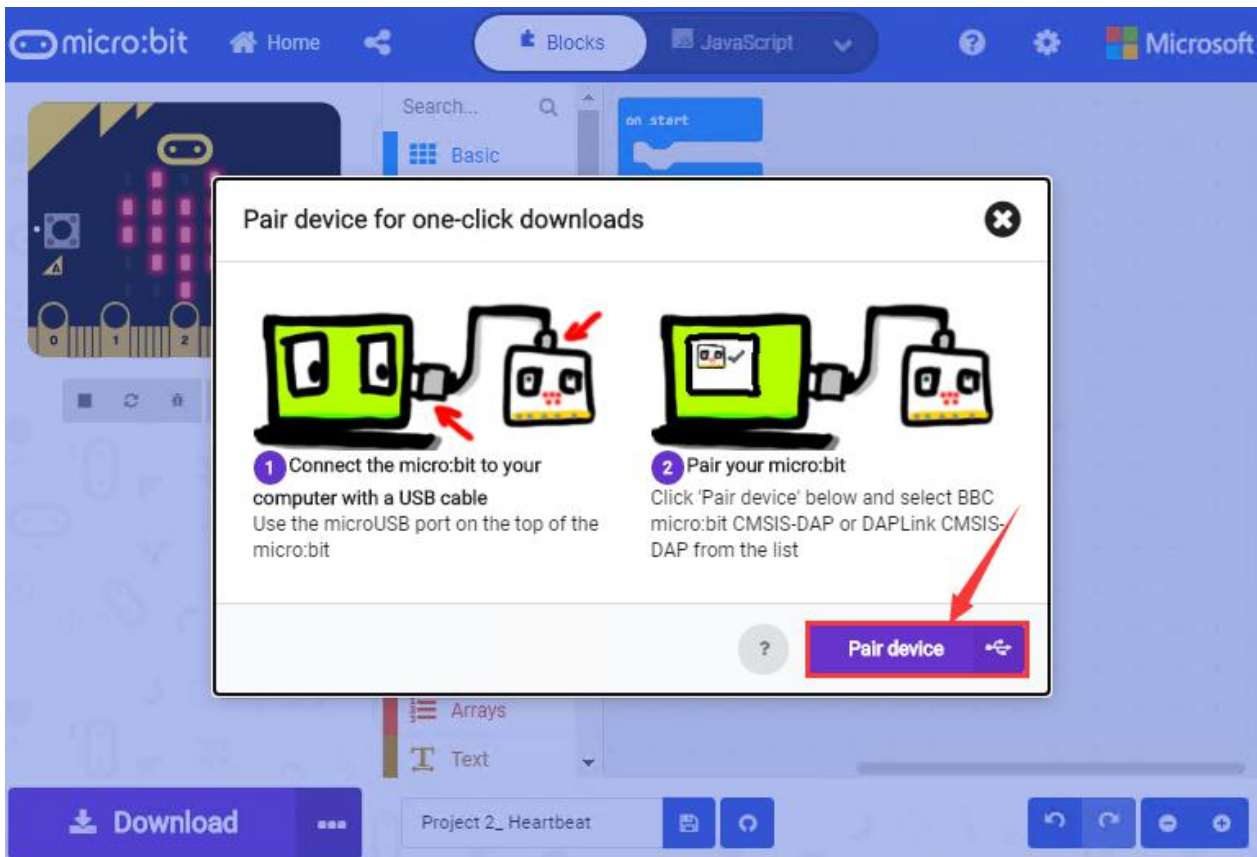
Interface micro:bit with computer using USB cable.

Click "..." beside "Download" and tap "Pair device" .



Continue to tap "Pair device"





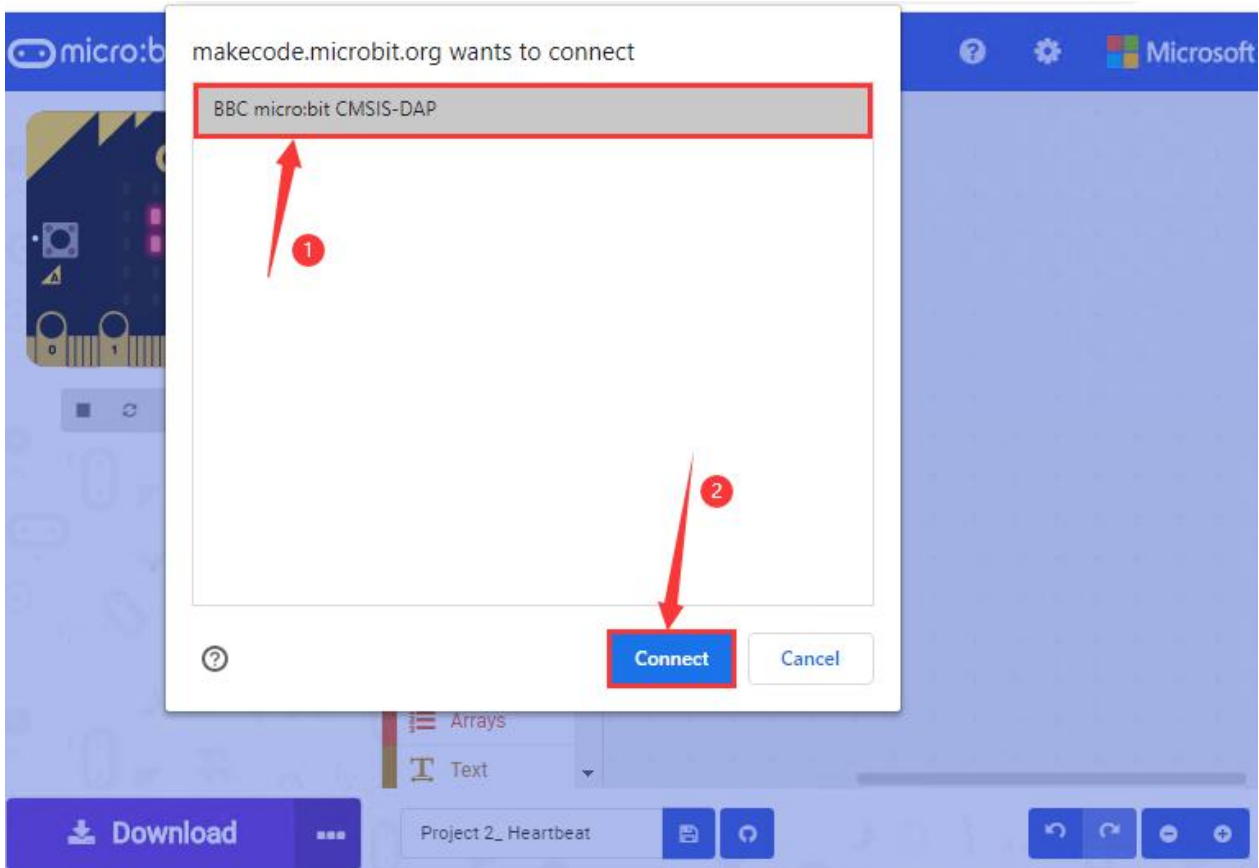
Then select the device you want to connect and tap “connect” in the window popping up.

If there is no device in the window, please refer to the following link:

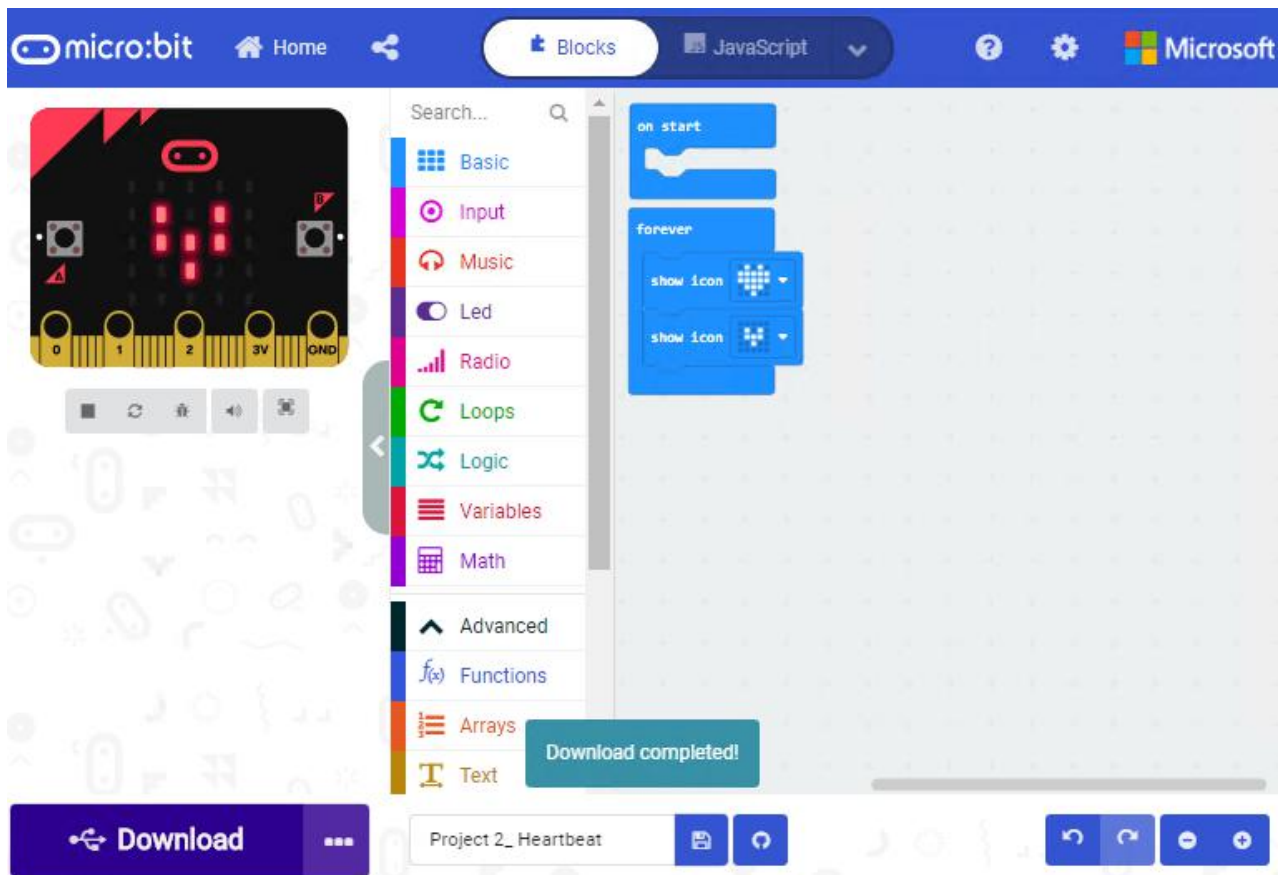
<https://makecode.microbit.org/device/usb/webusb/troubleshoot>

We also provide [6. Troubleshooting-WebUSB](#) in the resource link.

What’s more, if you don’t know how to update the firmware of micro:bit, refer to the link: <https://microbit.org/guide/firmware/> or browse folder [4.Upgrad the Firmware](#) we provide.



After connecting successfully, press buttons and download code to micro:bit.

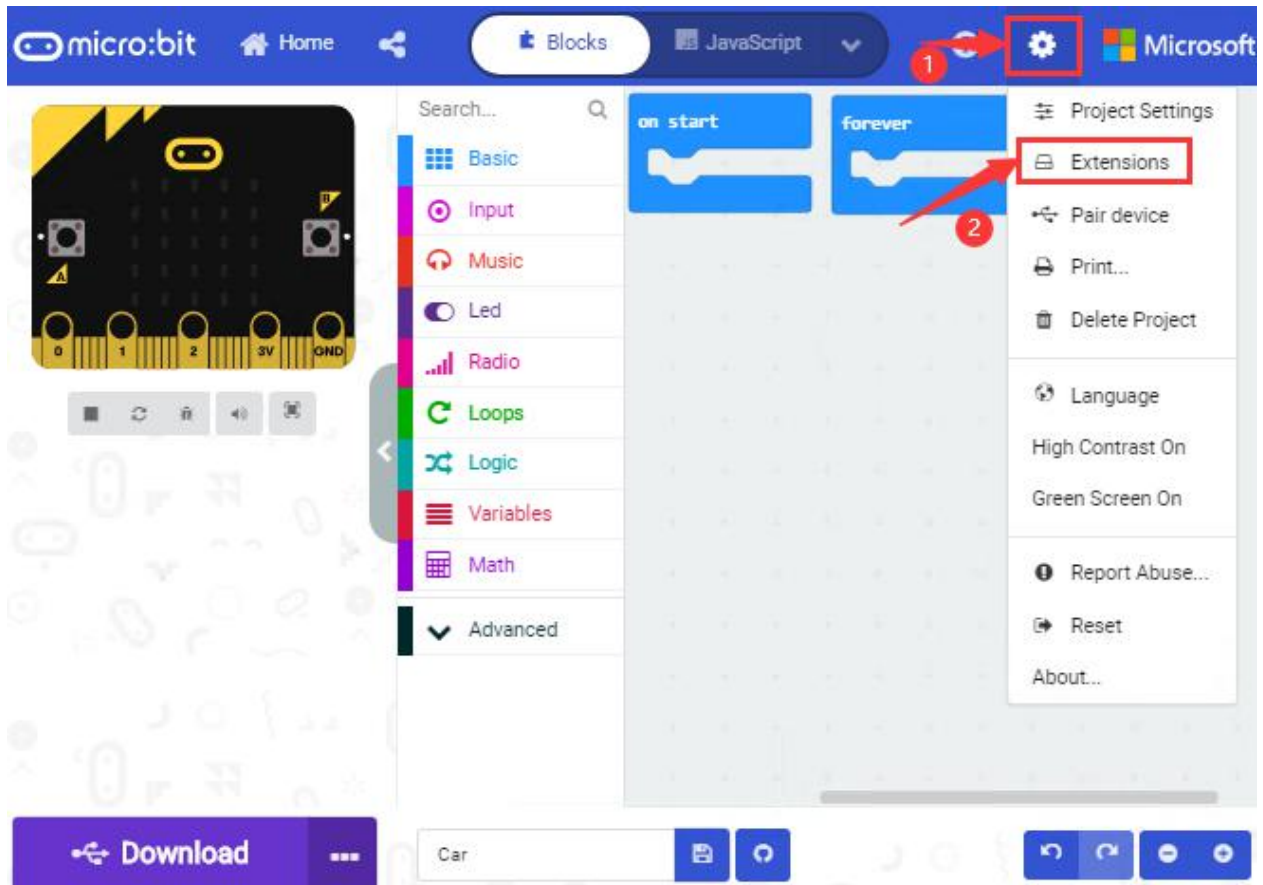


### 6.3 How to Import Extension Library on Makecode

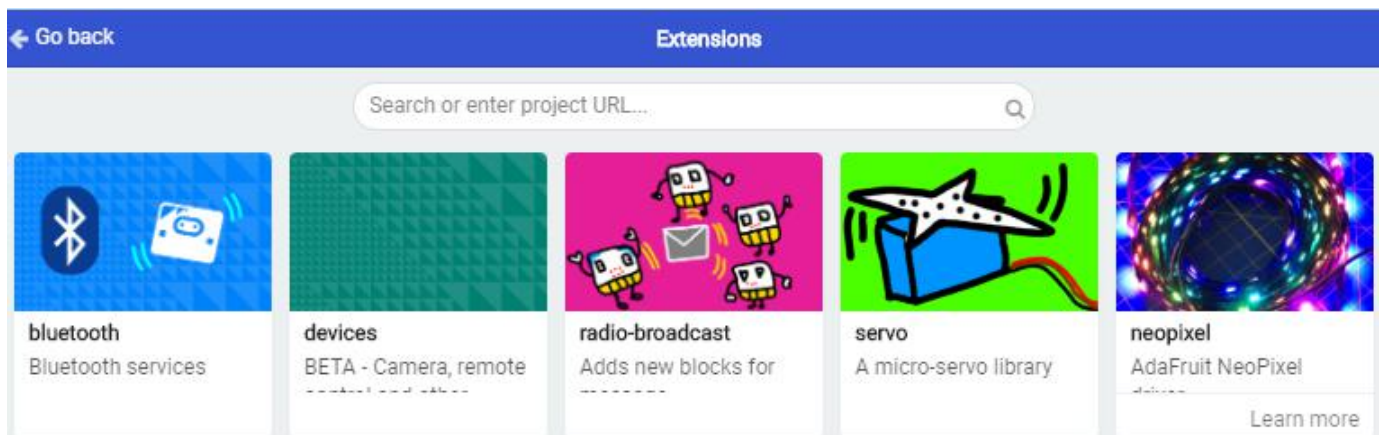
Next, we need to import turtle-bit extension library for further lessons.

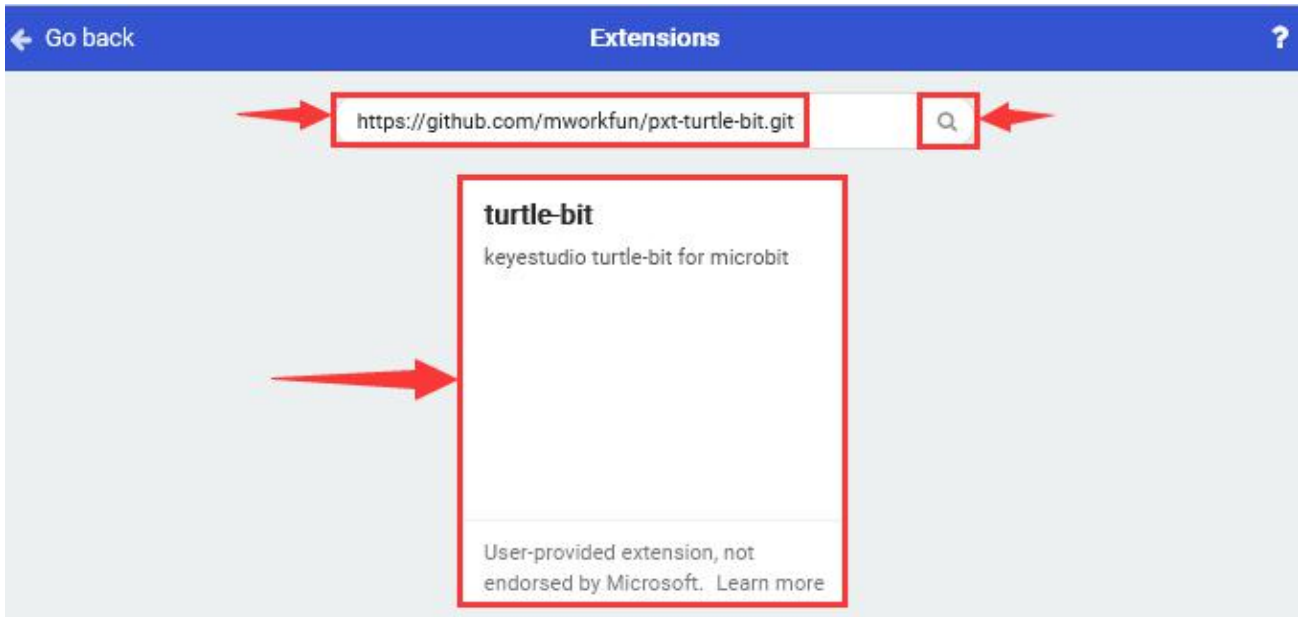
#### Add a Kit-bit extension library

Enter Makecode editor and click  icon and tap 

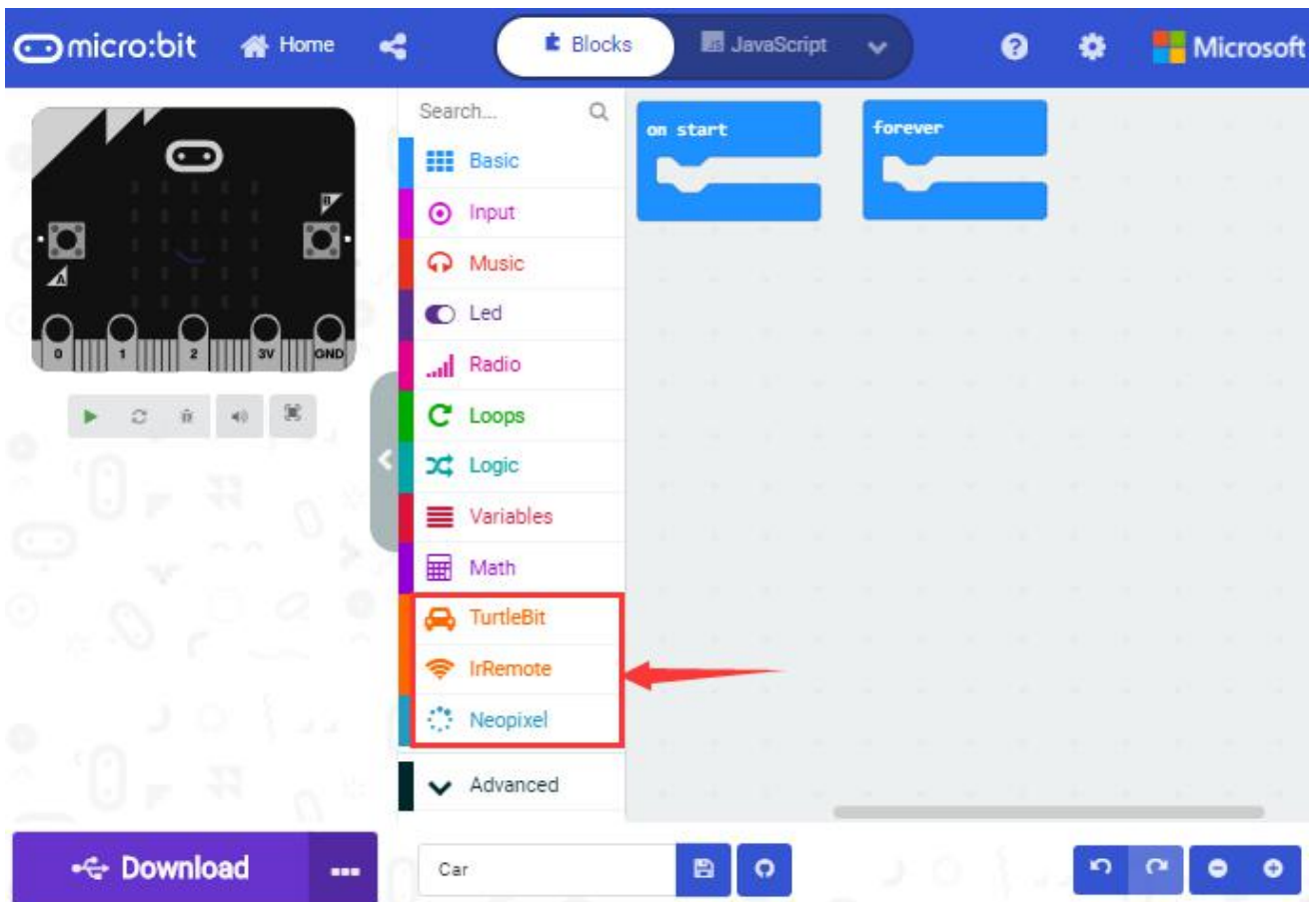


Copy <https://github.com/mworkfun/pxt-turtle-bit.git> in the searching box and search turtle-bit extension library.





After the installation, “turtle-bit” extension library will appear in the page





The image shows the Keyestudio IDE interface. On the left is a sidebar with a search bar and a list of extension categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, TurtleBit (highlighted), IrRemote, Neopixel, and Advanced. The main workspace displays a block-based code editor for the TurtleBit extension. The code is organized into sections: Motor, RGB-led, and Sensor. The Motor section contains blocks for 'Run\_forward' (speed: 0%), 'stop', and 'motor run' (LeftSide, Forward, speed: 0%). The RGB-led section contains blocks for 'LED brightness' (0), 'set left\_side RGBled red', 'set RGBled left\_side', and 'turn off all RGB-led'. The Sensor section contains blocks for 'Line Tracking' (Left), 'Ultrasonic', and 'Button'. A red arrow points from the 'Logic' category in the sidebar to the 'motor run' block in the code editor.

**Note: the extension library added is only valid to one project, therefore, it won't appear in other projects.**

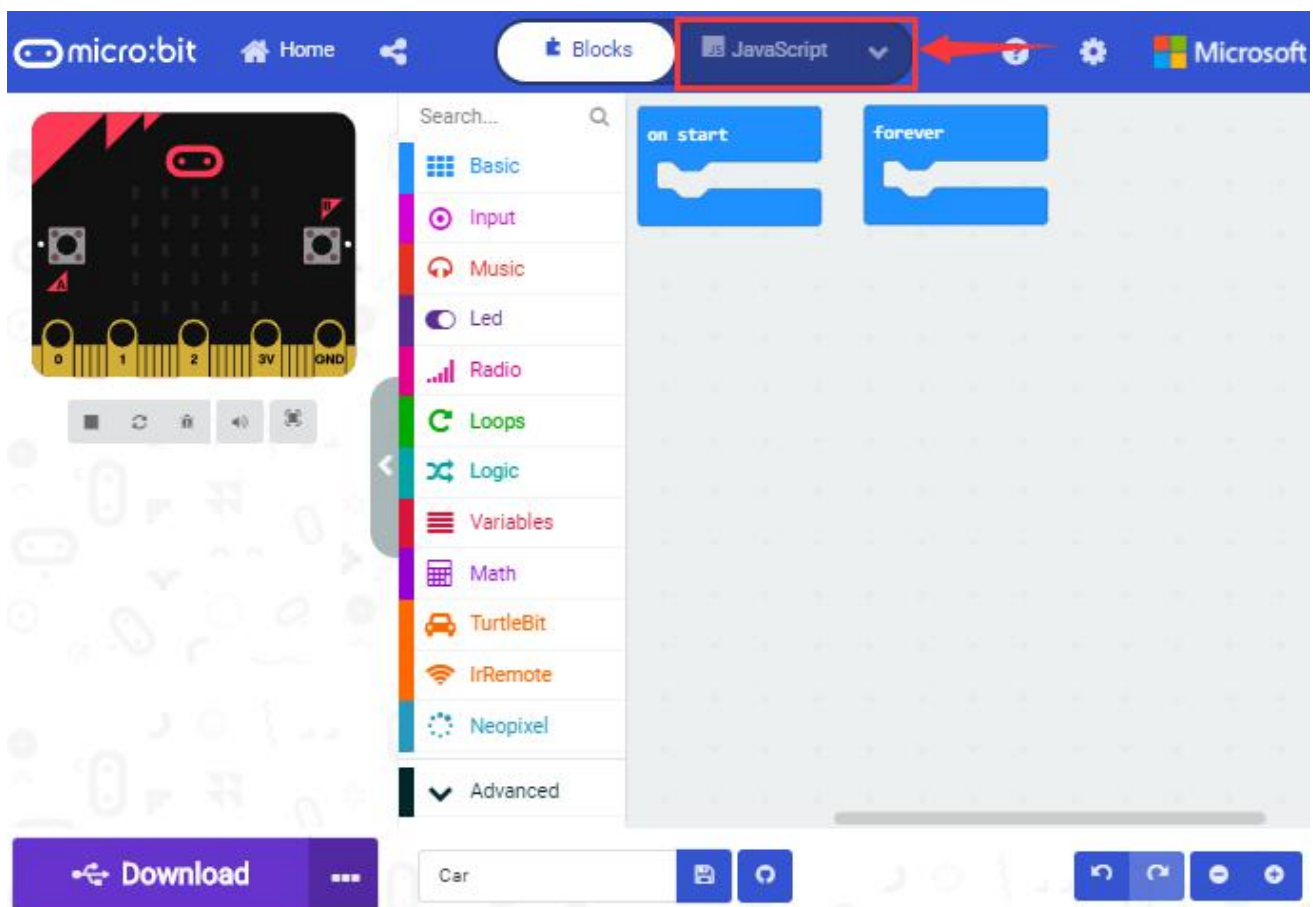


**You need to add the turtle-bit extension library again when creating new projects.**

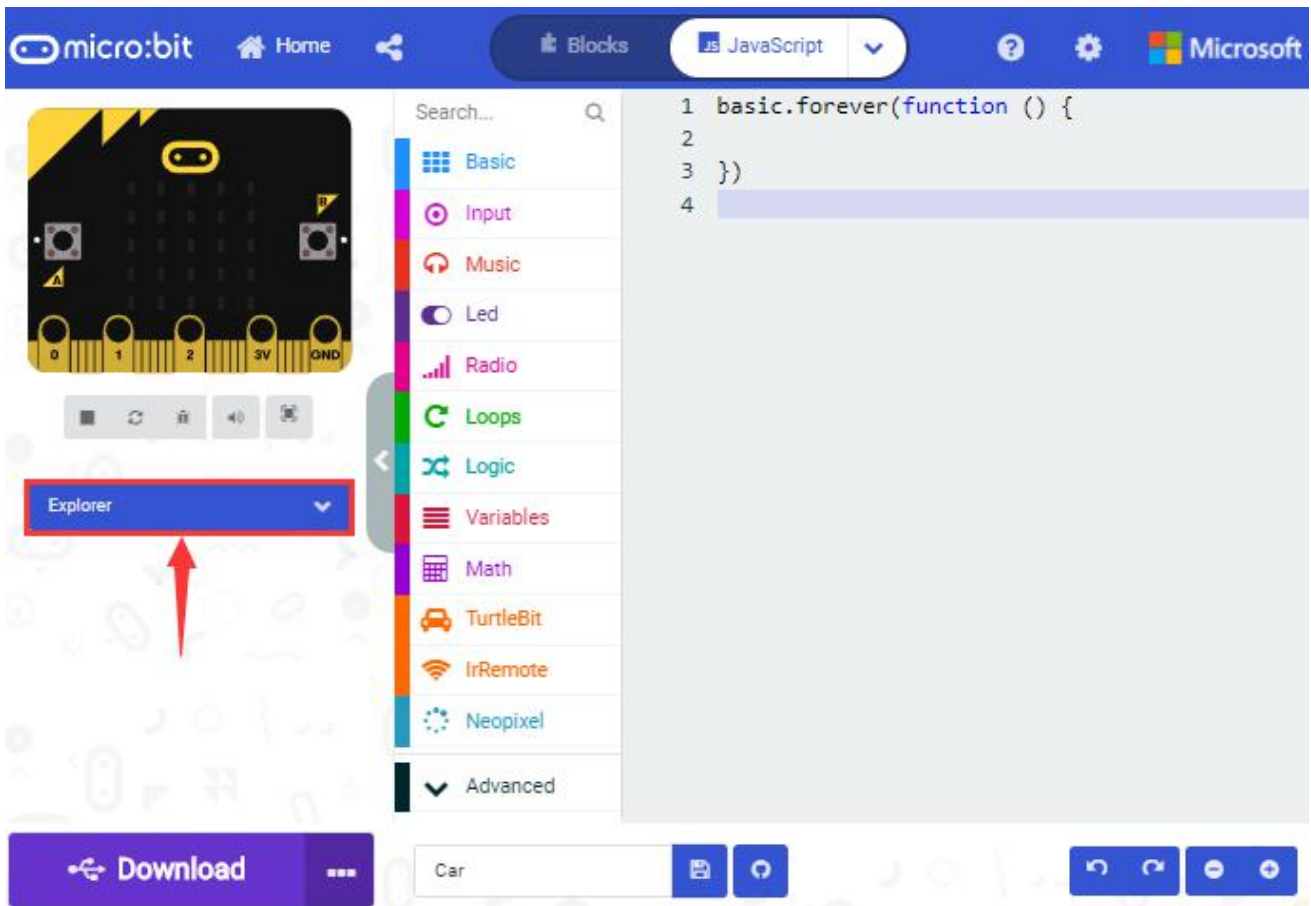
## Update or Delete Turtle-bit Extension Library



Refer to the following instruction please, if you intend to update or delete turtle-bit extension library.

Click "Js JavaScript" button to switch into text code

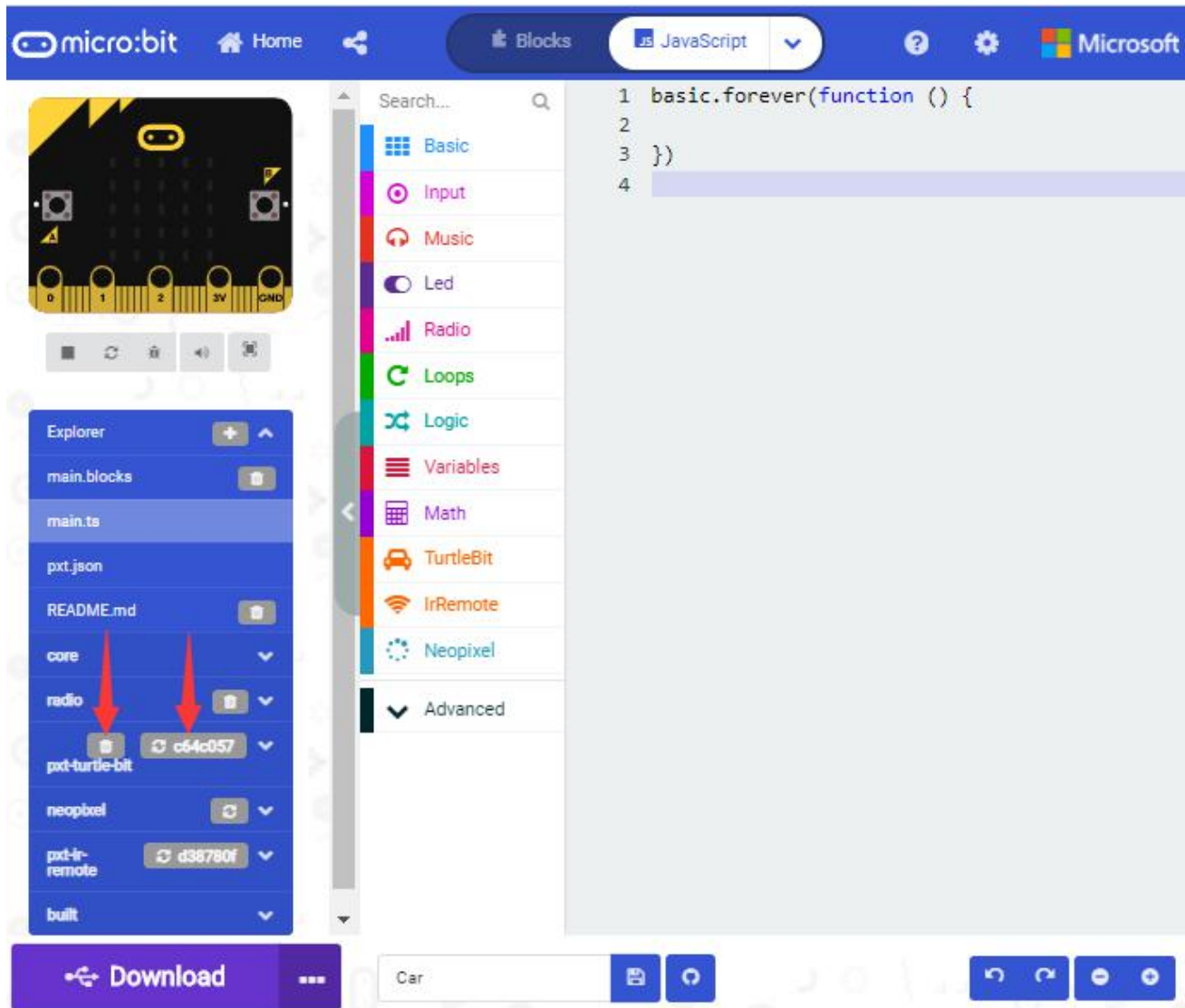


Click "Explorer" to get extension library .



Click "  " to delete turtle-bit extension program(TurtleBit IrRemote and Neopixel), next to tap "  " to update turtle-bit extension program.





## 6.4 Resources and Code

Download resources and code of tool package:

<https://fs.keyestudio.com/KS0426>

After the tool package is downloaded and unzipped, a file named KS0426 Micro:bit Mini Smart Robot Car V2 will be generated. It can be placed everywhere.

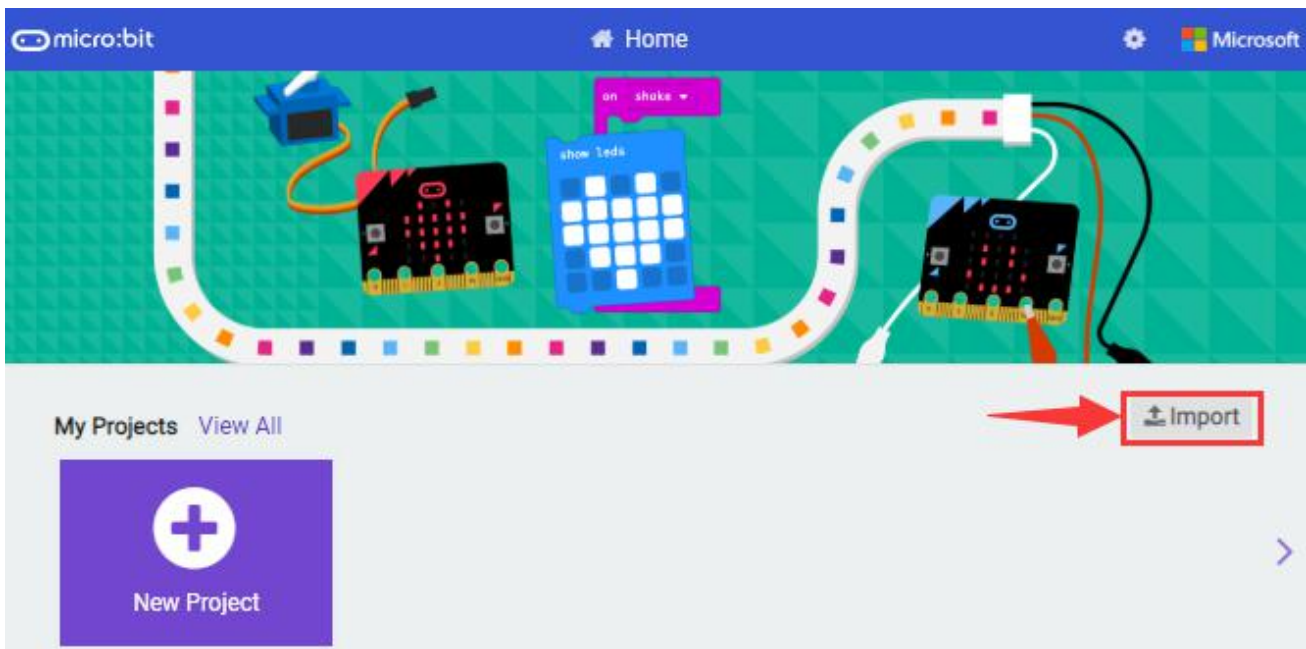


## 6.5 Import Code

We provide every program with hex file. You could import it directly or program in Makecode blocks area, therefore, the extension library must be added.[\(How to add extension?\)](#)

Next, we will take “heartbeat” as example to introduce how to import code

Open Makecode online editor on your computer

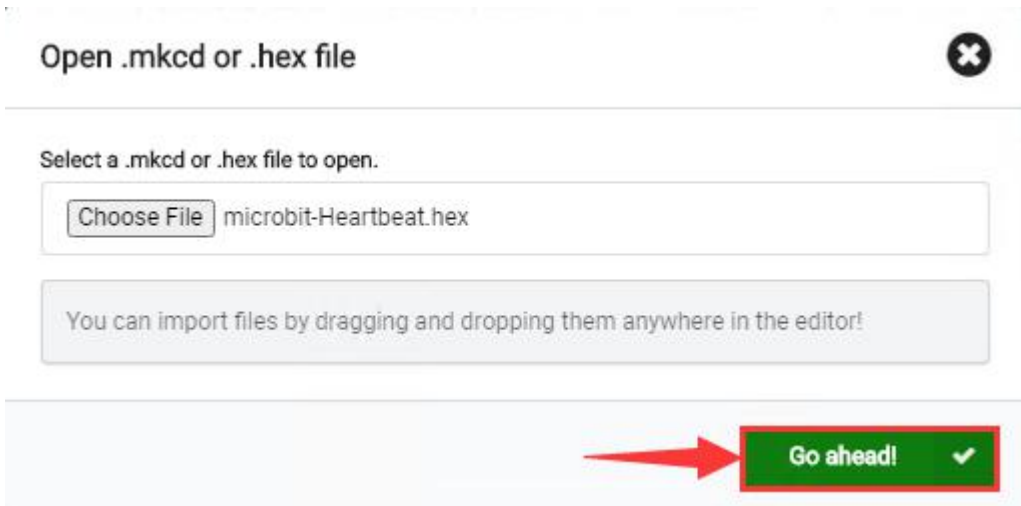
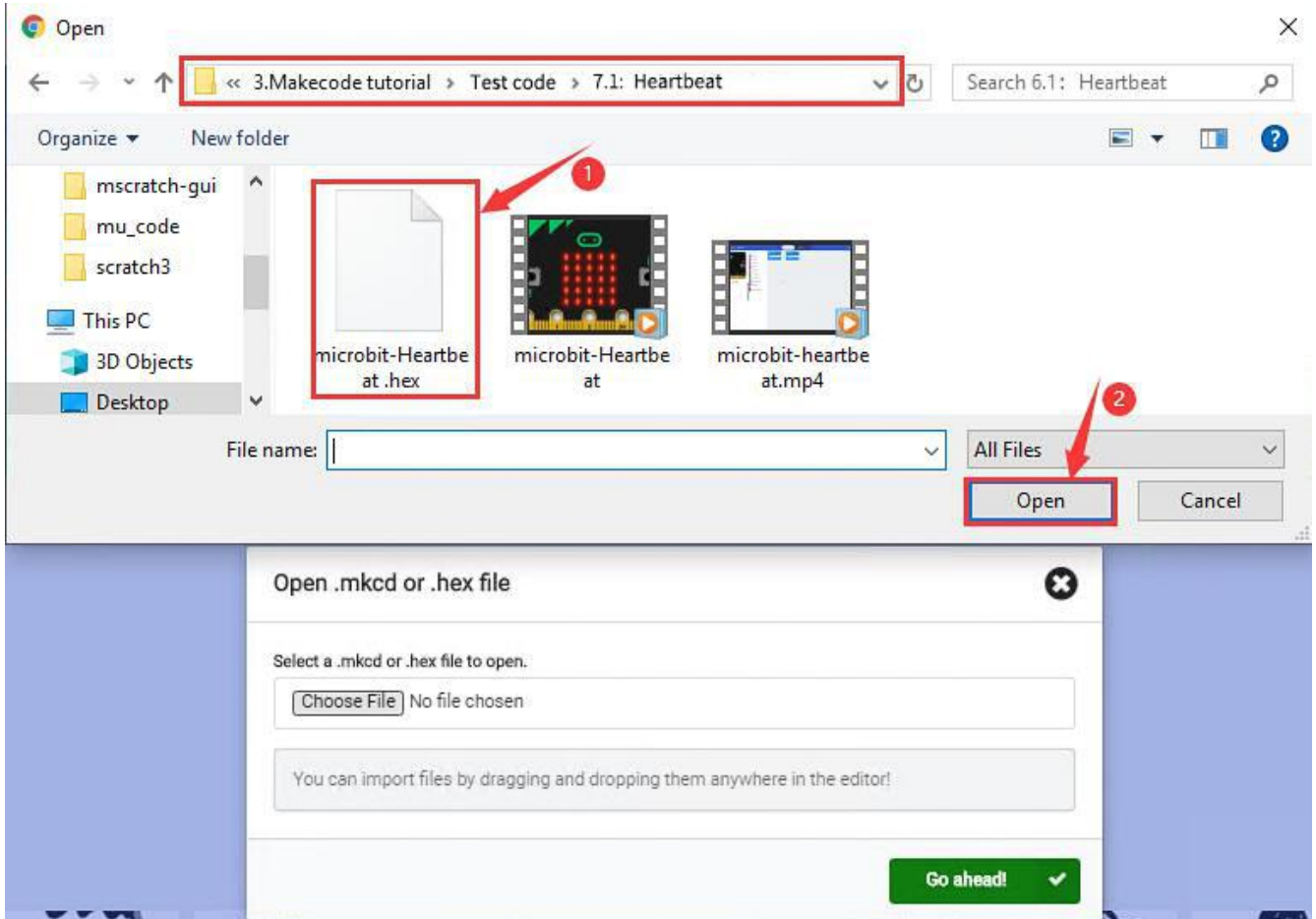


Click “Import” and “Import files”

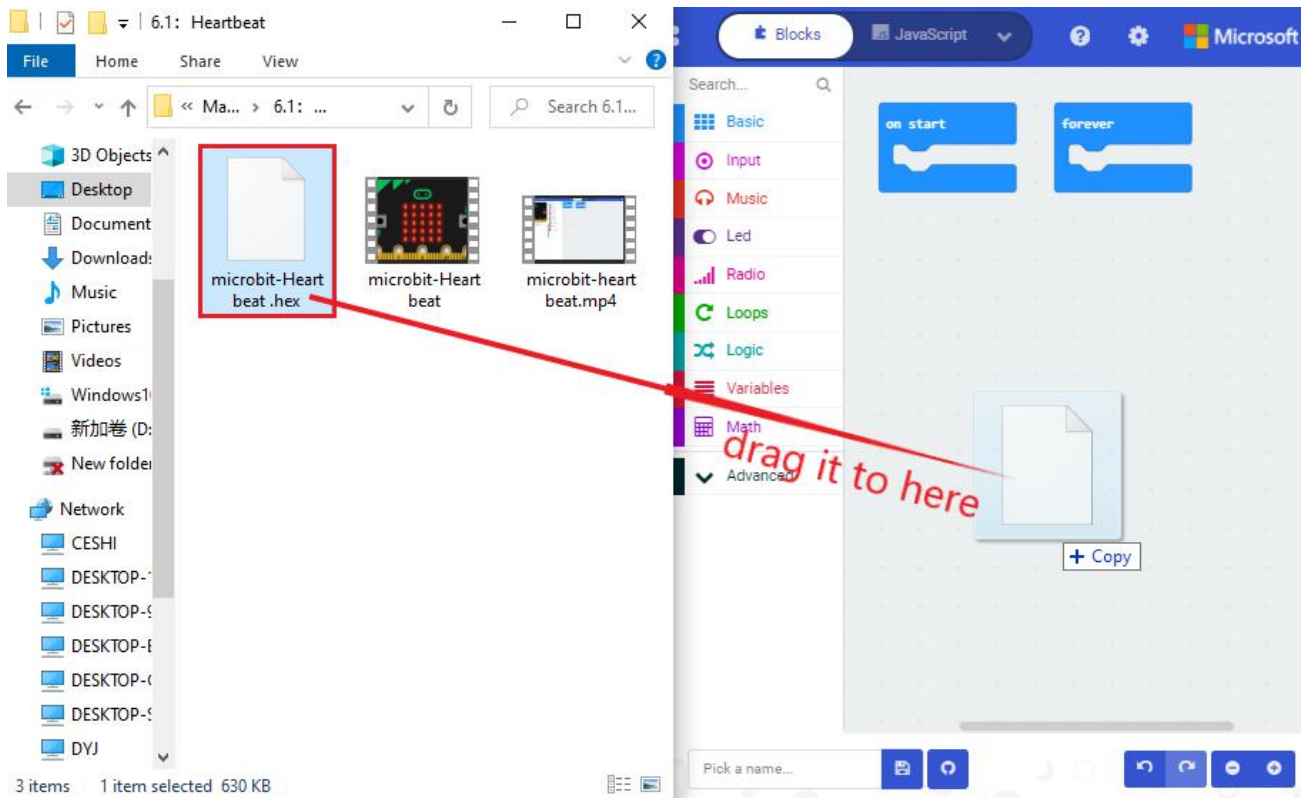


The screenshot shows the 'Import' dialog in Keyestudio. It has three main options: 'Import File...' (highlighted with a red box and arrow), 'Import URL...' (Open a shared project URL or GitHub repo), and 'Your GitHub Repo...' (Clone or create your own GitHub repository). Below these is a section titled 'Open .mkcd or .hex file' which contains a file selection field with a 'Choose File' button (also highlighted with a red box and arrow) and a 'No file chosen' status. A green 'Go ahead!' button with a checkmark is at the bottom right.

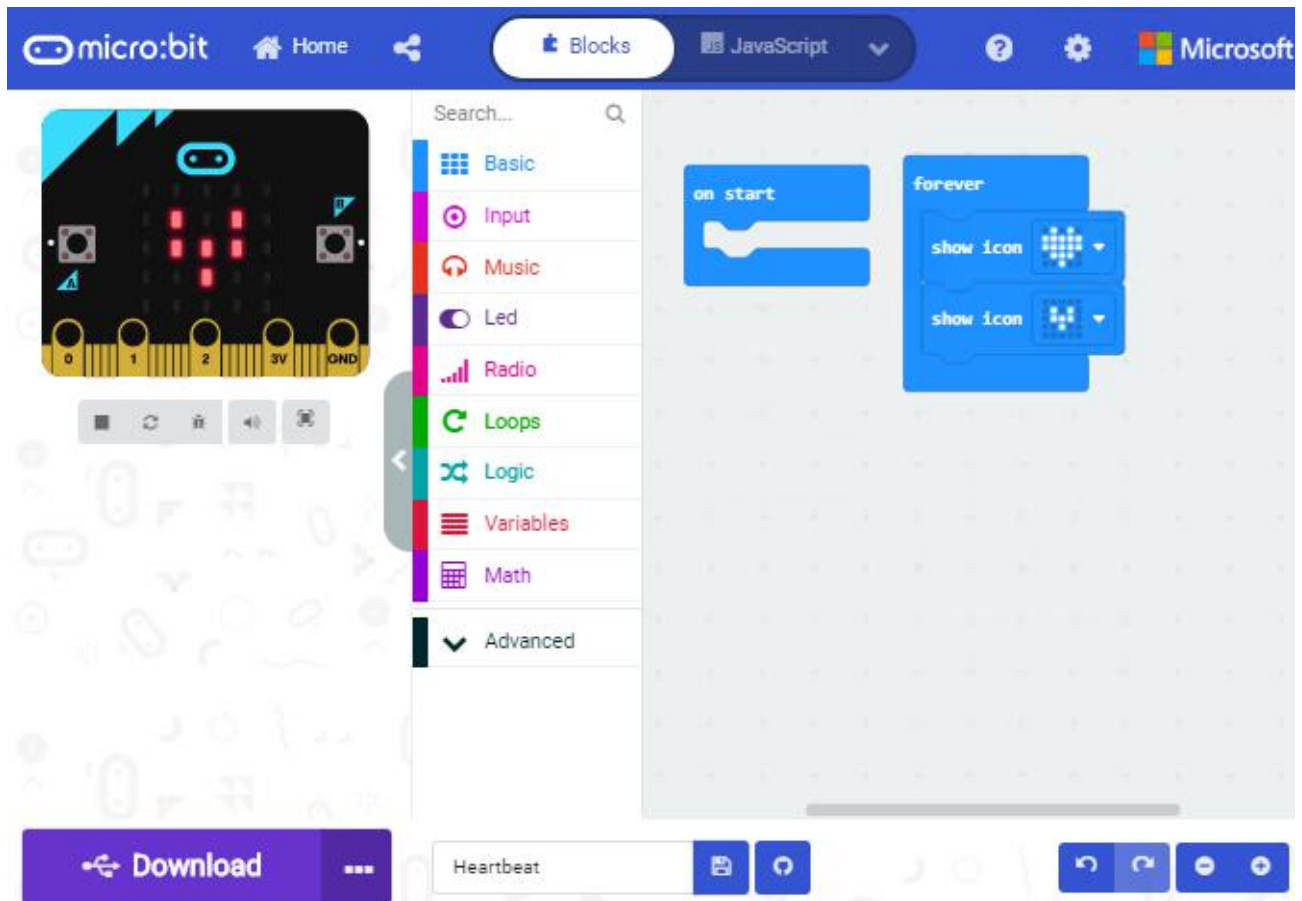
Choose file `../Test Code/7.1 : heartbeat/microbit-heartbeat.hex` , then tap "Go ahead"



In addition to the above method of importing code , you can also directly drag code into the Makecode compiler, as shown in the figure below:



The program is imported successfully after seconds



## 6.6 Install CoolTerm

If your computer system is Windows 7/8 rather than Windows 10, the device can't be paired in Google Chrome, as a result, the digital/analog signals can't be read.

Here, we need CoolTerm software to read data.

For the whole projects, we will use **CoolTerm** software.

Let's install it firstly.



CoolTerm program is used to read the serial communication.

Download CoolTerm program:

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

- (1) After the download, we need to install **CoolTerm program file**, the Window system is taken as an example.
- (2) Choose "win"
- (3) Unzip file and open it. **(also suitable for Mac and Linux system)**

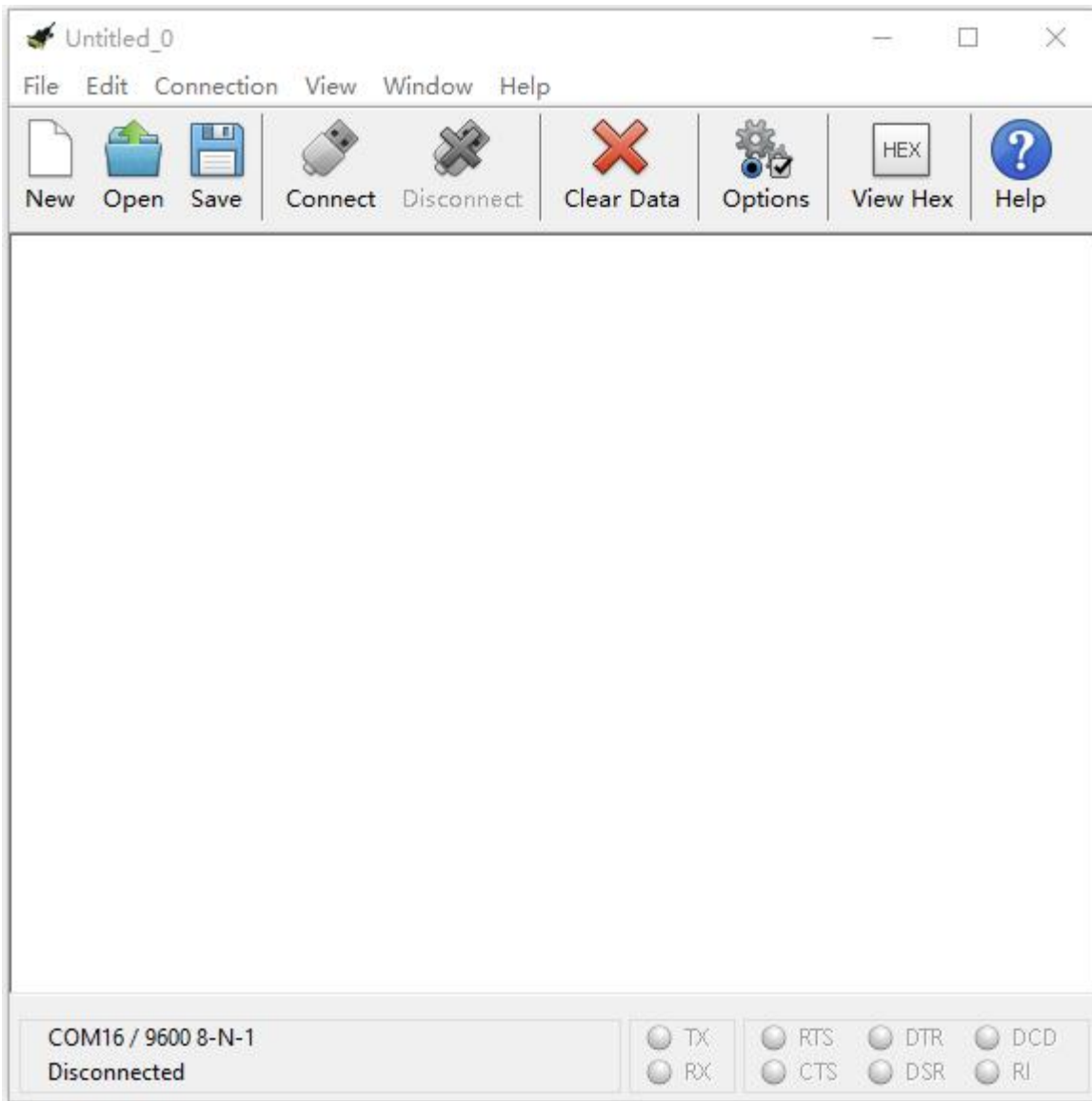


CoolTerm Libs	2020/4/21 11:20	File folder	
CoolTerm Resources	2020/4/21 11:20	File folder	
CoolTerm.exe	2019/5/17 22:56	Application	5,314 KB
msvcp120.dll	2019/4/3 14:33	Application extension	645 KB
msvcp140.dll	2019/4/3 14:33	Application extension	625 KB
msucr120.dll	2019/4/3 14:33	Application extension	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
XojoGUIFramework64.dll	2019/4/3 14:33	Application extension	30,801 KB

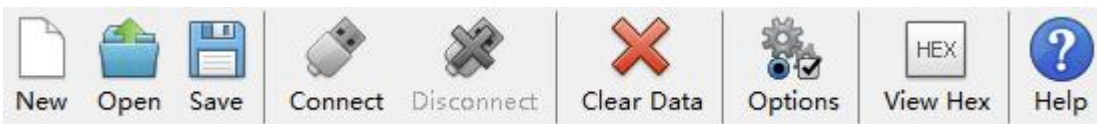
(4) Double-click CoolTerm.exe



Note: you have to install the driver of micro:bit and connect micro:bit to computer







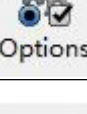
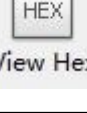
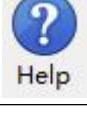


The functions of each button on the toolbar are listed below:





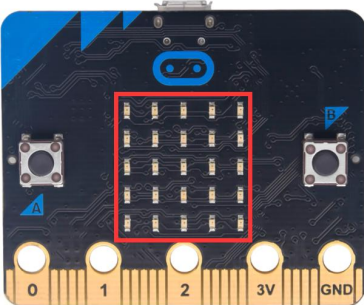


 New	Opens up a new Terminal
 Open	Opens a saved Connection
 Save	Saves the current Connection to disk
 Connect	Opens the Serial Connection
 Disconnect	Closes the Serial Connection
 Clear Data	Clears the Received Data
 Options	Opens the Connection Options Dialog
 View Hex	Displays the Terminal Data in Hexadecimal Format
 Help	Displays the Help Window



## 7. Projects

### 7.1: Heart beat



#### 1. Description:

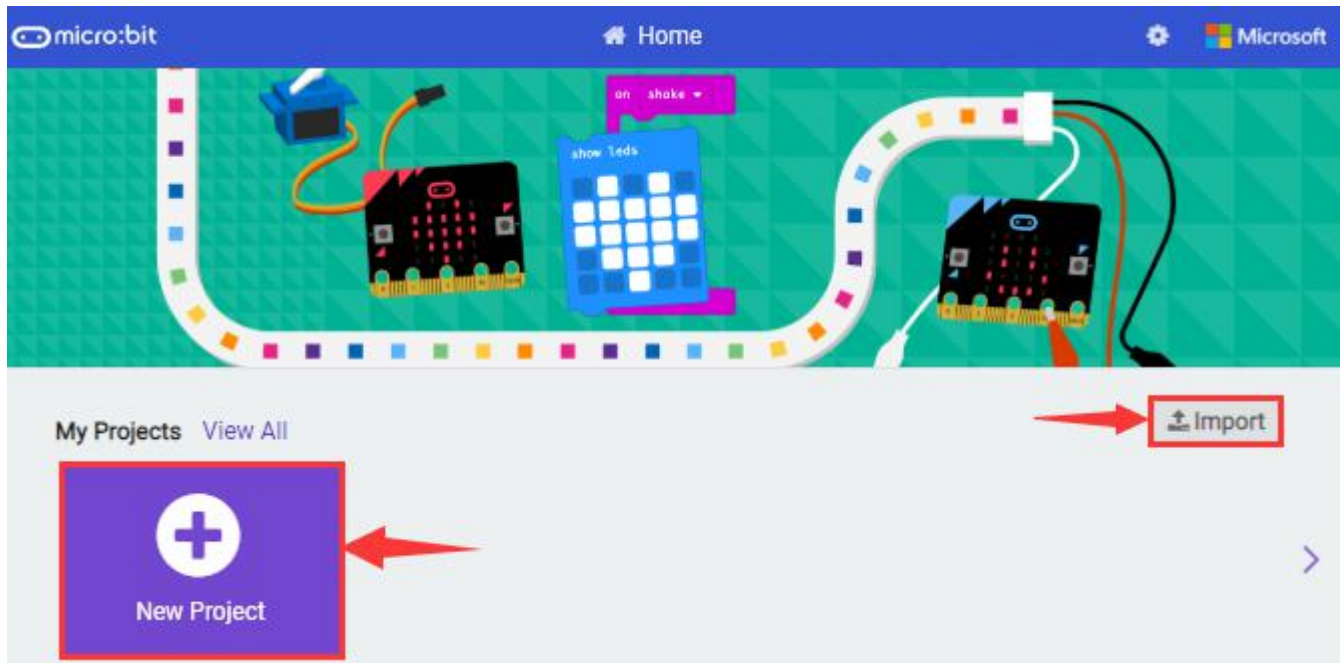
Prepare a Micro:bit board and USB cable. Next we will conduct a basic experiment that a heartbeat pattern flashes on micro:bit board.

#### 2. Experimental Preparation:

- (1) Connect micro:bit to computer with USB cable
- (2) Open online Makecode editor

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

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



### 3. Test Code:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.1: Heart beat	microbit-Heart beat.hex

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

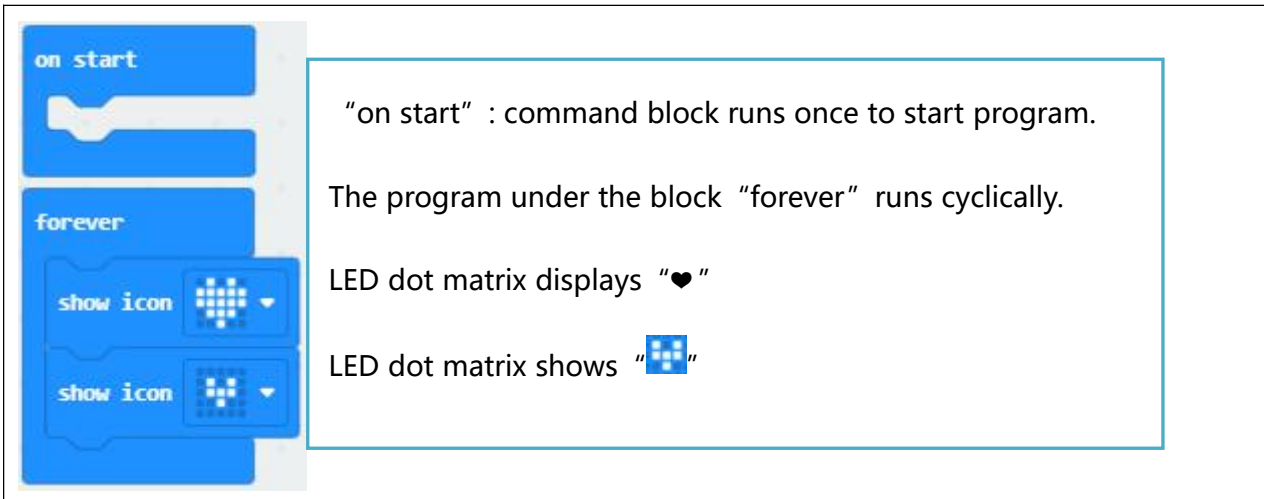
(1) Go to "Basic" → "show icon" .

Copy it again and place into "forever" block.

Click "♥" to select "🧩" .



## 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:



## 4. Test Result:

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

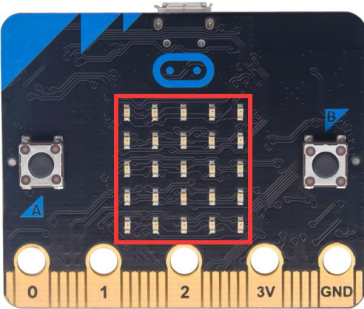


matrix will display  and  ceaselessly

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

If download unsuccessfully, disconnect micro:bit and reboot it

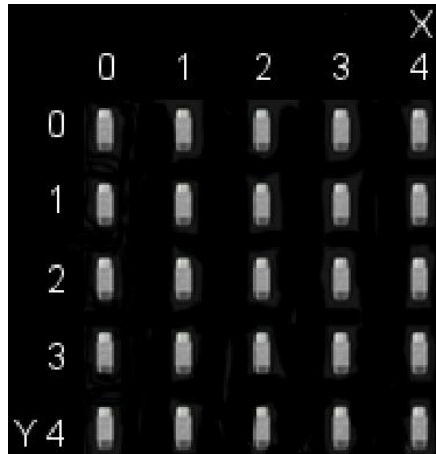
## 7.2: Light Up A Single LED



### 1. Description:

Micro:bit motherboard consists of 25 light-emitting diodes, 5 pcs in a group. They correspond to x and y axis. Then the 5\*5 matrix is formed. Moreover, every diode locates at the point of x and y axis.

Virtually, we could control an LED by setting coordinate points. For instance, set coordinate point (0, 0) to turn on the LED at row 1 and column 1; light up LED at the row 1 and column 3, we could set (2, 0) and so on.



## 2. Experimental Preparation:

- (1) Connect micro:bit to computer with USB cable.
- (2) Open online Makecode editor.

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

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

## 3. Test Code:

Import hex file:

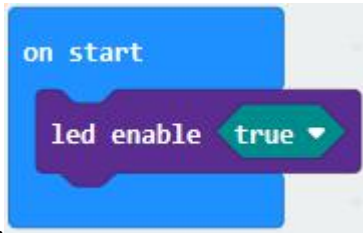
Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.2Light Up A Single LED	microbit-Light Up A Single LED.hex



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

(1) A. Click "Led" → "more" → "led enable false"

B. Put it into the "on start" block, 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"

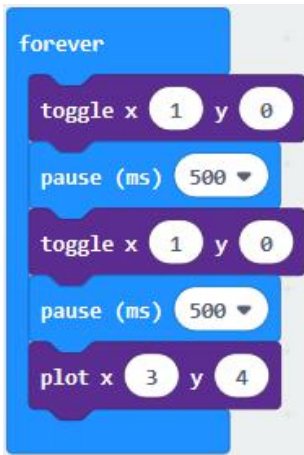


block.

\*\*\*\*\*

(1) A. Enter "Led" → "plot x 0 y 0"

B. Keep it beneath block "pause(ms)500" , then set to "plot x 3 y



4" .

\*\*\*\*\*

(2) Replicate "pause (ms) 500" once and keep it below the block "plot x3y4"





(3)

- A. Click "Led" → "unplot x 0 y 0" and set to "unplot x3 y 4" ;
- B. Lay down it beneath "pause (ms) 500" block
- C. Copy "pause (ms) 500" block once, and keep it below the "unplot x3 y

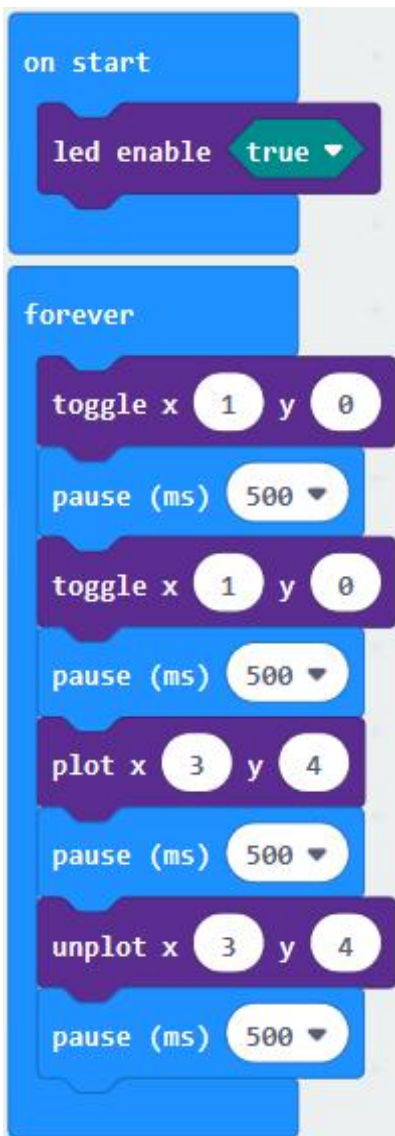
4" block.





## Complete Program:

Click "JavaScript" to switch into corresponding JavaScript code:



"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".



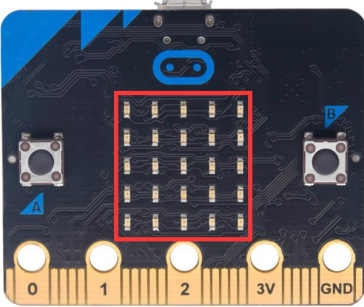
#### 4. Test Result:

Upload program and plug in micro:bit via USB port, the LED at coordinate point (1,0) flashes for 0.5s, then the LED at (3,4) blinks for 0.5s, alternately.

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



## 7.3: 5 x 5 LED Dot Matrix



### 1. Description:

Dot matrix gains popularity in our life, such as LED screen, bus station and the mini TV in the lift.

The dot matrix of Micro:bit board consists of 25 light emitting diodes. In previous lesson, we control LED of Micro:bit board to form patterns, numbers and character strings by setting the coordinate points. Moreover, we could adopt another way to complete the display of patterns, numbers and character strings.

### 2. Experimental Preparation:

- (1) Connect micro:bit to computer with USB cable
- (2) Open online Makecode editor

**Import Hex profile [\(How to import?\)](#) , or click “New Project” and drag blocks step by step.**



### 3. Test Code:

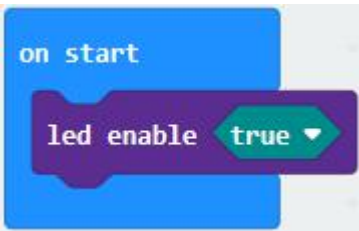
#### Code 1:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.3: 5 x 5 LED Dot Matrix/Code-1	microbit-Code-1.hex

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

(1) A. Enter "Led" → "more" → "led enable false"

B. Click the drop-down triangle button to select " true "



C. Combine it with "on start" block

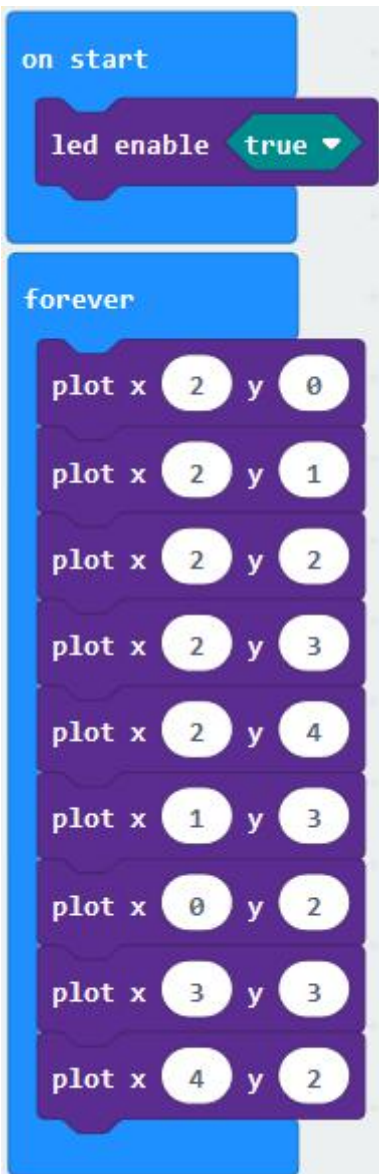
\*\*\*\*\*

(2) 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" , "x 4, y 2"



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



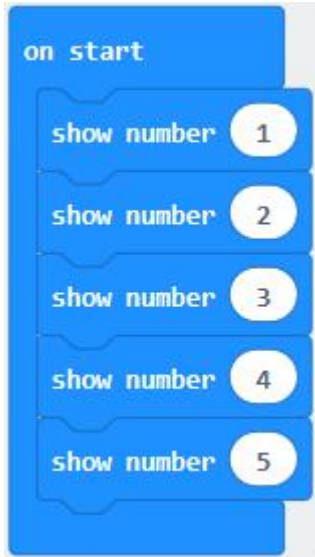
**Code 2:**

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.3: 5 x 5 LED Dot Matrix/Code-2	microbit-Code-2.hex

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

- (1) A. Enter "Basic" → "show number 0" block,
- B. Duplicate it for 4 times, then separately set to "show number 1" , "show number 2" , "show number 3" , "show number 4" , "show number 5" .





\*\*\*\*\*

(2) Click "Basic" → "show leds" , then put it into "forever" block, tick blue boxes to light LED and generate "↓" pattern.

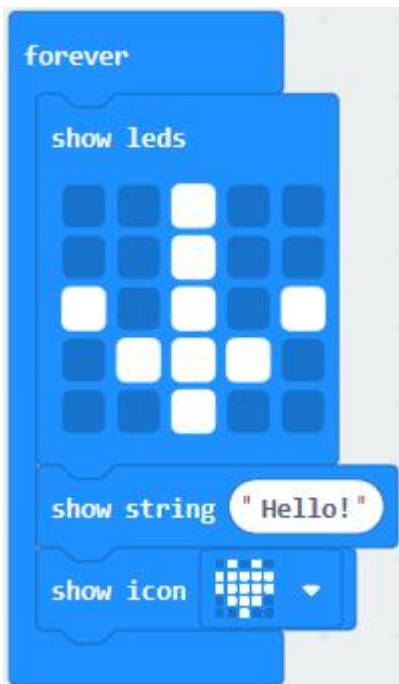


\*\*\*\*\*

(3) 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

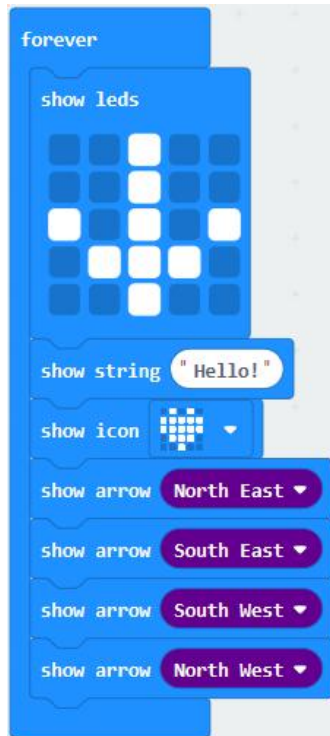


\*\*\*\*\*

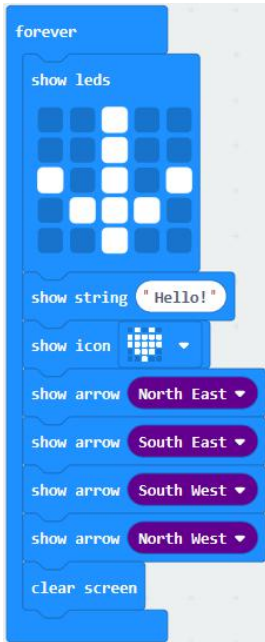
(4) A. Enter "Basic" → "show arrow North" ;



B. Leave it into "forever" block, replicate "show arrow North" for 3 times, respectively set to "North East" , "South East" , "South West" , "North West" .

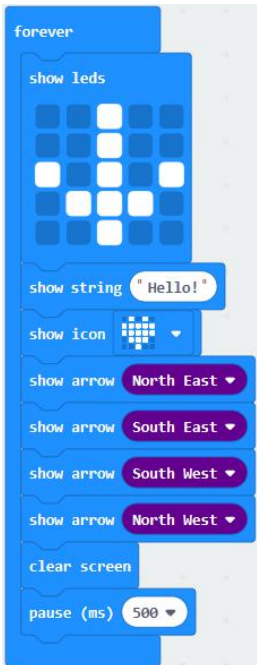


(5) Click "Basic" to get block "clear screen" then remain it below the block "show arrow North West"

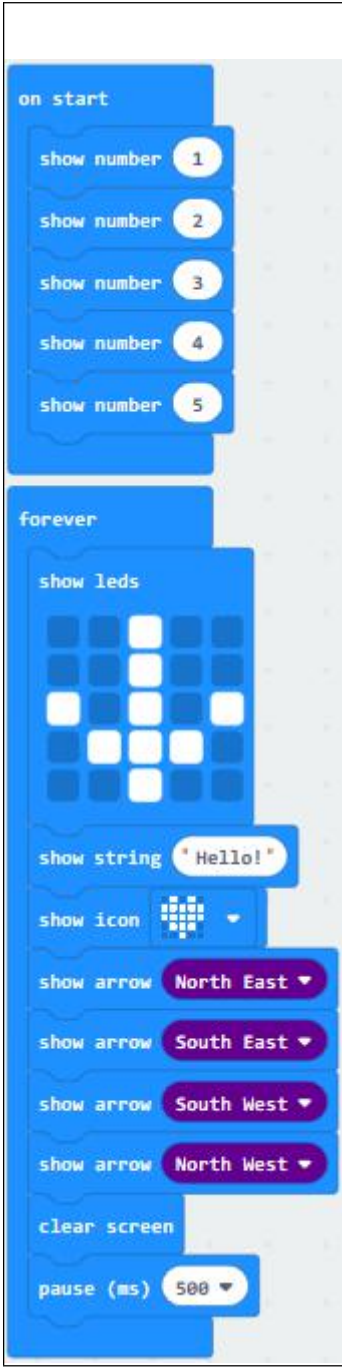


\*\*\*\*\*

(6) 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 displays "Hello!" in scroll way

"♥" 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" appears up on LED dot matrix

The "North West" is displayed on LED dot matrix

Clear the screen

Delay in 500ms

Click "JavaScript" to check the corresponding JavaScript code:



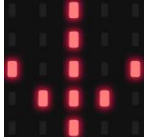
```

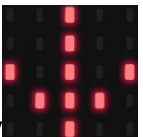
Blocks JavaScript
Search...
Basic
Input
Music
Led
Radio
Loops
Logic
Variables
Math
Advanced

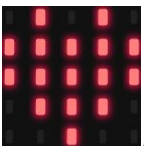
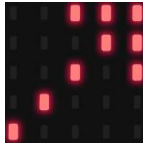



1 basic.showNumber(1)
2 basic.showNumber(2)
3 basic.showNumber(3)
4 basic.showNumber(4)
5 basic.showNumber(5)
6 basic.forever(function () {
7   basic.showLeds(`
8     . . # . .
9     . . # . .
10    # . # . #
11    . # # # .
12    . . # . .
13    `)
14   basic.showString("Hello!")
15   basic.showIcon(IconNames.Heart)
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 Result:

Upload code 1 and plug in micro:bit via USB cable , we will see the  icon.

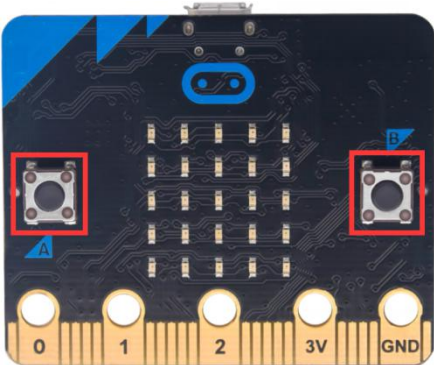
Upload code 2 and plug in micro:bit via USB cable. Micro: bit starts showing number 1, 2, 3, 4, and 5, then cyclically display , "Hello!" ,

, , ,  and  patterns.



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

## 7.4: Programmable Buttons



### 1. Description:

The button can control the on and off of the circuit. The button is attached to the circuit. The circuit is disconnected when the button is not pressed. The circuit is connected as soon as it is pressed, but it is disconnected after being released.

Both ends of button are like two mountains. There is a river in between.

The internal metal piece connect the two sides to let the current pass, just like building a bridge to connect the two mountains.

Micro:bit board has three buttons, the reset button on the back and two programmable buttons on the front. By pressing these buttons, the corresponding characters will be displayed on dot matrix.



## 2. Experimental Preparation:

- (1) Connect micro:bit to computer with USB cable
- (2) 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.

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.4: Programmable Buttons/Code-1	microbit-Code-1.hex

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

You could edit code step by step in the editing area.

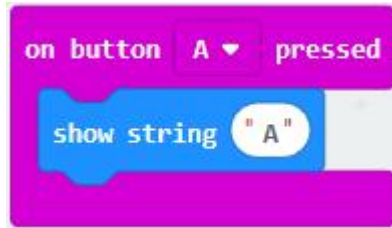
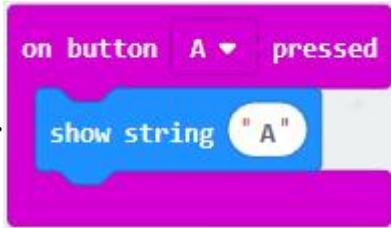
(1) A. Click “Basic” → “show string” ;

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



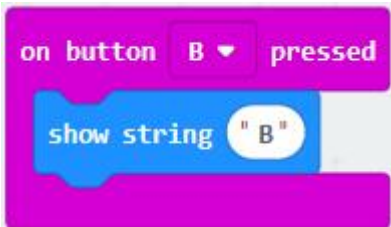


"A" .

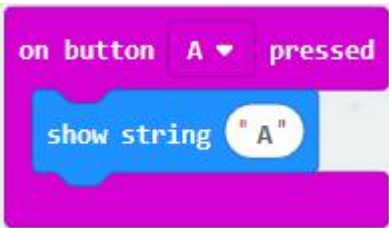


(2) Copy code string " A " once, tap the drop-down button " A " to select " B " and modify character " A " into

"B" .



\*\*\*\*\*

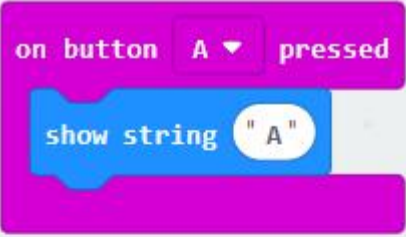
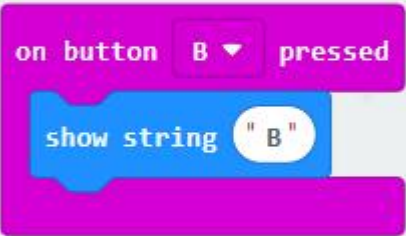
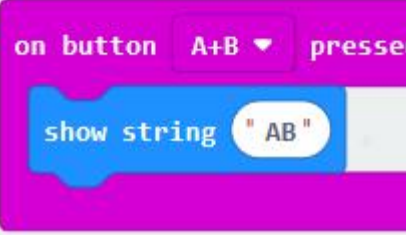


(3) Copy " A " once, and set to "on button A+B pressed" and "show string "AB"





## Complete Code:

	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"

Click "JavaScript" to switch into the corresponding 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 })
10
```



### Code 2:

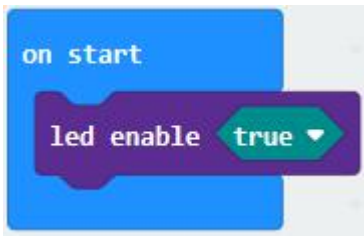
Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.4: Programmable Buttons/Code-2	microbit-Code-2.hex

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

You could edit code step by step in the editing area.

(1) A. Click "Led" → "more" → "led enable false" ,

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

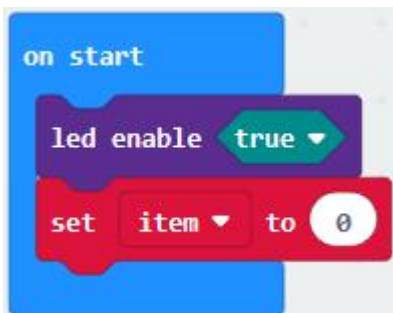


"true"

\*\*\*\*\*

(2) 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





(3) 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



\*\*\*\*\*



(4) 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"

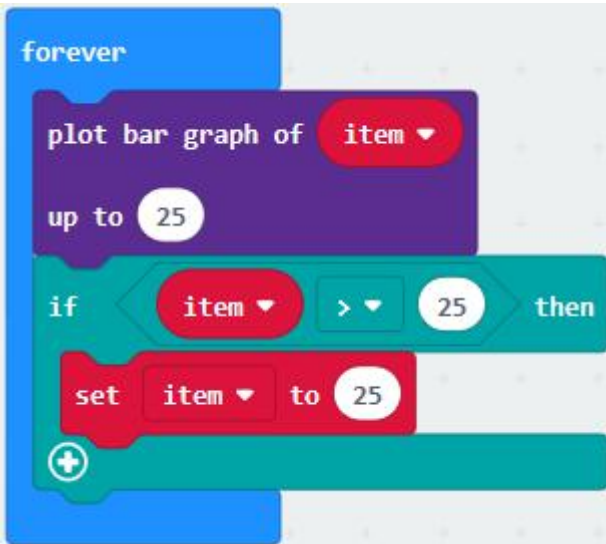
B. Keep it into "forever" block

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



\*\*\*\*\*

- (6) 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.



\*\*\*\*\*



```
if <item > 25 then  
  set item to 25
```

(7) A. Replicate code string once

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

```
if <item < 0 then  
  set item to 25
```

C. Leave it beneath code string.

```
forever  
  plot bar graph of item  
  up to 25  
  if <item > 25 then  
    set item to 25  
  if <item < 0 then  
    set item to 0
```



## Complete Program:

```
on start
  led enable true
  set item to 0

on button A pressed
  change item by 5

on button B pressed
  change item by -5

forever
  plot bar graph of item
  up to 25
  if item > 25 then
    set item to 25
  if item < 0 then
    set item to 0
```

“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 up to 25 LEDs

If item is greater than 25

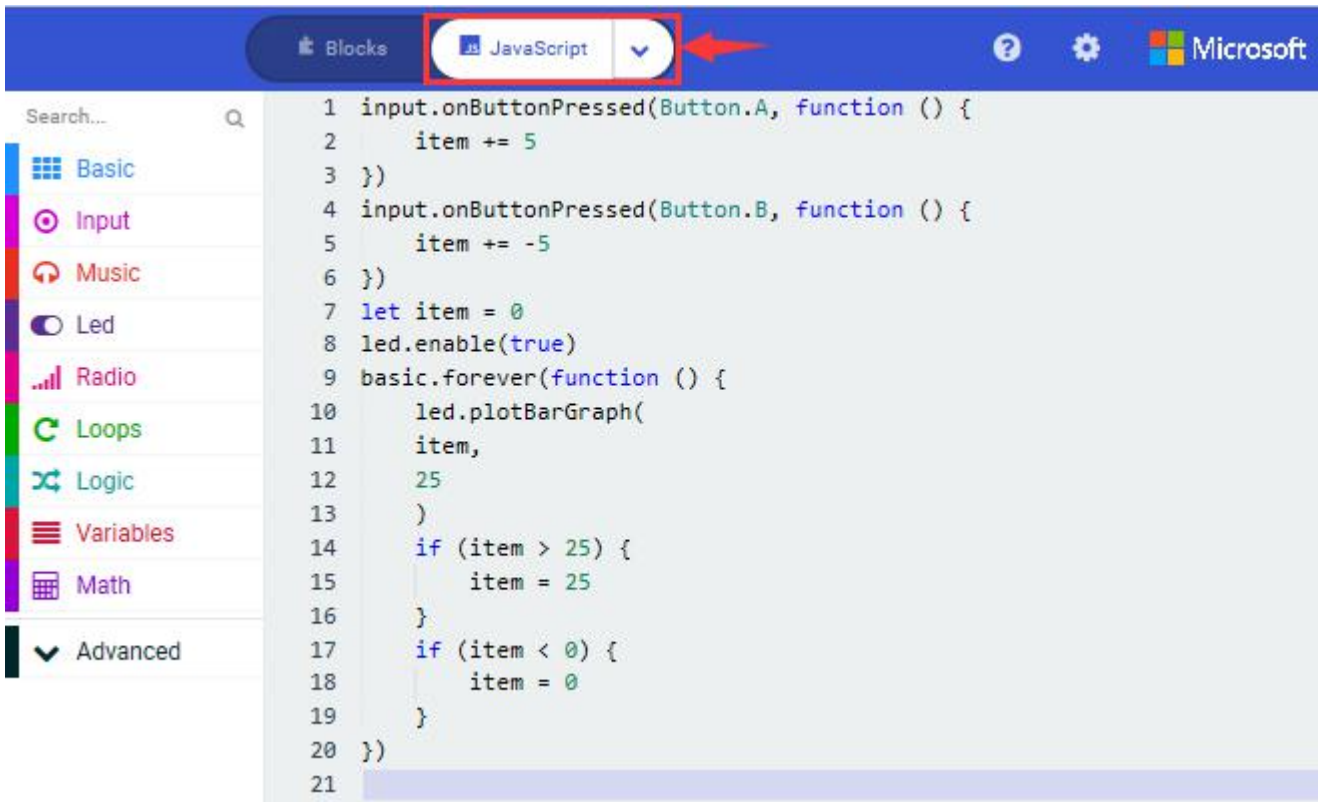
Then set item to 25

If item is less than 0

Then set item to 0



Click "JavaScript" to switch into JavaScript code:



#### 4. Test Result:

Upload code 1 and plug in micro:bit via USB cable, 5×5 LED dot matrix will show "A" if button A is pressed, in case that button B is pressed, "B" will appear. So will micro:bit show "AB" if you press A and B buttons simultaneously.

Upload code 2 and plug in board via USB cable. A row of luminous LEDs are added if button A is pressed, when B pressed, a row of luminous LEDs are deducted.





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

## 7.5: Temperature Measurement

### 1. Description:

Micro:bit main board doesn't come with temperature sensor actually, but detect temperature through built-in temperature of NFR51822 chip. Thereby, the detected temperature is more close to chip's temperature.

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



### 2. Experimental Preparation:

- (1) Connect micro:bit to computer with USB cable
- (2) 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

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.5: Temperature Measurement /Code-1	microbit-Code-1.hex

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

(1) Go to "Advanced" → "Serial" → "serial redirect to USB"

Place it into "on start"



\*\*\*\*\*

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

Move it into "forever" block





(3) Go to "Input" → "temperature(°C)"

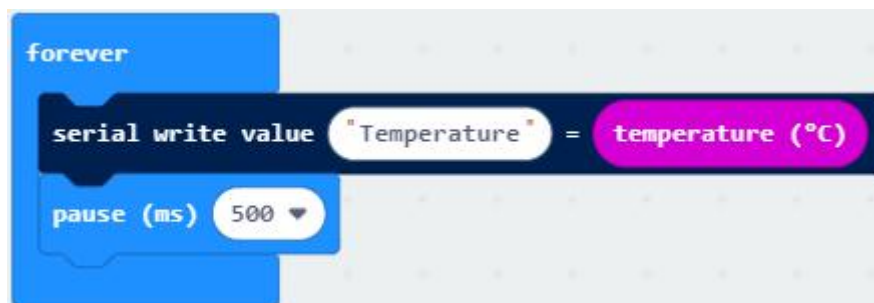
Place it into 0 box

Change x into Temperature

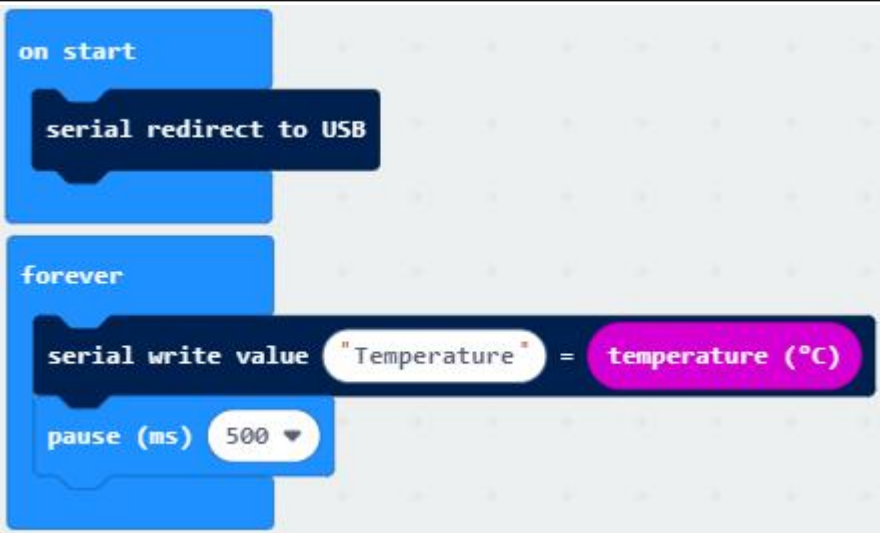


\*\*\*\*\*

(4) 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:

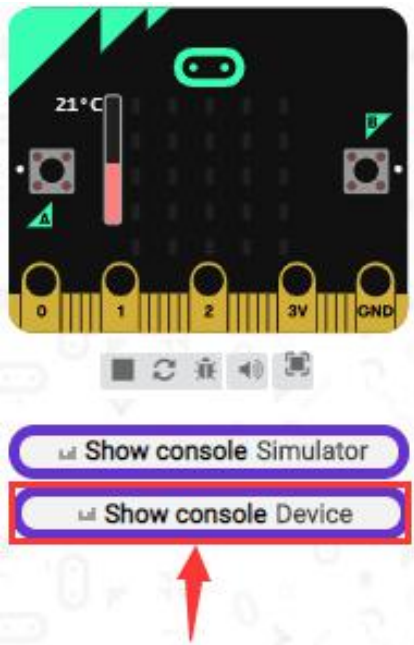


( [How to quick download?](#) )

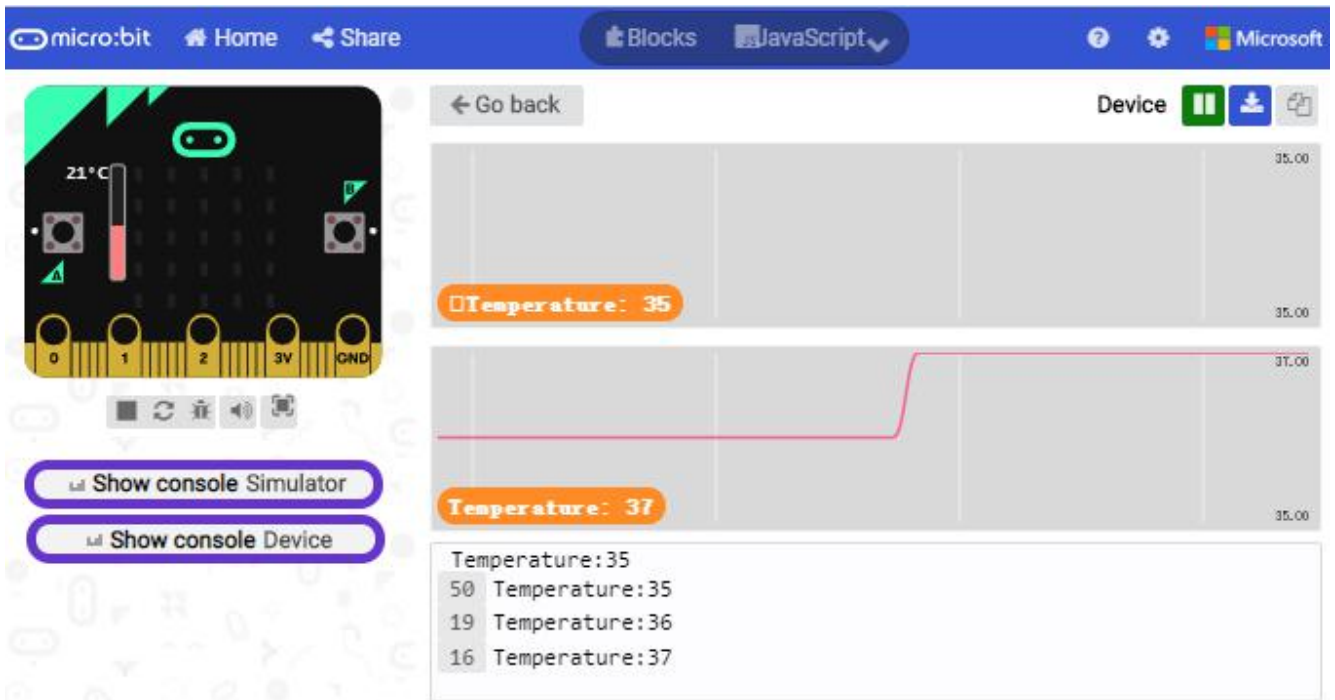


Download code 1 to micro:bit board and keep USB cable connected, then tap button :

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



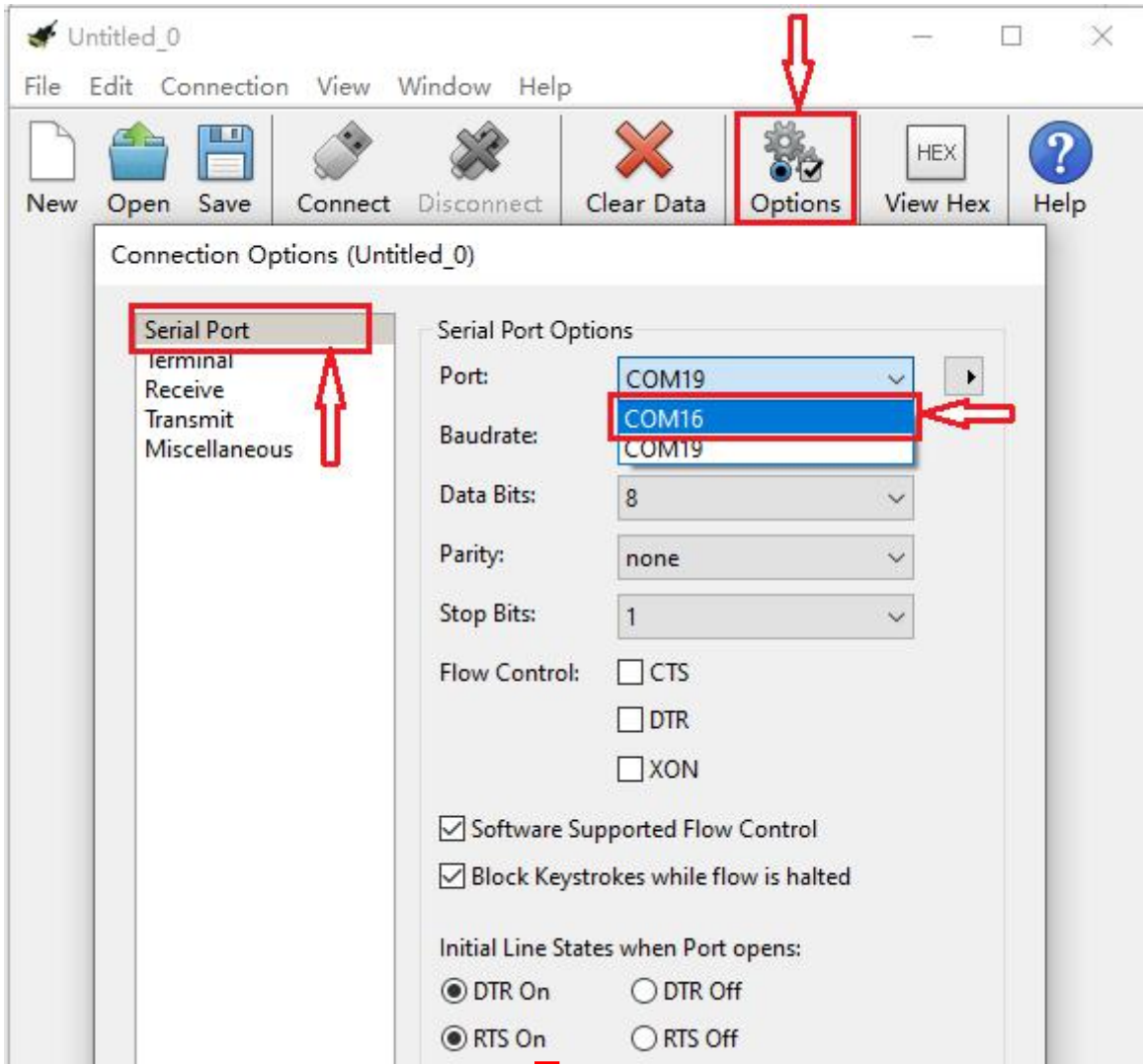
Temperature data is shown below:

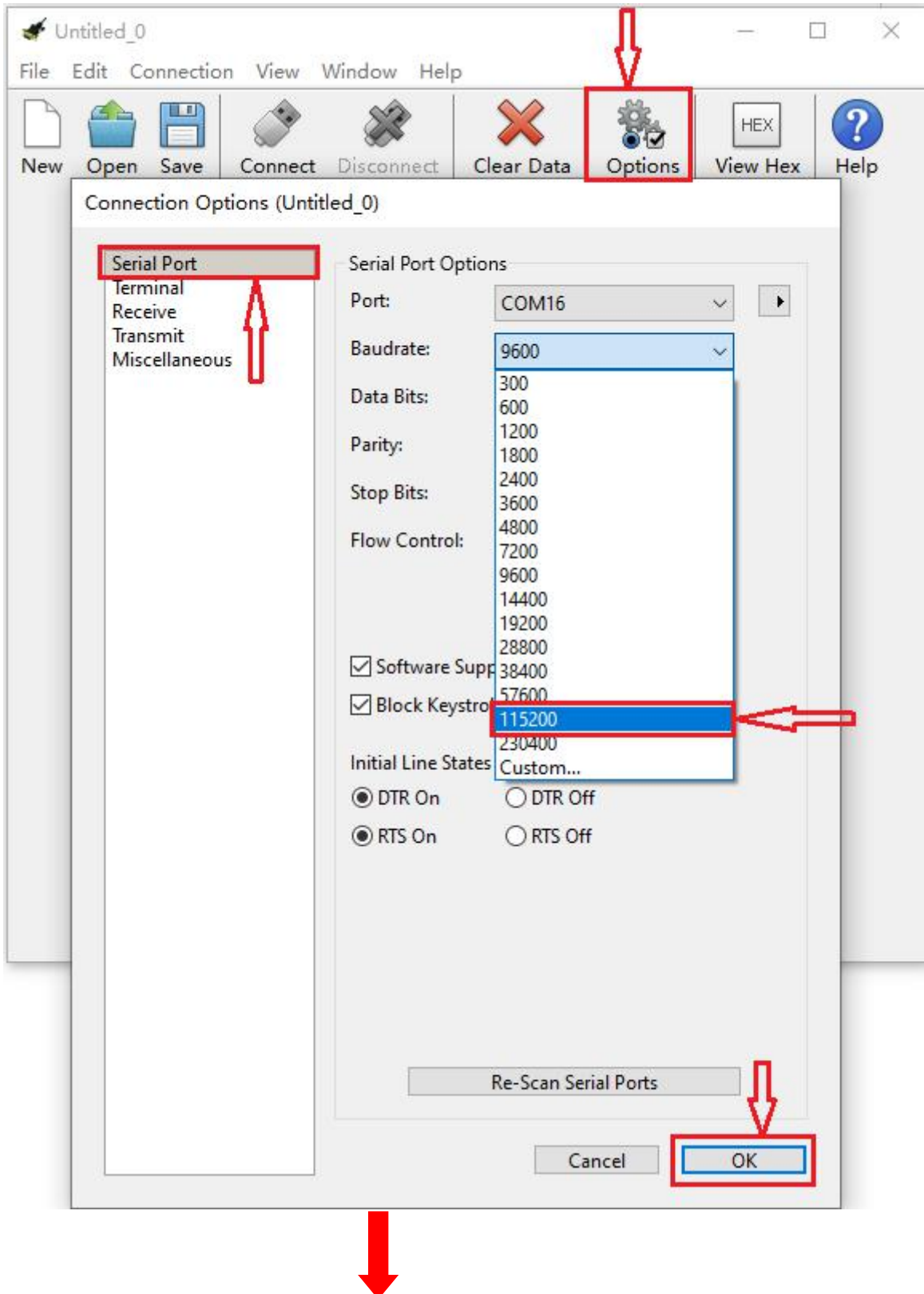


Through the test, the room temperature is 35 °C when touching the NFR51822 chip of micro:bit; however, the temperature rises to 37°C when it touches water cup.

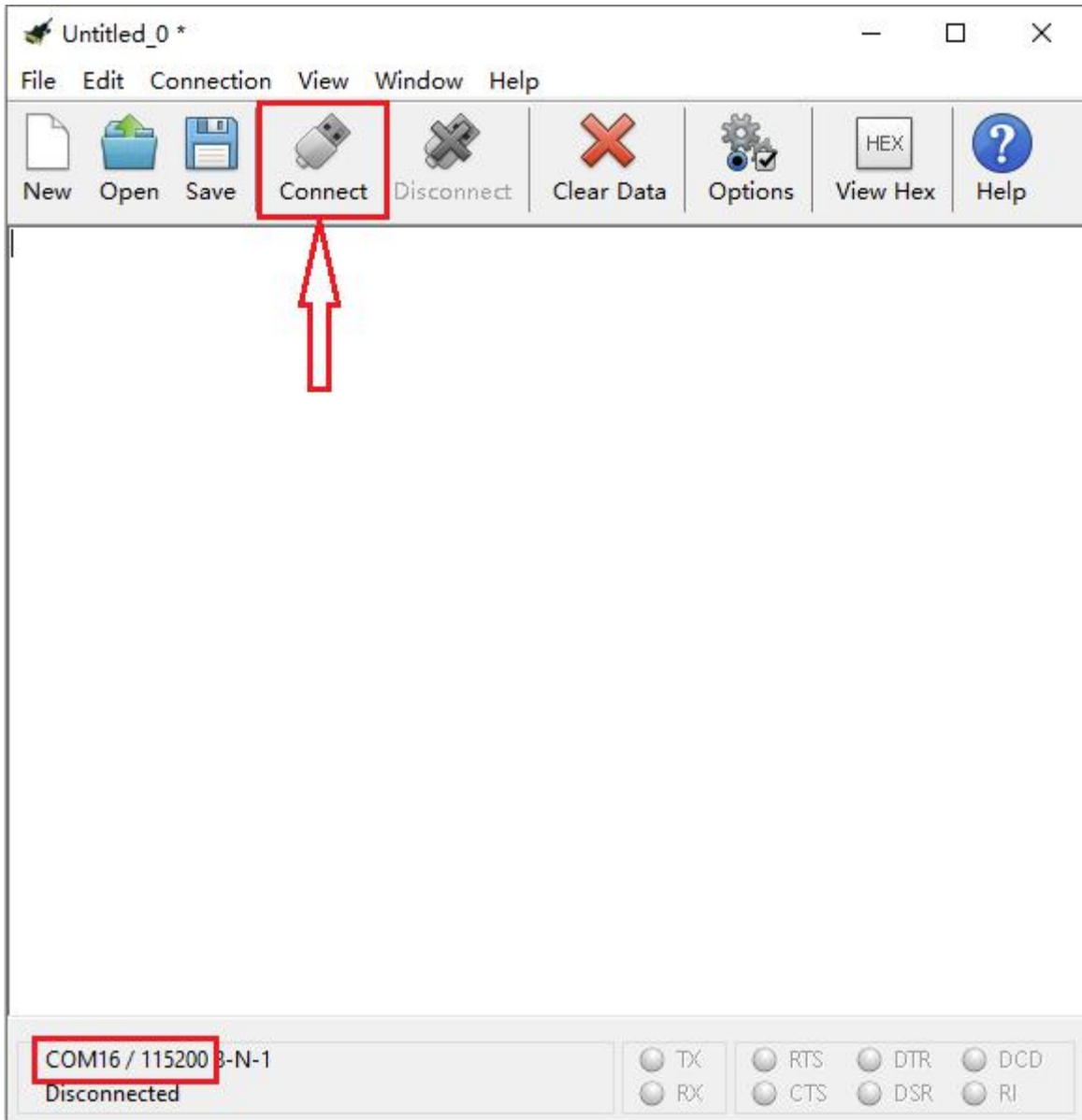
Open CoolTerm, click Options to select SerialPort. Set COM port and 115200 baud rate(the baud rate of USB serial communication of Micro:bit is 115200 through the test). Click "OK" and "Connect" .

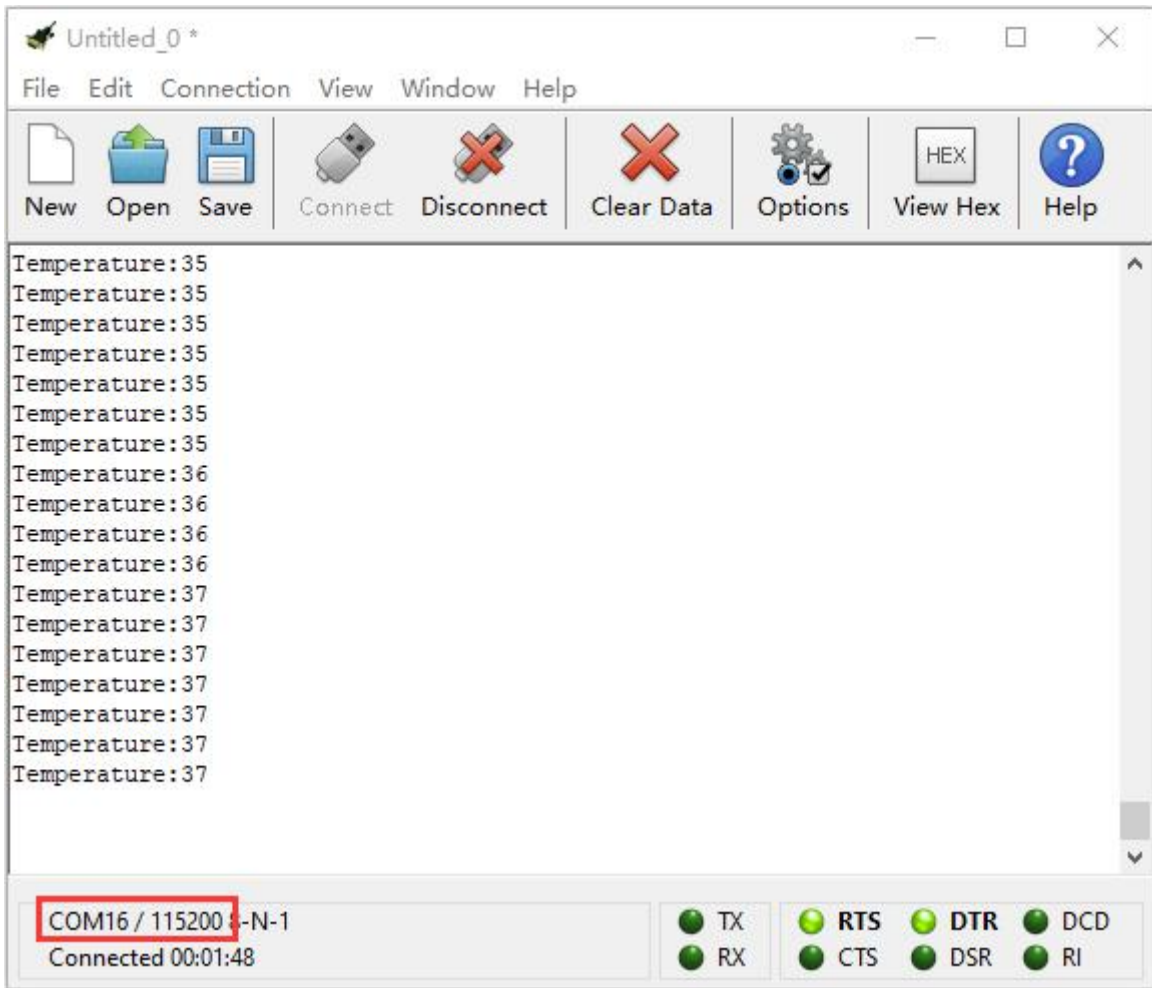
The serial monitor shows the current ambient temperature value, as shown below:











**Code 2:**

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

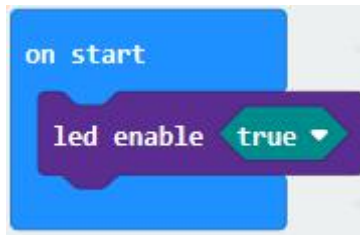
Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.5: Temperature Measurement/Code-2	microbit-Code-2.hex

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

You could set temperature based on real situation.



(1) Click "Led" → "more" → "led enable false" into "on start" , click drop-down

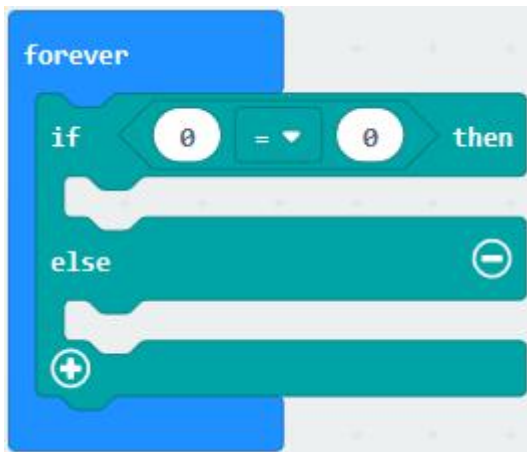


triangle button to select "true"

\*\*\*\*\*

(2) A. Go to "Logic" → "if.true...then...else" and "=" block;

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



\*\*\*\*\*

(3) A. Change "=" into "≥"

B. Go to "Input" → "temperature(°C)" and move it into left 0 box;

C. Change 0 into 35.



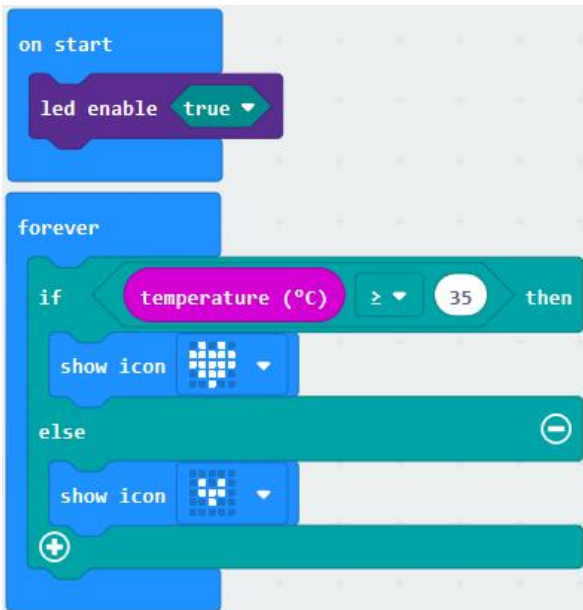
\*\*\*\*\*

(4) 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.

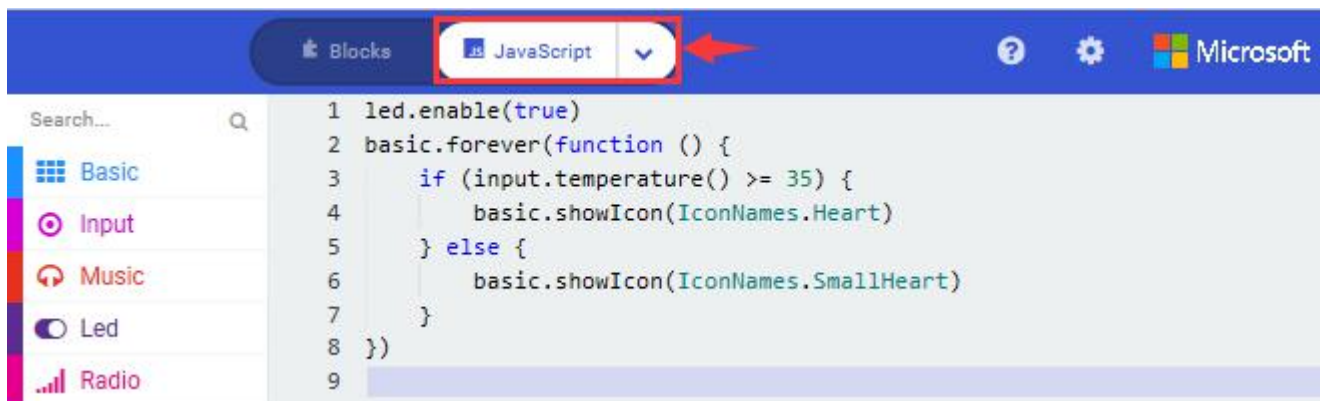
Turn on LED dot matrix

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:





#### 4. Test Result:

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



. When the

temperature is equivalent to or greater than 35°C, the pattern



will

appear.

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

### 7.6: Micro:bit' s Compass



#### 1. 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:

- (1) Connect micro:bit to computer with USB cable
- (2) 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.

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.6: Micro:bit' s Compass/Code-1	microbit-Code-1.hex

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

(1) A. Click "Input" → "more" → "calibrate compass"

B. Lay down it into block "on start" .



(2) A. Go to "Input" → "on button A pressed" .

B. Enter "Basic" → "show number" , put it into "on button A pressed" block;

C. Tap "Input" → "compass heading(°C)" , and place it into "show number"

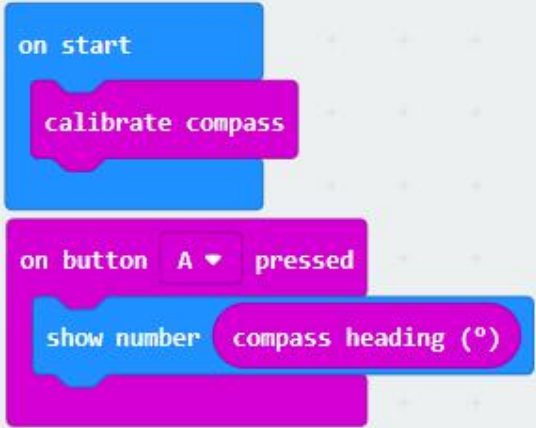


\*\*\*\*\*





## Complete Program:



- 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

Click "JavaScript", and view the corresponding JavaScript code:



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

### Code Description:

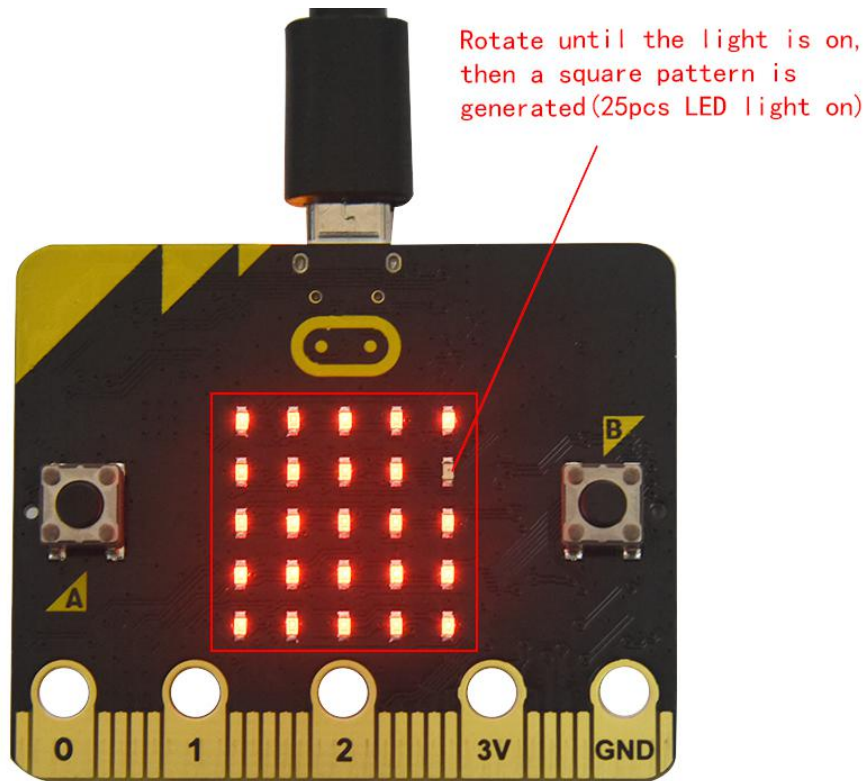
Upload the code 1, plug in micro:bit via USB cable.

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),

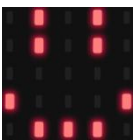


as shown in the following figure:

([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

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.6: Micro:bit' s Compass/Code-2	microbit-Code-2.hex

```
forever
  set x to compass heading (°)
  if compass heading (°) >= 293 and compass heading (°) < 338 then
    show leds
  else if compass heading (°) >= 23 and compass heading (°) < 68 then
```

This code string complies that we read the value detected incessantly and determine the direction by the value range. The direction is toward North at this time.

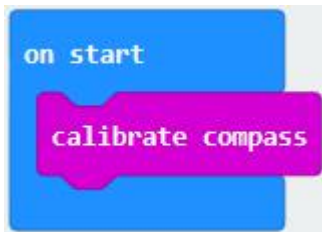


Or you could edit code step by step in the editing area.

(1)

A. Enter "Input" → "more" → "calibrate compass"

B. Move "calibrate compass" into "on start"



\*\*\*\*\*

(2) 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



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



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

(4) 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 "≥" ;

D. Then copy " $x \geq 293$ " once and leave it to the right "0" box and set to " $x < 338$ "



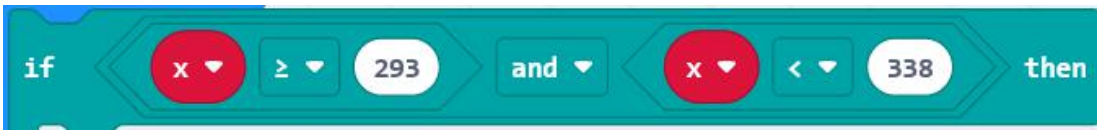
```
forever
  set x to compass heading (°)
  if x ≥ 293 and x < 338 then
  else if then
  else if then
  else if then
  else if then
  else if then
  else if then
  else if then
  else
  +
```

\*\*\*\*\*


(5) A. Go to "Basic" → "show leds"

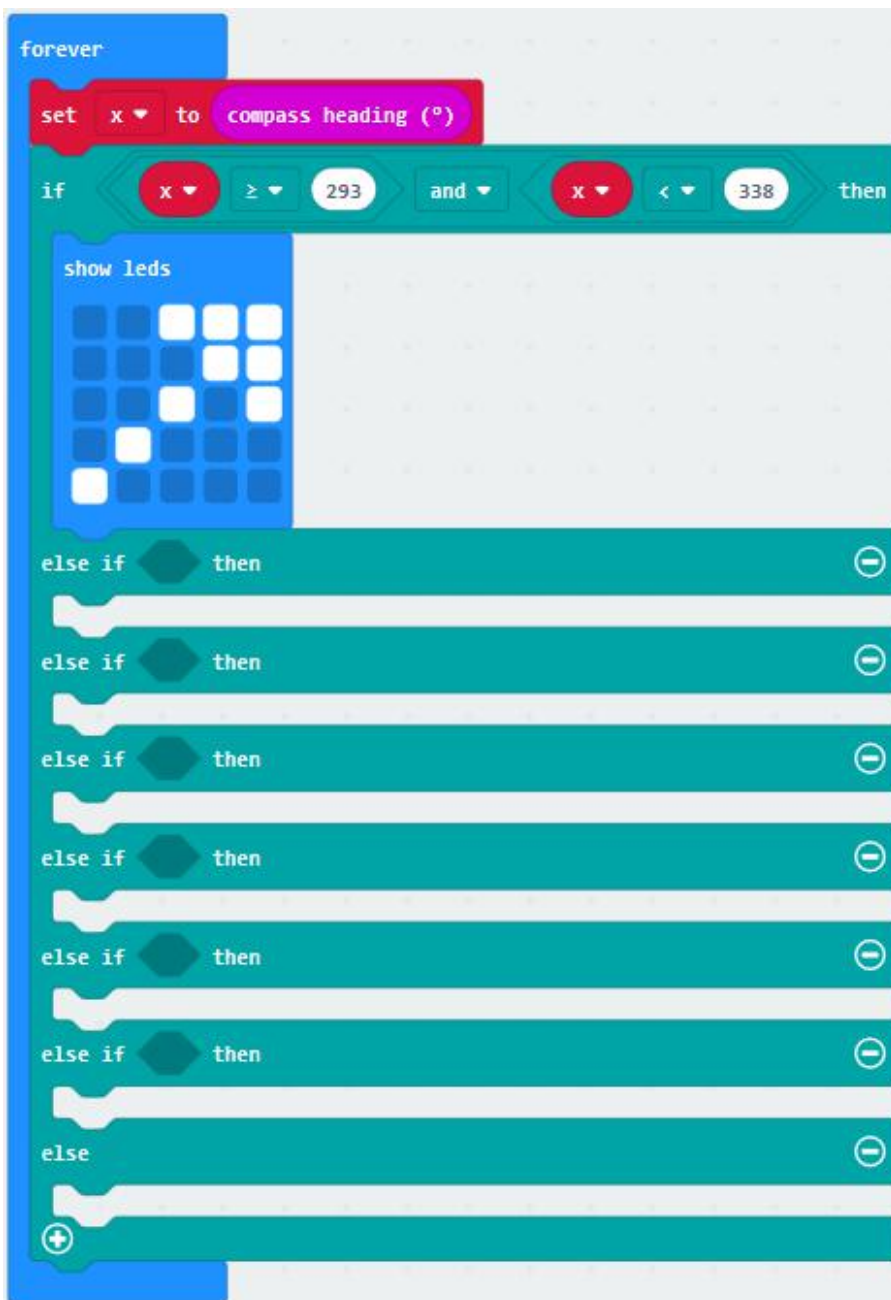


## B. Lay it down beneath



block, then

click "show leds" and the pattern  appears.












(6) 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 ≥ 158 and x < 203 " , "x ≥ 203 and x < 248 " , "x ≥ 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
  calibrate compass

forever
  set x to compass heading (°)
  if x >= 293 and x < 338 then
    show leds
  else if x > 23 and x < 68 then
    show leds
  else if x > 68 and x < 113 then
    show leds
  else if x >= 113 and x < 158 then
    show leds
```

“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

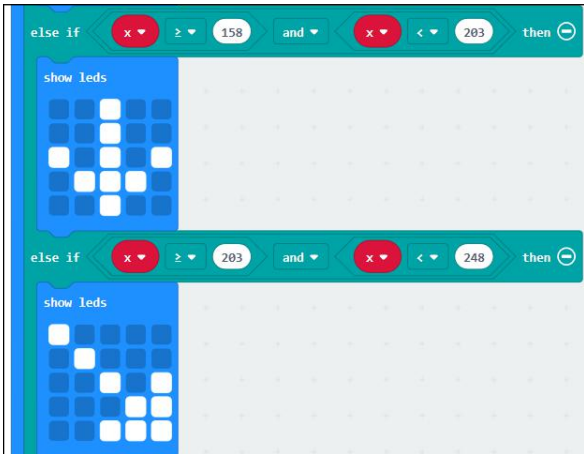


is shown on dot matrix


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




pattern appears

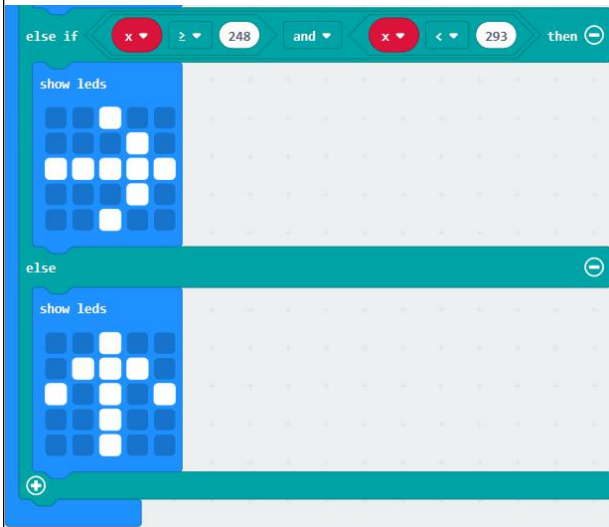


When  $158 \leq x < 203$ , the next program will be executed.


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

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



```
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 Result:

Download code 2 to micro:bit and keep USB cable connected.

After calibration, tilt Micro:bit board, micro:bit displays the direction signs.



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

## 7.7: Accelerometer



### 1. Description:

The micro:bit board has a built-in Freescale MMA8653FC three-axis acceleration sensor (accelerometer). Its I2C 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:

- (1) Connect micro:bit to computer with USB cable
- (2) 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:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.7: Accelerometer/Code-1	microbit-Code-1.hex

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;

B. 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

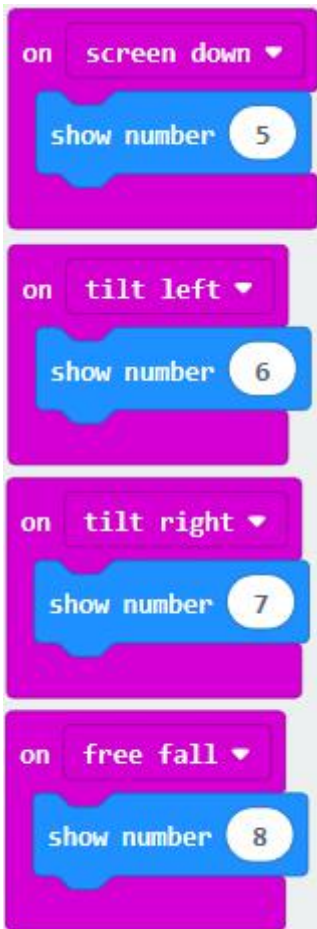
The logo is down

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

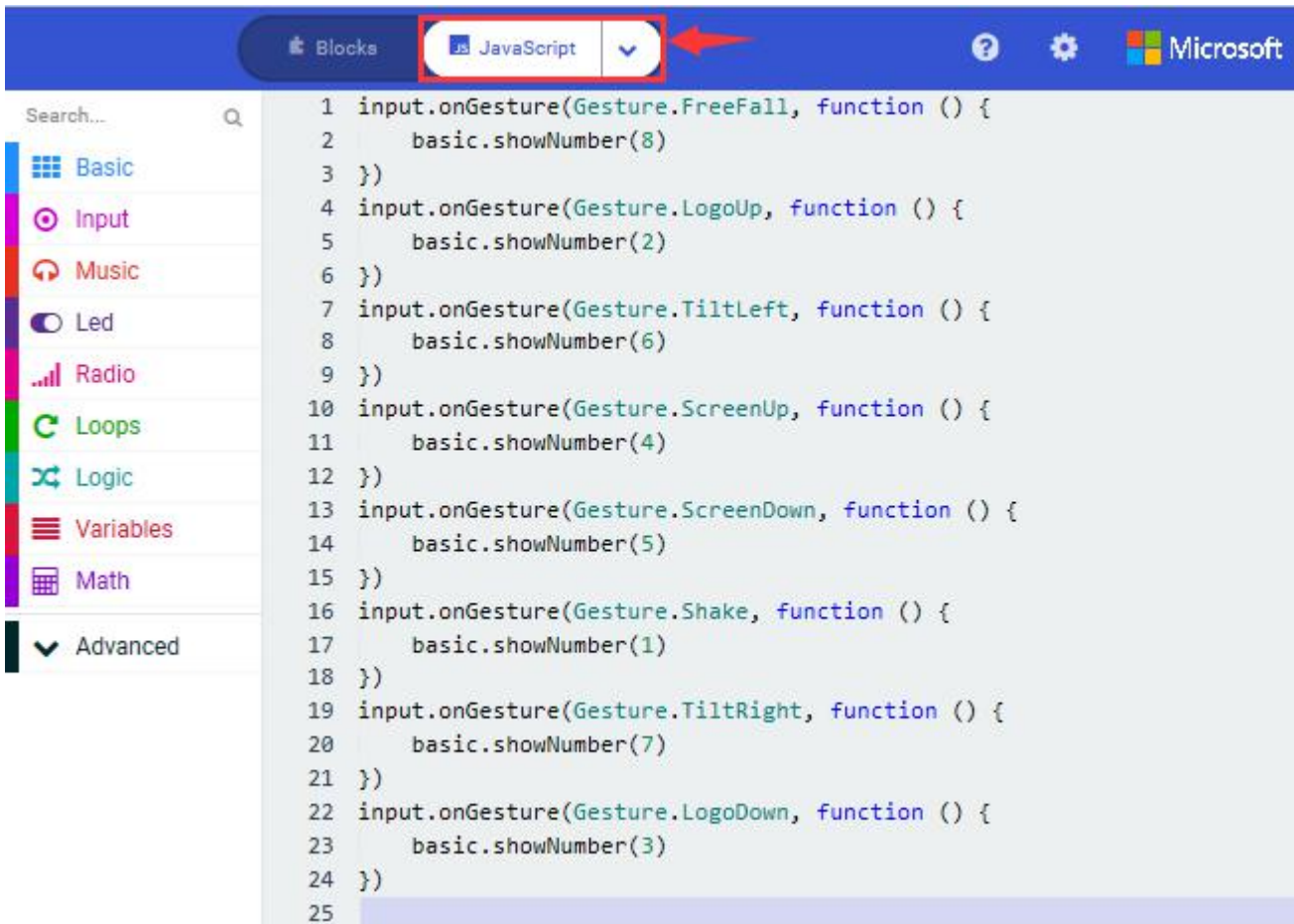
The Micro:bit board is tilt to the right

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:



### Code 2:

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

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.7: Accelerometer/Code-2	microbit-Code-2.hex

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

(1) A. Go to "Advanced" → "Serial" → "serial redirect to USB"

B. Drag it into "on start"



\*\*\*\*\*

(2) A. Enter "Serial" → "serial write value x =0"

B. Leave it into "forever" block



(3) \*\*\*\*\*

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

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



\*\*\*\*\*

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

block  , then set to 100ms.



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

\*\*\*\*\*

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

(5) Replicate code string 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
  serial redirect to USB

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

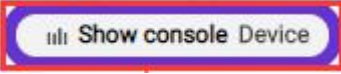
```

“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

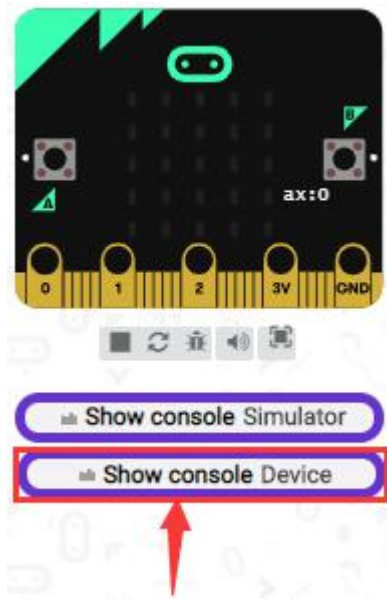
Click “JavaScript” to view the corresponding JavaScript code:

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

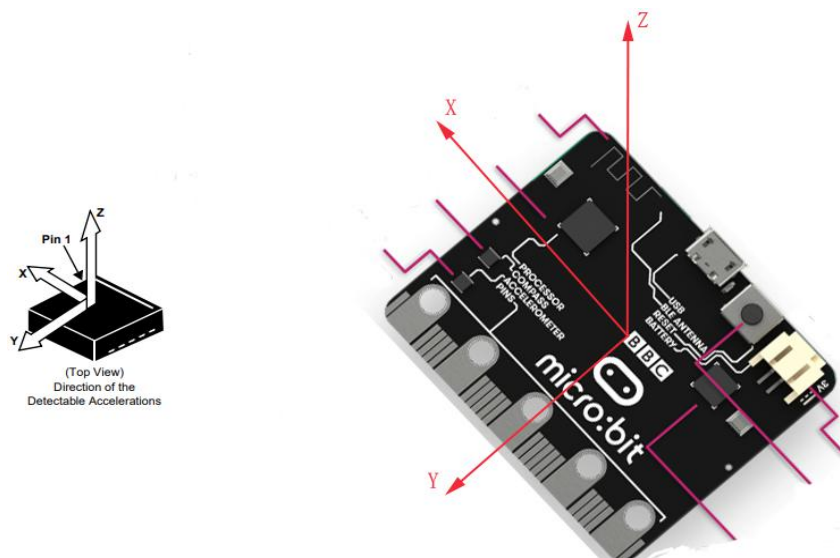
Download code 1 to micro:bit board, keep USB cable connected and

click 

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



The coordinates of the Micro:bit accelerometer are shown in the following figure:



The decomposition value of acceleration on the X-axis, Y-axis, and Z-axis, as well as the synthesis of acceleration (the synthesis of gravitational



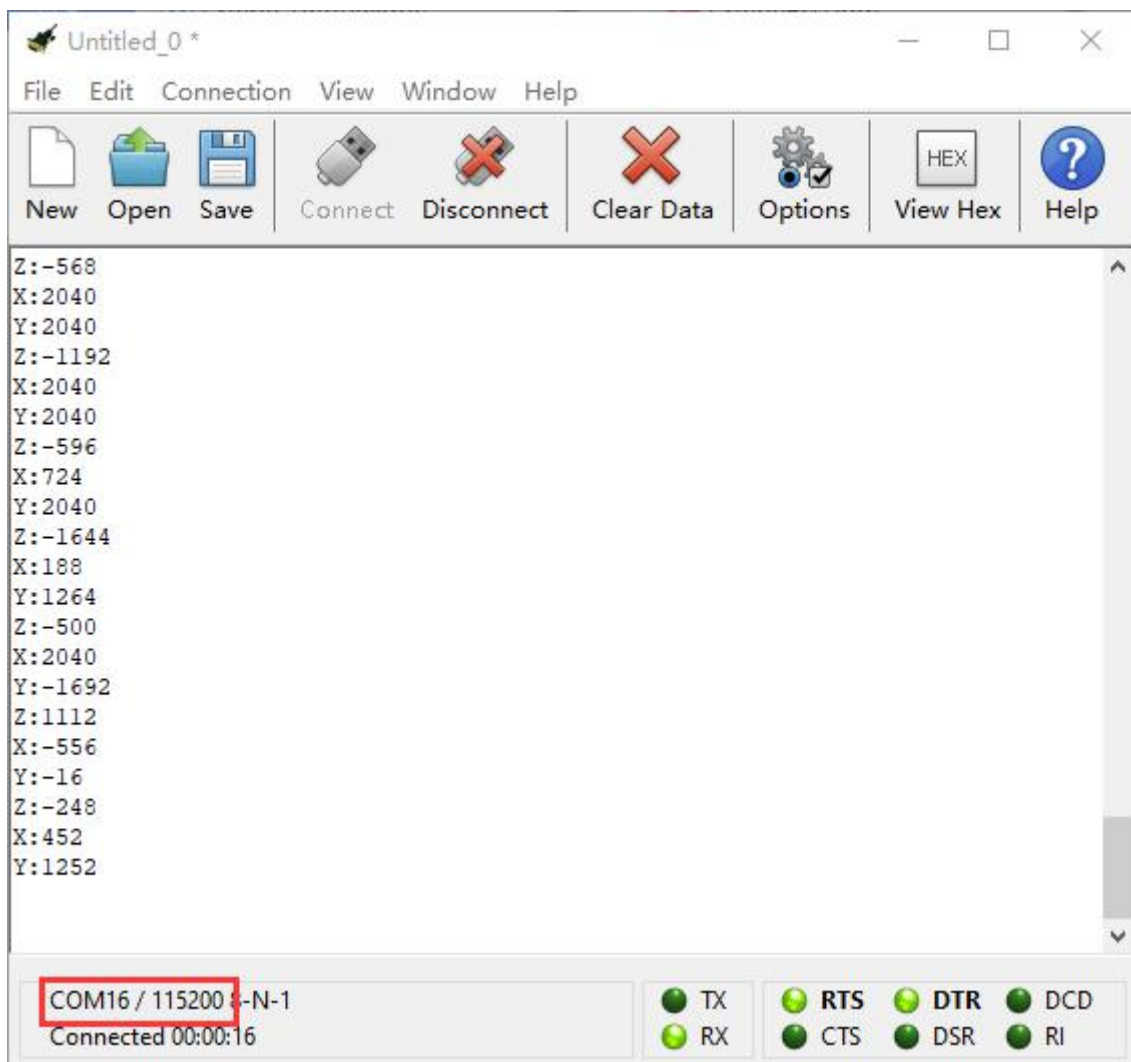
acceleration and other external forces). Then flip the micro:bit board, the data is shown below:





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 acceleration value on x, y and z axis





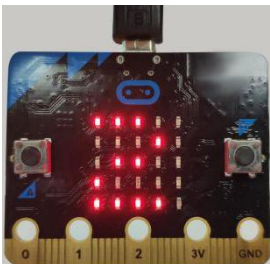


#### 4. Test Result:

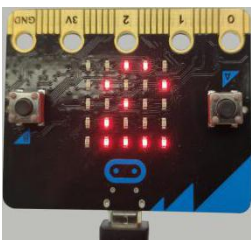
Download code 1 to micro:bit board and keep USB cable connected, shake the Micro:bit board then the number 1 appears.

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

Place micro:bit vertically (logo up), then the number 2 is displayed:



Place micro:bit vertically (logo down), then the number 3 is displayed:

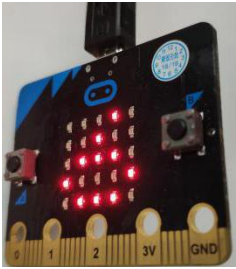


Place micro:bit horizontally (facing up), then the number 4 is displayed:

On the contrary, place micro:bit horizontally (facing down), then the number 5 is displayed:



When Micro:bit board is tilt to the left, number 6 is shown.



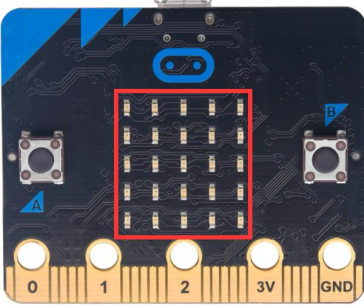
When Micro:bit board is inclined to the right, number 7 is displayed.



When it is free fall(accidentally making it fall), number 8 appears on dot matrix. (Note: we don' t recommend you to make it free fall, it will make board damage)



## 7.8: Detect Light Intensity by Micro:bit



### 1. Description:

This project will introduce how Micro:bit detects the external light intensity. Since Micro:bit doesn't come with photosensitive sensor, the detection of light intensity is completed through the LED matrix. 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:

- (1) Connect micro:bit to computer with USB cable
- (2) Open online Makecode editor

Import Hex profile [\(How to import?\)](#) , or click "New Project" and drag blocks step by step.



### 3. Test Code:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.8: Detect Light Intensity by Micro:bit	microbit-Detect Light Intensity by Micro:bit .hex

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"

(3) A. Click "Input" → "acceleration(mg) x"

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



```
forever
  serial write value "Light intensity" = light level
```

\*\*\*\*\*

(4) A. Click "Basic" → "pause (ms) 100" ;

B. Lay it down into "forever" and set to 100ms.

```
forever
  serial write value "Light intensity" = light level
  pause (ms) 100
```

\*\*\*\*\*

#### 4. Complete Program:

```
on start
  serial redirect to USB

forever
  serial write value "Light intensity" = light level
  pause (ms) 100
```



“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

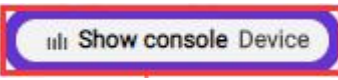
Delay in 100ms

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

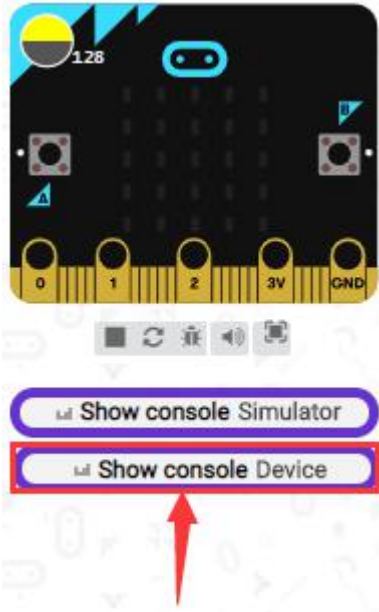
```
1 serial.redirectToUSB()
2 basic.forever(function () {
3   serial.writeValue("Light intensity", input.lightLevel())
4   basic.pause(100)
5 })
6
```

## 5. Test Result:

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

click 

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



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:



Light intensity: 220

```
Light intensity:33
Light intensity:34
Light intensity:39
Light intensity:43
Light intensity:48
Light intensity:57
Light intensity:70
Light intensity:92
Light intensity:120
Light intensity:150
Light intensity:196
Light intensity:220
```

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 View 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 RTS DTR DCD  
RX CTS DSR RI



## 7.9: Bluetooth Wireless Communication



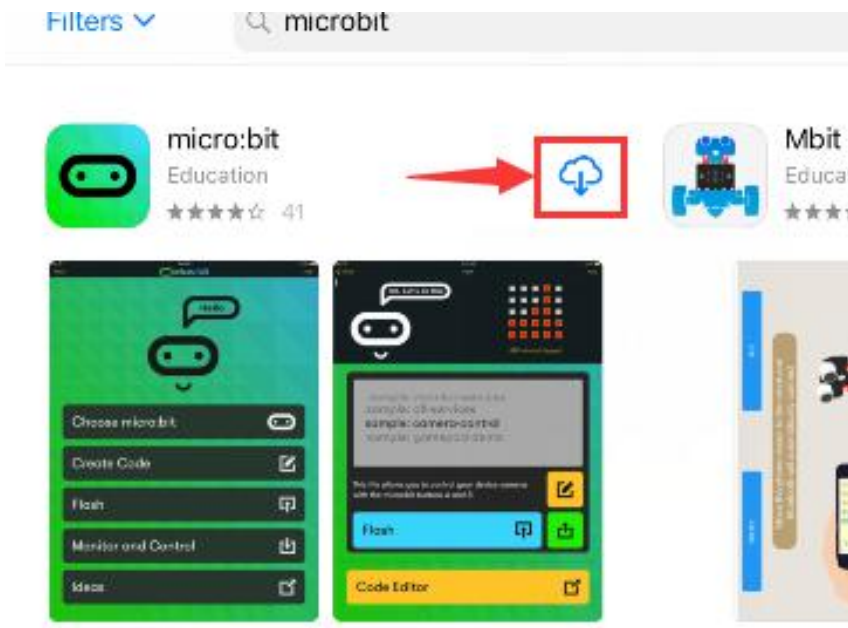
### 1. Description:

Micro:bit board comes with NRF51822 processor, Bluetooth and 2.4GHz RF antenna, which work with Bluetooth and 2.4G wireless communication.

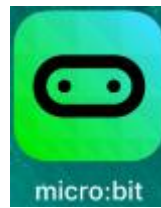
In this project, we connect cellphone to Micro:bit motherboard to complete the wireless connection.

### 2. Install Micro:bit APP:

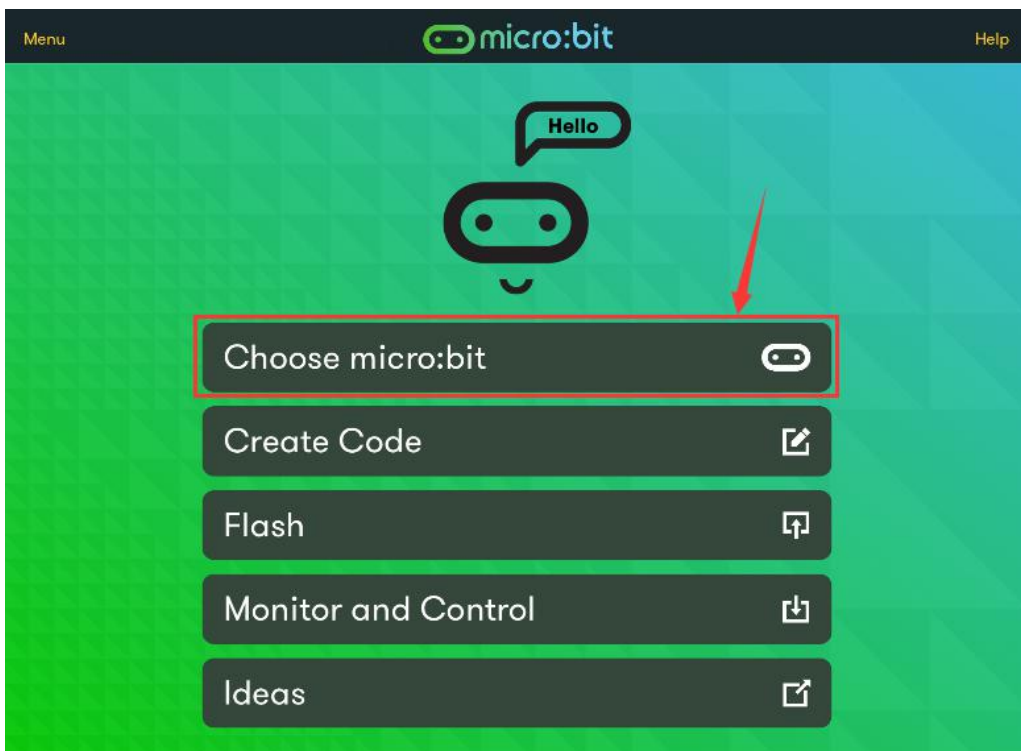
- (1) Connect the computer to micro:bit board by micro USB cable.
- (2) Search "micro:bit" in APP store and install it.



(3) Pair the Micro:bit motherboard with cellphone or iPad

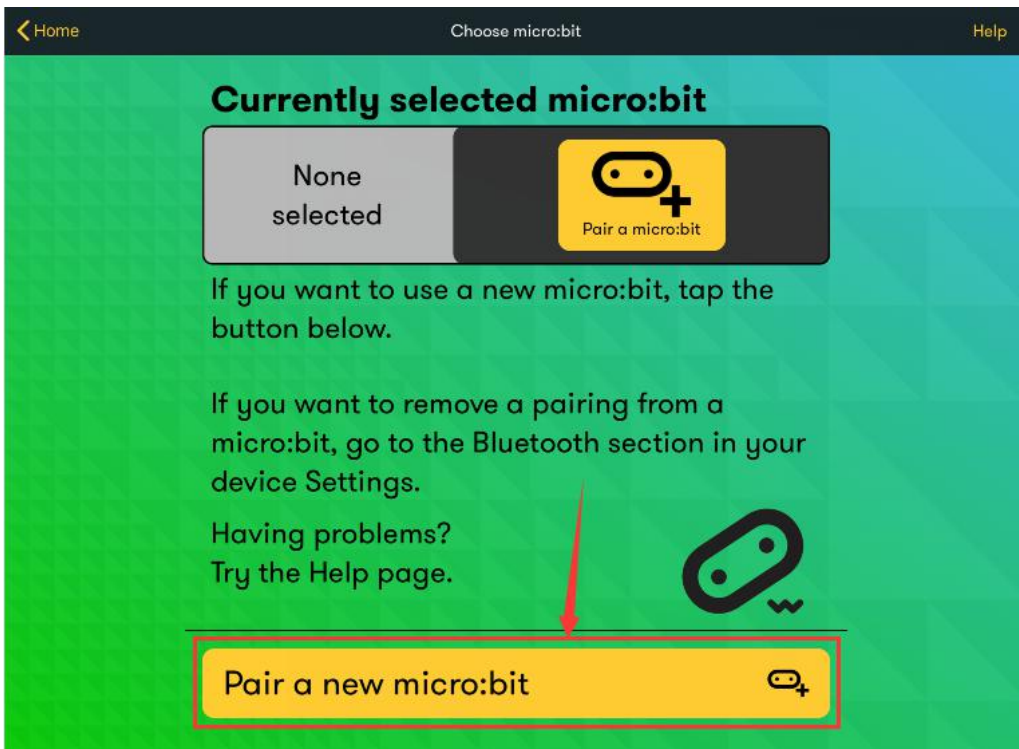


a. After installing APP successfully, click  to enter app. plug in Micro:bit motherboard, click "Choose Micro:bit" to start Bluetooth pairing.





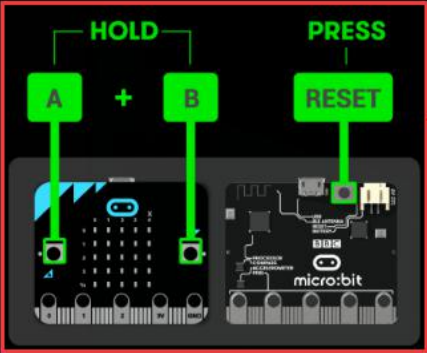
b. Click "Pair a new Micro:bit"



c. Press and hold button A and B, then press and release the reset button. LED dot matrix will show a pattern. At last, release the button A and B at same time and click "Next"



**How to pair your micro:bit**



Step 1  
HOLD the A and B buttons and  
PRESS and RELEASE RESET

Let's do this

Cancel X Next >





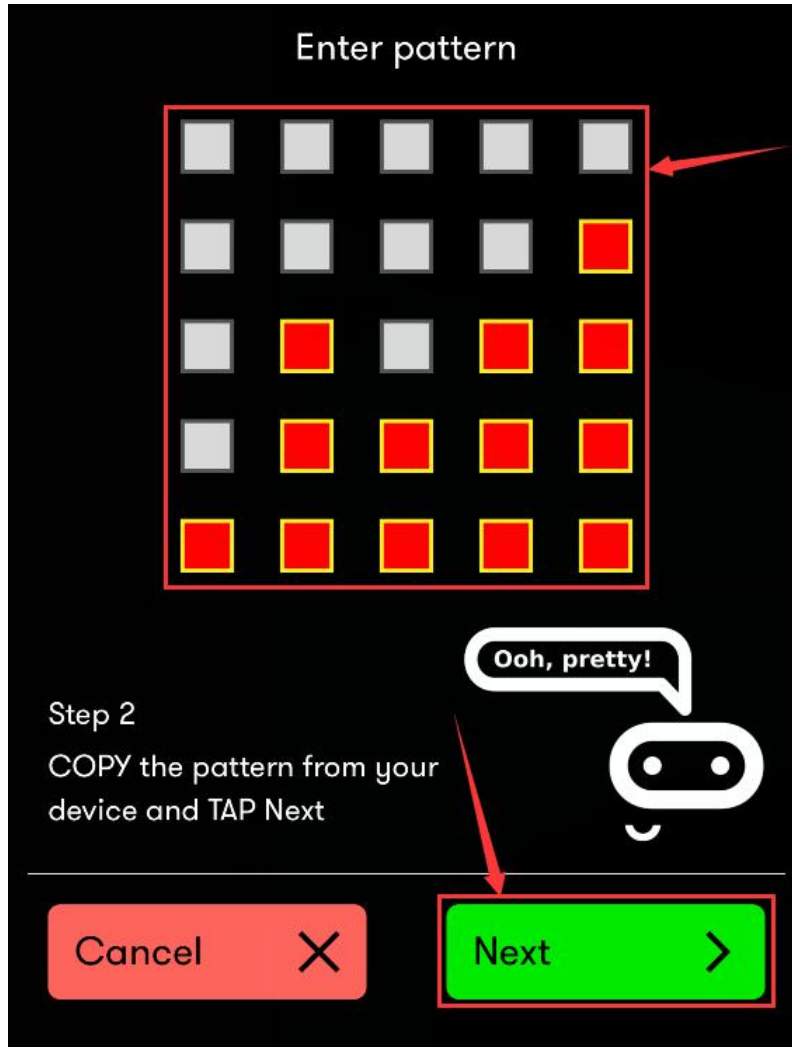
### How to pair your micro:bit

Step 1  
HOLD the A and B buttons and  
PRESS and RELEASE RESET

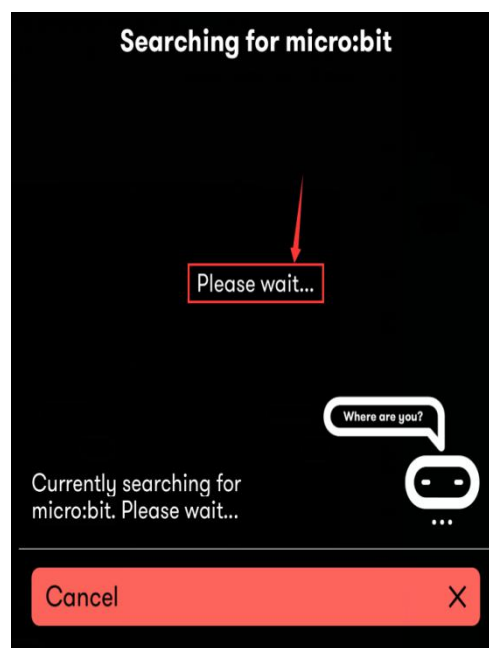
Let's do this

Cancel X Next >

d. Make the displayed pattern on Micro:bit board as same as the one on iPad, click "Next" .



e. Click "Next" and pair successfully.

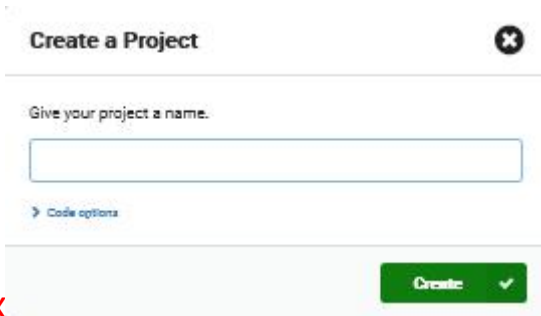




After connecting successfully, design code via APP .

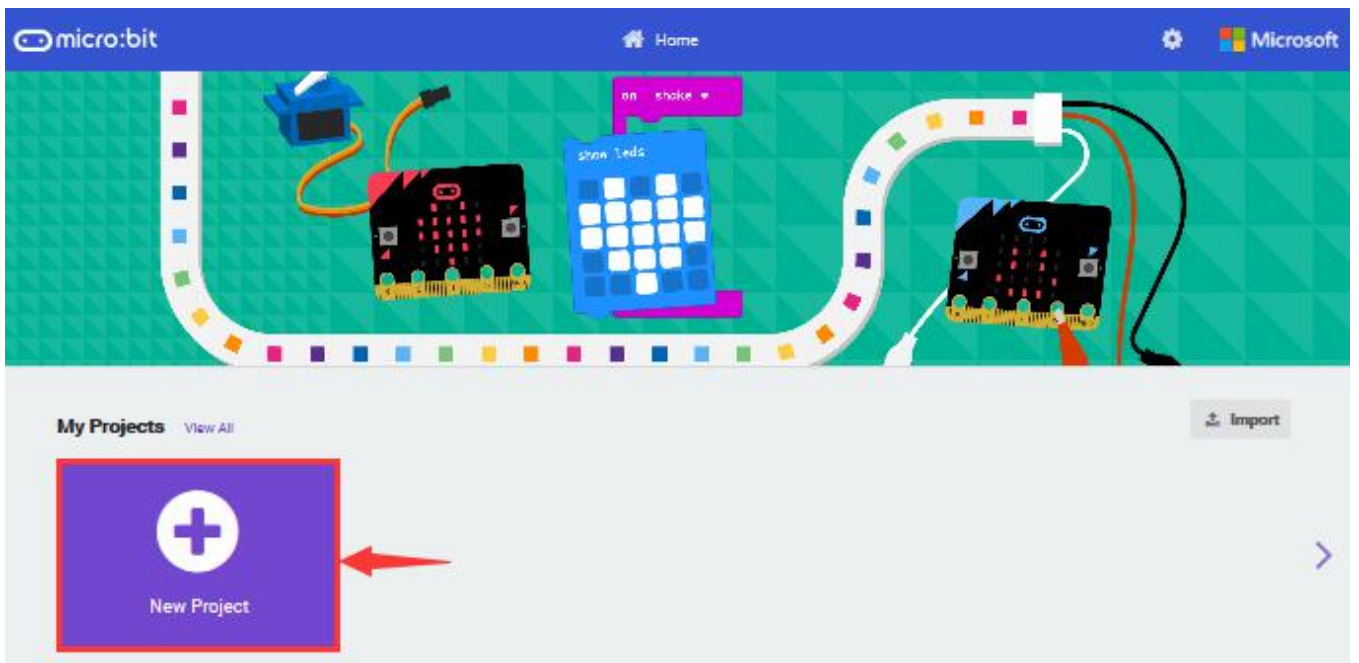
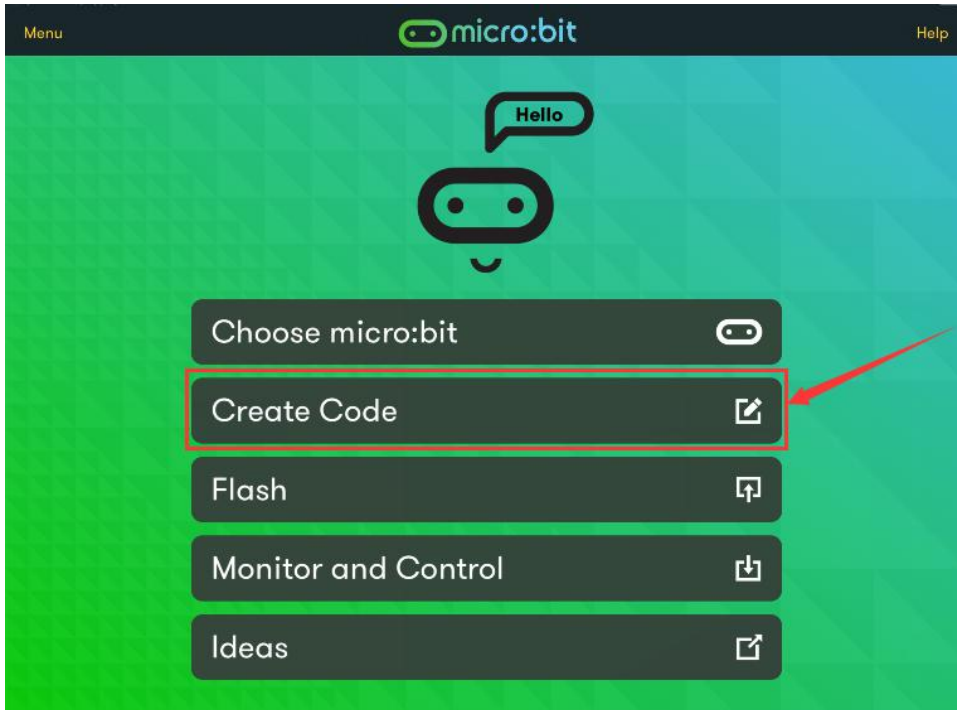


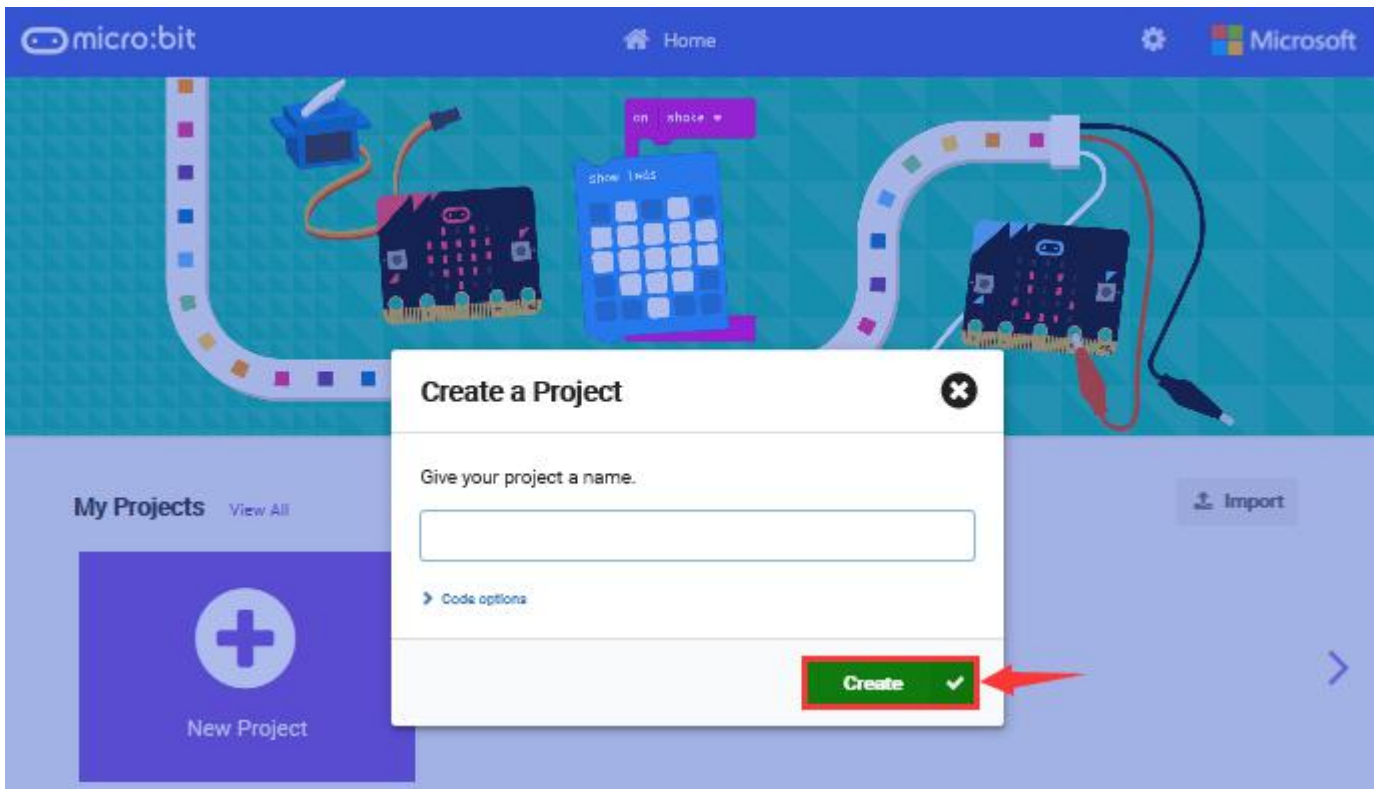
f. Select "Create Code" (Click **and appear the dialog**

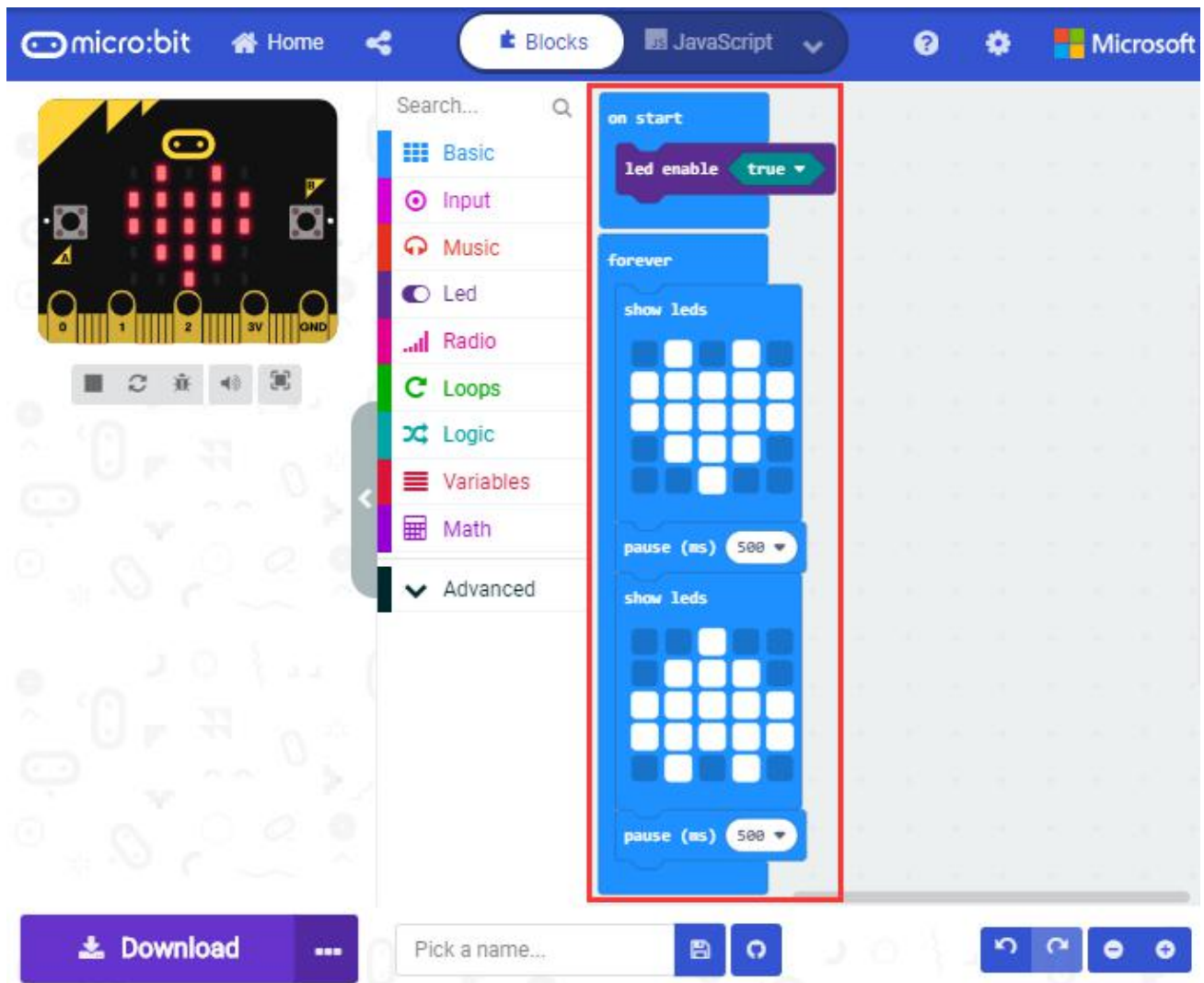


**box** , click icon **Create** to enter the programming interface.

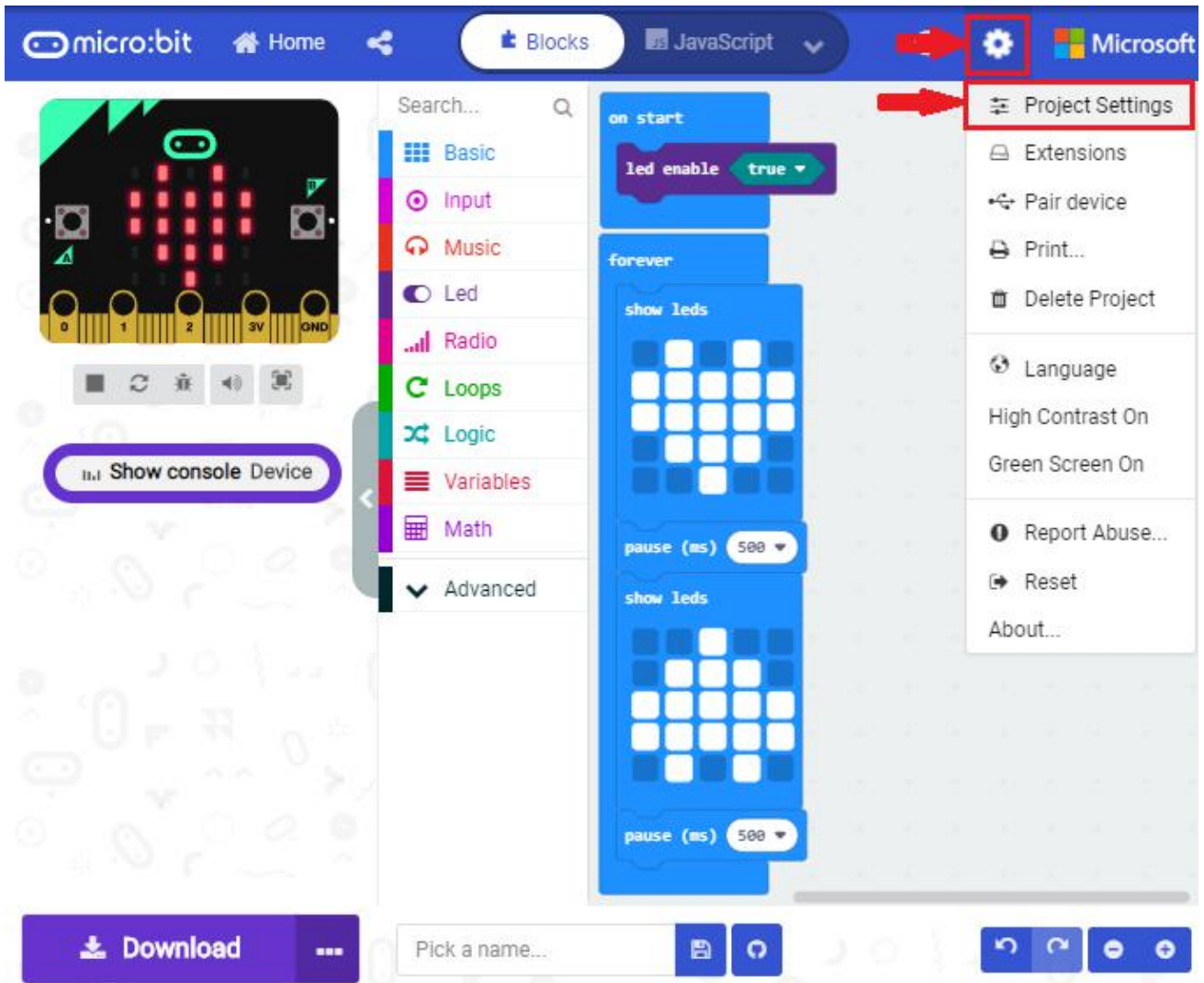





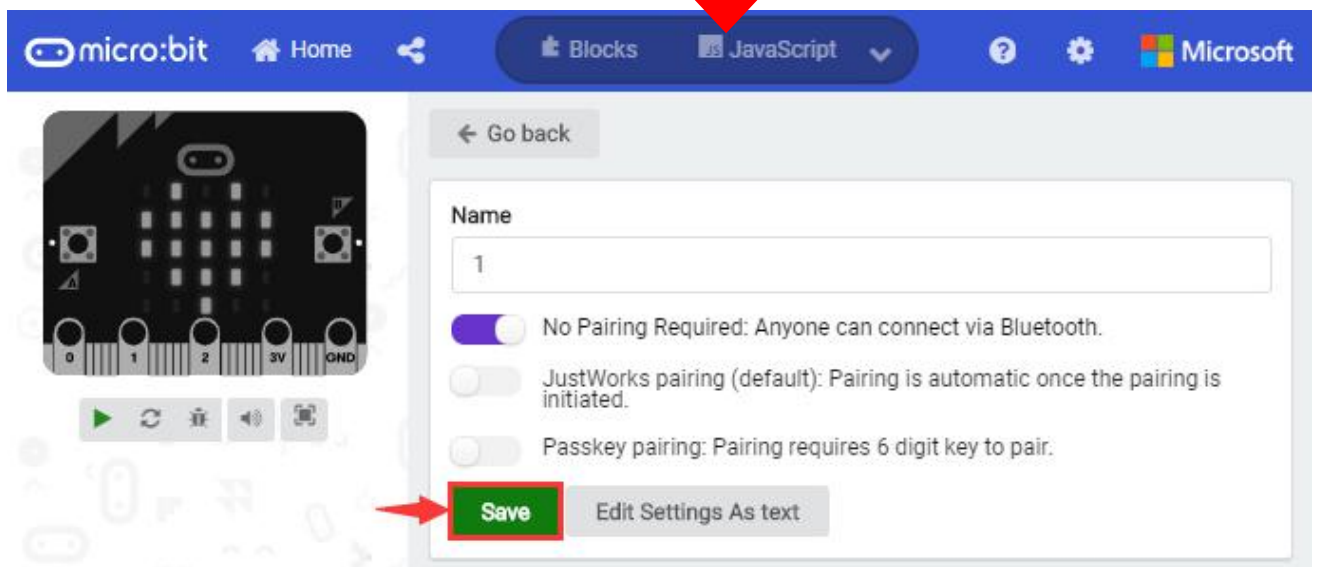
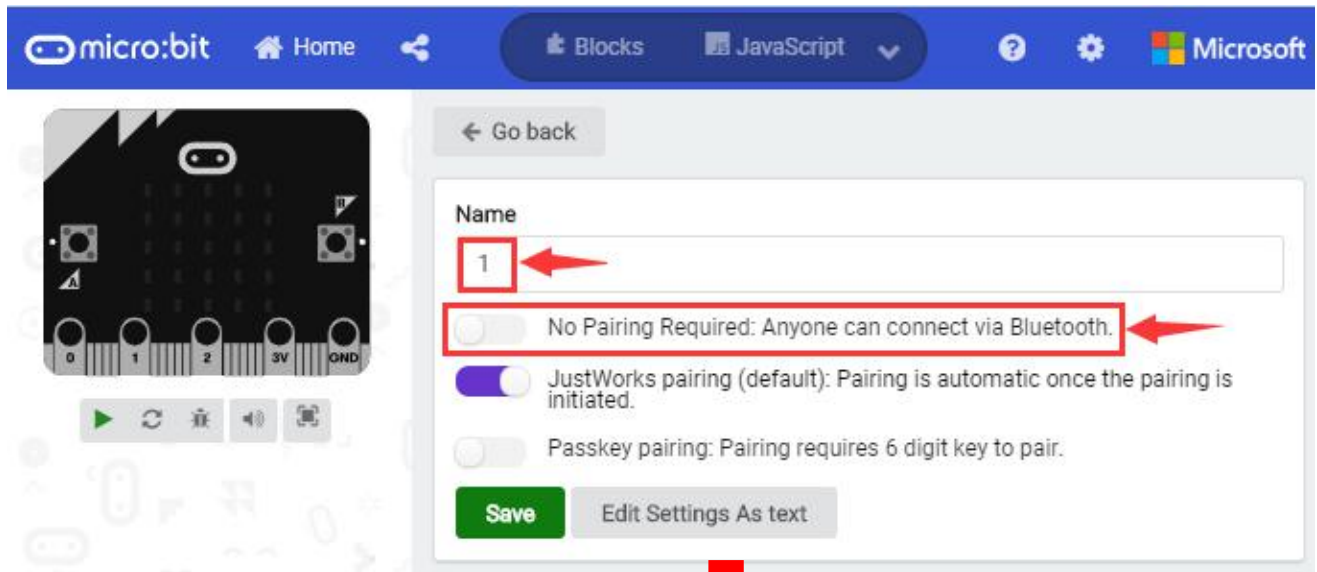




g. After finish programming, click  to select "Project Setting" .

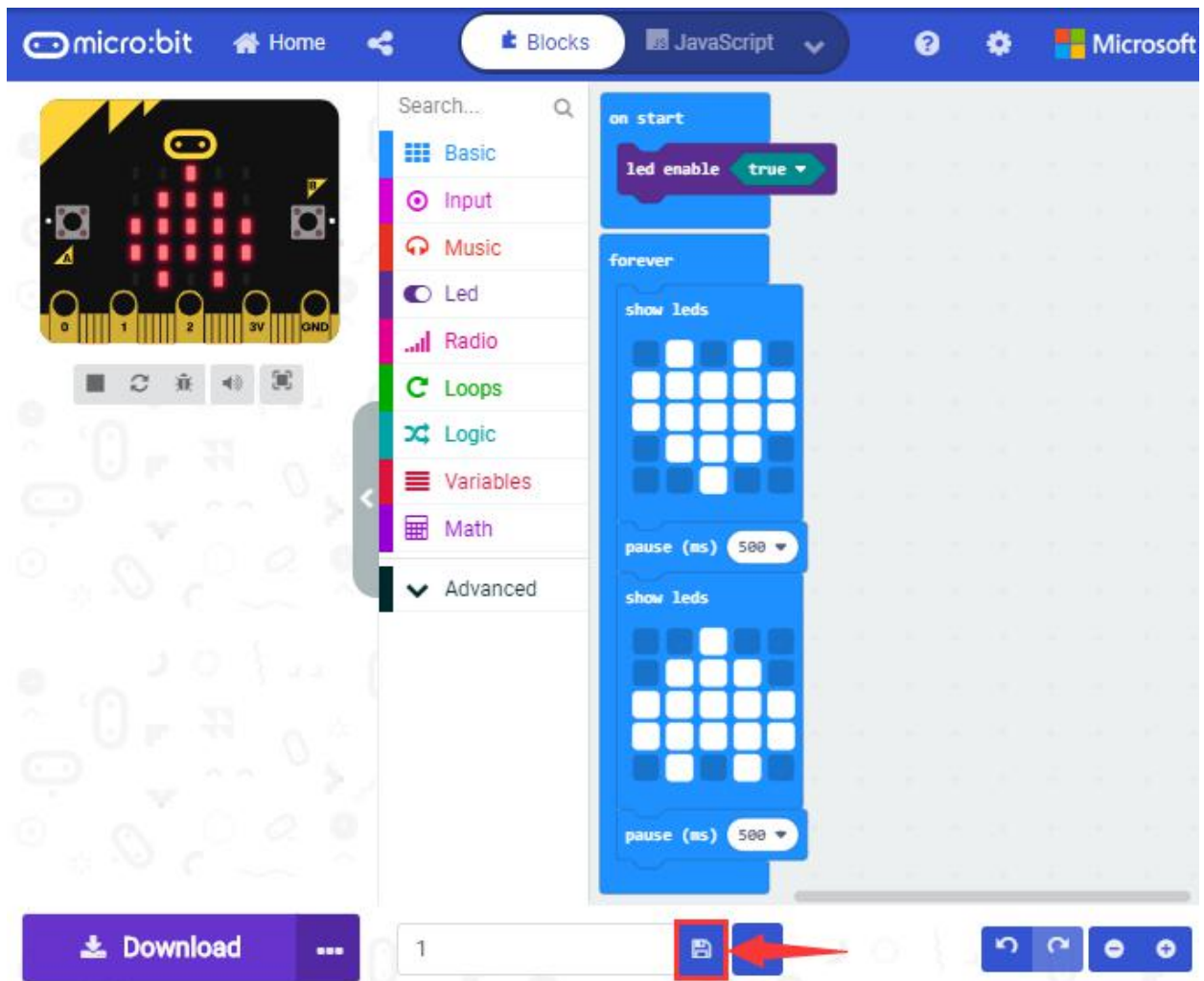


h. Input 1 in the blank box to name your project. Enable button  No Pairing Required: Anyone can connect via Bluetooth. The page will return to the initial page. Then click the button  and "Project Setting" to come back and tap "save"

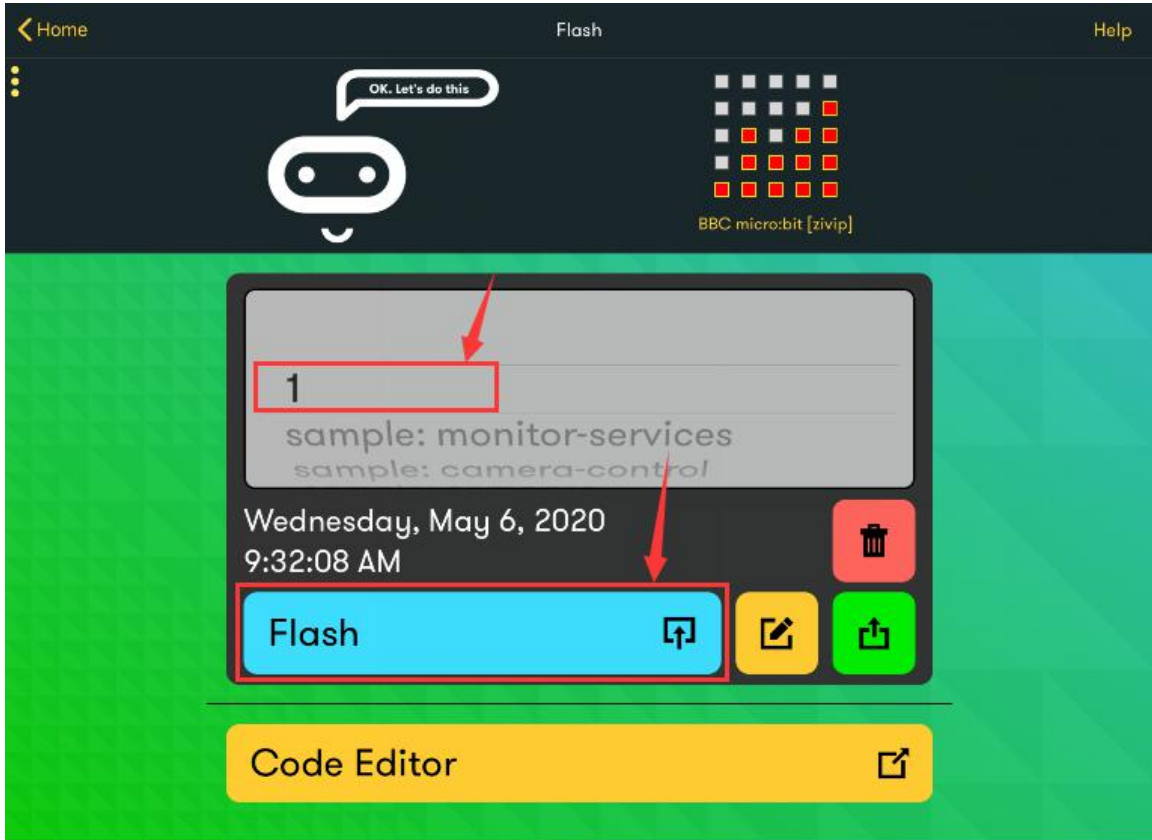


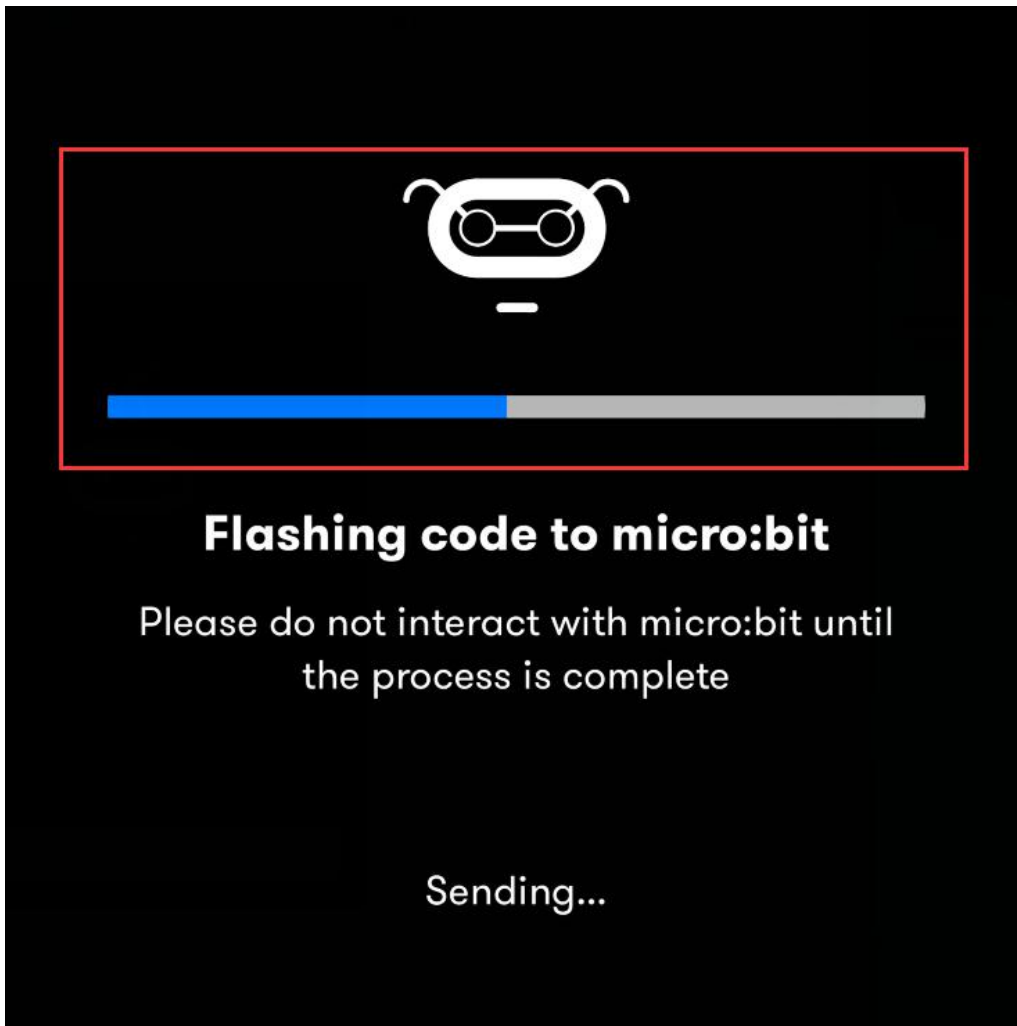
i. The program "1" is named, click

icon   to save the program.



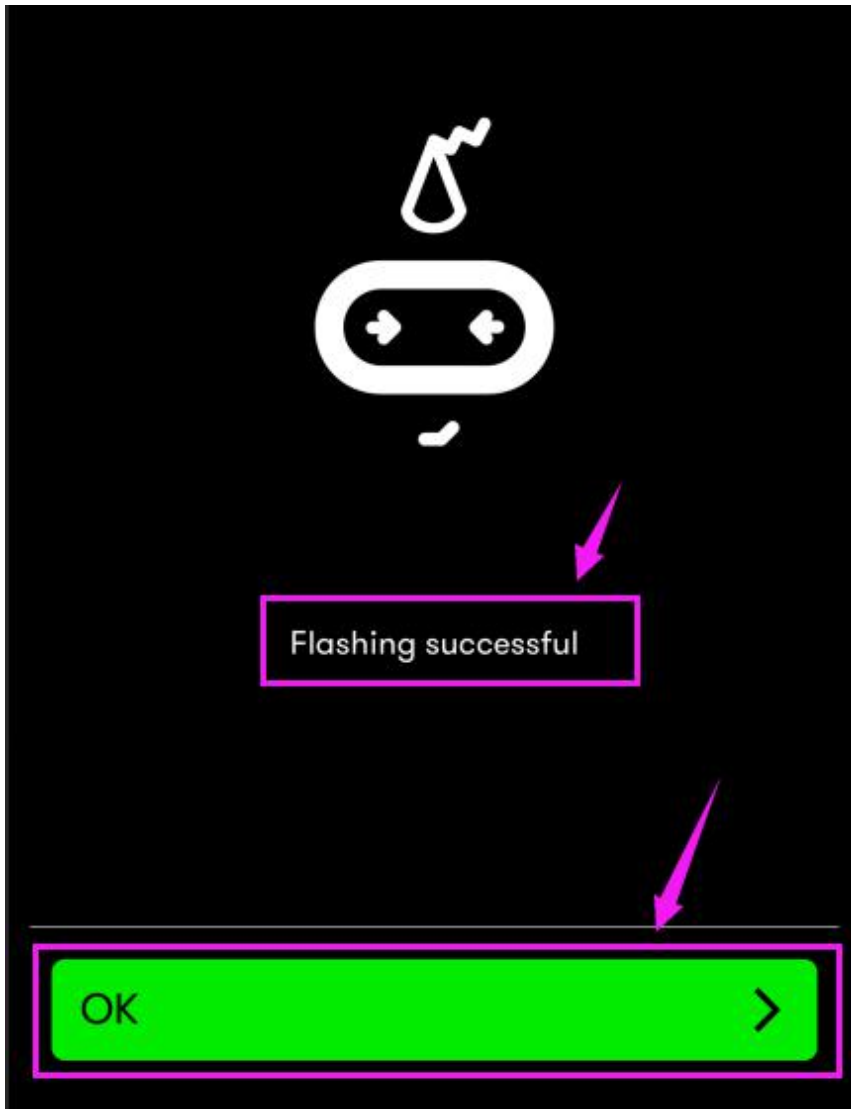
j. Then download the code directly, default the name of saved program as "1" and click "Flash" .





K. Upload the program successfully, as shown below:

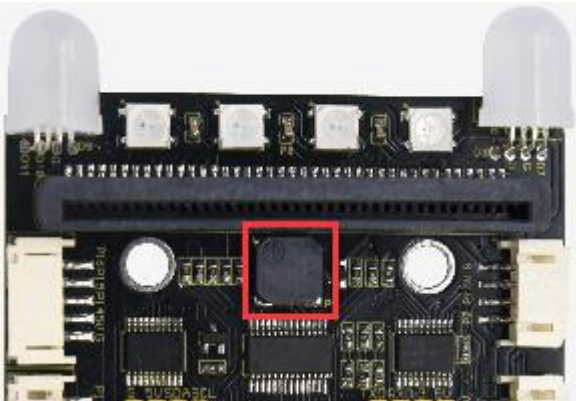




Caution: Disconnect power and turn off the switch at the back of micro: bit car when installing or removing micro:bit board on smart car.



## 7.10: Passive Sensor



### 1. Description:

We can use Micro:bit board to make many interactive works of which the most commonly used is acoustic-optic display. The previous lessons are related to LED. However, we will elaborate the Sound in this lesson.

Buzzer is inclusive of active buzzer and passive buzzer.

The passive buzzer doesn't carry with vibrator inside, so it need external sine or square wave to drive. It can produce slight sound when connecting directly to power supply. It features controlling sound frequency and producing the sound of "do re mi fa so la si" .

A diode should be connected in reverse when driving by the square wave signal source, which will hinder the high-voltage generated to damage other components or service life when the power breaks down.

Frequency is made of a series of pitch names in English letters and Numbers. You can choose different frequencies, that is, tone. The



frequency of sound is called pitch.

It involves music knowledge. In music lesson, our teacher taught "1 (Do) , 2 (Re) , 3(Mi), 4(Fa) , 5(Sol), 6(La), 7(Si)"

1 (Do)	2 (Re)	3(Mi)	4(Fa)	5(Sol)	6(La)	7(Si)
C	D	E	F	G	A	B

The number depends on high or low tone. The larger the number, the higher the tone. When the number is same, the frequency (tone) is getting higher and higher from C to \_B.

Beats are the time delay for each note. The larger the number, the longer the delay time. A note without a line in the spectrum is a beat, with a delay of 1000 milliseconds. while a beat with an underline is 1/2 of a beat without a line, and a beat with two underlines is 1/4 of a beat without a line.

1/4Beat 1Beat 1/2Beat

( 1   1   1 )

Here is the notation of Ode to Joy.



# Ode To Joy

Beethoven

1 =  $\flat$ B  $\frac{2}{4}$  ♩ = 120

[1]  
*f* 3̇ 3̇ 4̇ 5̇ | 5̇ 4̇ 3̇ 2̇ | 1̇ 1̇ 2̇ 3̇ | 3̇· 2̇2̇ 0 | 3̇ 3̇ 4̇ 5̇ |  
 5̇ 4̇ 3̇ 2̇ | 1̇ 1̇ 2̇ 3̇ | 2̇· 1̇1̇ 0 | 2̇ 2̇ 3̇ 1̇ |  
*mp* *crese*  
 2̇ 3̇4̇3̇ 1̇ | 2̇ 3̇4̇3̇ 2̇ | 1̇ 2̇ 5̇ 3̇<sup>V></sup> | 3̇ 3̇ 4̇ 5̇ |  
*f*  
 5̇ 4̇ 3̇ 2̇ | 1̇ 1̇ 2̇ 3̇ | 2̇· 1̇1̇ 0 | 2̇ 2̇ 3̇ 1̇ |  
*mp* *crese*  
 2̇ 3̇4̇3̇ 1̇ | 2̇ 3̇4̇3̇ 2̇ | 1̇ 2̇ 5̇ 3̇<sup>V></sup> | 3̇ 3̇ 4̇ 5̇ |  
*f*  
 5̇ 4̇ 3̇ 2̇ | 1̇ 1̇ 2̇ 3̇ | 2̇· 1̇1̇ 0 :|| 2̇· 1̇1̇ 5̇<sup>V></sup> |  
 4̇· 3̇3̇ 1̇<sup>V></sup> | 7̇· 6̇6̇ 4̇2̇ | 1̇7̇2̇7̇6̇5̇6̇7̇ | 1̇3̇2̇7̇1̇ 1̇·1̇ |  
 1̇ 0 0 0 ||

## 2. Experimental Preparation:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial POWER switch to ON end
- (4) Connect micro:bit to computer by USB cable and open online Makecode editor.



Import Hex profile [\(How to import?\)](#) , or click "New Project" and drag blocks step by step(add turtle-bit extension library first)

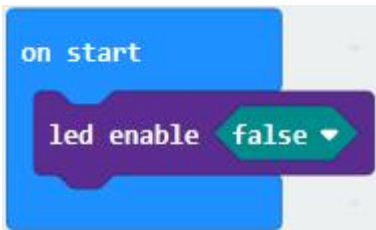
[\(How to add turtle-bit extension?\)](#)

### 3. Test Code:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.10: Passive Sensor	microbit-Passive Sensor.hex

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

(1) Click "Led" → " more" → "led enable false" , combine it with "on start"



start" .

\*\*\*\*\*

(2) Enter "Music" → "play tone Middle C for 1 beat" , leave it into "forever" block , then tap "Middle C" , then



appear

code.

Choose "High E" and set to "1 beat" .



(3) According to the above music score. Copy "play tone High E for 1 beat" 124 times, separately change "High E" of "play tone High E for 1 beat" into "High E" , "High F" , "High G" , "High G" , "High F" , "High E" , "High D" , "High C" , "High C" , "High D" , "High E" , "High E" , "High D" , "High D" , "High E" , "High E" , "High F" , "High G" , "High G" , "High F" , "High





"1" , "1" , "1" , "1" , "1" , "1" , "1" , "1" , "1" , "1" , "1" , "1" , "1" ,  
"1" , "1" , "1" , "1" , "1" , "1" , "1/2" , "1" , "1" , "1/2" , "1" , "1" , "1" ,  
"1/2" , "1" , "1" , "1" , "1/2" , "1" , "1/2" , "1/2" , "1/2" , "1/2" , "1/2" ,  
"1/2" , "1/2" , "1/2" , "1/2" , "1/2" , "1/2" , "1/2" , "1/2" , "1/2" , "1" ,  
"1/2" , "1/4" , "1" .

### Complete Program:

The image shows a Scratch code editor with two columns of code. The first column contains the following blocks:

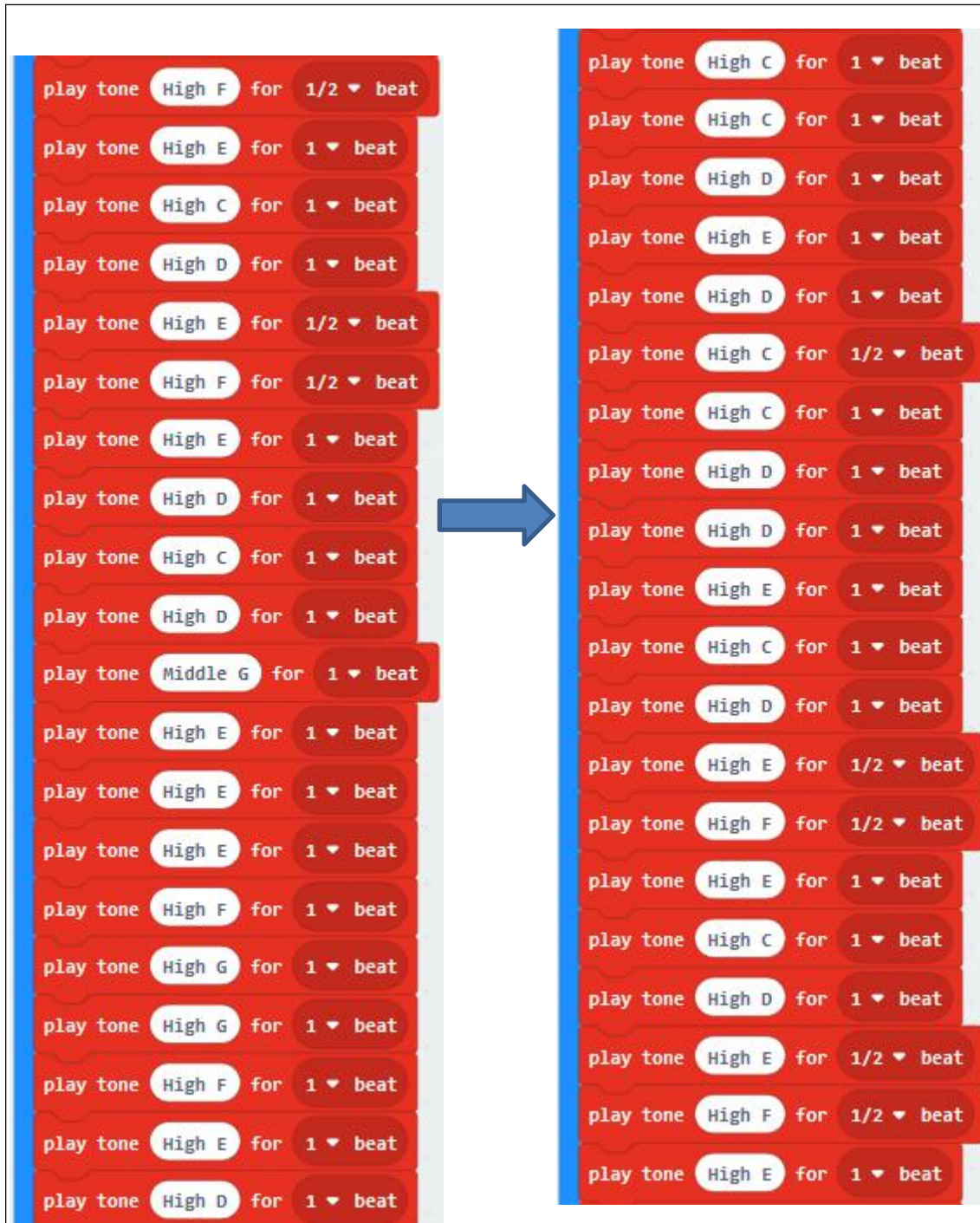
- on start
- led enable false
- forever loop:
  - play tone High E for 1 beat
  - play tone High E for 1 beat
  - play tone High F for 1 beat
  - play tone High G for 1 beat
  - play tone High G for 1 beat
  - play tone High F for 1 beat
  - play tone High E for 1 beat
  - play tone High D for 1 beat
  - play tone High C for 1 beat
  - play tone High C for 1 beat
  - play tone High D for 1 beat
  - play tone High E for 1 beat
  - play tone High E for 1 beat
  - play tone High D for 1/2 beat
  - play tone High D for 1 beat
  - play tone High E for 1 beat

The second column contains the following blocks:

- play tone High E for 1 beat
- play tone High F for 1 beat
- play tone High G for 1 beat
- play tone High G for 1 beat
- play tone High F for 1 beat
- play tone High E for 1 beat
- play tone High D for 1 beat
- play tone High C for 1 beat
- play tone High C for 1 beat
- play tone High D for 1 beat
- play tone High E for 1 beat
- play tone High D for 1 beat
- play tone High C for 1/2 beat
- play tone High C for 1 beat
- play tone High D for 1 beat
- play tone High E for 1 beat
- play tone High D for 1 beat
- play tone High E for 1 beat
- play tone High C for 1 beat
- play tone High D for 1 beat
- play tone High E for 1/2 beat

A blue arrow points from the first column to the second, indicating a transformation or continuation of the code.





The diagram illustrates a transformation of a Scratch script. A blue arrow points from the original script on the left to the modified script on the right. The original script consists of 20 'play tone' blocks with the following notes and durations: High F (1/2 beat), High E (1 beat), High C (1 beat), High D (1 beat), High E (1/2 beat), High F (1/2 beat), High E (1 beat), High D (1 beat), High C (1 beat), High D (1 beat), Middle G (1 beat), High E (1 beat), High E (1 beat), High E (1 beat), High F (1 beat), High G (1 beat), High G (1 beat), High F (1 beat), High E (1 beat), and High D (1 beat). The modified script consists of 20 'play tone' blocks with the following notes and durations: High C (1 beat), High C (1 beat), High D (1 beat), High E (1 beat), High D (1 beat), High C (1/2 beat), High C (1 beat), High D (1 beat), High D (1 beat), High E (1 beat), High C (1 beat), High D (1 beat), High E (1/2 beat), High F (1/2 beat), High E (1 beat), High C (1 beat), High D (1 beat), High E (1/2 beat), High F (1/2 beat), and High E (1 beat).

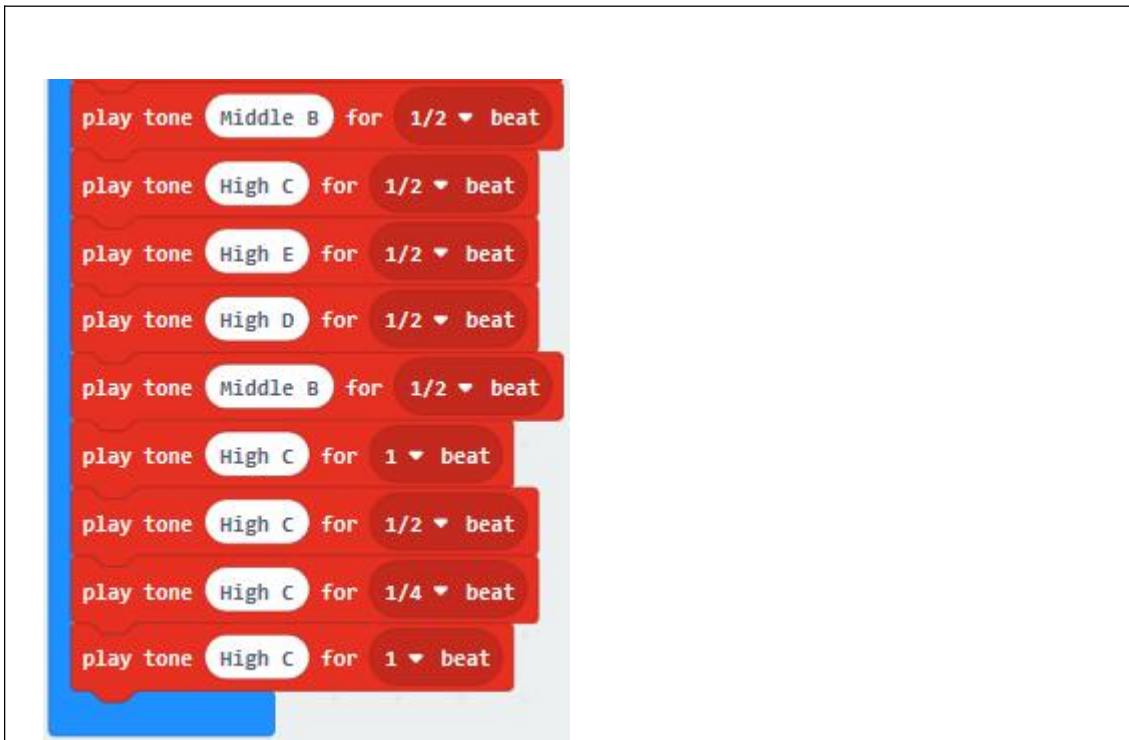


The diagram illustrates a transformation of a musical sequence. On the left, there are 20 red blocks, each containing the text "play tone [Note] for [Duration] beat". A blue arrow points from this sequence to a second sequence of 20 red blocks on the right, which contains modified notes and durations.

Block Index	Note	Duration
1	High D	1 beat
2	High C	1 beat
3	High D	1 beat
4	Middle G	1 beat
5	High E	1 beat
6	High E	1 beat
7	High E	1 beat
8	High F	1 beat
9	High G	1 beat
10	High G	1 beat
11	High F	1 beat
12	High E	1 beat
13	High C	1 beat
14	High C	1 beat
15	High C	1 beat
16	High D	1 beat
17	High E	1 beat
18	High D	1 beat
19	High C	1/2 beat
20	High C	1 beat

Block Index	Note	Duration
1	High D	1 beat
2	High C	1/2 beat
3	High C	1 beat
4	High G	1 beat
5	High F	1 beat
6	High E	1/2 beat
7	High E	1 beat
8	High C	1 beat
9	High B	1 beat
10	High A	1/2 beat
11	High A	1 beat
12	High F	1/2 beat
13	High D	1/2 beat
14	High C	1/2 beat
15	Middle B	1/2 beat
16	High D	1/2 beat
17	Middle B	1/2 beat
18	Middle A	1/2 beat
19	Middle G	1/2 beat
20	Middle A	1/2 beat



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



```
1 led.enable(false)
2 basic.forever(function () {
3     music.playTone(659, music.beat(BeatFraction.Whole))
4     music.playTone(659, music.beat(BeatFraction.Whole))
5     music.playTone(698, music.beat(BeatFraction.Whole))
6     music.playTone(784, music.beat(BeatFraction.Whole))
7     music.playTone(784, music.beat(BeatFraction.Whole))
8     music.playTone(698, music.beat(BeatFraction.Whole))
9     music.playTone(659, music.beat(BeatFraction.Whole))
10    music.playTone(587, music.beat(BeatFraction.Whole))
11    music.playTone(523, music.beat(BeatFraction.Whole))
12    music.playTone(523, music.beat(BeatFraction.Whole))
13    music.playTone(587, music.beat(BeatFraction.Whole))
14    music.playTone(659, music.beat(BeatFraction.Whole))
15    music.playTone(659, music.beat(BeatFraction.Whole))
16    music.playTone(587, music.beat(BeatFraction.Half))
17    music.playTone(587, music.beat(BeatFraction.Whole))
18    music.playTone(659, music.beat(BeatFraction.Whole))
19    music.playTone(659, music.beat(BeatFraction.Whole))
20    music.playTone(698, music.beat(BeatFraction.Whole))
21    music.playTone(784, music.beat(BeatFraction.Whole))
22    music.playTone(784, music.beat(BeatFraction.Whole))
23    music.playTone(698, music.beat(BeatFraction.Whole))
24    music.playTone(659, music.beat(BeatFraction.Whole))
25    music.playTone(587, music.beat(BeatFraction.Whole))
26    music.playTone(523, music.beat(BeatFraction.Whole))
27    music.playTone(523, music.beat(BeatFraction.Whole))
28    music.playTone(587, music.beat(BeatFraction.Whole))
29    music.playTone(659, music.beat(BeatFraction.Whole))
30    music.playTone(587, music.beat(BeatFraction.Whole))
31    music.playTone(523, music.beat(BeatFraction.Half))
32    music.playTone(523, music.beat(BeatFraction.Whole))
33    music.playTone(587, music.beat(BeatFraction.Whole))
34    music.playTone(587, music.beat(BeatFraction.Whole))
35    music.playTone(659, music.beat(BeatFraction.Whole))
36    music.playTone(523, music.beat(BeatFraction.Whole))
37    music.playTone(587, music.beat(BeatFraction.Whole))
38    music.playTone(659, music.beat(BeatFraction.Half))
39    music.playTone(698, music.beat(BeatFraction.Half))
40    music.playTone(659, music.beat(BeatFraction.Whole))
41    music.playTone(523, music.beat(BeatFraction.Whole))
42    music.playTone(587, music.beat(BeatFraction.Whole))
43    music.playTone(659, music.beat(BeatFraction.Half))
```



```
44 music.playTone(698, music.beat(BeatFraction.Half))
45 music.playTone(659, music.beat(BeatFraction.Whole))
46 music.playTone(587, music.beat(BeatFraction.Whole))
47 music.playTone(523, music.beat(BeatFraction.Whole))
48 music.playTone(587, music.beat(BeatFraction.Whole))
49 music.playTone(392, music.beat(BeatFraction.Whole))
50 music.playTone(659, music.beat(BeatFraction.Whole))
51 music.playTone(659, music.beat(BeatFraction.Whole))
52 music.playTone(659, music.beat(BeatFraction.Whole))
53 music.playTone(698, music.beat(BeatFraction.Whole))
54 music.playTone(784, music.beat(BeatFraction.Whole))
55 music.playTone(784, music.beat(BeatFraction.Whole))
56 music.playTone(698, music.beat(BeatFraction.Whole))
57 music.playTone(659, music.beat(BeatFraction.Whole))
58 music.playTone(587, music.beat(BeatFraction.Whole))
59 music.playTone(523, music.beat(BeatFraction.Whole))
60 music.playTone(523, music.beat(BeatFraction.Whole))
61 music.playTone(587, music.beat(BeatFraction.Whole))
62 music.playTone(659, music.beat(BeatFraction.Whole))
63 music.playTone(587, music.beat(BeatFraction.Whole))
64 music.playTone(523, music.beat(BeatFraction.Half))
65 music.playTone(523, music.beat(BeatFraction.Whole))
66 music.playTone(587, music.beat(BeatFraction.Whole))
67 music.playTone(587, music.beat(BeatFraction.Whole))
68 music.playTone(659, music.beat(BeatFraction.Whole))
69 music.playTone(523, music.beat(BeatFraction.Whole))
70 music.playTone(587, music.beat(BeatFraction.Whole))
71 music.playTone(659, music.beat(BeatFraction.Half))
72 music.playTone(698, music.beat(BeatFraction.Half))
73 music.playTone(659, music.beat(BeatFraction.Whole))
74 music.playTone(523, music.beat(BeatFraction.Whole))
75 music.playTone(587, music.beat(BeatFraction.Whole))
76 music.playTone(659, music.beat(BeatFraction.Half))
77 music.playTone(698, music.beat(BeatFraction.Half))
78 music.playTone(659, music.beat(BeatFraction.Whole))
79 music.playTone(587, music.beat(BeatFraction.Whole))
80 music.playTone(523, music.beat(BeatFraction.Whole))
81 music.playTone(587, music.beat(BeatFraction.Whole))
82 music.playTone(392, music.beat(BeatFraction.Whole))
83 music.playTone(659, music.beat(BeatFraction.Whole))
84 music.playTone(659, music.beat(BeatFraction.Whole))
85 music.playTone(659, music.beat(BeatFraction.Whole))
86 music.playTone(698, music.beat(BeatFraction.Whole))
```



```
87 music.playTone(784, music.beat(BeatFraction.Whole))
88 music.playTone(784, music.beat(BeatFraction.Whole))
89 music.playTone(698, music.beat(BeatFraction.Whole))
90 music.playTone(659, music.beat(BeatFraction.Whole))
91 music.playTone(523, music.beat(BeatFraction.Whole))
92 music.playTone(523, music.beat(BeatFraction.Whole))
93 music.playTone(523, music.beat(BeatFraction.Whole))
94 music.playTone(587, music.beat(BeatFraction.Whole))
95 music.playTone(659, music.beat(BeatFraction.Whole))
96 music.playTone(587, music.beat(BeatFraction.Whole))
97 music.playTone(523, music.beat(BeatFraction.Half))
98 music.playTone(523, music.beat(BeatFraction.Whole))
99 music.playTone(587, music.beat(BeatFraction.Whole))
100 music.playTone(523, music.beat(BeatFraction.Half))
101 music.playTone(523, music.beat(BeatFraction.Whole))
102 music.playTone(784, music.beat(BeatFraction.Whole))
103 music.playTone(698, music.beat(BeatFraction.Whole))
104 music.playTone(659, music.beat(BeatFraction.Half))
105 music.playTone(659, music.beat(BeatFraction.Whole))
106 music.playTone(523, music.beat(BeatFraction.Whole))
107 music.playTone(988, music.beat(BeatFraction.Whole))
108 music.playTone(880, music.beat(BeatFraction.Half))
109 music.playTone(880, music.beat(BeatFraction.Whole))
110 music.playTone(698, music.beat(BeatFraction.Half))
111 music.playTone(587, music.beat(BeatFraction.Half))
112 music.playTone(523, music.beat(BeatFraction.Half))
113 music.playTone(494, music.beat(BeatFraction.Half))
114 music.playTone(587, music.beat(BeatFraction.Half))
115 music.playTone(494, music.beat(BeatFraction.Half))
116 music.playTone(440, music.beat(BeatFraction.Half))
117 music.playTone(392, music.beat(BeatFraction.Half))
118 music.playTone(440, music.beat(BeatFraction.Half))
119 music.playTone(494, music.beat(BeatFraction.Half))
120 music.playTone(523, music.beat(BeatFraction.Half))
121 music.playTone(659, music.beat(BeatFraction.Half))
122 music.playTone(587, music.beat(BeatFraction.Half))
123 music.playTone(494, music.beat(BeatFraction.Half))
124 music.playTone(523, music.beat(BeatFraction.Whole))
125 music.playTone(523, music.beat(BeatFraction.Half))
126 music.playTone(523, music.beat(BeatFraction.Quarter))
127 music.playTone(523, music.beat(BeatFraction.Whole))
128 })
129
```

#### 4 .Test Result:

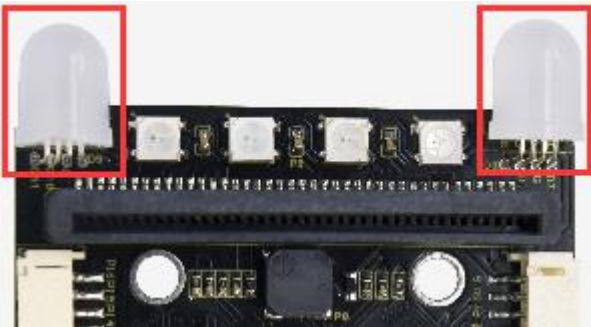
Download code to micro:bit board and dial POWER switch to ON end. The



“Ode to Joy” is played by passive buzzer

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

## 7.11: RGB Experiments



### 1. Description:

The RGB color mode is a color standard in the industry. It obtains various colors by changing the three color channels of red (R), green (G), and blue (B) and integrating them. RGB denotes the three colors of red, green and blue.

The monitors mostly adopt the RGB color standard, and all the colors on the computer screen are composed of the three colors of red, green and blue mixed in different proportions. A group of red, green and blue is the smallest display unit. Any color on the screen can be recorded and expressed by a set of RGB values.

Each of the three color channels of red, green, and blue is divided into 256 levels of brightness. At 0, the "light" is the weakest-it is turned off, and at



255, the "light" is the brightest. When the three-color gray values are the same, the gray tones with different gray values are produced, that is, when the three-color gray is 0, the darkest black is generated; when the three-color gray is 255, it is the brightest white tone .

Color	RGB value (R,G,B)	Color code	Color	RGB value (R,G,B)	Color code
Black	0,0,0	#000000	Red	255,0,0	#FF0000
Green	0,255,0	#00FF00	Blue	0,0,255	#0000FF
indigo	0,255,255	#00FFFF	Dark red	255,0,255	#FF00FF
Yellow	255,255,0	#FFFF00	White	255,255,255	#FFFFFF
.....	.....	.....	.....	.....	.....
Adjust the numbers to get gradient colors					

RGB colors are called additive colors since the adding of R, G, and B together (that is, all light reflect back to the eye) produces white color. Additive colors are used for lighting, television and computer displays. For example, displays produce color by emitting red, green, and blue rays. Most visible spectra can be expressed as a mixture of red, green and blue (RGB) light in different proportions and intensities. If these colors overlap, they produce cyan, magenta and yellow.

We will make two experiments, one is that two RGB LEDs light up red,





green, blue, indigo, dark red, yellow and white color, another one is that RGB lights display color in gradient way.

## 2. Experimental Preparation:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end
- (4) Connect micro:bit to computer by USB cable and open online Makecode editor.

Import Hex profile [\(How to import?\)](#), or click "New Project" and drag blocks step by step([add turtle-bit extension library first](#))

[\(How to add turtle-bit extension?\)](#)

## 3. Test Code:

### Code 1

RGB lights show seven colors



Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.11: RGB Experiments/Code-1	microbit-Code-1.hex

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

(1) Click "TurtleBit" to drag "LED brightness 0" into "on start" block

The larger the number you set, the brighter the RGB gets

Here, we set to 70



(2) Click "TurtleBit" to drag "set RGBled left\_side R: 0 G: 0 B: 0" block into "forever" "forever"

Set R:255 G:0 B:0



(3)Click "TurtleBit" to move "set RGBled left\_side R: 0 G: 0 B: 0" into "forever" block.

Tap left\_side to choose right\_side, change R:0 into R:255



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

Delay in 1000ms





(5) Copy code string for six times and separately place

them into "forever" block.

Respectively set to "R: 0 G: 255 B: 0" , " R: 0 G: 0 B: 255" , " R: 0 G: 255 B: 255" , " R: 255 G: 0 B: 255" , " R: 255 G: 255 B: 0" and " R: 255 G: 0 B: 255" .



```
forever
  set RGBled left_side
  R: 255
  G: 0
  B: 0
  set RGBled right_side
  R: 255
  G: 0
  B: 0
  pause (ms) 1000
  set RGBled left_side
  R: 0
  G: 255
  B: 0
  set RGBled right_side
  R: 0
  G: 255
  B: 0
  pause (ms) 1000
  set RGBled left_side
  R: 0
  G: 0
  B: 255
  set RGBled right_side
  R: 0
  G: 0
  B: 255
  pause (ms) 1000
```

```
set RGBled left_side
R: 0
G: 255
B: 255
set RGBled right_side
R: 0
G: 255
B: 255
pause (ms) 1000
set RGBled left_side
R: 255
G: 0
B: 255
set RGBled right_side
R: 255
G: 0
B: 255
pause (ms) 1000
set RGBled left_side
R: 255
G: 255
B: 0
set RGBled right_side
R: 255
G: 255
B: 0
pause (ms) 1000
```



```
set RGBled left_side ▾  
R: 255  
G: 255  
B: 255  
set RGBled right_side ▾  
R: 255  
G: 255  
B: 255  
pause (ms) 1000 ▾
```

Complete Code



The image displays a Scratch script for controlling two RGB LEDs. The script is organized into three vertical columns of code blocks.

- Column 1 (Left):**
  - on start** block: LED brightness 70.
  - forever** loop:
    - set RGBled left\_side** block: R: 255, G: 0, B: 0.
    - set RGBled right\_side** block: R: 255, G: 0, B: 0.
    - pause (ms)** block: 1000.
    - set RGBled left\_side** block: R: 0, G: 255, B: 0.
    - set RGBled right\_side** block: R: 0, G: 255, B: 0.
    - pause (ms)** block: 1000.
- Column 2 (Middle):**
  - set RGBled left\_side** block: R: 0, G: 0, B: 255.
  - set RGBled right\_side** block: R: 0, G: 0, B: 255.
  - pause (ms)** block: 1000.
  - set RGBled left\_side** block: R: 0, G: 255, B: 255.
  - set RGBled right\_side** block: R: 0, G: 255, B: 255.
  - pause (ms)** block: 1000.
  - set RGBled left\_side** block: R: 255, G: 0, B: 255.
  - set RGBled right\_side** block: R: 255, G: 0, B: 255.
  - pause (ms)** block: 1000.
- Column 3 (Right):**
  - set RGBled left\_side** block: R: 255, G: 255, B: 0.
  - set RGBled right\_side** block: R: 255, G: 255, B: 0.
  - pause (ms)** block: 1000.
  - set RGBled left\_side** block: R: 255, G: 255, B: 255.
  - set RGBled right\_side** block: R: 255, G: 255, B: 255.
  - pause (ms)** block: 1000.



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

Set the brightness of two RGB lights on car to 70

The program under the block “forever” runs cyclically.

Set R: 255 G: 0 B: 0 to make left RGB light display red color

Set R: 255 G: 0 B: 0 to make right RGB light display red color

Delay in 1000ms

Set R: 0 G: 255 B: 0 to make left RGB light display green color

Set R: 0 G: 255 B: 0 to make right RGB light display green color

Delay in 1000ms

Set R: 0 G: 0 B: 255 to make the left RGB light display blue color

Set R: 0 G: 0 B: 255 to make the right RGB light display blue color

Delay in 1000m

Set R: 0 G: 255 B: 255 to make the left RGB light display indigo color

Set R: 0 G: 255 B: 255 to make the right RGB light display indigo color

Delay in 1000ms

Set R: 255 G: 0 B: 255 to make the left RGB light display dark red color

Set R: 255 G: 0 B: 255 to make the right RGB light display dark red color

Delay in 1000ms

Set R: 255 G: 255 B: 0 to make the left RGB light display yellow color

Set R: 255 G: 255 B: 0 to make the right RGB light display yellow color

Delay in 1000ms

Set R: 255 G: 255 B: 255 to make the left RGB light display white color

Set R: 255 G: 255 B: 255 to make the right RGB light display white color

Delay in 1000ms





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

The screenshot shows the Keyestudio IDE interface. At the top, there is a blue header bar with a "Blocks" tab and a "JavaScript" tab. The "JavaScript" tab is selected and highlighted with a red box and a red arrow pointing to it. Below the header, there is a search bar and a sidebar on the left with various modules: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, TurtleBit, IrRemote, Neopixel, and Advanced. The main area is a code editor showing JavaScript code for controlling an LED strip. The code is as follows:

```
1 turtleBit.LED_brightness(70)
2 basic.forever(function () {
3     turtleBit.SetLed(
4         RGBLED.left_side,
5         255,
6         0,
7         0
8     )
9     turtleBit.SetLed(
10        RGBLED.right_side,
11        255,
12        0,
13        0
14    )
15    basic.pause(1000)
16    turtleBit.SetLed(
17        RGBLED.left_side,
18        0,
19        255,
20        0
21    )
22    turtleBit.SetLed(
23        RGBLED.right_side,
24        0,
25        255,
26        0
27    )
28    basic.pause(1000)
29    turtleBit.SetLed(
30        RGBLED.left_side,
31        0,
32        0,
33        255
34    )
35    turtleBit.SetLed(
36        RGBLED.right_side,
37        0,
38        0,
39        255
40    )
41    basic.pause(1000)
```



```
42 turtleBit.SetLed(  
43 RGBLED.left_side,  
44 0,  
45 255,  
46 255  
47 )  
48 turtleBit.SetLed(  
49 RGBLED.right_side,  
50 0,  
51 255,  
52 255  
53 )  
54 basic.pause(1000)  
55 turtleBit.SetLed(  
56 RGBLED.left_side,  
57 255,  
58 0,  
59 255  
60 )  
61 turtleBit.SetLed(  
62 RGBLED.right_side,  
63 255,  
64 0,  
65 255  
66 )  
67 basic.pause(1000)  
68 turtleBit.SetLed(  
69 RGBLED.left_side,  
70 255,  
71 255,  
72 0  
73 )  
74 turtleBit.SetLed(  
75 RGBLED.right_side,  
76 255,  
77 255,  
78 0  
79 )  
80 basic.pause(1000)
```



```
81 turtleBit.SetLed(  
82   RGBLED.left_side,  
83   255,  
84   255,  
85   255  
86 )  
87 turtleBit.SetLed(  
88   RGBLED.right_side,  
89   255,  
90   255,  
91   255  
92 )  
93 basic.pause(1000)  
94 })  
95
```

### Code 2:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.11: RGB Experiments/Code-2	microbit-Code-2.hex

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

- (1) Go to "TurtleBit" to move block "LED brightness 0" into "on start"  
Change 0 into 200



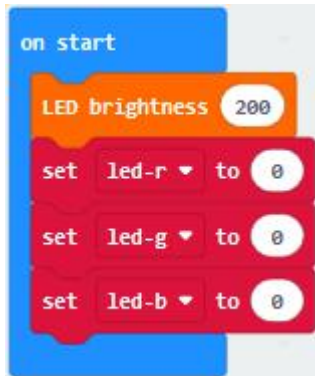
- (2) Go to "Variables" → "Make a Variable..." → "New variable name: " dialog box,



(3) Input "led-r" and click "OK" to produce variable "led-r" ,  
Then set variable "led-g" and "led-b" in same way.

Move block "set led-b to 0" into "on start"

Copy "set led-b to 0" block twice and set to led-r, led\_g and led\_b



(4) Go to "Loops" to drag block "repeat 4 times do" into "forever"



(5) Tap "TurtleBit" to drag block "set RGBled left\_side R: 0 G: 0 B: 0" into  
"repeat 4 times do" block.

Copy it once, and change blocks as follows:



```
forever
  repeat 4 times
    do
      set RGBled left_side
      R: led-r
      G: 0
      B: 0
      set RGBled right_side
      R: led-r
      G: 0
      B: 0
```

(6) Move block "pause (ms) 100" from "Basic" and place it into "repeat 4 times do" .

```
forever
  repeat 4 times
    do
      set RGBled left_side
      R: led-r
      G: 0
      B: 0
      set RGBled right_side
      R: led-r
      G: 0
      B: 0
      pause (ms) 100
```



(7) Enter "Variables" to move block "change led-b by 1" under block "pause (ms) 100" .

Click triangle to change led-b into led-r.

The R value is in the range of 0-255, and we make variable "led-r" increases by 5 every time. Therefore, 51 times in total.

Set "repeat 51 times" and "by 5" .





(8) Copy code string

```
repeat 51 times
do
set RGBled left_side
R: led-r
G: 0
B: 0
set RGBled right_side
R: led-r
G: 0
B: 0
pause (ms) 100
change led-r by 5
```

once and leave it into "forever" .

To make RGB get darker gradually, we set "led-r by -5" , 51 times in total  
So we change 5 into -5.



```
forever
  repeat 51 times
    do
      set RGBled left_side
      R: led-r
      G: 0
      B: 0
      set RGBled right_side
      R: led-r
      G: 0
      B: 0
      pause (ms) 100
      change led-r by 5
    repeat 51 times
      do
        set RGBled left_side
        R: led-r
        G: 0
        B: 0
        set RGBled right_side
        R: led-r
        G: 0
        B: 0
        pause (ms) 100
        change led-r by -5
```





```
repeat 51 times
do
  set RGBled left_side
  R: led-r
  G: 0
  B: 0
  set RGBled right_side
  R: led-r
  G: 0
  B: 0
  pause (ms) 100
  change led-r by 5

repeat 51 times
do
  set RGBled left_side
  R: led-r
  G: 0
  B: 0
  set RGBled right_side
  R: led-r
  G: 0
  B: 0
  pause (ms) 100
  change led-r by -5
```

(9) Replicate once and keep them into "forever" block.

Set R:0 and G: led-g, as shown below:



```
repeat 51 times
do
  set RGBled left_side
  R: 0
  G: led-g
  B: 0
  set RGBled right_side
  R: 0
  G: led-g
  B: 0
  pause (ms) 100
  change led-g by 5

repeat 51 times
do
  set RGBled left_side
  R: 0
  G: led-g
  B: 0
  set RGBled right_side
  R: 0
  G: led-g
  B: 0
  pause (ms) 100
  change led-g by -5
```



```
repeat 51 times
do
  set RGBled left_side
  R: led-r
  G: 0
  B: 0
  set RGBled right_side
  R: led-r
  G: 0
  B: 0
  pause (ms) 100
  change led-r by 5
```

(10) Copy again, and set code string as follows:

```
repeat 51 times
do
  set RGBled left_side
  R: led-r
  G: 0
  B: 0
  set RGBled right_side
  R: led-r
  G: 0
  B: 0
  pause (ms) 100
  change led-r by -5
```



```
repeat 51 times
do
  set RGBled left_side
  R: 0
  G: 0
  B: led-b
  set RGBled right_side
  R: 0
  G: 0
  B: led-b
  pause (ms) 100
  change led-b by 5

repeat 51 times
do
  set RGBled left_side
  R: 0
  G: 0
  B: led-b
  set RGBled right_side
  R: 0
  G: 0
  B: led-b
  pause (ms) 100
  change led-b by -5
```



## Complete Code

```
on start
  LED brightness 200
  set led-r to 0
  set led-g to 0
  set led-b to 0
```

The image shows a Scratch script starting with an 'on start' block. It contains four blocks: an orange 'LED brightness' block set to 200, and three red 'set' blocks for 'led-r', 'led-g', and 'led-b', each set to 0.



```
forever
repeat 51 times
do
set RGBled left_side
R: led-r
G: 0
B: 0
set RGBled right_side
R: led-r
G: 0
B: 0
pause (ms) 100
change led-r by 5

repeat 51 times
do
set RGBled left_side
R: led-r
G: 0
B: 0
set RGBled right_side
R: led-r
G: 0
B: 0
pause (ms) 100
change led-r by -5
```

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

Set the light intensity of 2 RGB to 200

Set led-r to 0

Set led-g to 0

Set led-b to 0

The program under the block "forever" runs cyclically.

The program in the do block repeats 51 times

Set left RGB: R: led-r G: 0 B: 0

Set right RGB : R: led-r G: 0 B: 0

Delay in 100ms

Change led-r by 5

The program in the do block repeats 51 times

Set left RGB : R: led-r G: 0 B: 0

Set right RGB : R: led-r G: 0 B: 0

Delay in 100ms

Change led-r by -5



```
repeat 51 times
do
  set RGBled left_side
  R: 0
  G: led-g
  B: 0
  set RGBled right_side
  R: 0
  G: led-g
  B: 0
  pause (ms) 100
  change led-g by 5

repeat 51 times
do
  set RGBled left_side
  R: 0
  G: led-g
  B: 0
  set RGBled right_side
  R: 0
  G: led-g
  B: 0
  pause (ms) 100
  change led-g by -5
```

The program in the do block repeats 51 times

Set left RGB: R: 0 G: led-g B: 0

Set right RGB: R: 0 G: led-g B: 0

Delay in 100ms

Change led-g by 5

The program in the do block repeats 51 times

Set left RGB: R: 0 G: led-g B: 0

Set right RGB: R: 0 G: led-g B: 0

Delay in 100ms

Change led-g by -5



```
repeat 51 times
do
  set RGBled left_side
  R: 0
  G: 0
  B: led-b
  set RGBled right_side
  R: 0
  G: 0
  B: led-b
  pause (ms) 100
  change led-b by 5

repeat 51 times
do
  set RGBled left_side
  R: 0
  G: 0
  B: led-b
  set RGBled right_side
  R: 0
  G: 0
  B: led-b
  pause (ms) 100
  change led-b by -5
```

The program in the do block repeats 51 times

Set right RGB: R: 0 G: 0 B: led-b

Set right RGB: R: 0 G: 0 B: led-b

Delay in 100ms

Change led-b by 5

The program in the do block repeats 51 times

Set right RGB: R: 0 G: 0 B: led-b

Set right RGB: R: 0 G: 0 B: led-b

Delay in 100ms

Change led-g by -5





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

```
1 turtleBit.LED_brightness(200)
2 let ledr = 0
3 let ledg = 0
4 let ledb = 0
5 basic.forever(function () {
6   for (let index = 0; index < 51; index++) {
7     turtleBit.SetLed(
8       RGBLED.left_side,
9       ledr,
10      0,
11      0
12     )
13     turtleBit.SetLed(
14       RGBLED.right_side,
15       ledr,
16       0,
17       0
18     )
19     basic.pause(100)
20     ledr += 5
21   }
22   for (let index = 0; index < 51; index++) {
23     turtleBit.SetLed(
24       RGBLED.left_side,
25       ledr,
26       0,
27       0
28     )
29     turtleBit.SetLed(
30       RGBLED.right_side,
31       ledr,
32       0,
33       0
34     )
35     basic.pause(100)
36     ledr += -5
37   }
}
```



```
38     for (let index = 0; index < 51; index++) {
39         turtleBit.SetLed(
40             RGBLED.left_side,
41             0,
42             ledg,
43             0
44         )
45         turtleBit.SetLed(
46             RGBLED.right_side,
47             0,
48             ledg,
49             0
50         )
51         basic.pause(100)
52         ledg += 5
53     }
54     for (let index = 0; index < 51; index++) {
55         turtleBit.SetLed(
56             RGBLED.left_side,
57             0,
58             ledg,
59             0
60         )
61         turtleBit.SetLed(
62             RGBLED.right_side,
63             0,
64             ledg,
65             0
66         )
67         basic.pause(100)
68         ledg += -5
69     }
```



```
70   for (let index = 0; index < 51; index++) {
71     turtleBit.SetLed(
72       RGBLED.left_side,
73       0,
74       0,
75       ledb
76     )
77     turtleBit.SetLed(
78       RGBLED.right_side,
79       0,
80       0,
81       ledb
82     )
83     basic.pause(100)
84     ledb += 5
85   }
86   for (let index = 0; index < 51; index++) {
87     turtleBit.SetLed(
88       RGBLED.left_side,
89       0,
90       0,
91       ledb
92     )
93     turtleBit.SetLed(
94       RGBLED.right_side,
95       0,
96       0,
97       ledb
98     )
99     basic.pause(100)
100    ledb += -5
101  }
102 })
103
```

#### 4.Test Result:

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

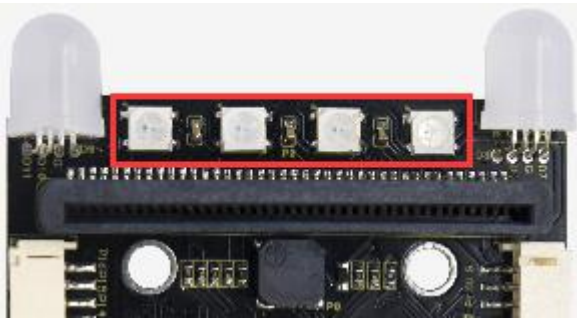
Download code 1 to micro:bit board and dial POWER switch to ON end, 2 RGB lights of smart car emit red, green, blue, indigo, dark red, yellow and white color cyclically.

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



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

## 7.12: WS2812 RGB



### 1.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

### 2. Experimental Preparation:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end
- (4) Connect micro:bit to computer by USB cable and open online Makecode editor.

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

[\(How to add turtle-bit extension?\)](#)



### 3. Test Code:

#### Code 1:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.12: WS2812 RGB/Code-1	microbit-Code-1.hex

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

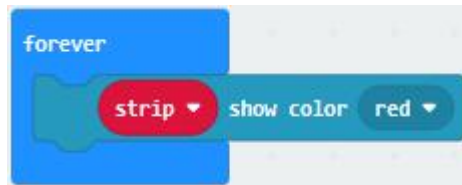
- (1) a. Enter "Neopixel" → "set strip to Neopixel at pin P0 with 24 leds as RGB (GRB format)"
- b. Place it into "on start" block,
- c. Signal end P8 of WS2812 RGB is controlled by P8 of micro:bit . So we set to P8.
- d. Smart car has 4 pcs WS2812 RGB lights, so set to 4 leads



- (2) Click "Neopixel" to move block "strip clear" into "on start" block.

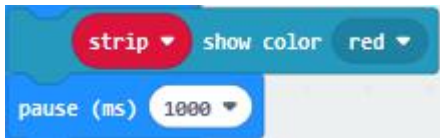


(3) Enter "Neopixel" to move block "strip show color red" into "forever" block



(4) Click "Basic" to move "pause (ms) 100" block into "forever" block  
Then set to 1000ms



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

(6) 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

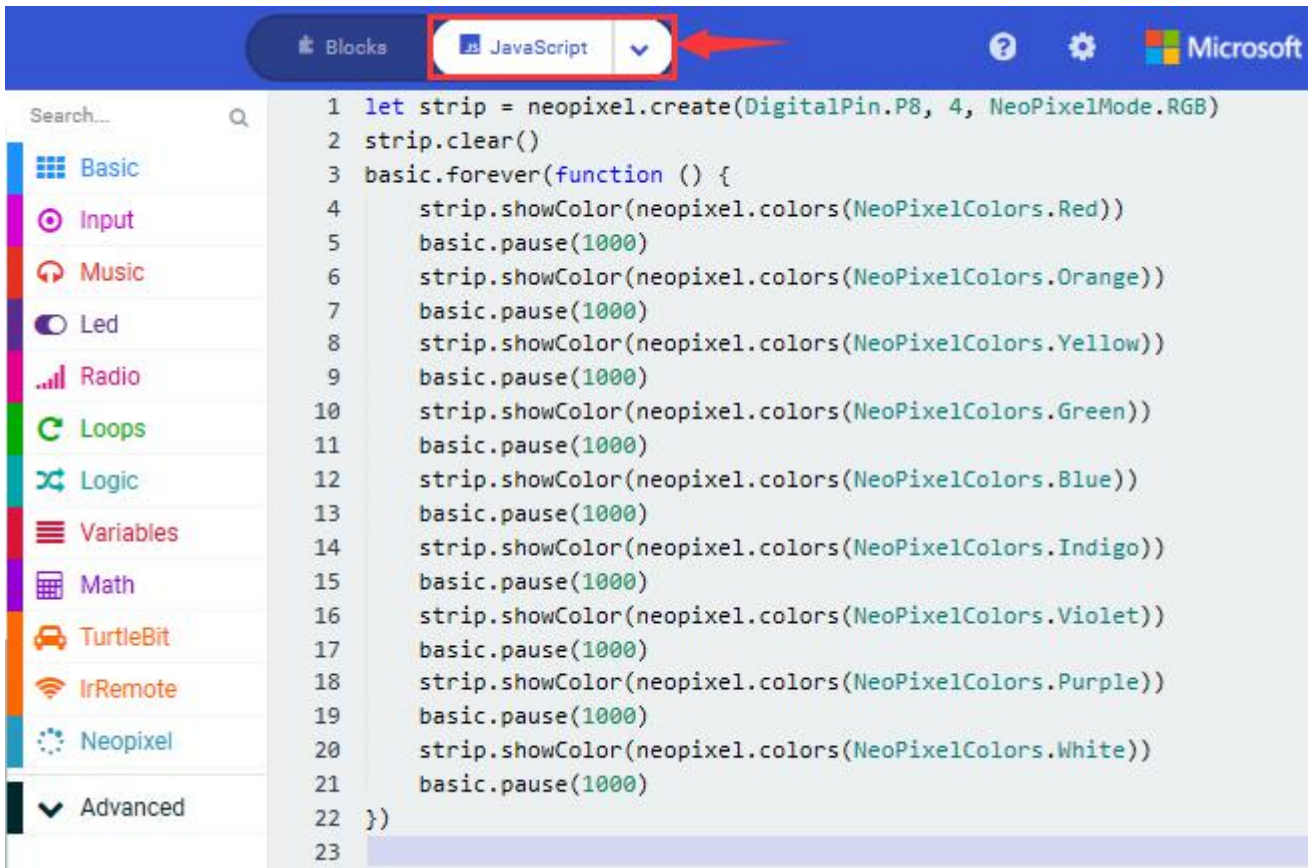
forever
  strip show color red
  pause (ms) 1000
  strip show color orange
  pause (ms) 1000
  strip show color yellow
  pause (ms) 1000
  strip show color green
  pause (ms) 1000
  strip show color blue
  pause (ms) 1000
  strip show color indigo
  pause (ms) 1000
  strip show color violet
  pause (ms)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



### Code 2:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.12: WS2812 RGB/Code-2	microbit-Code-2.hex

(1) 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



(2) Click "Loops" to drag "for index from 0 to 4...do" into "forever" block  
Change 4 into 3



(3) Click "Neopixel" to move block "strip clear" into block "for index from 0 to 3...do"



(4) 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

```
forever
  for index from 0 to 3
  do
    strip clear
    strip set pixel color at index to red
```

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

```
forever
  for index from 0 to 3
  do
    strip clear
    strip set pixel color at index to red
    strip show
```

(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
```



(7) Replicate code string

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

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:

```
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:**

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.12: WS2812 RGB/Code-3	microbit-Code-3.hex

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

- (1) 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, set to 4 leads



(2) 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



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

Leave it into "forever" and change 4 into 3






(4) Move block "set B to 0" into "for index from 0 to 3...do" block,  
Click B to choose R

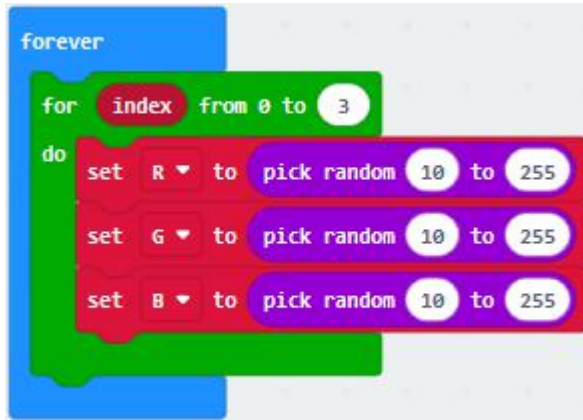
Go to "Math" to drag block "pick random 0 to 10" into 0 box

Change 0 into 10, 10 into 255

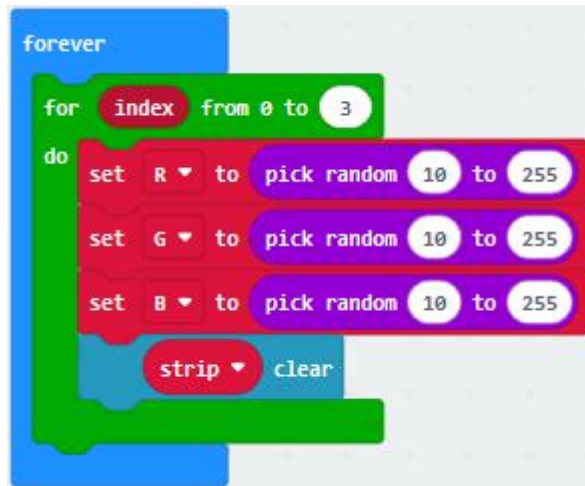


(5) Replicate block  twice and place them into "for index from 0 to 3...do" block.

Click R to select G and B



(6) Tap "Neopixel" and move "strip clear" into "for index from 0 to 3...do" block.



(7) 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:



(8) Click "Basic" to drag "pause (ms) 100" under block "strip....B"

Set to 500ms.



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

(9) Click "Neopixel" to move "strip show" block under "pause(as) 500"

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

Complete Code:





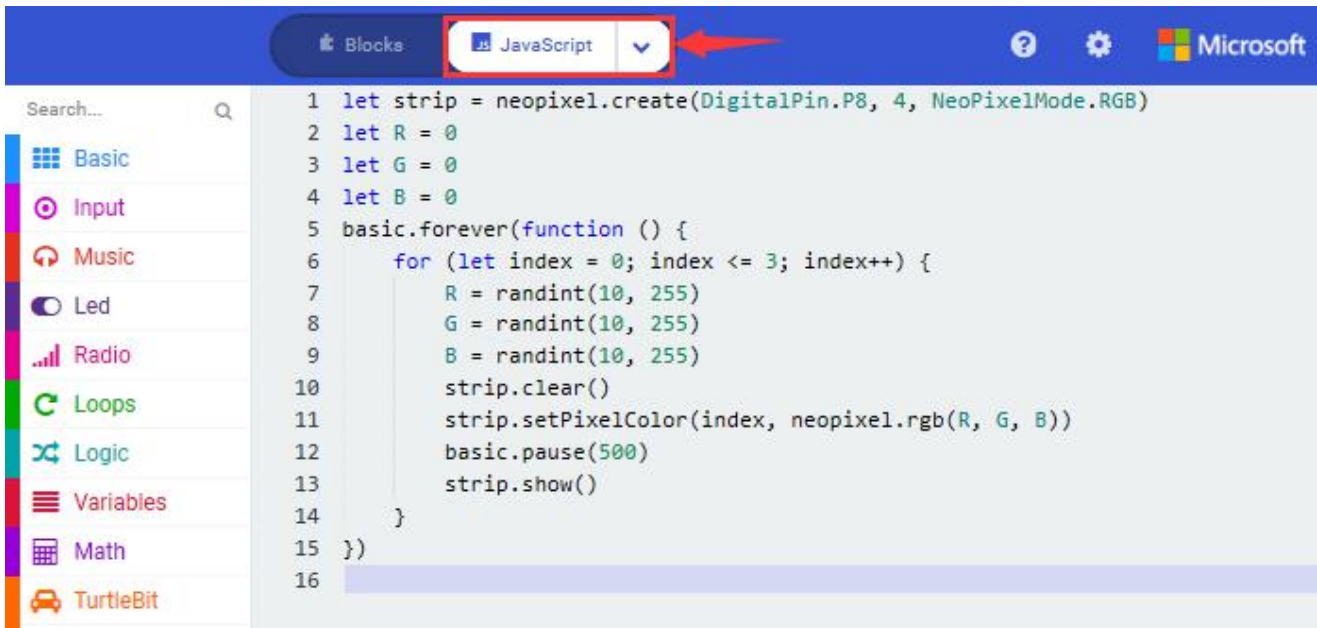
```
on start
  set strip to Neopixel at pin P8 with 4 leds as RGB (GRB format)
  set R to 0
  set G to 0
  set B to 0

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



#### 4.Test Result:

Download code 1 to micro: bit, and dial POWER to ON end. WS2812RGB LEDs light up different colors cyclically.

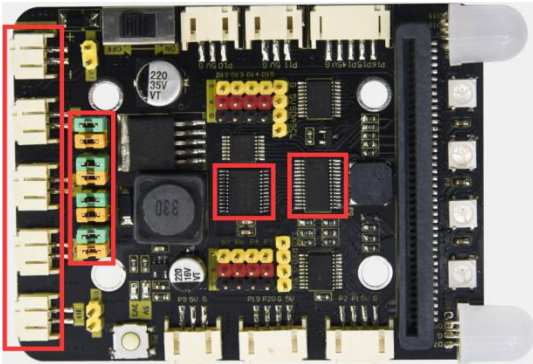
Download code 2 to micro: bit, WS2812RGB LEDs display like flow light.

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

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



## 7.13: Motor Driving

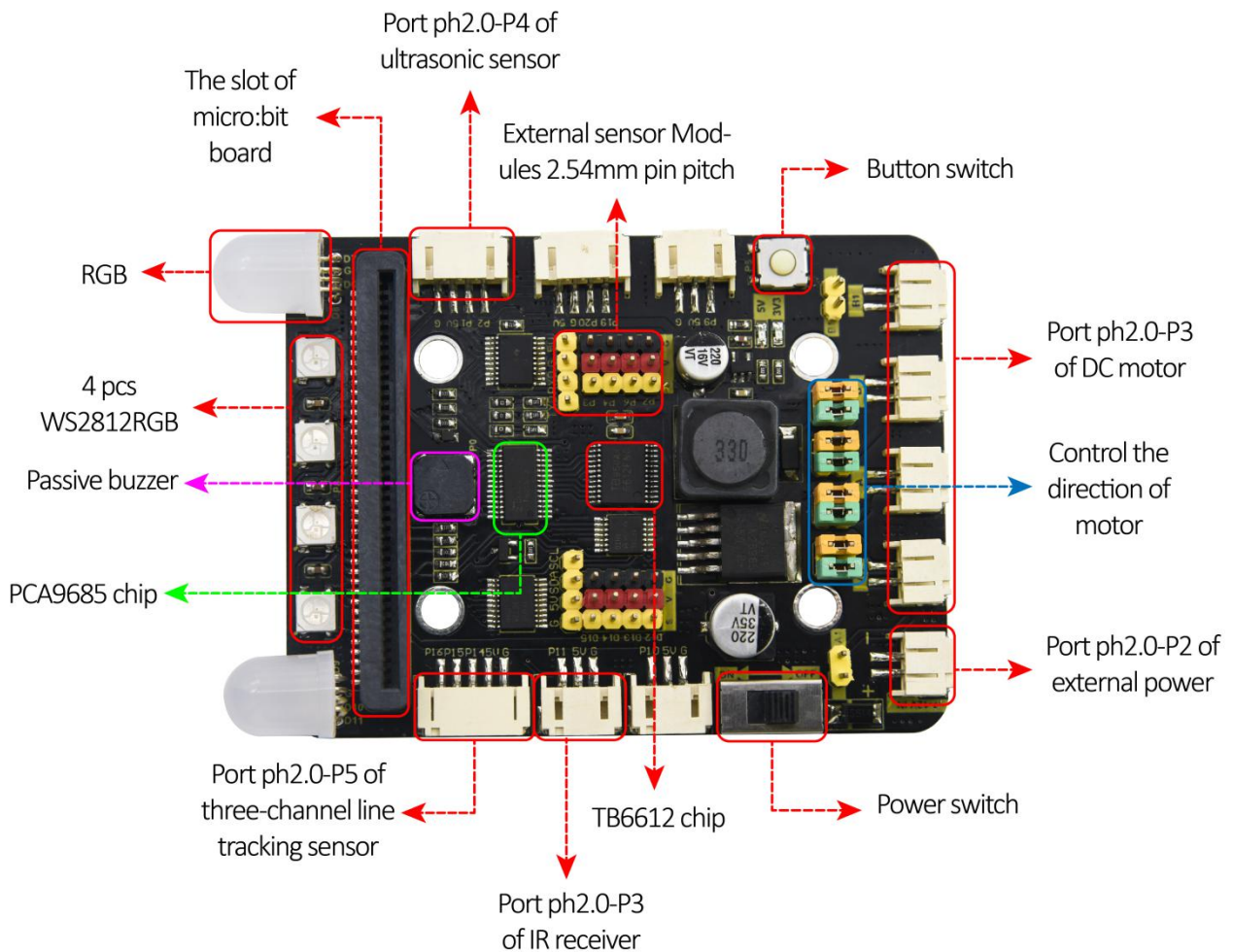


### 1. Description:

Keyestudio micro: bit smart car is equipped with two DC geared motors which are added a gearbox based on regular DC motors.

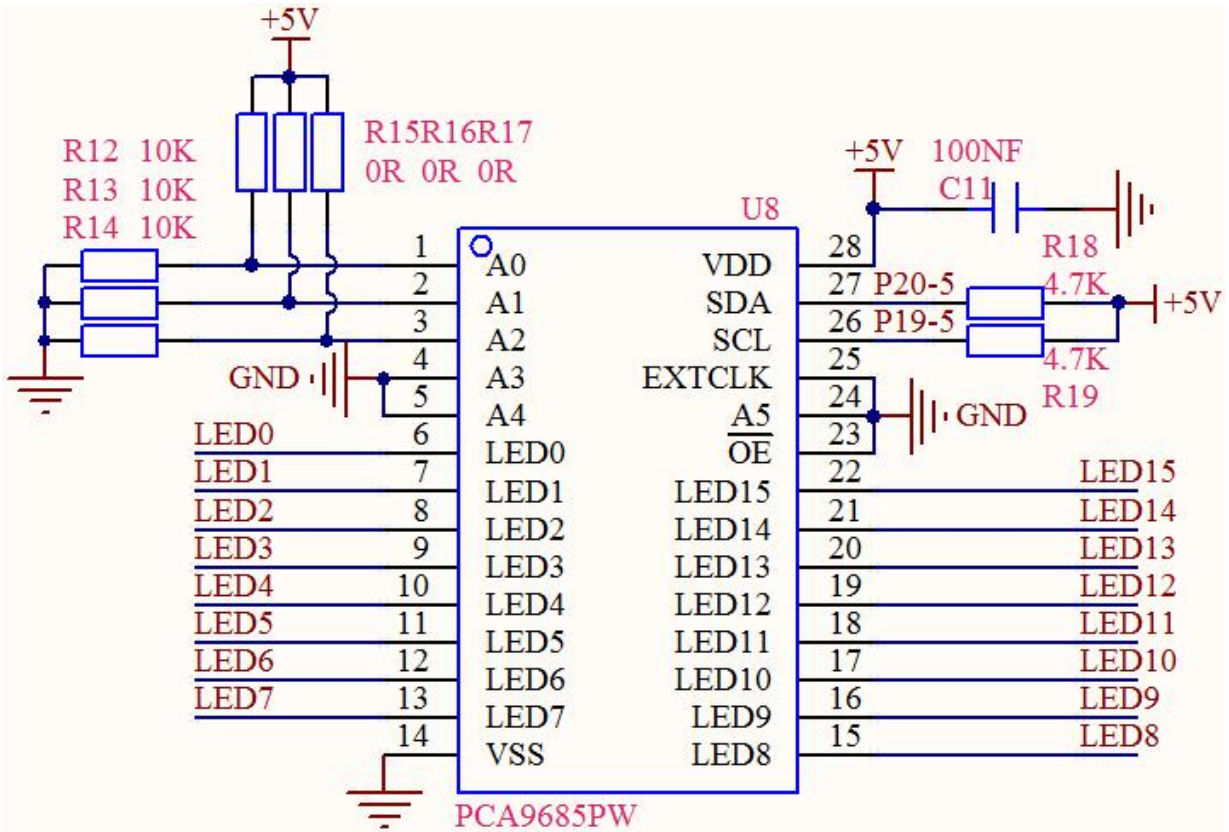
Gear motor is the integration of gearmotor and motor, which is applied widely in steel and machine industry

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

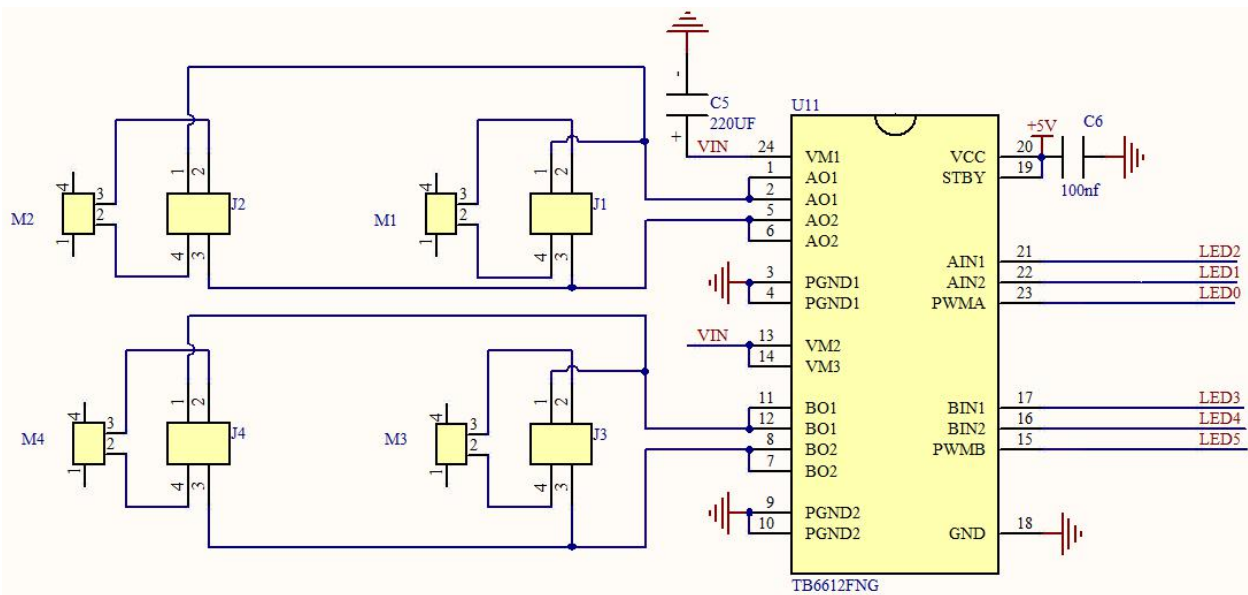


PCA9685 chip is controlled by IIC port of micro:bit board and used to be output port and expand the IO ports of micro:bit board.

TB6612 chip is controlled by LEDx pin of PCA9685 chip(pin LED1 and LED2 control the direction of left motor, speed is LED0 pin; LED3 and LED4 pin control direction of right motor, speed is LED5



# PCA9685PW IC

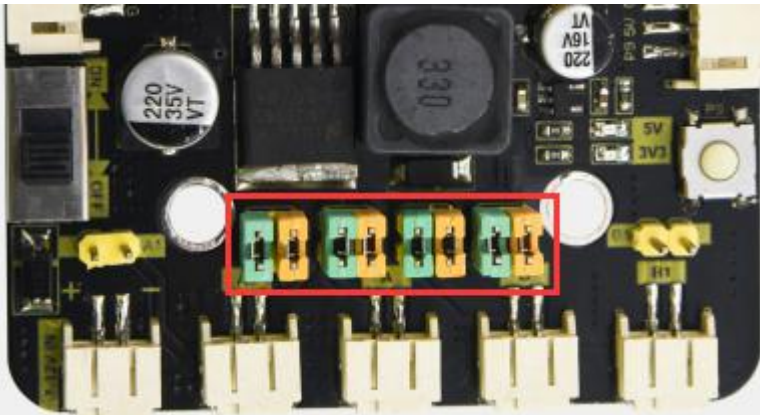


# TB6612FNG Motor control chip

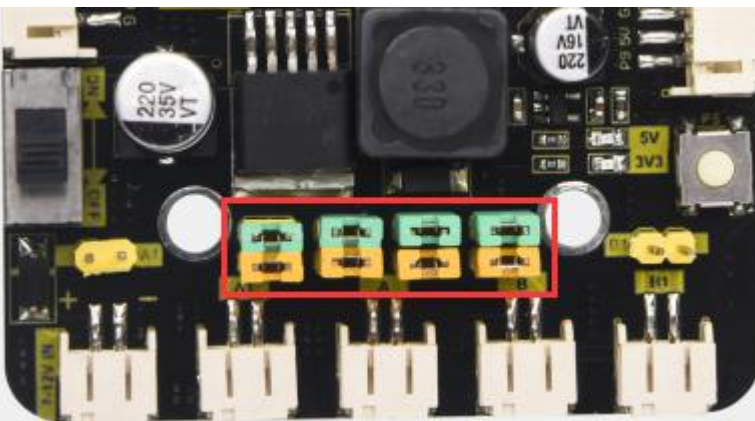


**Note: please follow the direction of eight jumper caps inserted.**

**In this way, the rotation direction is as same as the set rotation orientation in the code**



**The picture below is wrong inserted direction of jumper caps**



## 2. Experimental Preparation:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end



(4) Connect micro:bit to computer by USB cable and open online Makecode editor.

Import Hex profile [\(How to import?\)](#) , or click "New Project" and drag blocks step by step([add turtle-bit extension library first](#))

[\(How to add turtle-bit extension?\)](#)

### 3. Test Code:

#### Code 1:

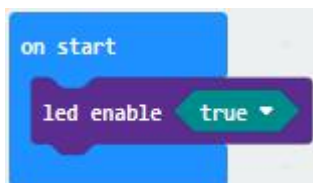
Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.13: Motor Driving/Code-1	microbit-Code-1.hex

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

(1) Click "Led" → "more" → "led enable fasle"

Leave it into "on start"

Tap "false" to select "true"





(2) Click "Basic" to get block "show arrow North"

Leave it into "forever" block

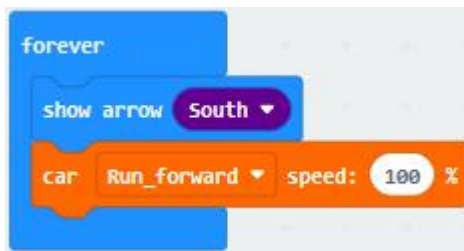
Click North to select South



(3) Click "TurtleBit" to drag "car Run\_forward speed : 0 %" block

Place it into "forever" block

Change 0 into 100.



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

Delay in 100ms



(5) Replicate code string  once and leave it under

"pause (ms) 100" block





Change South into North and Run\_forward into Run\_back



(6) Copy "show arrow South" once and keep it under block "pause (ms) 100" .

Change South into East.



(7) Click "TurtleBit" to move "LeftSide motor run Forward speed : 0 %" and copy it once

Place them under "show arrow East" block



And copy "pause (ms) 1000" block once

Set the code string as follows:

```
forever
  show arrow South
  car Run_forward speed: 100 %
  pause (ms) 1000
  show arrow North
  car Run_back speed: 100 %
  pause (ms) 1000
  show arrow East
  LeftSide motor run Forward speed: 50 %
  RightSide motor run Forward speed: 100 %
  pause (ms) 1000
```

(8) Duplicate code string

```
show arrow East
LeftSide motor run Forward speed: 50 %
RightSide motor run Forward speed: 100 %
pause (ms) 1000
```

once and

keep it under "pause (ms) 1000" block

Change East into West, 50 into 100 and 100 into 50



```
forever
  show arrow South
  car Run_forward speed: 100 %
  pause (ms) 1000
  show arrow North
  car Run_back speed: 100 %
  pause (ms) 1000
  show arrow East
  LeftSide motor run Forward speed: 50 %
  RightSide motor run Forward speed: 100 %
  pause (ms) 1000
  show arrow West
  LeftSide motor run Forward speed: 100 %
  RightSide motor run Forward speed: 50 %
  pause (ms) 1000
```

```
show arrow South
car Run_forward speed: 100 %
pause (ms) 1000
```

(9) Copy code string twice,

Place them into "forever" block

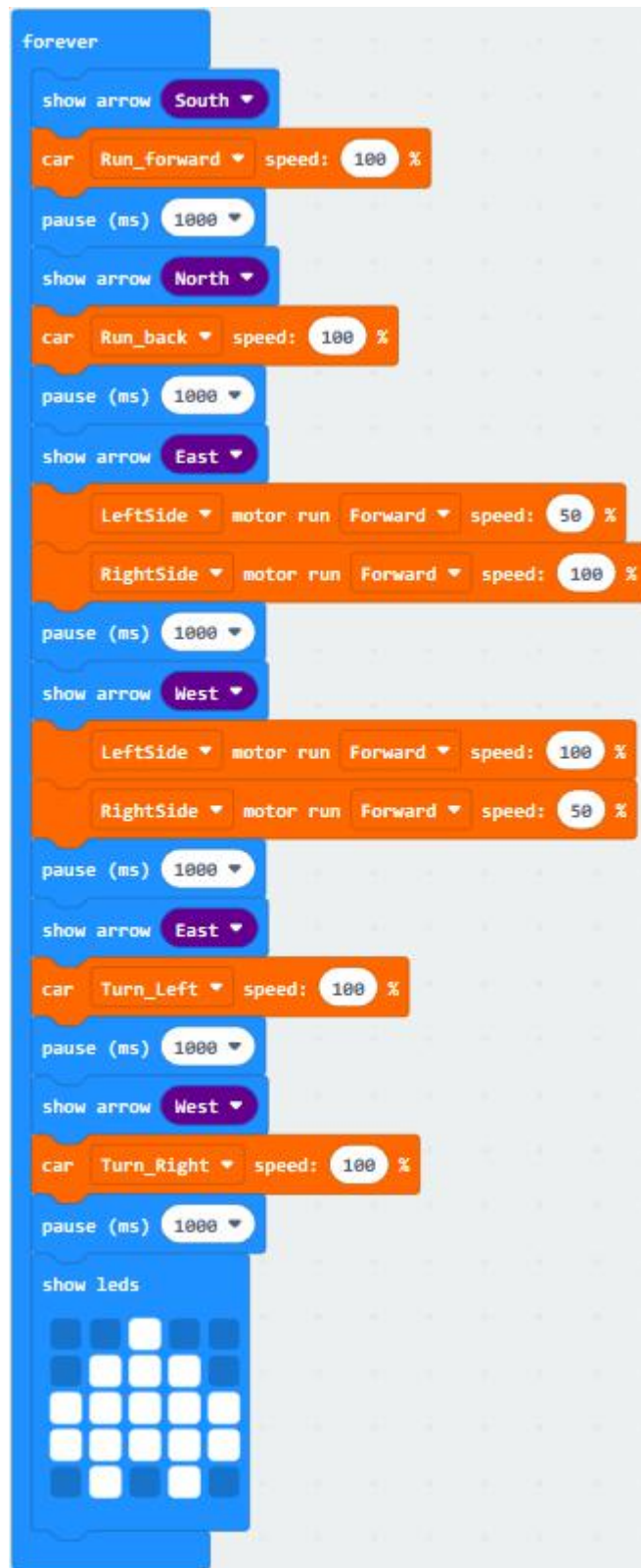
Click South to select East and West

Tap Run\_forward to select Turn\_Left and Turn\_Right



(10) Tap "Basic" to drag block "show leds" into "forever"

Tick blue boxes to generate "♥" pattern



(11) Tap "TurtleBit" to drag "car stop" into "forever" block

Copy "pause (ms) 1000" once and leave it under "car stop" block



```
forever
  show arrow South
  car Run_forward speed: 100 %
  pause (ms) 1000
  show arrow North
  car Run_back speed: 100 %
  pause (ms) 1000
  show arrow East
  LeftSide motor run Forward speed: 50 %
  RightSide motor run Forward speed: 100 %
  pause (ms) 1000
  show arrow West
  LeftSide motor run Forward speed: 100 %
  RightSide motor run Forward speed: 50 %
  pause (ms) 1000
  show arrow East
  car Turn_Left speed: 100 %
  pause (ms) 1000
  show arrow West
  car Turn_Right speed: 100 %
  pause (ms) 1000
  show leds
  car stop
  pause (ms) 1000
```



## Complete Code

```
on start
  led enable true

forever
  show arrow South
  car Run_forward speed: 100 %
  pause (ms) 1000
  show arrow North
  car Run_back speed: 100 %
  pause (ms) 1000
  show arrow East
  LeftSide motor run Forward speed: 50 %
  RightSide motor run Forward speed: 100 %
  pause (ms) 1000
  show arrow West
  LeftSide motor run Forward speed: 100 %
  RightSide motor run Forward speed: 50 %
  pause (ms) 1000
```

“on start” : command block runs once to start program.  
Turn on LED dot matrix  
The program under the block “forever” runs cyclically.  
Micro:bit shows the southward arrow  
Car goes forward at 100% speed  
Delay in 1000ms  
Micro:bit shows the northward arrow  
Car goes back at 100% speed  
Delay in 1000ms  
Micro:bit shows the eastward arrow  
The left car wheels go forward at 50% speed  
The right car wheels go forward at 100% speed  
Delay in 1000ms  
Micro:bit shows the westward arrow  
The left car wheels go forward at 100% speed  
The right car wheels go forward at 50% speed  
Delay in 1000ms  
Micro:bit shows the eastward arrow  
Car rotates anticlockwise at 100% speed  
Delay in 1000ms  
Micro:bit shows the westward arrow  
Car rotates anticlockwise at 100% speed  
Delay in 1000ms  
Micro:bit shows the eastward arrow  
Car stops  
Delay in 1000ms



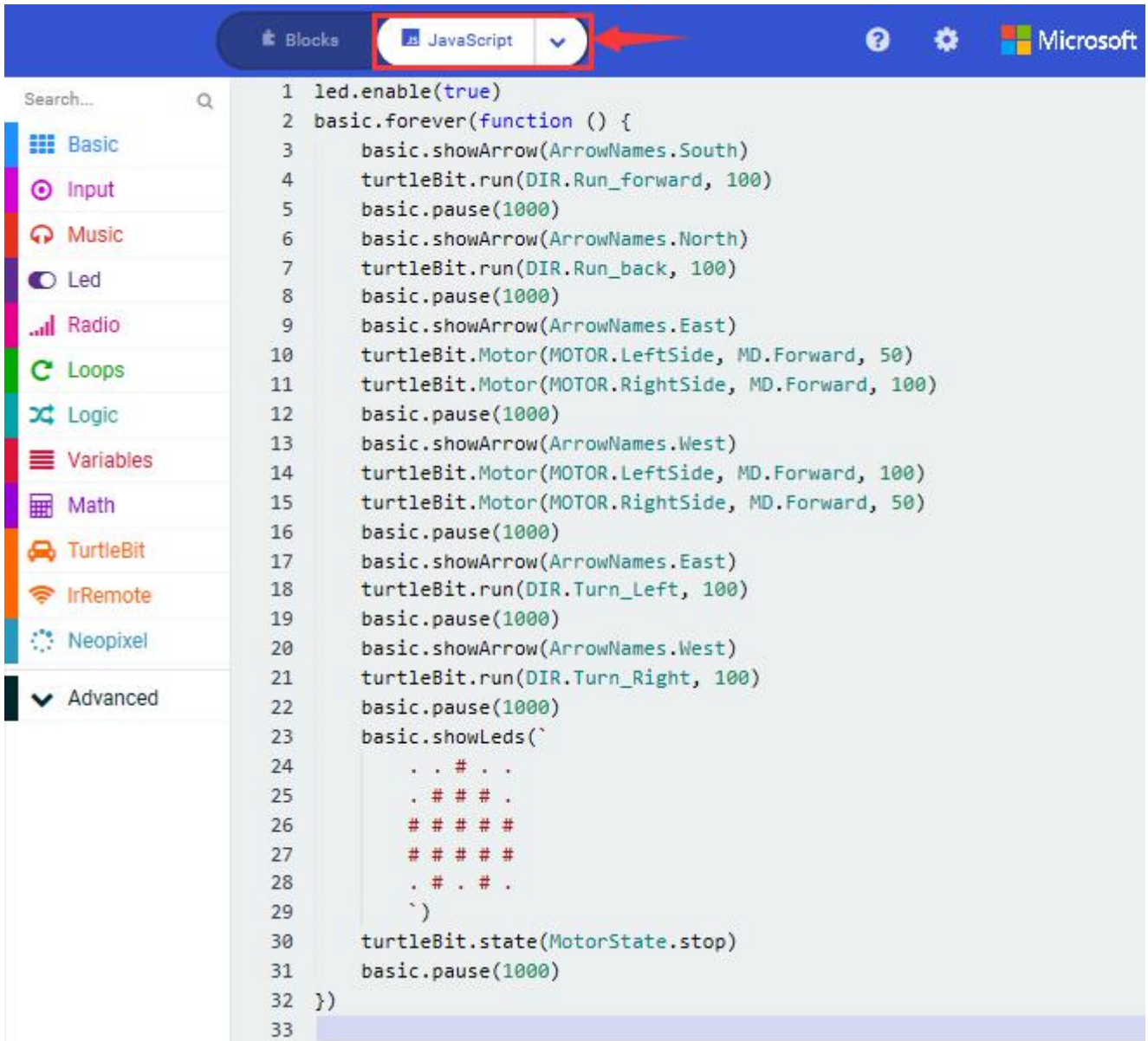
The image shows a Scratch script for a car simulation. The script consists of the following blocks:

- show arrow** (purple block) with a dropdown menu set to **East**.
- car Turn\_Left** (orange block) with a speed of **100 %**.
- pause (ms)** (blue block) with a value of **1000**.
- show arrow** (purple block) with a dropdown menu set to **West**.
- car Turn\_Right** (orange block) with a speed of **100 %**.
- pause (ms)** (blue block) with a value of **1000**.
- show leds** (blue block) with a 5x5 grid of LED indicators. The top row has the second and fourth LEDs lit. The second and fourth rows have all LEDs lit. The bottom row has the second and fourth LEDs lit.
- car stop** (orange block).
- pause (ms)** (blue block) with a value of **1000**.





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



**Code 2:**

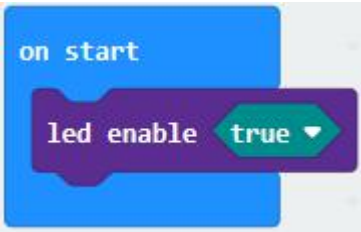
Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.13: Motor Driving/Code-2	microbit-Code-2.hex



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

(1) Click "Led" → "more" → "led enable false" ,

Put it into block "on start" , click drop-down triangle button to select "true"



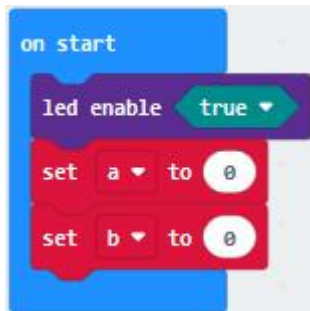
(2) Click "Variables" → "Make a Variable..."

Input "a" and click "OK" , variable "a" is built

Then create the variable "b" in same way

Drag "set b to 0" into "on start" block, change b into a

Copy "set b to 0" once and leave it under "set a to 0"



(3) Click "Input" to drag out "on button A pressed"

Tap "Variables" to move "change b by 1" into "on button A pressed" block, and change b into a.



(4) Copy code string  once, delete "change a by 1" block



Change A into B

Tap "Variables" to drag "set b to 0" into "on button B pressed"

Alter 0 into 1



(5) Click "Logic" to move "if true then..." into "forever" block

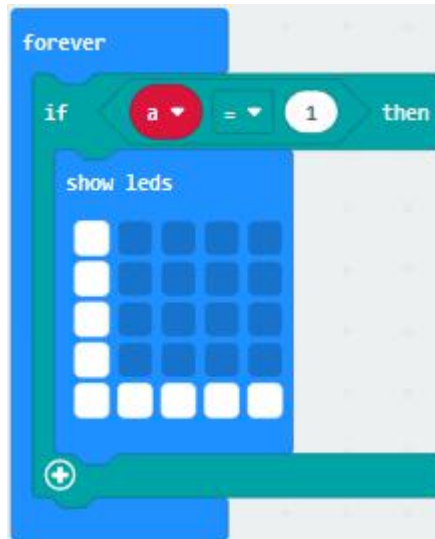
Drag "=" block into "true" box

Tap "Variables" to move variable "a" to left box of "=", and change 0 into 1.



(6) Click "Basic" to drag "show leds" under then block

Tick blue box to generate "L"

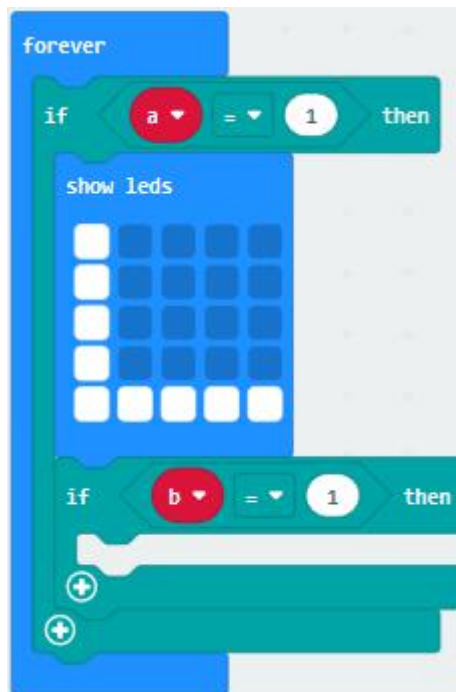


(7) Click "Logic" to get "if true then..." block

Leave it under block "if..then" block

Go to "Variables" to move block "b" to left box of "=" block

Change 0 into 1



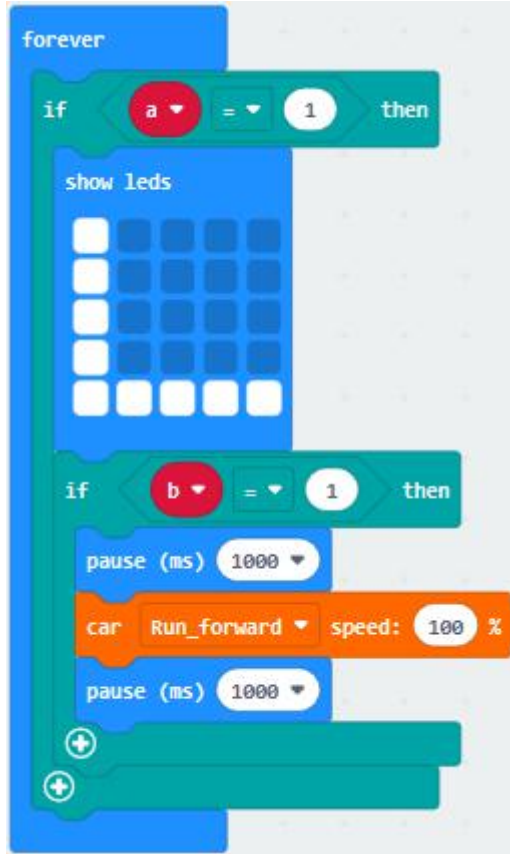
(8) Click "Basic" to get block "pause (ms) 100"

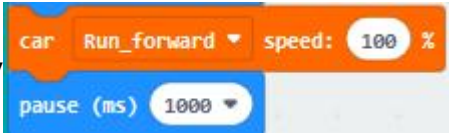
Leave it into "if..then" block



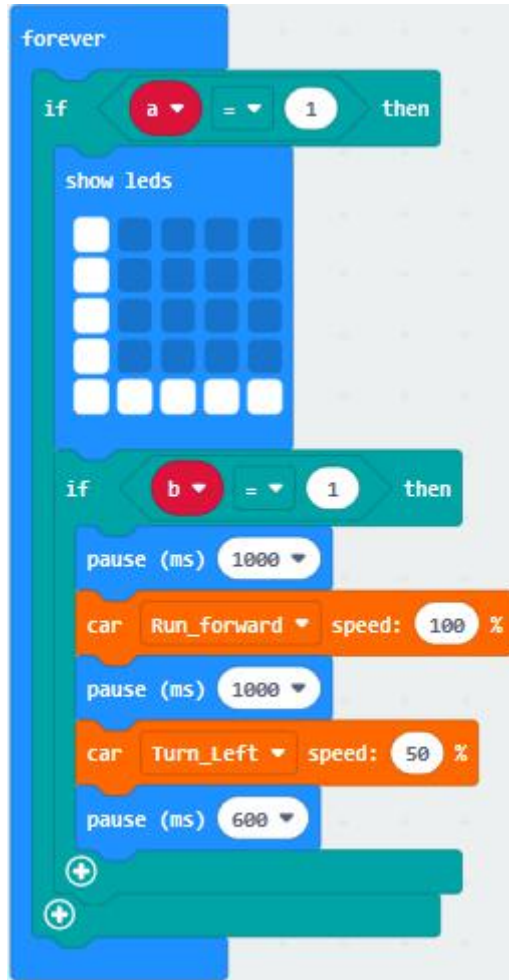
Click "TurtleBit" to move "car Run\_forward speed: 0 %" under block "pause (ms) 100" and change 0 into 100

Copy "pause (ms) 1000" block and leave it under "car...100%" block.

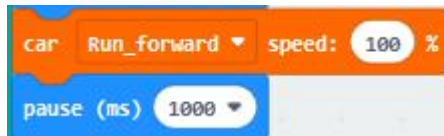


(9) Copy  once and leave it under "pause (ms) 100" block

Change Run\_Forward into Turn\_Left, 100 into 50 and delay in 600ms

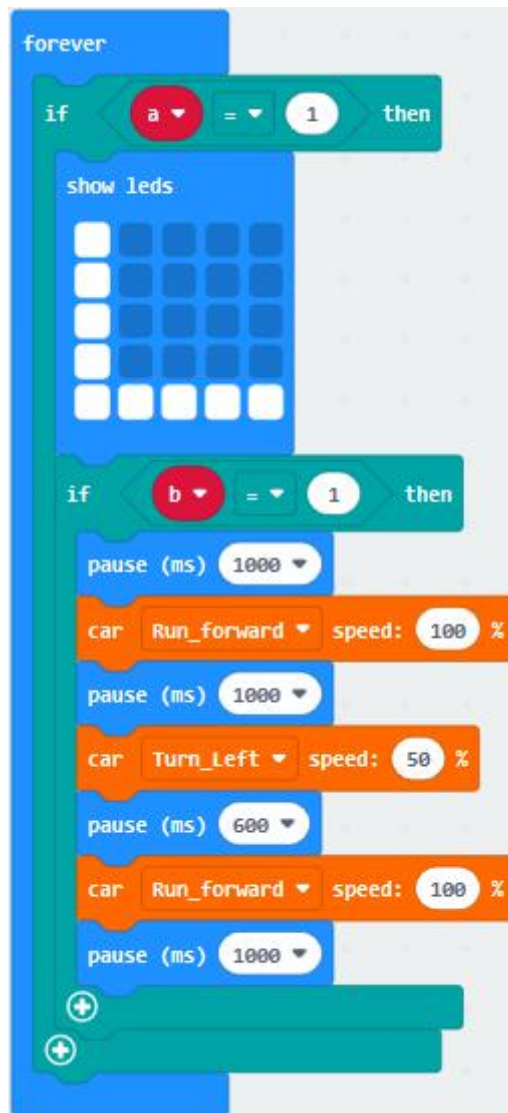


(10) Duplicate code string

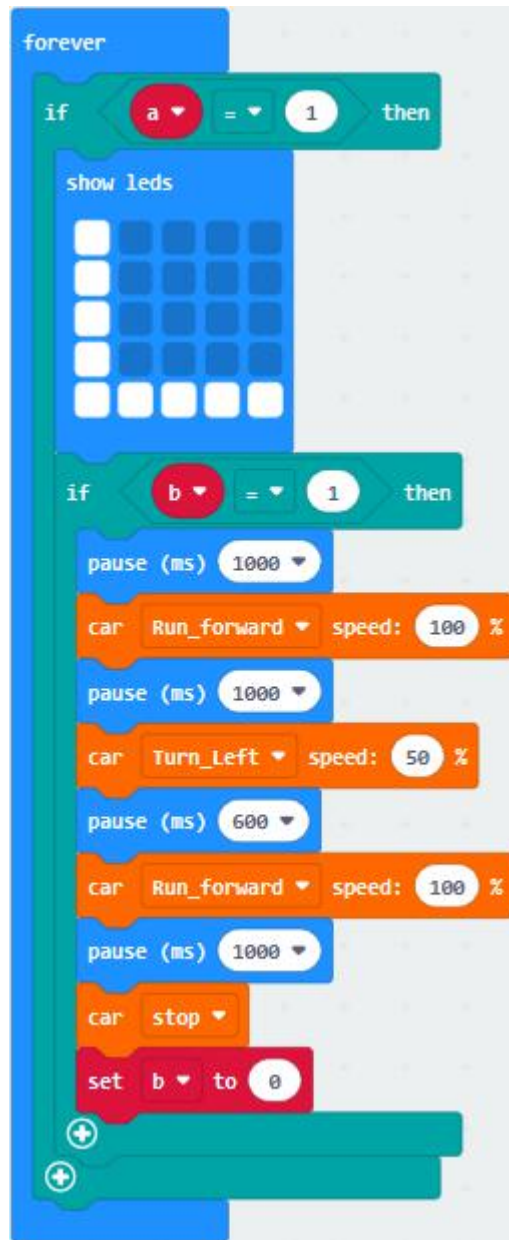


once, keep it under

"pause (ms) 600" block



(11) Tap "TurtleBit" to drag "car stop" block under block "pause (ms) 1000"  
Tap "Variables" to move "set b to 0" block under "car stop" block

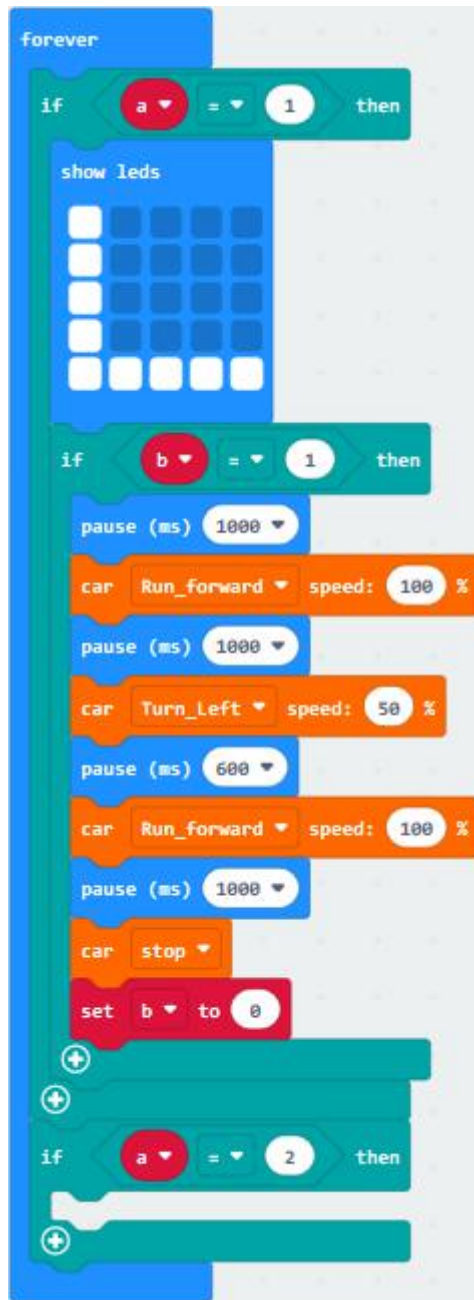


(12) Click "Logic" to drag "if true then..." into "forever"

Move "=" block into "true" box

Click "Variables" to move "a" to left box of "=" and change 0 into 20





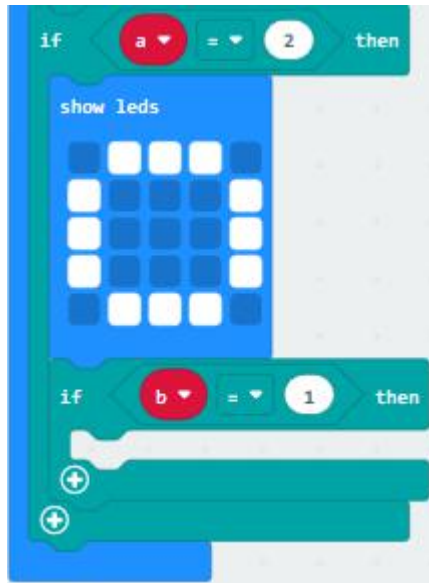
(13) Click "Basic" to drag "show leds" into the third "if..then" block  
Tick blue boxes to produce "□"



(14) Go to “Logic” to drag “if true then...” block

Keep it under “show leds □” block

Move variable b into left box of “=” block and change 0 into 1



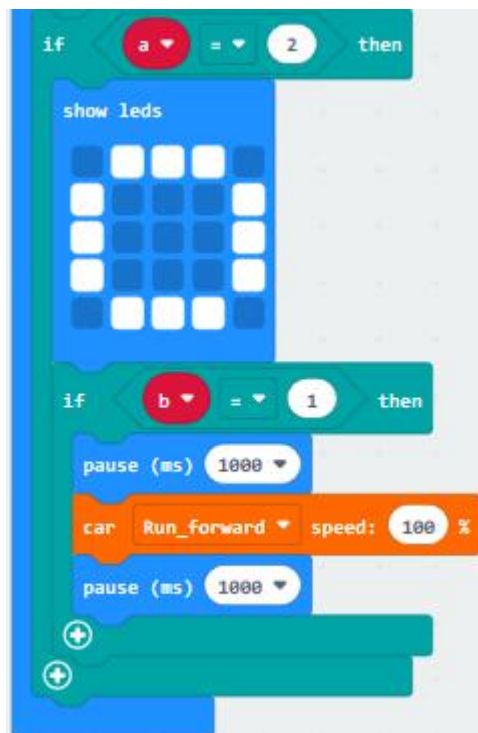
(15) Tap "Basic" to drag "pause (ms) 100" and copy it once

Set to 1000ms

Tap "TurtleBit" block to drag out "car Run\_forward speed: 0 %" block

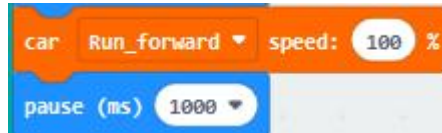
Change 0 into 100

Then set the code string as follows:





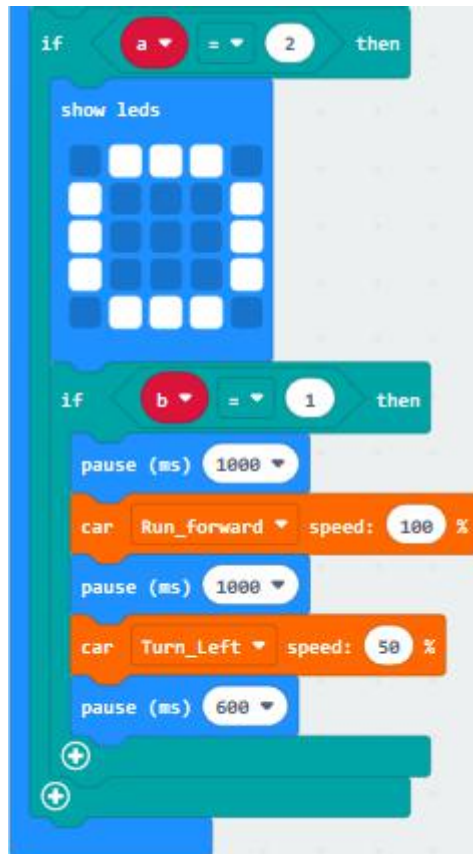
(16) Duplicate code string



once

Change Run\_Forward into Turn\_Left, 100 into 50 and 1000 into 600

Place it into code sting, as shown below



(17) Replicate code string



twice and keep them in

the code string as follows:



```
if a = 2 then
  show leds
  if b = 1 then
    pause (ms) 1000
    car Run_forward speed: 100 %
    pause (ms) 1000
    car Turn_Left speed: 50 %
    pause (ms) 600
    car Run_forward speed: 100 %
    pause (ms) 1000
    car Turn_Left speed: 50 %
    pause (ms) 600
    car Run_forward speed: 100 %
    pause (ms) 1000
    car Turn_Left speed: 50 %
    pause (ms) 600
```

```
car Run_forward speed: 100 %
pause (ms) 1000
car stop
set b to 0
```

(18) Copy code string once, and leave it under " pause(as) 600" block



(20) Copy "if a=2 then" block once and change 2 into 3

Tap "Variables" to move "set b to 0" into "if a=3 then" block

Alter b into a and 0 into 1



```
if b = 1 then
  pause (ms) 1000
  car Run_forward speed: 100 %
  pause (ms) 1000
  car Turn_Left speed: 50 %
  pause (ms) 600
  car Run_forward speed: 100 %
  pause (ms) 1000
  car Turn_Left speed: 50 %
  pause (ms) 600
  car Run_forward speed: 100 %
  pause (ms) 1000
  car Turn_Left speed: 50 %
  pause (ms) 600
  car Run_forward speed: 100 %
  pause (ms) 1000
  car stop
  set b to 0
+
+
if a = 3 then
  set a to 1
+
```

Complete Code:



```
on start
  led enable true
  set a to 0
  set b to 0

on button A pressed
  change a by 1

on button B pressed
  set b to 1

forever
  if a = 1 then
    show leds
  if b = 1 then
    pause (ms) 1000
    car Run_forward speed: 100 %
    pause (ms) 1000
    car Turn_Left speed: 50 %
    pause (ms) 600
```

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

Turn on LED dot matrix  
Set variable a to 0  
Set variable b to 0  
When button A is pressed  
Change variable a by 1  
When A is pressed  
Set variable b to 1  
The program under the block "forever" runs cyclically.  
When variable a=1, run the program under then block  
Micro:bit shows "L"  
When variable b=1, run the program under then block  
Delay in 1000ms  
Go forward at 100% speed  
Delay in 1000ms  
  
Turn anticlockwise at 50% speed  
Delay in 600ms





Car goes forward at 100% speed  
Delay in 1000ms  
Car stops  
Set variable b to 0  
When variable a=2, run the program under then block  
Matrix shows "□"  
When variable b=1, execute the program under then block  
Delay in 1000ms  
Car goes forward at 100% speed  
Delay in 1000ms  
Car turns anticlockwise at 50% speed  
Car turns anticlockwise at 50% speed  
Delay in 600ms  
Car goes forward at 100% speed  
Delay in 1000ms  
Car turns anticlockwise at 50% speed  
Delay in 600ms  
Car goes forward at 100% speed  
Delay in 1000ms  
Car stops  
Set variable b to 0  
When variable a=3, execute the program under then block  
Set variable a to 1



The image shows a Scratch script for a car simulation. The script consists of the following blocks:

- car Turn\_Left speed: 50 %
- pause (ms) 600
- car Run\_forward speed: 100 %
- pause (ms) 1000
- car stop
- set b to 0
- if a = 3 then
- set a to 1

Click "JavaScript" to view the corresponding JavaScript code:



The screenshot shows the Keyestudio IDE interface. At the top, there is a blue header bar with a 'Blocks' tab and a 'JavaScript' tab. A red arrow points to the 'JavaScript' tab. Below the header, on the left, is a block palette with a search bar and various categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, TurtleBit, IrRemote, Neopixel, and Advanced. The main area is a code editor with the following JavaScript code:

```
1 input.onButtonPressed(Button.A, function () {
2     a += 1
3 })
4 input.onButtonPressed(Button.B, function () {
5     b = 1
6 })
7 let b = 0
8 led.enable(true)
9 let a = 0
10 b = 0
11 basic.forever(function () {
12     if (a == 1) {
13         basic.showLeds(`
14             # . . . .
15             # . . . .
16             # . . . .
17             # . . . .
18             # # # # #
19             `)
20     if (b == 1) {
21         basic.pause(1000)
22         turtleBit.run(DIR.Run_forward, 100)
23         basic.pause(1000)
24         turtleBit.run(DIR.Turn_Left, 50)
25         basic.pause(600)
26         turtleBit.run(DIR.Run_forward, 100)
27         basic.pause(1000)
28         turtleBit.state(MotorState.stop)
29         b = 0
30     }
31 }
```



```
32   if (a == 2) {
33       basic.showLeds(`
34           . # # # .
35           # . . . #
36           # . . . #
37           # . . . #
38           . # # # .
39       `)
40   if (b == 1) {
41       basic.pause(1000)
42       turtleBit.run(DIR.Run_forward, 100)
43       basic.pause(1000)
44       turtleBit.run(DIR.Turn_Left, 50)
45       basic.pause(600)
46       turtleBit.run(DIR.Run_forward, 100)
47       basic.pause(1000)
48       turtleBit.run(DIR.Turn_Left, 50)
49       basic.pause(600)
50       turtleBit.run(DIR.Run_forward, 100)
51       basic.pause(1000)
52       turtleBit.run(DIR.Turn_Left, 50)
53       basic.pause(600)
54       turtleBit.run(DIR.Run_forward, 100)
55       basic.pause(1000)
56       turtleBit.state(MotorState.stop)
57       b = 0
58   }
59   }
60   if (a == 3) {
61       a = 1
62   }
63 })
64
```

#### 4.Test Result:

Download code 1 to micro:bit board, dial POWER switch to ON end. Smart car goes forward for 1s, backward for 1s, turns left for 1s, turns right for 1s, rotates anticlockwise for 1s, clockwise for 1 and stops for 1s. And dot matrix displays the corresponding patterns



Download code 2 to micro:bit board, "L" will be shown on dot matrix when A button is pressed, then press B, the route of smart car is "L" type.

"□" will be displayed when the Button A is pressed again, then press B, the route of smart car is "□" type.

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

## 7.14: Line Tracking Smart Car

### 7.14.1: Detect Line Tracking Sensor



#### 1. Description:

The V2 expansion board of keyestudio micro : bit mini smart robot car comes with two line tracking sensors which adopt TCRT5000 IR tubes.

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.

The left and right line tracking sensors are respectively controlled by P12 and P13.

Put a paper under the bottom of car, adjust the potentiometers on shield to adjust sensitivity. When D2 and D6 are on, then pull up the universal wheels for 0.5cm off the paper. The sensitivity is set well if D2 and D6 are off

## 2. Experimental Preparation:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end
- (4) Connect micro:bit to computer by USB cable and open online Makecode editor.
- (5) Import Hex profile ([How to import?](#)), or click "New Project" and drag blocks step by step([add turtle-bit extension library first](#))



## (How to add turtle-bit extension?)

### 3. Test Code:

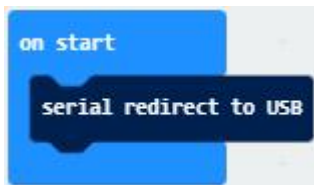
#### Code 1:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.14: Line Tracking Smart Car/7.14.1:Detect Line Tracking Sensor/Code-1	microbit-Code-1.hex

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

(1) Click "Advanced" → "Serial" → "serial redirect to USB"

Place it into "on start"



(2) Enter "Advanced" → "Serial" → "serial write value "x" =0"

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" .



```
forever
  serial write value "digital signal" = digital read pin P14
```

(3) Go to "Basic" → "pause (ms) 100"

Keep it into "forever" block and set to 200ms.

```
forever
  serial write value "digital signal" = digital read pin P14
  pause (ms) 200
```

### Complete Program:

```
on start
  serial redirect to USB

forever
  serial write value "digital signal" = digital read pin P14
  pause (ms) 200
```



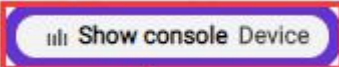


“on start”: command block runs once to start program.  
Serial redirect to USB  
The program under the block “forever” runs cyclically.  
Serial write value “digital signal”=digital red pin P14  
Delay in 200ms

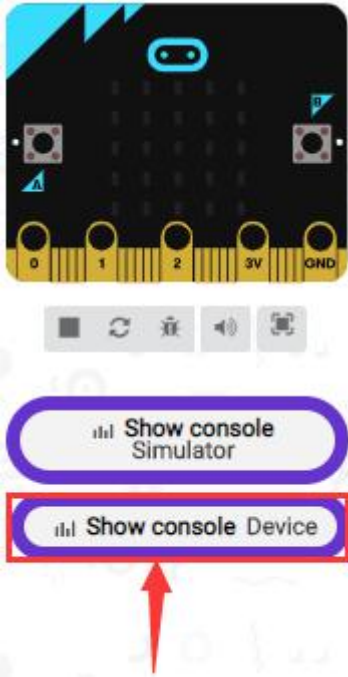
Click “JavaScript” to view the corresponding JavaScript code:

```
1 serial.redirectToUSB()
2 basic.forever(function () {
3   serial.writeValue("digital signal", pins.digitalReadPin(DigitalPin.P14))
4   basic.pause(200)
5 })
6
```

Download code 1 to micro:bit board, don't plug off USB cable and

click 

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



Serial monitor will display low level(0) and left indicator will be on when the left TCRT5000 IR tube detects white objects, black objects or no object are detected by left TCRT5000 IR tube, 1(high level) will be shown on serial monitor and indicator will be off, as shown below:

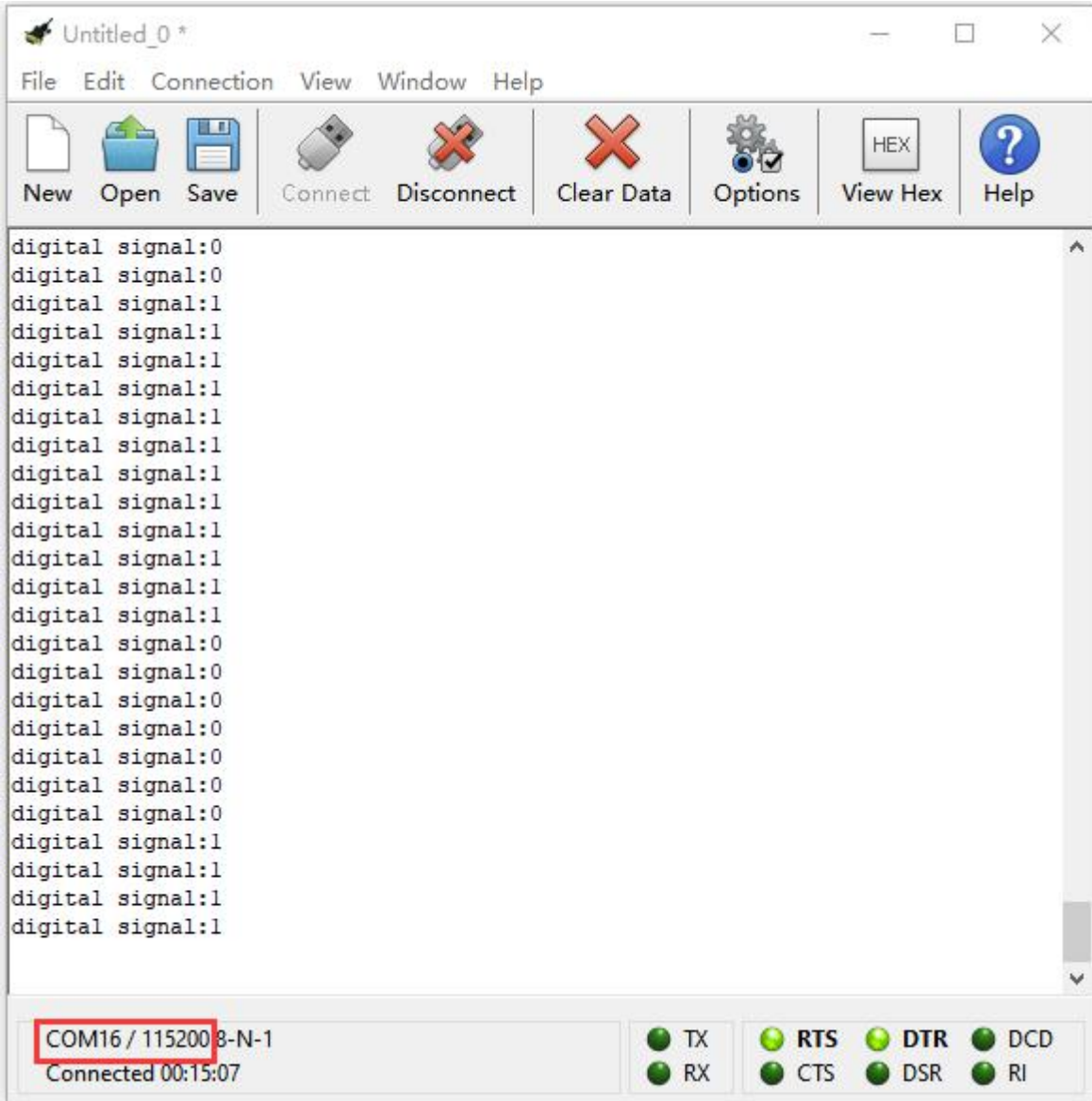


The screenshot shows the Keyestudio simulator interface. On the left, there is a breadboard with a sensor. On the right, there is a digital signal waveform graph showing a square wave. Below the graph is a console window displaying a sequence of digital signal readings.

```
21 digital signal:1
15 digital signal:0
7 digital signal:1
13 digital signal:0
7 digital signal:1
15 digital signal:0
```

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 sensor.



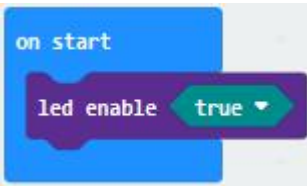
### Code 2:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.14: Line Tracking Smart Car/7.14.1:Detect Line Tracking Sensor/Code-2	microbit-Code-2.hex



(1) Click "Led" → "more" → "led enable false" ,

Put it into block "on start" , click drop-down triangle button to select "true"



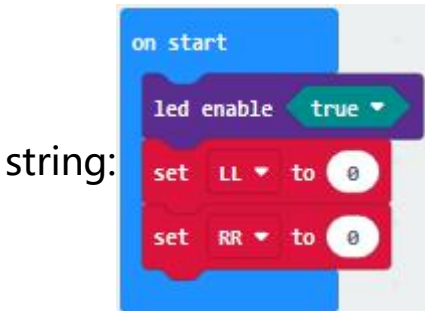
\*\*\*\*\*

(2) Go to "Variables" → "Make a Variable..." → "New variable name: " dialog box.

Enter LL and click "OK" to create variable "LL" .

Next to produce variable "RR" in same way.

Drag block "set RR to 0" into "on start" block and copy it once. Then set code



string:

(3) Click "Variables" to drag block "set RR to 0" into block "forever"

Tap RR to select LL

Tap "Advanced" → "Pins" → "digital read pin P0"

Place "digital read pin P0" into 0 box

Copy "set LL to digital read pin P0" twice and keep them into "forever" block.

Separately change into CC and RR



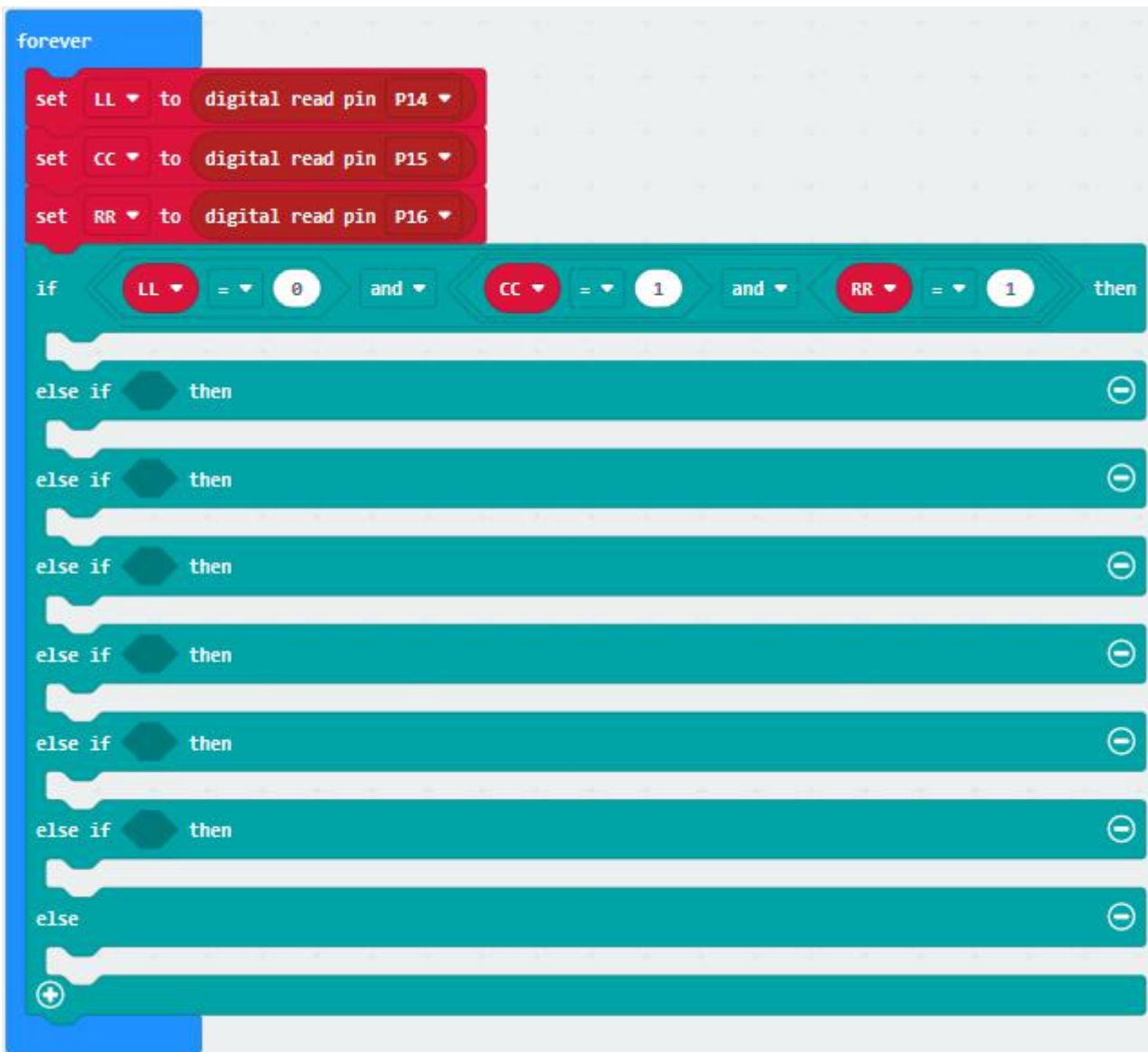
The three TCRT5000 IR tubes of line tracking sensor are controlled by P14, P15 and P16, therefore we set to code string as follows:



(4) Click "Logic" to drag "if true then...else" block into "forever" block  
Tap "+" for six times and move "and" block twice and place them into true boxes, as shown below:



(5) Go to "Logic" to move block "=" into left box of the first and block  
Click "Variables" to drag "LL" into left box of "=" block  
Copy "LL=0" twice and respectively leave them into boxes of the second  
and block and change 0 into 1, as shown below



(6) Click "Basic" to move block "show leds" under "if...then" block  
Copy it twice and respectively leave them under the first and second "else if...then" block

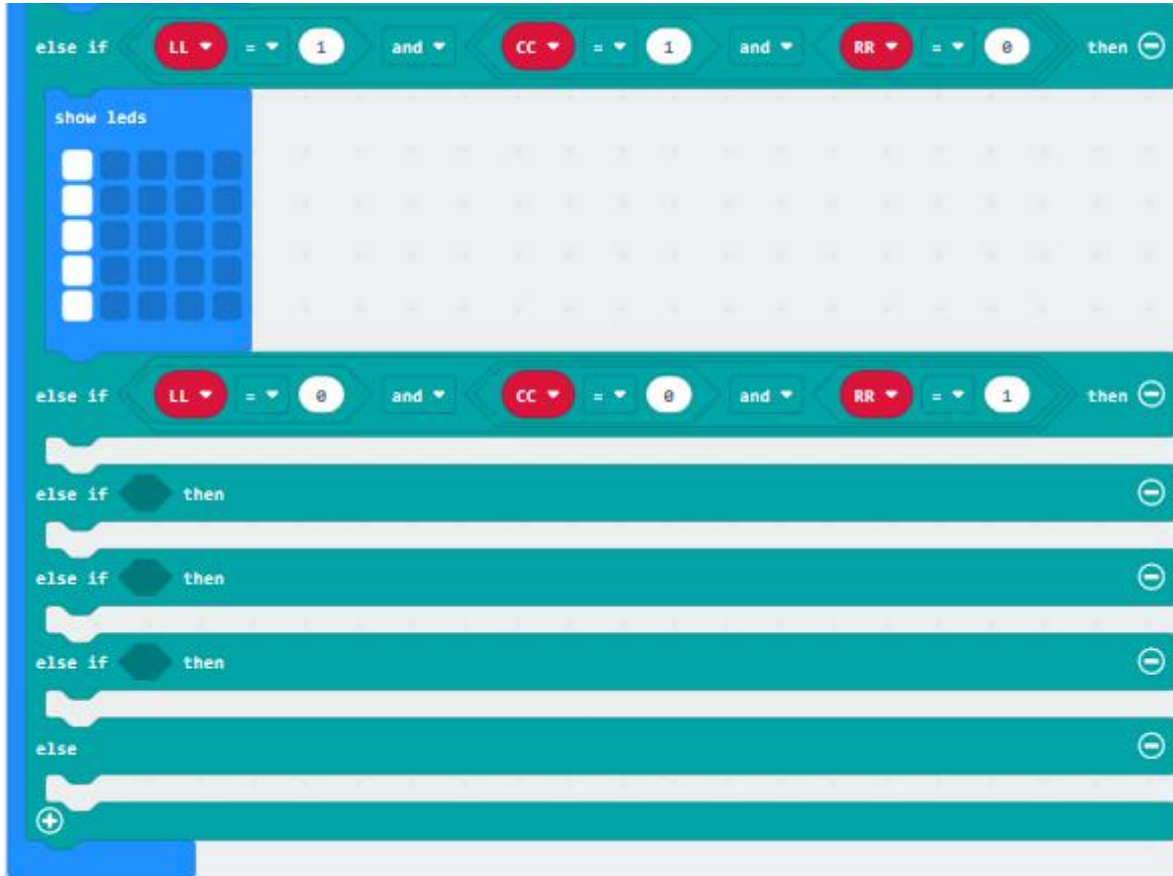
Replicate "LL=0 and CC=1 and RR=1" twice and edit code string as follows:





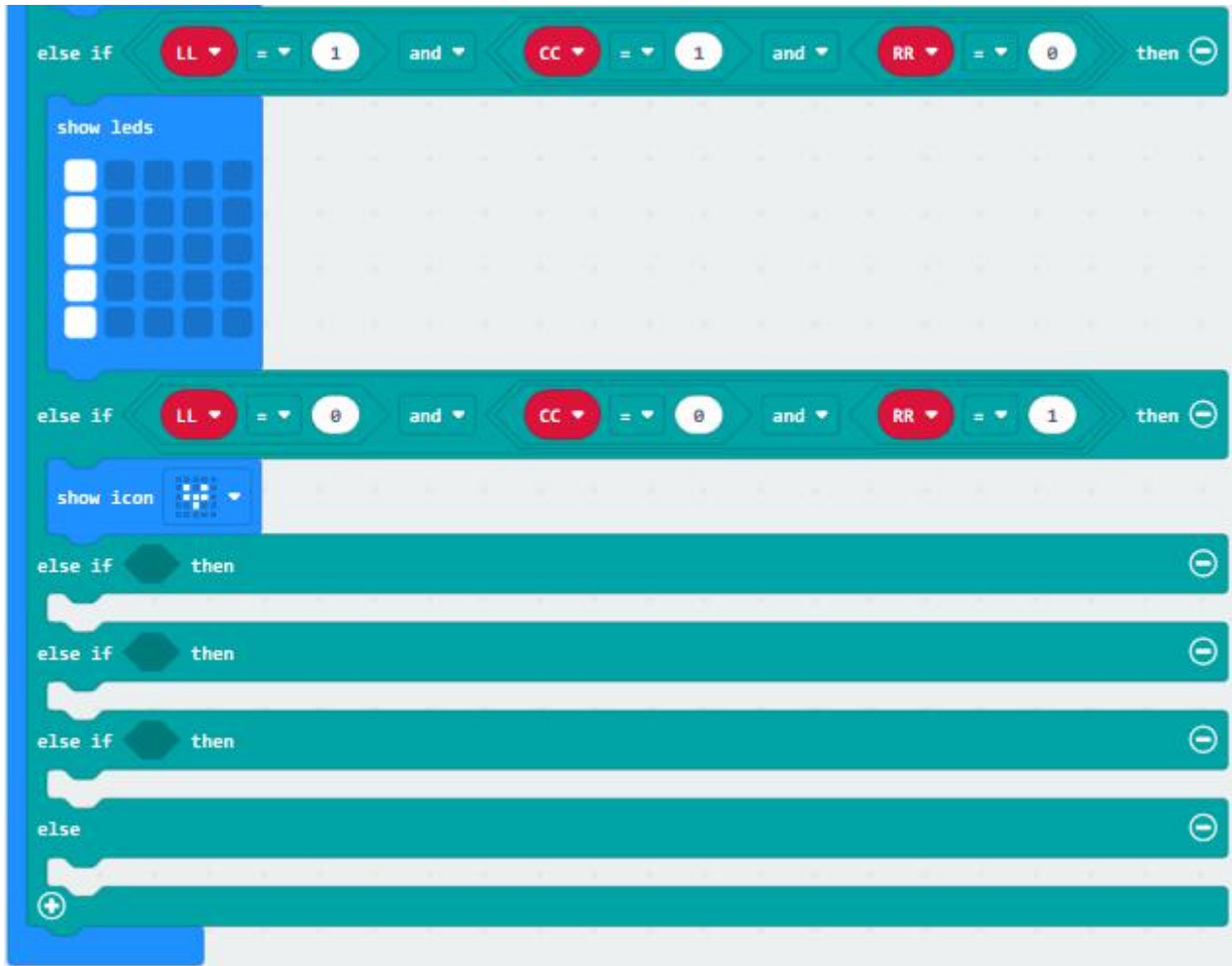
(7) Duplicate "LL=1 and CC=1 and RR=0" again and leave it into box behind the third else if block

Set "LL=0 and CC=0 and RR=1"



(8) Click "Basic" to move block "show icon" under the fourth "else if...then" block

Click the triangle button to select "📊"



(9) Copy “LL=0 and CC=0 and RR=1” for three times and block “show icon” for four times

(10) Edit the code string as follows:



```
else if < LL = 0 and CC = 0 and RR = 1 > then
  show icon [icon]
else if < LL = 0 and CC = 1 and RR = 0 > then
  show icon [icon]
else if < LL = 1 and CC = 0 and RR = 0 > then
  show icon [icon]
else if < LL = 1 and CC = 1 and RR = 1 > then
  show icon [icon]
else
  show icon [icon]
```

Complete Code:



```
on start
  led enable true
  set LL to 0
  set CC to 0
  set RR to 0

forever
  set LL to digital read pin P14
  set CC to digital read pin P15
  set RR to digital read pin P16

  if LL = 0 and CC = 1 and RR = 1 then
    show leds
  else if LL = 1 and CC = 0 and RR = 1 then
    show leds
```



The image shows a Scratch script for controlling a 3x3 LED matrix. The script consists of several conditional blocks, each with a specific logic condition and a corresponding LED pattern icon.

- Block 1:** Condition:  $LL = 1$  and  $CC = 1$  and  $RR = 0$ . Action: show leds (a 3x3 grid with the top row lit).
- Block 2:** Condition:  $LL = 0$  and  $CC = 0$  and  $RR = 1$ . Action: show icon (a 3x3 grid with the right column lit).
- Block 3:** Condition:  $LL = 0$  and  $CC = 1$  and  $RR = 0$ . Action: show icon (a 3x3 grid with the middle column lit).
- Block 4:** Condition:  $LL = 1$  and  $CC = 0$  and  $RR = 0$ . Action: show icon (a 3x3 grid with the left column lit).
- Block 5:** Condition:  $LL = 1$  and  $CC = 1$  and  $RR = 1$ . Action: show icon (a 3x3 grid with all LEDs lit).
- Block 6:** Condition: else. Action: show icon (a 3x3 grid with all LEDs lit).



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

Turn on dot matrix

Set variable LL to 0

Set variable CC to 0

Set variable RR to 0

The program under the block “forever” runs cyclically.

Set LL to the digital signals read by P14

Set CC to the digital signals read by P15

Set RR to the digital signals read by P16

When LL=0, CC=1 and RR=1, execute the program under then block

Set to display “I”

When LL=1, CC=0 and RR=1, execute the program under then block

Micro:bit shows “I”


When LL=1, CC=1 and RR=0, execute the program under then block

Micro:bit shows “I”


When LL=1, CC=0 and RR=1, execute the program under then block

Micro:bit shows 


When LL=1, CC=1 and RR=0, execute the program under then block

Micro:bit shows 

When LL=1, CC=0 and RR=0, execute the program under then block

Micro:bit shows 

When LL=1, CC=1 and RR=1, execute the program under then block

Micro:bit shows 

When the above condition is not met, execute the program under else block

Micro:bit shows “♥”

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




```
1 led.enable(true)
2 let LL = 0
3 let CC = 0
4 let RR = 0
5 basic.forever(function () {
6   LL = pins.digitalReadPin(DigitalPin.P14)
7   CC = pins.digitalReadPin(DigitalPin.P15)
8   RR = pins.digitalReadPin(DigitalPin.P16)
9   if (LL == 0 && (CC == 1 && RR == 1)) {
10    basic.showLeds(`
11      . . . . #
12      . . . . #
13      . . . . #
14      . . . . #
15      . . . . #
16    `)
17  } else if (LL == 1 && (CC == 0 && RR == 1)) {
18    basic.showLeds(`
19      . . # . .
20      . . # . .
21      . . # . .
22      . . # . .
23      . . # . .
24    `)
25  } else if (LL == 1 && (CC == 1 && RR == 0)) {
26    basic.showLeds(`
27      # . . . .
28      # . . . .
29      # . . . .
30      # . . . .
31      # . . . .
32    `)
```

```
33 } else if (LL == 0 && (CC == 0 && RR == 1)) {
34   basic.showIcon(IconNames.SmallHeart)
35 } else if (LL == 0 && (CC == 1 && RR == 0)) {
36   basic.showIcon(IconNames.Yes)
37 } else if (LL == 1 && (CC == 0 && RR == 0)) {
38   basic.showIcon(IconNames.No)
39 } else if (LL == 1 && (CC == 1 && RR == 1)) {
40   basic.showIcon(IconNames.Sad)
41 } else {
42   basic.showIcon(IconNames.Heart)
43 }
44 })
45
```





### Code 3:

We could use block “” to simplify code 2. The digital signal read by line tracking sensor is 0(low level) and 1(high level).

Then we transfer the digital signals from line tracking sensor into binary and decimal system, as shown below:

Level of left, middle and right TCRT5000 IR Tube			Binary	Decimal system
Low (0)	Low (0)	High (1)	001	1
Low (0)	High (1)	Low (0)	010	2
Low (0)	High (1)	High (1)	011	3
High (1)	Low (0)	Low (0)	100	4
High (1)	Low (0)	High (1)	101	5
High (1)	High (1)	Low (0)	110	6
High (1)	High (1)	High (1)	111	7
Low (0)	Low (0)	Low (0)	000	0



Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.14: Line Tracking Smart Car/7.14.1: Detect Line Tracking Sensor/Code-3	microbit-Code-3.hex

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

(1) Click "Advanced" → "Serial" → "serial redirect to USB"

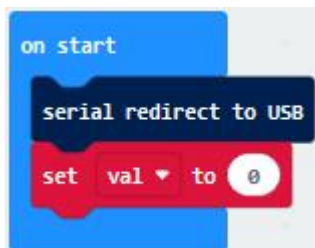
Place it into "on start" block.



(2) Click "Variables" → "Make a Variable..." → "New variable name: " dialog box;

Input val and click "OK" to generate variable "val"

Move out "set val to 0" and keep it into block "on start"



(3) Enter "Variables" → "set val to 0"

Keep it into "forever" block

Move "Line Tracking" from "TurtleBit" and place it into 0 box



(4) Click "Advanced" → "Serial" → "serial write value "x" =0"  
Put it into "forever" , and move variable "val" into 0 box.



\*\*\*\*\*

(5) Click "Logic" → "if true then...else"  
Move it into "forever" , tap "⊕" six times and move out "=" block into  
"true" block.



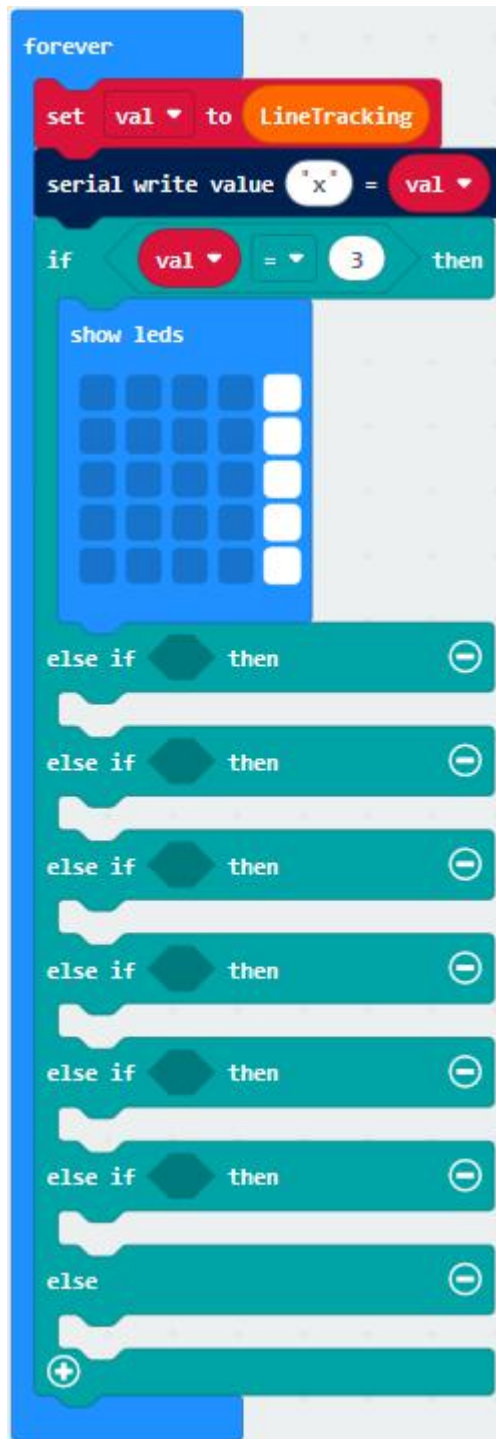
(6) Enter "Variables" to move block "val" into left box of "=" block  
And change 0 into 3.



(7) Enter "Basic" → "show leds"

Place it under the first block "if..val...then" block

Tick dark blue boxes to generate "I" pattern.



(8) Copy block "val=3 " once and place it into box behind the first "else of ... then" block and change 3 into 5



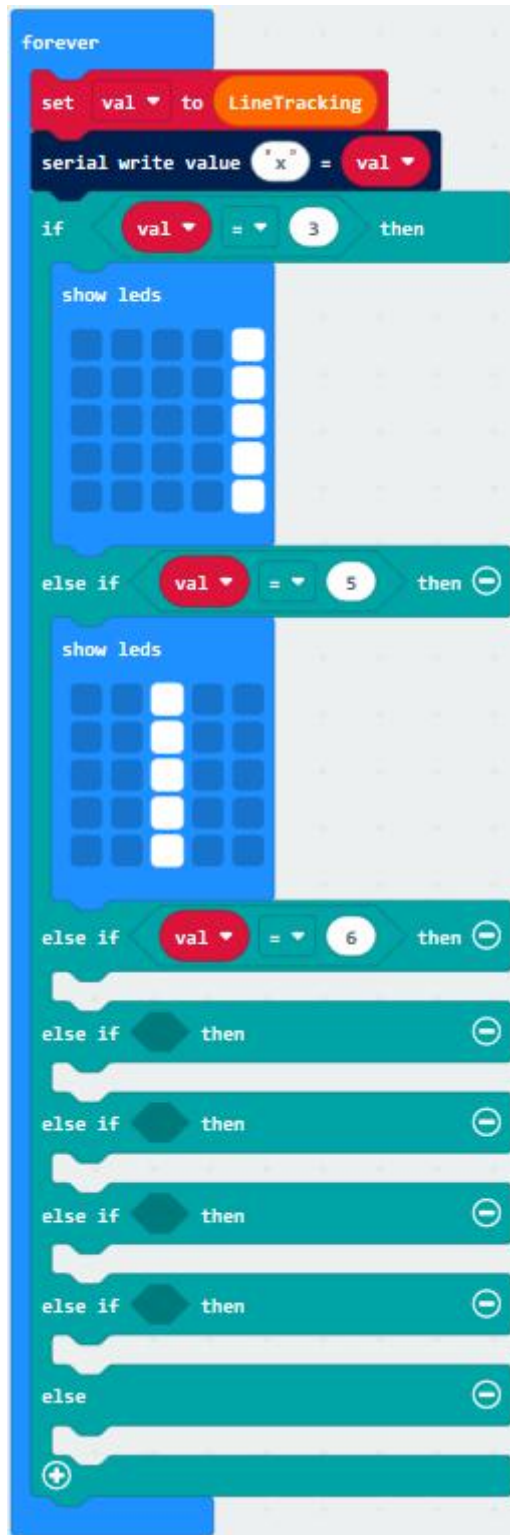
(9) Replicate “show leds” block and leave it under the first “else of ... then” block.

Tick blue boxes to produce “1”



(4) Duplicate block "val=5 " and place it into the second box behind the second "else if...then" block, change 5 into 6





(5) Duplicate "show leds" once and leave it into box behind the second "else if...then" block

Tick blue box to create "I"

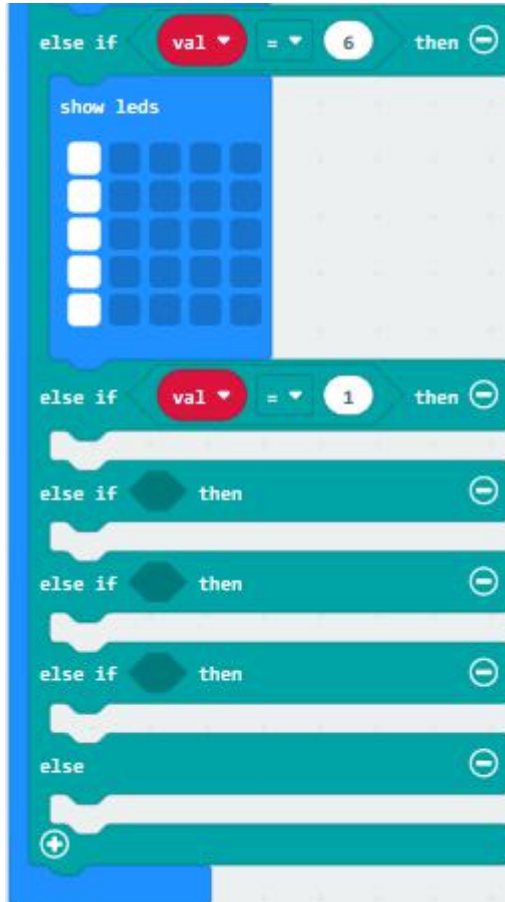


(6) Copy "val=6" block once and leave it into box behind the third "else"



if...then" block

Change 6 into 1



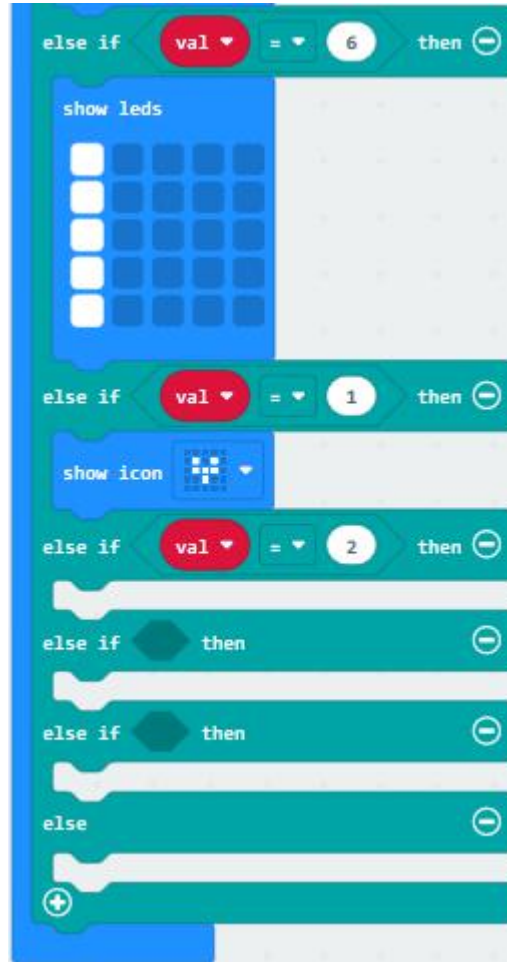
(7) Drag out "show icon" under the third "else if...then" block

Click "♥" to set "🎮"



(8) Copy "val=1" once and leave it into box behind the the fourth "else if ..then" block.

Change 1 into 2.



(9) Copy block "show icon" for four times and click the triangle button to select " " , " " , " " and " " .  
Set to val=1, 2, 4 and 7




```
else if (val == 1) then
  show icon [1]
else if (val == 2) then
  show icon [2]
else if (val == 4) then
  show icon [4]
else if (val == 7) then
  show icon [7]
else
  show icon [8]
```

Complete Code



```
on start
  serial redirect to USB
  set val to 0

forever
  set val to LineTracking
  serial write value "x" = val
  if val = 3 then
    show leds
  else if val = 5 then
    show leds
```



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

Serial redirects to USB

Set variable val to 0

The program under the block "forever" runs cyclically.

Set val to Line Tracking

Serial redirects x=val

When val=3, execute the program under then block

Micro:bit shows "l" at right

When val=5, execute the program under then block

Micro:bit shows "l" in the middle

When val=6, execute the program under then block

Micro:bit shows "l" at left

When val=1, execute the program under then block

Micro:bit shows " "

When val=2, execute the program under then block

Micro:bit shows " "

When val=4, execute the program under then block

Micro:bit shows " "

When val=7, execute the program under then block

Micro:bit shows " "

When the above condition is not met, execute the program under else block

Micro:bit shows "♥"

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





```
1 serial.redirectToUSB()
2 let val = 0
3 basic.forever(function () {
4   val = turtleBit.LineTracking()
5   serial.writeValue("x", val)
6   if (val == 3) {
7     basic.showLeds(`
8       . . . . #
9       . . . . #
10      . . . . #
11      . . . . #
12      . . . . #
13      `)
14   } else if (val == 5) {
15     basic.showLeds(`
16      . . # . .
17      . . # . .
18      . . # . .
19      . . # . .
20      . . # . .
21      `)
22   } else if (val == 6) {
23     basic.showLeds(`
24      # . . . .
25      # . . . .
26      # . . . .
27      # . . . .
28      # . . . .
29      `)
30   } else if (val == 1) {
31     basic.showIcon(IconNames.SmallHeart)
32   } else if (val == 2) {
33     basic.showIcon(IconNames.Yes)
34   } else if (val == 4) {
35     basic.showIcon(IconNames.No)
36   } else if (val == 7) {
37     basic.showIcon(IconNames.Sad)
38   } else {
39     basic.showIcon(IconNames.Heart)
40   }
41 })
42
```

### 3.Test Result:



Download code 2 to micro:bit board, "I" will be shown at left and indicator will be on when only left TCRT5000 IR tube on line tracking sensor detects white objects.

"I" will be shown in the middle and indicator will be on when only middle TCRT5000 IR tube on line tracking sensor detects white objects.

"I" will be shown at right and indicator will be on when only right TCRT5000 IR tube on line tracking sensor detects white objects.

As only left and middle TCRT5000 IR tubes detect white objects, micro:bit shows "II" and indicators at left and middle are on.

Micro:bit shows "IR" and indicators at left and right are on when only left and right TCRT5000 IR tubes detect white objects.

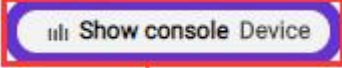
Micro:bit shows "RI" and indicators at right and middle are on when only middle and right TCRT5000 IR tubes detect white objects.

Micro:bit displays "00" and none of indicator is on when both of them detect black objects or no object is detected.

Both of them detect white objects, micro:bit shows "II" and indicators are on.



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

Download code 3 to micro:bit, and keep micro : bit connected and click 

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



Number 3 will be displayed when only left TCRT5000 IR tube detects white objects.

Number 5 will be displayed when only middle TCRT5000 IR tube detects white objects.

Number 6 will be displayed when only right TCRT5000 IR tube detects white objects.



Number 1 will be displayed when only left and middle TCRT5000 IR tubes detect white objects.

Number 2 will be displayed when only left and right TCRT5000 IR tubes detect white objects.

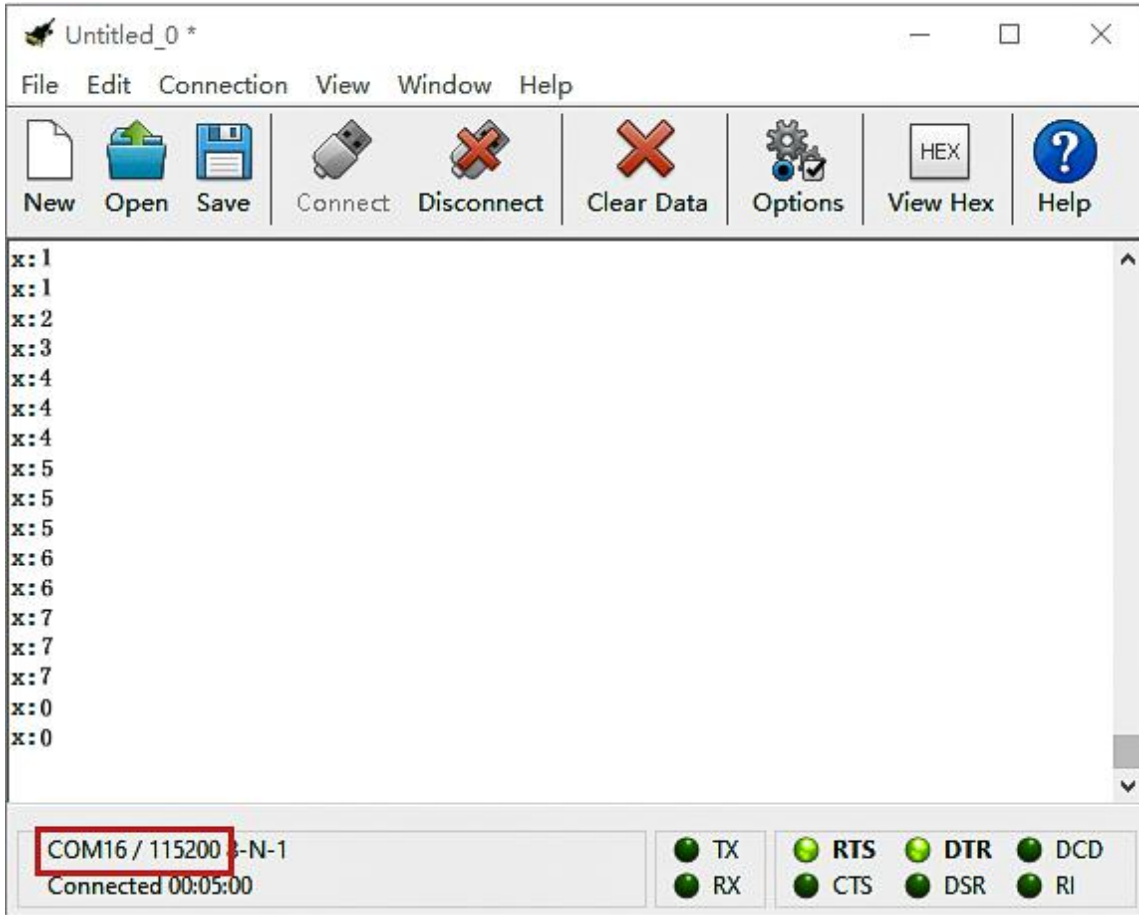
Number 4 will be displayed when only right and middle TCRT5000 IR tubes detect white objects.

Number 7 will be displayed when all TCRT5000 IR tubes detect black objects or not object is detected.



```
8 x:2
9 x:3
16 x:4
x:5
8 x:7
15 x:6
13 x:7
3 x:0
```

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 display the value of decimal system.



## 7.14.2: Line Tracking Smart Car



### 1.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.

Left/Middle/Right TCRT5000 IR Tunes (Level)			binary	Decimal system (a)	Turtle Smart Car
LOW (0)	LOW (0)	HIGH (1)	001	1	Turn Right
LOW (0)	HIGH (1)	LOW (0)	010	2	Go forward
LOW (0)	HIGH (1)	HIGH (1)	011	3	Go forward
HIGH (1)	LOW (0)	LOW (0)	100	4	Turn Left
HIGH (1)	LOW (0)	HIGH (1)	101	5	Go forward



HIGH (1)	HIGH (1)	LOW (0)	110	6	Go forward
HIGH (1)	HIGH (1)	HIGH (1)	111	7	Go forward
LOW (0)	LOW (0)	LOW (0)	000	0	Stop

**Black Line**



The three-channel line tracking sensor is connected to integrated pin(G,5V P14, P15, P16) and controlled by P14, P15 and P16.

## **2. Experimental Preparation:**

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.





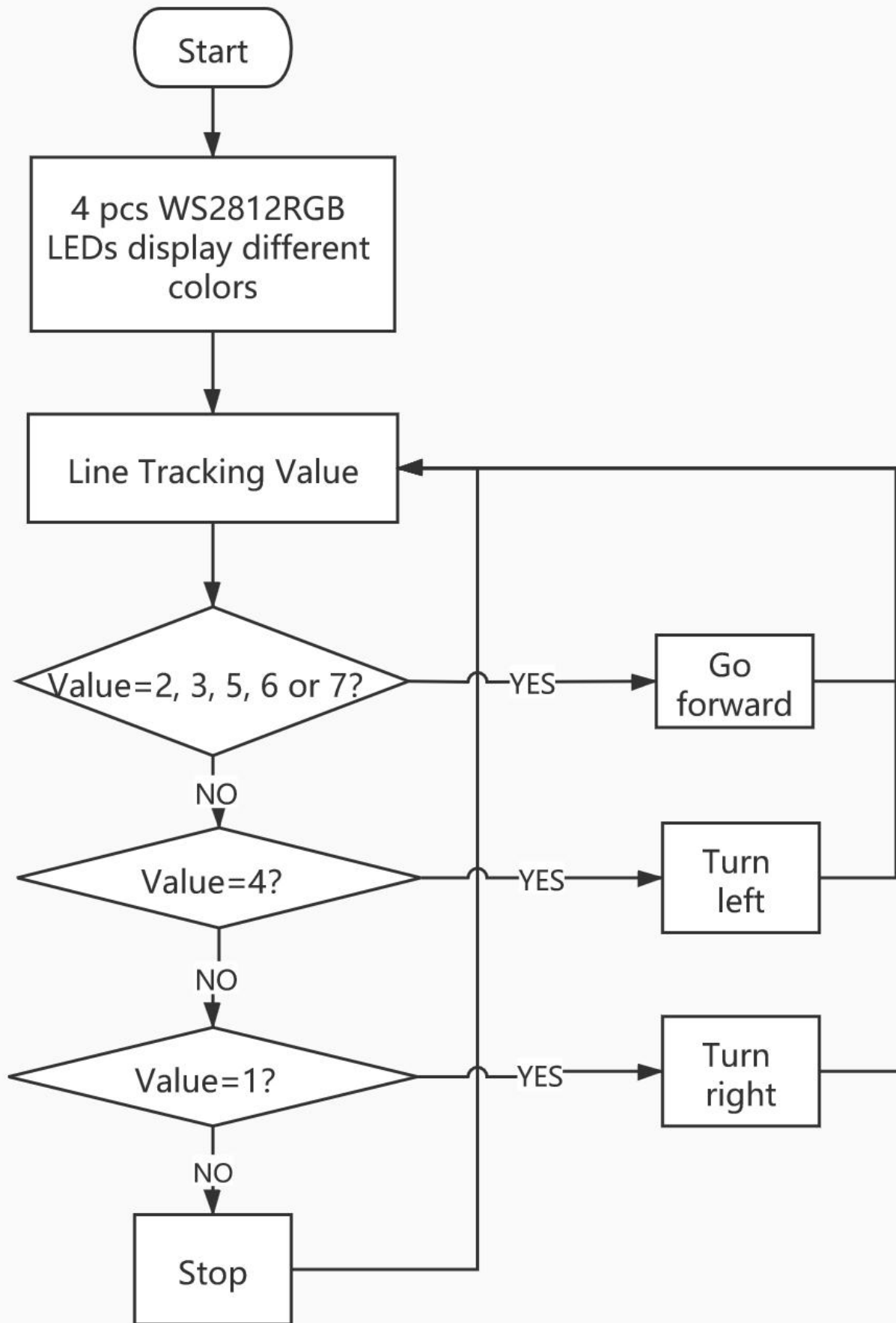
(3) Dial power switch to ON end

(4) Connect micro:bit to computer by USB cable and open online

Makecode editor.

Warning: the line tracking sensor can't work normally under strong light because there is a mountain of invisible light including IR and ultraviolet rays.

### 3. Flow Chart





#### 4. Test Code:

Type	Route	File Name
Hex file	../Test Code/7.14: Line Tracking Smart Car/7.14.2: Line Tracking Smart Car	microbit-Line Tracking Smart Car.hex

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

(1) Enter "Basic" → "show icon ♥"

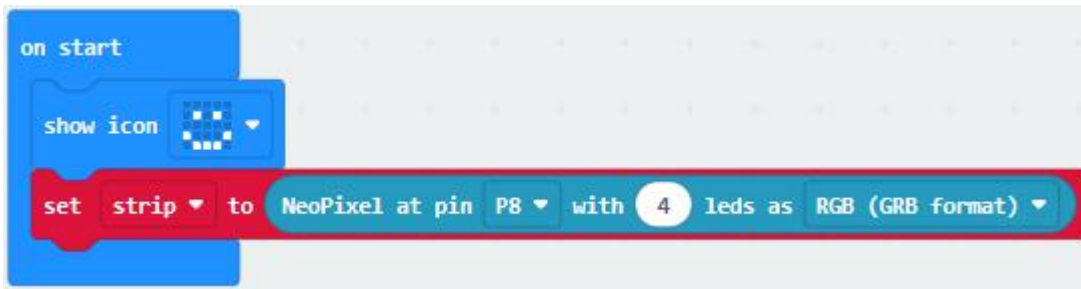
Combine it with "on start" block, and click triangle button to select "🔲".



(2) Click "Neopixel" → "set strip to Neopixel at pin P0 with 24 leds as RGB (GRB format)"

Put it into "on start" block, P8 of 4pcs WS2812RGB lights is connected by P8 of micro:bit.

Click the triangle button to set to P8 and set to 4 leads



(3) Enter "Neopixel" → "strip clear"

Place it into "on start"



(4) Go to "Variables" → "Make a Variable..." → "New variable name: " Enter "tracking values" and click "OK" , then variable "tracking values" is set up.

Drag "set tracking values to 0" into forever block.

Tap turtle-bit to move "Line Tracking" block into 0 box.





(5) Click "Logic" and drag "if true then...else" under the block "set...tracking" .

Tap "+" twice, move "or" block into true box, then copy "or" block for three times and leave them into right box of "or" block.

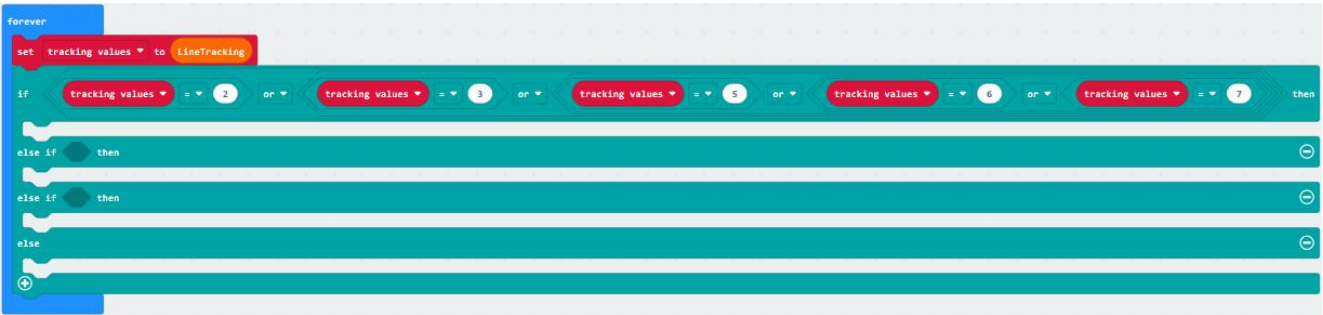


(6) Tap "Logic" to drag out "=" and leave it at left box of "or" block  
Click "Variables" to move "tracking values" into right box of "=" block.

Change 0 into 2

Copy "tracking values=2" for four times, and separately leave them into boxes as follows:

Change 2 into 3, 5, 6 and 7



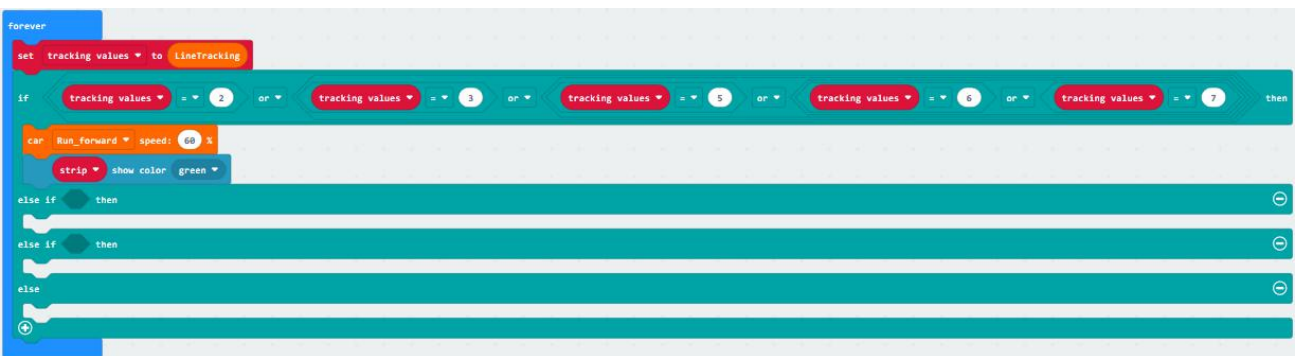
(7) Click "TurtleBit" to drag "car Run\_forward speed: 0 %" under "if...then" block

Change 0 into 60

Tap "Neopixel" to get block "strip show color red"

Change red into green

Leave "strip show color green" under "car...68%" block



(8) Duplicate block "tracking values=2" once and place it into box behind the first "else if" block, change 2 into 4.



```
forever
set tracking values to LineTracking
if tracking values = 2 or tracking values = 3 or tracking values = 5 or tracking values = 6 or tracking values = 7 then
car Run_forward speed: 68 %
strip show color green
else if tracking values = 4 then
else if then
else
```

(9) Click "TurtleBit" to drag "LeftSide motor run Forward speed: 0 %"

Copy it once and leave them under the "else if...then" block

Then set blocks as follows:

Copy "strip show color green" once and change green into blue.

```
forever
set tracking values to LineTracking
if tracking values = 2 or tracking values = 3 or tracking values = 5 or tracking values = 6 or tracking values = 7 then
car Run_forward speed: 68 %
strip show color green
else if tracking values = 4 then
LeftSide motor run Back speed: 88 %
RightSide motor run Forward speed: 88 %
strip show color blue
else if then
else
```

(10) Copy "tracking values=4" once and place it into box behind the second "else if" , change 4 into 1.



```
forever
set tracking values to LineTracking
if tracking values = 2 or tracking values = 3 or tracking values = 5 or tracking values = 6 or tracking values = 7 then
car Run_forward speed: 80 %
strip show color green
else if tracking values = 4 then
LeftSide motor run Back speed: 80 %
RightSide motor run Forward speed: 80 %
strip show color blue
else if tracking values = 1 then
else
```

(11) Copy

```
LeftSide motor run Back speed: 80 %
RightSide motor run Forward speed: 80 %
strip show color blue
```

once, and place it under the

“second else..then” block

Set the code string as follows:

```
forever
set tracking values to LineTracking
if tracking values = 2 or tracking values = 3 or tracking values = 5 or tracking values = 6 or tracking values = 7 then
car Run_forward speed: 80 %
strip show color green
else if tracking values = 4 then
LeftSide motor run Back speed: 80 %
RightSide motor run Forward speed: 80 %
strip show color blue
else if tracking values = 1 then
LeftSide motor run Forward speed: 80 %
RightSide motor run Back speed: 80 %
strip show color yellow
else
```

(12) Click “TurtleBit” to move “car stop” under the else block

Copy “strip show color yellow” once and alter yellow into red.

Place blocks as follows:






```
forever
  set tracking values to LineTracking
  if tracking values = 2 or tracking values = 3 or tracking values = 5 or tracking values = 6 or tracking values = 7 then
    car Run_forward speed: 80 %
    strip show color green
  else if tracking values = 4 then
    LeftSide motor run Back speed: 80 %
    RightSide motor run Forward speed: 80 %
    strip show color blue
  else if tracking values = 1 then
    LeftSide motor run Forward speed: 80 %
    RightSide motor run Back speed: 80 %
    strip show color yellow
  else
    car stop
    strip show color red
```

## 5. Complete Code

```
on start
  show icon
  set strip to NeoPixel at pin P8 with 4 leds as RGB (GRB format)
  strip clear
forever
  set tracking values to LineTracking
  if tracking values = 2 or tracking values = 3 or tracking values = 5 or tracking values = 6 or tracking values = 7 then
    car Run_forward speed: 80 %
    strip show color green
  else if tracking values = 4 then
    LeftSide motor run Back speed: 80 %
    RightSide motor run Forward speed: 80 %
    strip show color blue
  else if tracking values = 1 then
    LeftSide motor run Forward speed: 80 %
    RightSide motor run Back speed: 80 %
    strip show color yellow
  else
    car stop
    strip show color red
```



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

Micro:bit shows “”

Set strip to Neopixel pin 8 with 4 leads(RGB format)

Turn off all RGB on strip

The program under the block “forever” runs cyclically.

Set tracking values to the value read by line tracking sensor

When tracking values=2, 3, 5, 6 or 7, execute the program under then block

Go forward at the 60% speed

4pcs WS2812RGB display green color

When tracking values=4, execute the program under then block.

The left car wheels go back at 80% speed

The right car wheels go forward at 80% speed

4pcs WS2812RGB display blue color

When tracking values=1, execute the program under then block

The left car wheels go backward at 80% speed

The right car wheels go backward at 80% speed

4pcs WS2812RGB display yellow color


When tracking values is not met, execute the program under else block

turtle car stops

4pcs WS2812RGB display red color



Click "JavaScript" to view the corresponding JavaScript code:



```
1 let tracking_values = 0
2 basic.showIcon(IconNames.Happy)
3 let strip = neopixel.create(DigitalPin.P8, 4, NeoPixelMode.RGB)
4 strip.clear()
5 basic.forever(function () {
6   tracking_values = turtleBit.LineTracking()
7   if (tracking_values == 2 || (tracking_values == 3 || (tracking_values == 5 || (tracking_values == 6 || tracking_values == 7)))) {
8     turtleBit.run(DIR.Run_forward, 60)
9     strip.showColor(neopixel.colors(NeoPixelColors.Green))
10  } else if (tracking_values == 4) {
11    turtleBit.Motor(MOTOR.LeftSide, MD.Back, 80)
12    turtleBit.Motor(MOTOR.RightSide, MD.Forward, 80)
13    strip.showColor(neopixel.colors(NeoPixelColors.Blue))
14  } else if (tracking_values == 1) {
15    turtleBit.Motor(MOTOR.LeftSide, MD.Forward, 80)
16    turtleBit.Motor(MOTOR.RightSide, MD.Back, 80)
17    strip.showColor(neopixel.colors(NeoPixelColors.Yellow))
18  } else {
19    turtleBit.state(MotorState.stop)
20    strip.showColor(neopixel.colors(NeoPixelColors.Red))
21  }
22 })
23
```

## 6. Test Result:

Download code to micro:bit and dial POWER to ON end, line tracking car goes forward along black line and turn on WS2812 RGB lights

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

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.



## 7.15: Ultrasonic Follow Smart Car

### 7.15.1: Ultrasonic Ranging

#### 1. Description:

The HC-SR04 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 HC-SR04 or 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.

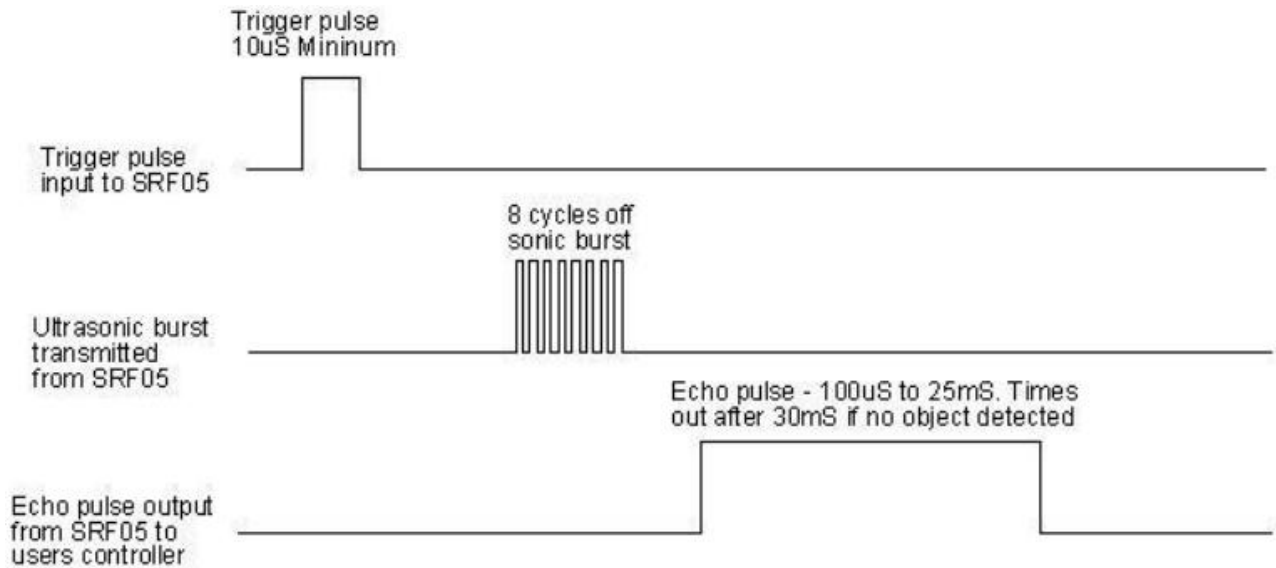


As the above picture shown, it is like two eyes. One is transmitting end, the other is receiving end.

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 and echo signal.



## 2. Working Principle:



1. Pull down TRIG then trigger high level signals with least 10µs
2. After triggering, the module will automatically send eight 40KHz ultrasonic pulses and detect whether there is a signal return.
3. The propagation speed of sound in the air is about 343m/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. Specification:

- Working voltage: 3-5.5V (DC)
- Power Supply :+5V DC
- Working Current: 15mA
- Working frequency: 40khz



- Maximum Ranging Distance : around 3m
- Minimum Ranging Distance: 2-3cm
- Resolution : 0.3 cm
- Measuring Angle:  $\leq 15$  degree
- Trigger Input Pulse width: 10uS
- Accuracy: up to 0.2cm
- Output echo signal : output TTL level signal(high), which is proportion to range.

#### **4. Experimental Preparation:**

Insert micro:bit board into slot of V2 shield.

Place batteries into battery holder.

Dial power switch to ON end

Connect micro:bit to computer by USB cable and open online Makecode editor.

Import Hex profile ([How to import?](#)) , or click "New Project" and drag blocks step by step([add turtle-bit extension library first](#))

[\(How to add turtle-bit extension?\)](#)



### 5. Test Code:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.15: Ultrasonic Follow Smart Car/7.15.1: Ultrasonic Ranging	microbit-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

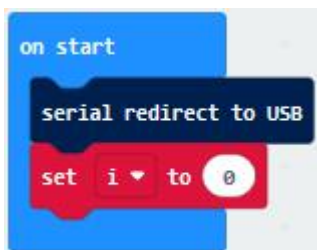


\*\*\*\*\*

(2) Go to "Variables" → "Make a Variable..." → "New variable name:" dialog box,

Input i and click "OK" to produce variable "i" ,

Move "set i to 0" from "Variables" and integrate with "on start" block



\*\*\*\*\*



(3) Tap "Advanced" → "Serial" → "serial write value x=0"

Place it under "set i to 0" block

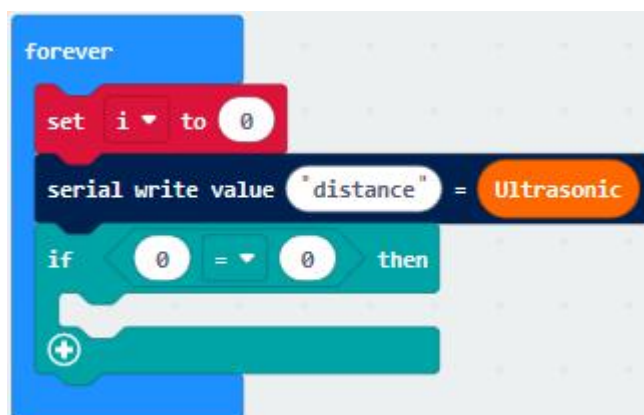
Click "TurtleBit" to move "Ultrasonic" into 0 box of "=" block

Change X into distance



(4) Click "Logic" to drag "if true then" block into "forever" block.

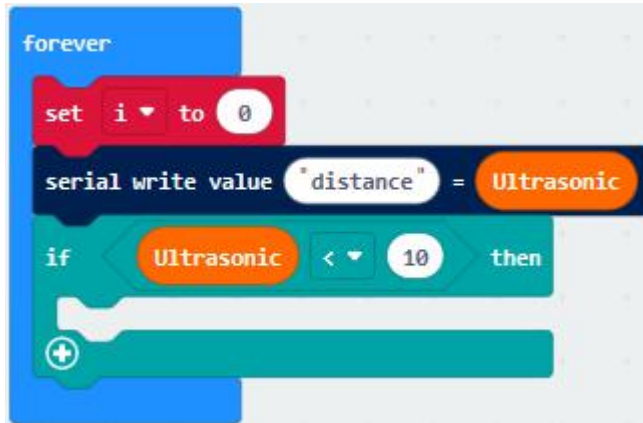
Move "=" into true box



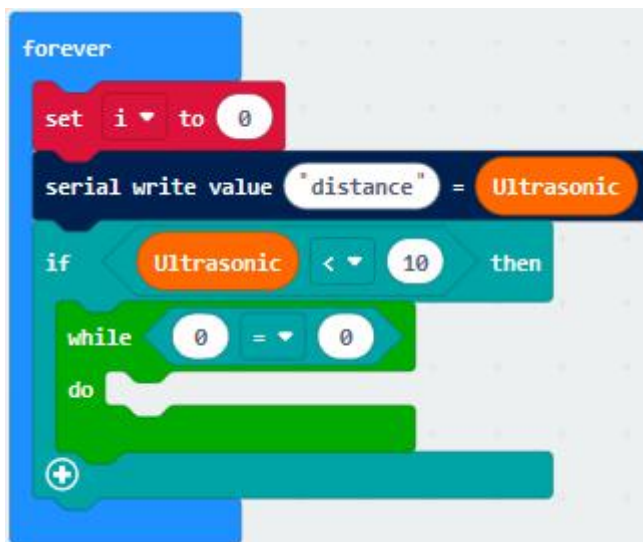
(5) Click "TurtleBit" to drag "Ultrasonic" to left box of "="

Change "=" into "<" , 0 into 10.





(6) Tap "Loops" to get block "while true do" and place it into forever block  
Click "Logic" to move "=" block into true box



(7) Click "Variables" to move variable "i" to left box of "=" block  
Change "=" into "<" and 0 into 1



(8) Click "Music" and move "play tone Middle C for 1 beat" block into do block

Tap "Basic" to move "pause(ms)100" under "play...beat" block

Change 100 into 200

Copy

once and place it into do block



```
forever
  set i to 0
  serial write value "distance" = Ultrasonic
  if Ultrasonic < 10 then
    while i < 1
      do
        play tone Middle C for 1 beat
        pause (ms) 200
        play tone Middle C for 1 beat
        pause (ms) 200
```

Click "Variables" to drag "change i by 1" into do block

```
forever
  set i to 0
  serial write value "distance" = Ultrasonic
  if Ultrasonic < 10 then
    while i < 1
      do
        play tone Middle C for 1 beat
        pause (ms) 200
        play tone Middle C for 1 beat
        pause (ms) 200
        change i by 1
```

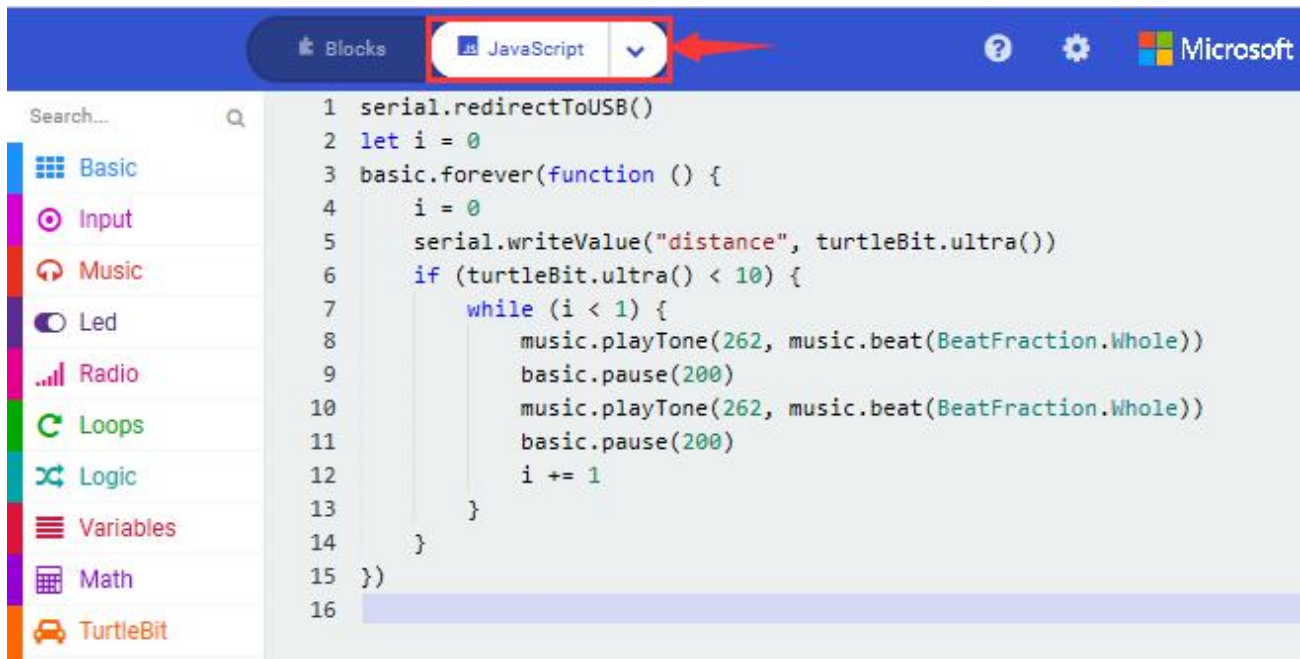


## Complete Code

```
on start
  serial redirect to USB
  set i to 0
forever
  set i to 0
  serial write value "distance" = Ultrasonic
  if Ultrasonic < 10 then
    while i < 1
      do
        play tone Middle C for 1 beat
        pause (ms) 200
        play tone Middle C for 1 beat
        pause (ms) 200
        change i by 1
```

"on start" : command block runs once to start program.  
Serial redirect to USB  
Set variable i to 0  
The program under the block "forever" runs cyclically.  
Set variable i to 0  
Serial writes distance=Ultrasonic  
When Ultrasonic<10, execute the program under then block  
When variable i <1, execute the program under do block  
Play tone C for 1 beat  
Delay in 200ms  
Play tone C for 1 beat  
Delay in 200ms  
Change i by 1

Click "JavaScript" to view the corresponding JavaScript code:



## 6. Test Result:

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.



The screenshot displays the Keyestudio IDE interface. On the left is a 3D model of a Micro:bit board with a USB cable connected. Below it are control buttons: a square stop button, a circular refresh button, a speaker icon for volume, and a square button with a play symbol. Two purple buttons are visible: "Show console Simulator" and "Show console Device".

On the right, there is a "Go back" button and a "Device" section with a green play button, a blue download button, and a blue share button. Below these is a graph showing a green line representing distance over time. The y-axis ranges from 4.00 to 8.00. An orange callout box on the graph says "distance: 8".

At the bottom right is a serial console window with a scroll bar, displaying the following text:

```
4 distance:5  
distance:4  
5 distance:3  
4 distance:4  
distance:5  
4 distance:6  
distance:7  
4 distance:8
```

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 View Window Help

New Open Save Connect Disconnect Clear Data Options View Hex Help

```
distance (cm) :2
distance (cm) :2
distance (cm) :2
distance (cm) :2
distance (cm) :3
distance (cm) :3
distance (cm) :3
distance (cm) :3
distance (cm) :3
distance (cm) :3
distance (cm) :4
distance (cm) :5
distance (cm) :5
distance (cm) :5
distance (cm) :5
distance (cm) :5
distance (cm) :6
distance (cm) :8
distance (cm) :6
distance (cm) :6
distance (cm) :6
distance (cm) :6
distance (cm) :7
distance (cm) :7
distance (cm) :8
distance (cm) :8
```

COM16 / 115200 -N-1  
Connected 00:20:39

TX  RTS  DTR  DCD  
 RX  CTS  DSR  RI





## 7.15.2: Ultrasonic Avoidance Car



### 1. 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:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end

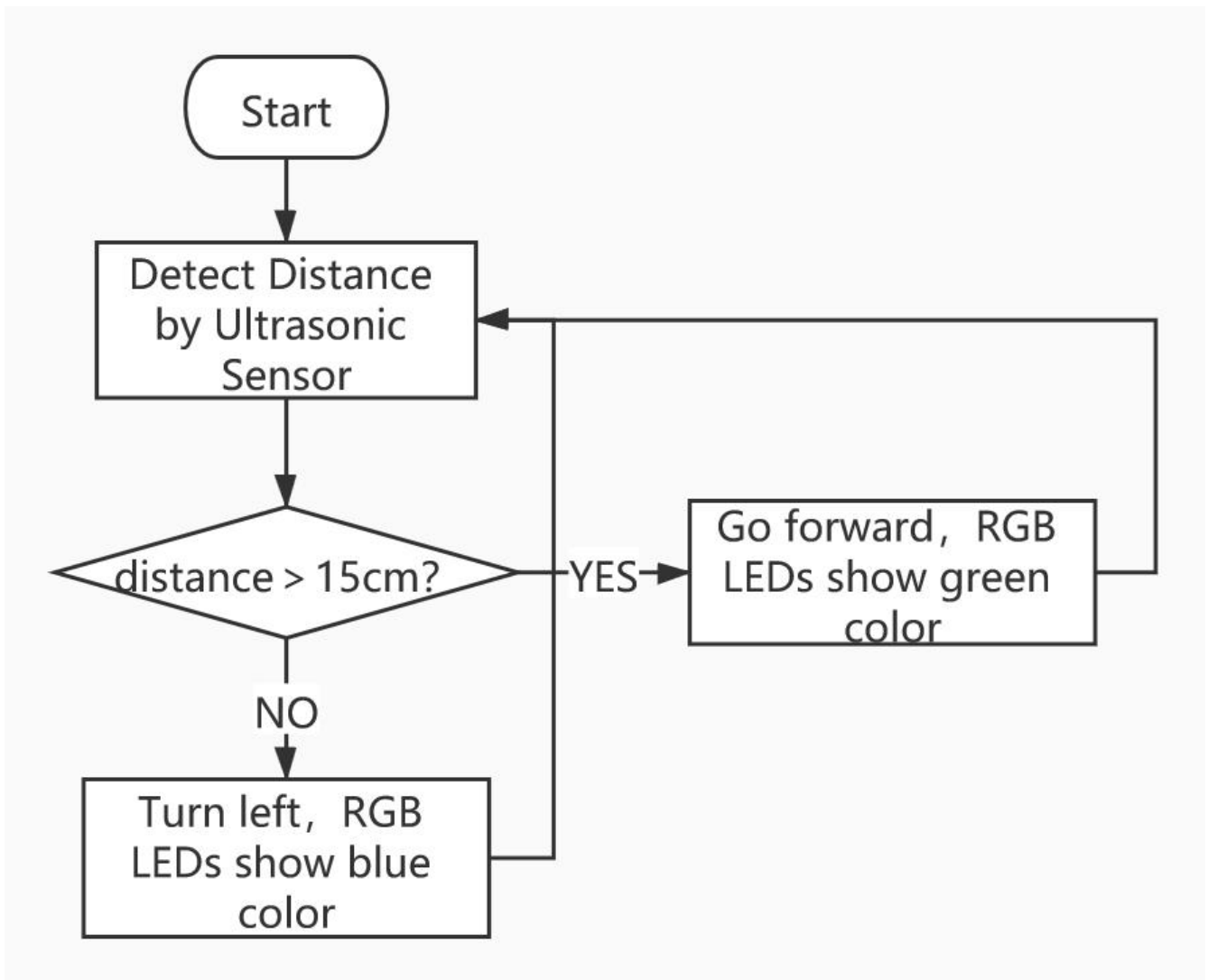


(4) Connect micro:bit to computer by USB cable and open online Makecode editor.

(5) Import Hex profile ([How to import?](#)), or click "New Project" and drag blocks step by step([add turtle-bit extension library first](#))

[\(How to add turtle-bit extension?\)](#)

### 3. Flow Chart





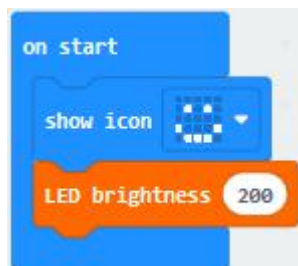
#### 4. Test Code:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.15: Ultrasonic Follow Smart Car/7.15.2: Ultrasonic Avoidance Car	microbit-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.



(2) Click "TurtleBit" to move "LED brightness 0" into "on start" block



Change 0 into 200

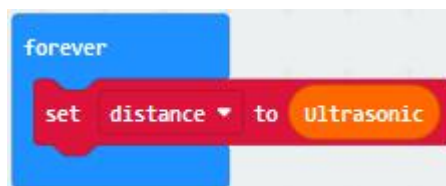


(3) Tap "Variables" → "Make a Variable..."

Put "distance" in the search bar

Click "OK" to set up variable "distance"

Place it under "LED brightness 2000" block

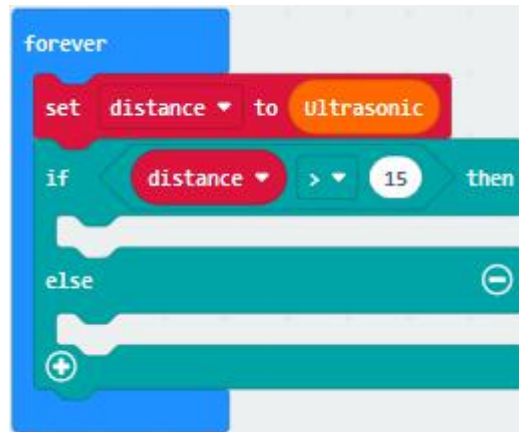


(4) Go to "Logic" to move "if true then...else" under "set...ultrasonic" block

Drag "=" block into true box

Click "Variables" to move variable "distance" into left box of "=" block

Change "=" into ">" , 0 into 15.



(5) Tap "TurtleBit" to move "car Run\_forward speed: 0%" block under "if...then" block

Change 0 into 80

Drag "set left\_side RGBled red" and place it under "car...80%" block

Copy it once and change red into green

Set code string as follows



(6) Click "TurtleBit" to drag "LeftSide motor run forward speed: 0%" block into else block

Change "forward" into back, 0 into 60



Copy "LeftSide motor run back speed: 60%" again

Set the code string as follows:

```
forever
  set distance to Ultrasonic
  if distance > 15 then
    car Run_forward speed: 80%
    set left_side RGBled green
    set right_side RGBled green
  else
    LeftSide motor run Back speed: 60%
    RightSide motor run Forward speed: 80%
```

(7) Duplicate  once and leave it under else block



Click green to select blue



```
forever
  set distance to Ultrasonic
  if distance > 15 then
    car Run_forward speed: 80 %
    set left_side RGBled green
    set right_side RGBled green
  else
    LeftSide motor run Back speed: 60 %
    RightSide motor run Forward speed: 80 %
    set left_side RGBled blue
    set right_side RGBled blue
```

Complete Code

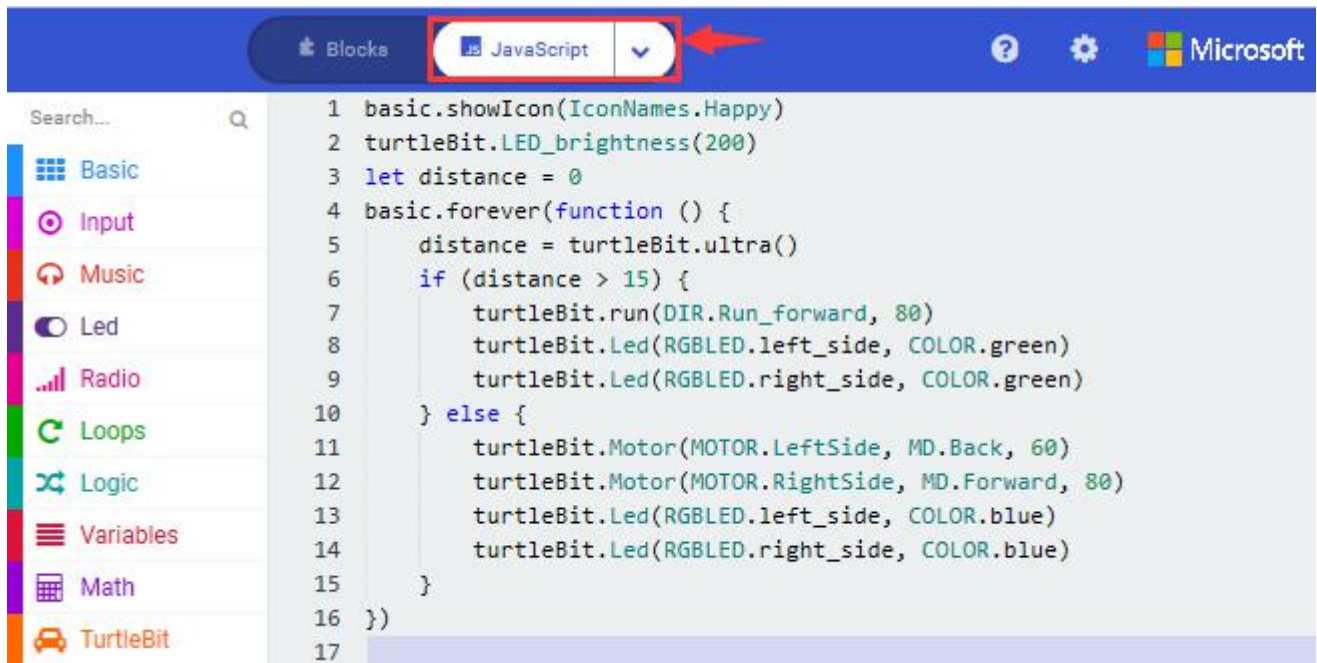


```
on start
  show icon [LED icon]
  LED brightness 200
  set distance to 0

forever
  set distance to Ultrasonic
  if distance > 15 then
    car Run_forward speed: 80 %
    set left_side RGBled green
    set right_side RGBled green
  else
    LeftSide motor run Back speed: 60 %
    RightSide motor run Forward speed: 80 %
    set left_side RGBled blue
    set right_side RGBled blue
```

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





## 5. Test Result:

Download code to micro:bit, dial to ON end, and dial POWER to ON end. When the obstacle distance is greater than 15cm, turtle car goes forward and 2 RGB lights show green color; on the contrary, smart car turns left and 2 RGB lights show blue color.

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



### 7.15.3: Ultrasonic Follow Smart Car



#### 1. 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:

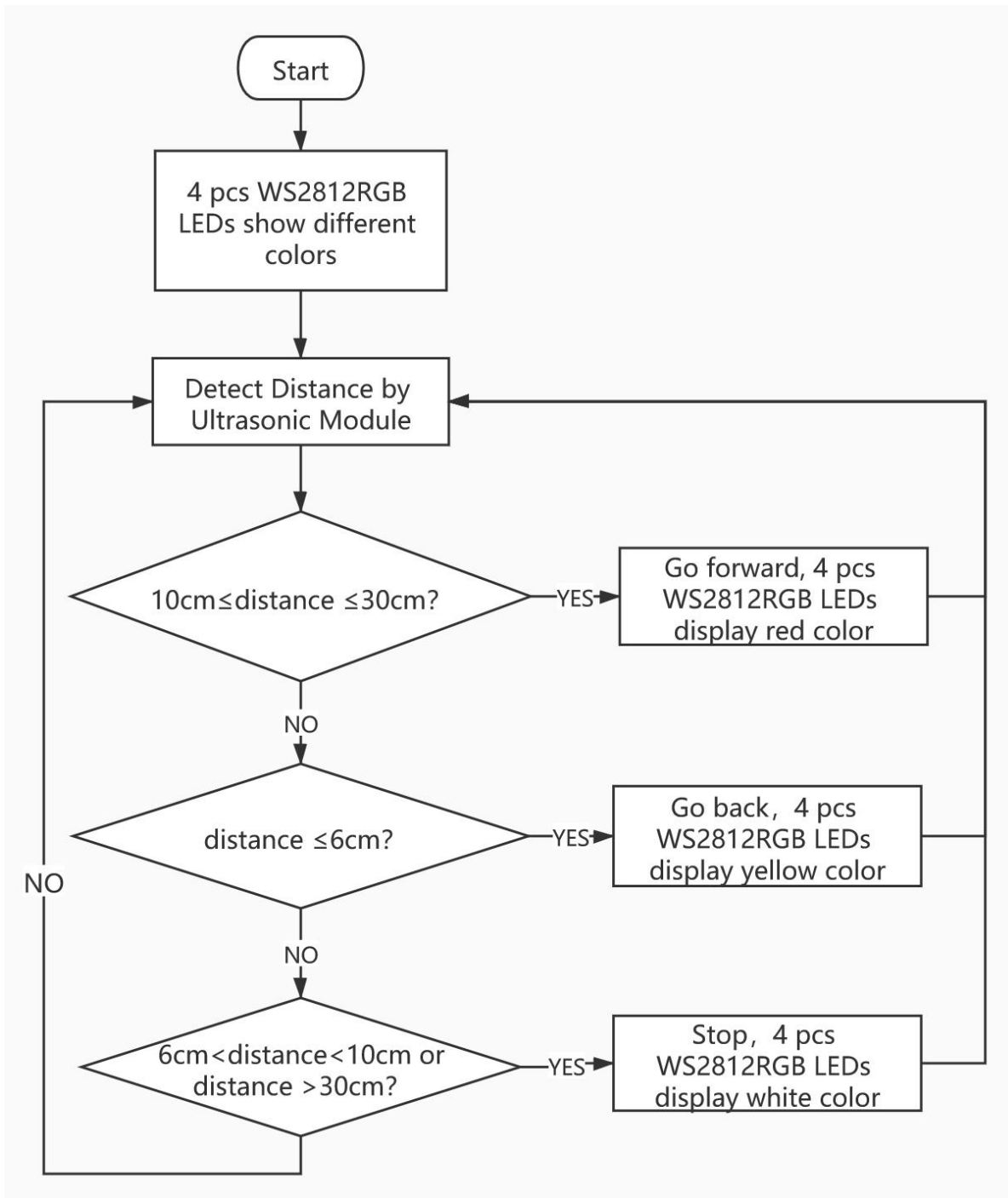
- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end
- (4) Connect micro:bit to computer by USB cable and open online Makecode editor.
- (5) Import Hex profile ([How to import?](#)), or click "New Project" and drag



blocks step by step(add turtle-bit extension library first)

(How to add turtle-bit extension?)

### 3. Flow Chart





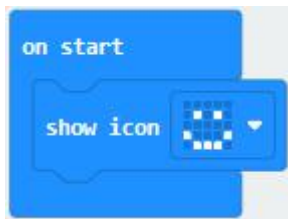
#### 4. Test Code:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.15: Ultrasonic Follow Smart Car/7.15.3: Ultrasonic Follow Smart Car	microbit-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 "🔲" pattern.



\*\*\*\*\*

(2) 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. 4 pcs WS2812 RGB lights are controlled by P8 of micro:bit board . So we set to P8 and 4 leads



(3) Tap "Neopixel" → "more" → "strip set brightness 255"

Leave it under the block "set...(RGB format)"

To reduce the brightness of WS2812 RGB lights, change 255 into 100.



(4) Click "Neopixel" to find "strip clear"

Keep it into "on start" block.



(5) Go to "Variables" → "Make a Variable..." → "New variable name:" dialog



box,

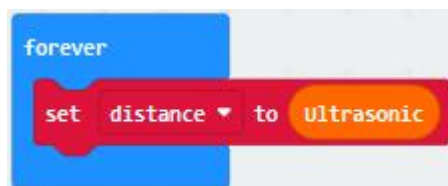
Input distance and click "OK" to produce variable "distance" ,

Drag out "set distance to 0" into "on start"



(6) Click "Variables" to move "set distance to 0" into forever block

Click "TurtleBit" to drag block "Ultrasonic" into 0 box.



(7) Enter "Logic" to move out "if true then" into "forever" ,

Then place "and" block into true box.



(8) Click "Logic" to move out "=" into left box of "and" block  
Move out variable "distance" into left box of "=" block  
Change "=" into "≥" and 0 into 10.  
Replicate "distance ≥ 10" once and leave it into right box of "and" block.  
Then we set distance ≤ 30



(5) Click "TurtleBit" → "car Run\_Forward speed: 0 %"  
Leave it under "if...30 then" block. Then change 0 into 80.  
Go to "Neopixel" → "strip show color red"  
Place it under "car Run\_Forward speed: 80 %" block.



```
forever
  set distance to Ultrasonic
  if distance >= 10 and distance <= 30 then
    car Run_forward speed: 80 %
    strip show color red
```

(6) Duplicate code string

```
if distance >= 10 and distance <= 30 then
  car Run_forward speed: 80 %
  strip show color red
```

once and

keep it into "forever" ,

Delete block "distance≥10" and "and" block,

Change 30 into 6, Run\_Forward into Run\_Back , 80 into 60 and red into yellow.





```
forever
  set distance to Ultrasonic
  if distance >= 10 and distance <= 30 then
    car Run_forward speed: 80 %
    strip show color red
  if distance <= 6 then
    car Run_back speed: 60 %
    strip show color yellow
```

```
if distance <= 6 then
  car Run_back speed: 60 %
  strip show color yellow
```

(7) Copy code string once and place it into "forever" ,

Delete block "distance≤6" and "car RunBack speed: 60 %"

Go to "Logic" to drag out block "or" into true box,

Move block "and" into left box of "or" block,

Replicate block "distance≤6" once and place it into left box of "and" block,

Change "≤" into ">" , and duplicate block "distance≥10" once and move



into right box of "and" ,

Change " $\geq$ " into " $<$ " , and copy block "distance $\leq$ 30" once and drag into right box of "or" ,

Change " $\leq$ " into " $>$ " , and move block "car stop" from "TurtleBit"

Keep "car stop" block under "if...then" block

Alter yellow into white.

```
forever
  set distance to Ultrasonic
  if distance >= 10 and distance <= 30 then
    car Run_forward speed: 80 %
    strip show color red
  +
  if distance <= 6 then
    car Run_back speed: 60 %
    strip show color yellow
  +
  if distance > 6 and distance < 10 or distance > 30 then
    car stop
    strip show color white
  +
```

Complete Code:



```
on start
  show icon
  set strip to NeoPixel at pin P8 with 4 leds as RGB (GRB format)
  strip set brightness 100
  strip clear
  set distance to 0

forever
  set distance to Ultrasonic
  if distance ≥ 10 and distance ≤ 30 then
    car Run_forward speed: 80 %
    strip show color red
  +
  if distance ≤ 6 then
    car Run_back speed: 60 %
    strip show color yellow
  +
  if distance > 6 and distance < 10 or distance > 30 then
    car stop
    strip show color white
  +
```

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



```
1 basic.showIcon(IconNames.Happy)
2 let strip = neopixel.create(DigitalPin.P8, 4, NeoPixelMode.RGB)
3 strip.setBrightness(100)
4 strip.clear()
5 let distance = 0
6 basic.forever(function () {
7   distance = turtleBit.ultra()
8   if (distance >= 10 && distance <= 30) {
9     turtleBit.run(DIR.Run_forward, 80)
10    strip.showColor(neopixel.colors(NeoPixelColors.Red))
11  }
12  if (distance <= 6) {
13    turtleBit.run(DIR.Run_back, 60)
14    strip.showColor(neopixel.colors(NeoPixelColors.Yellow))
15  }
16  if (distance > 6 && distance < 10 || distance > 30) {
17    turtleBit.state(MotorState.stop)
18    strip.showColor(neopixel.colors(NeoPixelColors.White))
19  }
20 })
21
```

## 5. Test Result:

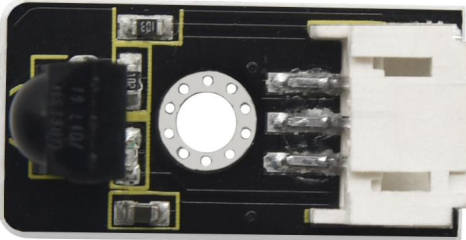
Download code to micro:bit, dial POWER switch to ON end on shield, smart car could follow the obstacle to move and WS2812 RGB lights show different color

**Note: the obstacle only moves in front of smart car, not turning**



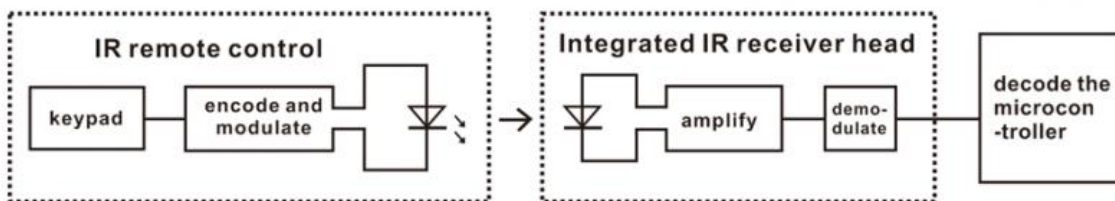
## 7.16: IR Remote Control Smart Car

### 7.16.1: Decode IR Remote Control



#### 1. 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.

## **2. Specification:**

- 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:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end
- (4) Connect micro:bit to computer by USB cable and open online Makecode editor.
- (5) Import Hex profile ([How to import?](#)), or click "New Project" and drag blocks step by step(**add turtle-bit extension library first**)  
[\(How to add turtle-bit extension?\)](#)

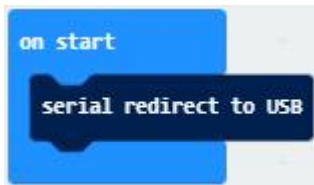
### 4. Test Code:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.16: IR Remote Control Smart Car/7.16.1: Decode IR Remote Control	microbit-Decode IR Remote Control.hex

- (1) Click "Advanced" → "Serial" → "serial redirect to USB"



Place it into "on start" block.



(2) Enter "IrRemote" → "connect IR receiver at P0"

Put it into "on start" block

IR receiving module is controlled by P11 of micro:bit board, so click P0 to select P11.



(3) 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.



(4) Go to "Ir Remote" → "IR button"

Place it into 0 box



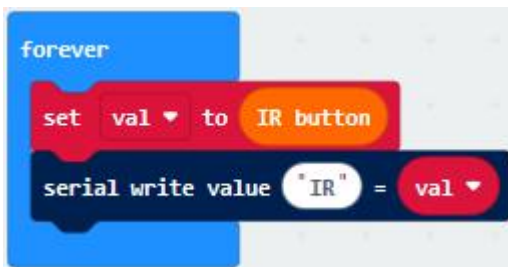


(5) Click "Advanced" → "Serial" → "serial write value "x" =0"

Put it into "forever" block

Change "x" into "IR"

Enter "Variables" to move block "val" into 0 box behind "="




(6) Drag out block "pause (ms) 100" from "Basic" and delay in 1000ms

Leave it into "forever" block





## Complete Program:



The diagram shows a Scratch-style block-based program. It starts with an "on start" block containing "serial redirect to USB" and "connect IR receiver at P11". This is followed by a "forever" loop containing "set val to IR button", "serial write value 'IR' = val", and "pause (ms) 1000".

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

Serial redirect to USB

Connect IR receiver to P11

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:



The screenshot shows the software interface with the "JavaScript" tab selected. A red arrow points to the "JavaScript" tab. The code in the editor is as follows:

```
1 let val = 0
2 serial.redirectToUSB()
3 irRemote.connectInfrared(DigitalPin.P11)
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



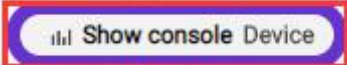
constantly shows 0; when pressed, the corresponding key values are displayed.

**Note:**

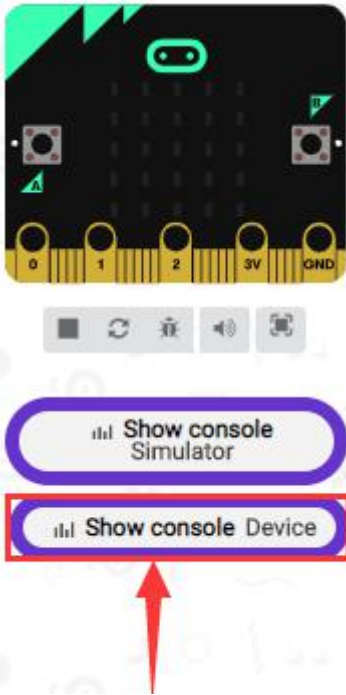
The remote control in this kit is not inclusive of batteries. We recommend you to purchase them online.(battery type:CR2025).

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 

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



Make IR remote control point at IR receiver and press the button, the serial monitor will display the corresponding key values, as shown below:



← Go back Device [Pause] [Download] [Share]

IR: 0 0.00

IR: 0 94.00

IR:0  
IR:13  
IR:0  
IR:94  
IR:8  
IR:28  
IR:66  
IR:0  
IR:74  
2 IR:0

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:



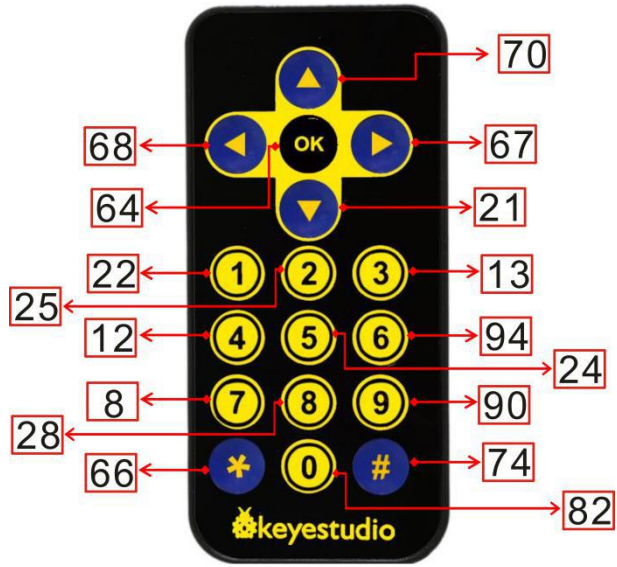
The screenshot shows the Keyestudio software interface. At the top, there is a menu bar with 'File', 'Edit', 'Connection', 'View', 'Window', and 'Help'. Below the menu is a toolbar with icons for 'New', 'Open', 'Save', 'Connect', 'Disconnect', 'Clear Data', 'Options', 'View Hex', and 'Help'. The main area is a terminal window displaying a list of IR data points, each starting with 'IR:'. At the bottom, there is a status bar showing 'COM16 / 115200 8-N-1' (highlighted with a red box) and 'Connected 00:04:47'. To the right of the status bar are several status indicators: TX, RX, RTS, CTS, DTR, DSR, DCD, and RI, each with a green dot.

```
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:





## 7.16.2: IR Remote Control



### 1. 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:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end

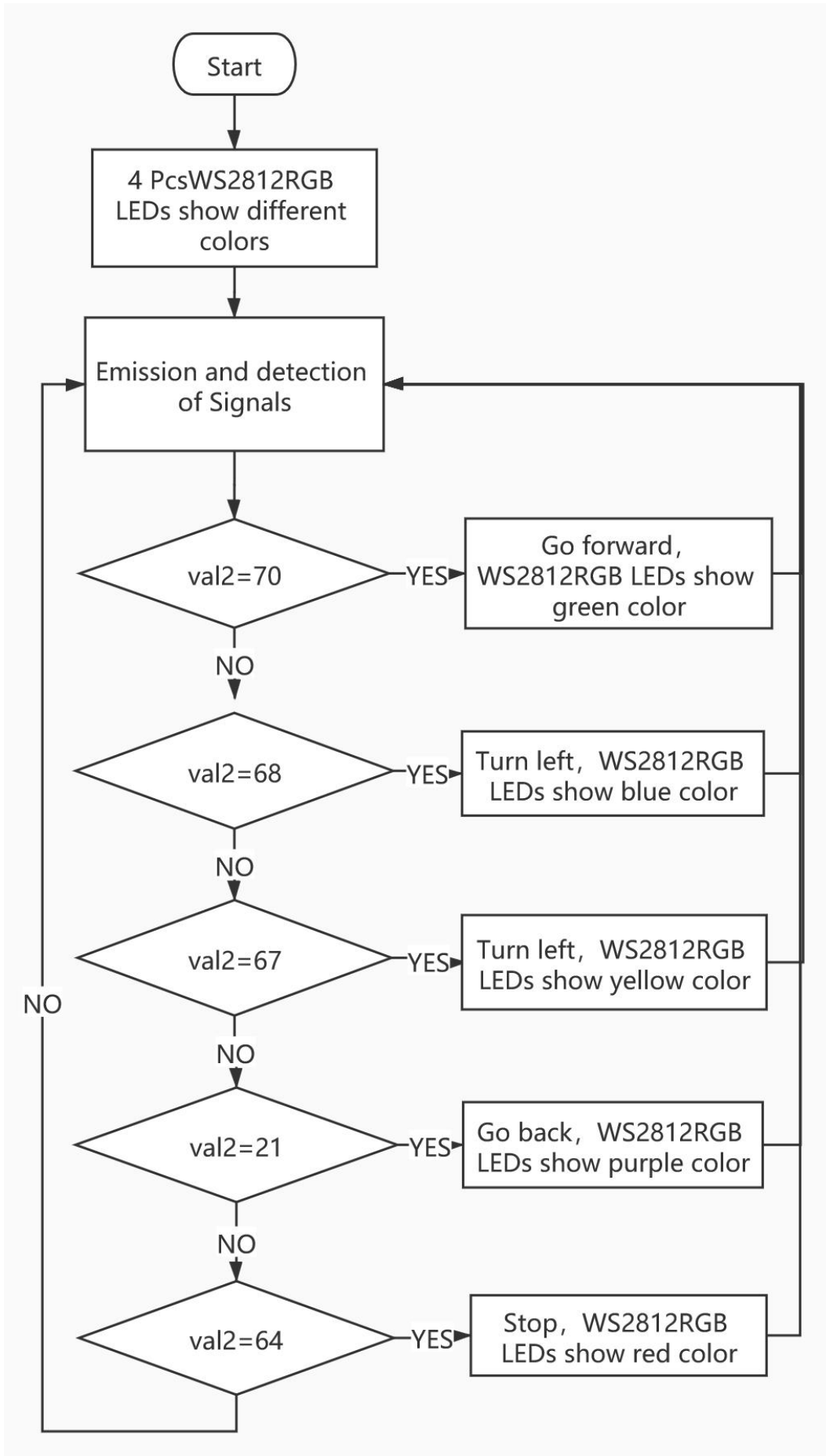




(4) Connect micro:bit to computer by USB cable and open online Makecode editor.

(5) Import Hex profile ([How to import?](#)), or click "New Project" and drag blocks step by step([add turtle-bit extension library first](#))  
[\(How to add turtle-bit extension?\)](#)

### 3. Flow Chart





#### 4. Test Code:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.16: IR Remote Control Smart Car/7.16.2: IR Remote Control	microbit-IR Remote Control .hex

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

(1) Enter "IrRemote" to get block "connect IR receiver at P0"

Put it into "on start" block

IR receiving module is controlled by P11 of micro:bit board, so click P0 to select P11.



(2) 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 of WS2812 RGB is connected to P8 of micro:bit board . So we set to P8.
- d. The robot has 4 pcs WS2812 RGB lights, therefore, change 24 into 4.

```
on start
  connect IR receiver at P11
  set strip to NeoPixel at pin P8 with 4 leds as RGB (GRB format)
```

(3) Go to "Variables" → "Make a Variable..." → "New variable name: " dialog box.

Enter "val" and click "OK" to produce variable "val"

Move "set val to 0" under "set strip...RGB(RGB format)" block

Then create variable "val2" in same way

Drag "set val2 to 0" into "on start"

Edit the code as follows:

```
on start
  connect IR receiver at P16
  set strip to NeoPixel at pin P5 with 18 leds as RGB (GRB format)
  set val to 0
  set val2 to 0
```



(4) Copy "set val2 to 0" once and move it into "forever" block.

Click the triangle button to select "val"

Go to "IrRemote" to drag block "IR button" into 0 box.



\*\*\*\*\*

(5) Enter "Logic" → "if true then" and place it into "forever"

Drag block "=" block into "true" box

Go to "Variables" and move "val" block into left box of "=" .

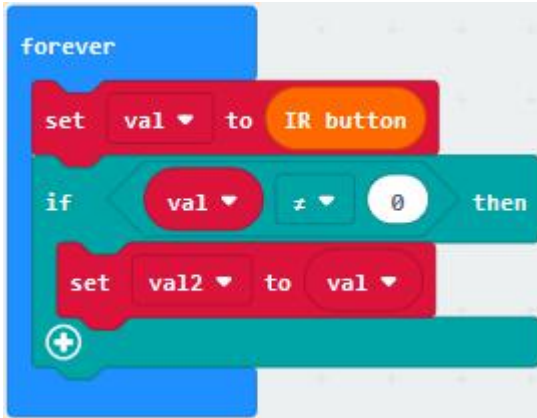
Then click "=" to set "≠"



(6) Duplicate "set val2 to 0" block again and leave it under the block "if...val...then" block.



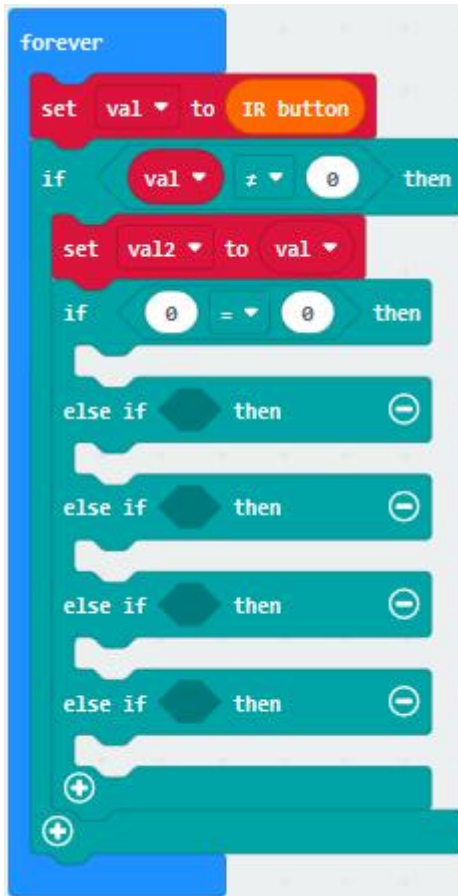
Then drag out variable "val" into 0 box.



(7) Go to "Logic" to drag block "if.true..then...else" under block "set val2 to val" block.

Then tap "+" four times and delete "-" behind "else"

Move "=" block into "true" box.



(8) Go to "Variables" to move block "val2" into left box of "="  
Change 0 into 70.



(9) Enter "TurtleBit" → "car Run\_Forward speed: 0%"

Leave it under the second block "if...val2..then" block and change 0 into 90.

Go to "Neopixel" to move block "strip show color red" under block "car Run\_Forward speed: 90%"

Change red into green.

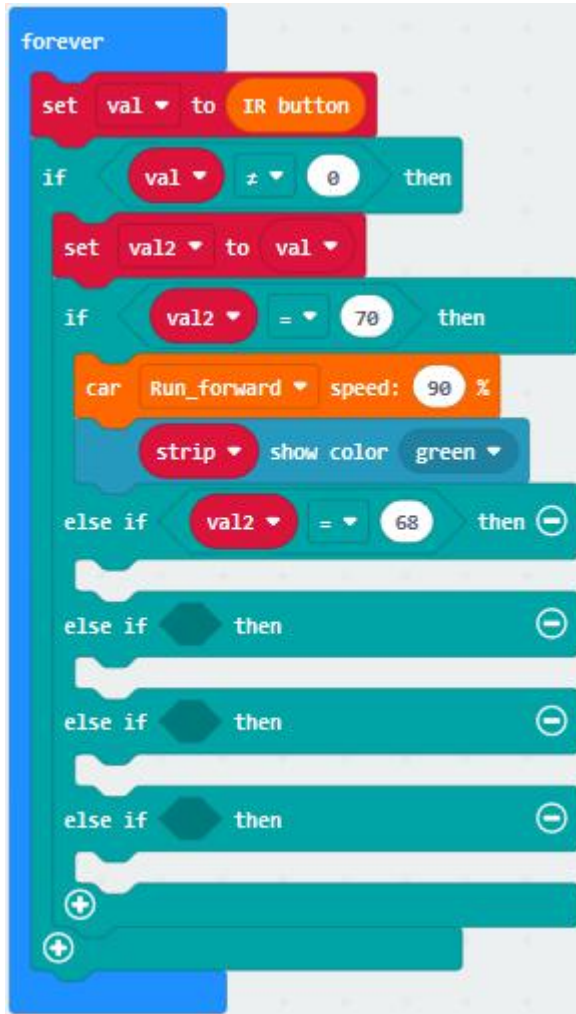




```
forever
  set val to IR button
  if val ≠ 0 then
    set val2 to val
    if val2 = 70 then
      car Run_forward speed: 90 %
      strip show color green
    else if then
    else if then
    else if then
    else if then
```

(10) Replicate code "val2=70" once and leave it into box behind "else if...then" .

Change 70 into 68



(11) Click "TurtleBit" to drag "LeftSide motor run Forward speed: 0%" block under the first "else if ...then" block. And alter 0 into 60  
Duplicate "LeftSide motor run Forward speed: 60%" once and alter LeftSide into RightSide and 60 into 85.  
Drag "strip show color red" block under the block "RightSide motor run Forward speed: 85%"  
Change red into blue.



```
forever
  set val to IR button
  if val ≠ 0 then
    set val2 to val
    if val2 = 70 then
      car Run_forward speed: 90 %
      strip show color green
    else if val2 = 68 then
      LeftSide motor run Forward speed: 60 %
      RightSide motor run Forward speed: 85 %
      strip show color blue
    else if then
    else if then
    else if then
  +
  +
```

(12) Replicate "val2=68" block and code string

```
LeftSide motor run Forward speed: 60 %
RightSide motor run Forward speed: 85 %
strip show color blue
```

Then edit the code string as follows:



```
forever
  set val to IR button
  if val ≠ 0 then
    set val2 to val
    if val2 = 70 then
      car Run_forward speed: 90 %
      strip show color green
    else if val2 = 68 then
      LeftSide motor run Forward speed: 60 %
      RightSide motor run Forward speed: 85 %
      strip show color blue
    else if val2 = 67 then
      LeftSide motor run Forward speed: 85 %
      RightSide motor run Forward speed: 60 %
      strip show color yellow
    else if then
    else if then
  +
  +
```

(13) Copy "val2=67" again and keep it into the third box behind "else if...then" block

Change 67 into 21.

Replicate code string

```
car Run_forward speed: 90 %
strip show color green
```

and place it under the



third “else if...then” block.

Change Run\_Forward into RunBack and green into purple

```
forever
  set val to IR button
  if val ≠ 0 then
    set val2 to val
    if val2 = 70 then
      car Run_forward speed: 90 %
      strip show color green
    else if val2 = 68 then
      LeftSide motor run Forward speed: 60 %
      RightSide motor run Forward speed: 85 %
      strip show color blue
    else if val2 = 67 then
      LeftSide motor run Forward speed: 85 %
      RightSide motor run Forward speed: 60 %
      strip show color yellow
    else if val2 = 21 then
      car Run_back speed: 90 %
      strip show color purple
    else if then
```



(14) Then replicate "val2=21" once and "strip show color purple" block again

Change 21 into 64 and purple into red

Click "TurtleBit" to move out block "car stop"

Edit the code string as follows:



```
forever
  set val to IR button
  if val ≠ 0 then
    set val2 to val
    if val2 = 70 then
      car Run_forward speed: 90 %
      strip show color green
    else if val2 = 68 then
      LeftSide motor run Forward speed: 60 %
      RightSide motor run Forward speed: 85 %
      strip show color blue
    else if val2 = 67 then
      LeftSide motor run Forward speed: 85 %
      RightSide motor run Forward speed: 60 %
      strip show color yellow
    else if val2 = 21 then
      car Run_back speed: 90 %
      strip show color purple
    else if val2 = 64 then
      car stop
      strip show color red
  
```

Complete Code



```
on start
  connect IR receiver at P11
  set strip to NeoPixel at pin P8 with 4 leds as RGB (GRB format)
  set val to 0
  set val2 to 0

forever
  set val to IR button
  if val > 0 then
    set val2 to val
    if val2 = 70 then
      car Run_forward speed: 90 %
      strip show color green
    else if val2 = 68 then
      LeftSide motor run Forward speed: 60 %
      RightSide motor run Forward speed: 85 %
      strip show color blue
```





```
else if (val2 == 67) then
  LeftSide motor run Forward speed: 85 %
  RightSide motor run Forward speed: 60 %
  strip show color yellow
else if (val2 == 21) then
  car Run_back speed: 90 %
  strip show color purple
else if (val2 == 64) then
  car stop
  strip show color red
```



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

Connect IR receiver to P16

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

Set val to 0

Set val2 to 0

The program under the block "forever" runs cyclically.

Set val to IR button

If val≠0, execute the program under then block

Set val2 to val

When val2=70, execute the program under then block

Smart car goes forward at 40% speed

18 pcs RGB lights show green color

When val2=68, execute the program under then block

The left car wheels go forward at 15% speed

The right car wheels go forward at 35% speed

18 pcs RGB lights show blue color on strip

When val2=67, execute the program under then block

The left wheel go forward at 35% speed

The right wheels go forward at 15% speed

18 pcs RGB show yellow color on strip

When val2=21, execute the program under then block

When val2=21

Car goes back at 40% speed

18 pcs RGB lights show purple color on strip

When val2=64, execute the program under then block

Car stops

18 pcs RGB lights show red color on strip

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



```

1  irRemote.connectInfrared(DigitalPin.P11)
2  let strip = neopixel.create(DigitalPin.P8, 4, NeoPixelMode.RGB)
3  let val = 0
4  let val2 = 0
5  basic.forever(function () {
6      val = irRemote.returnIrButton()
7      if (val != 0) {
8          val2 = val
9          if (val2 == 70) {
10             turtleBit.run(DIR.Run_forward, 90)
11             strip.showColor(neopixel.colors(NeoPixelColors.Green))
12         } else if (val2 == 68) {
13             turtleBit.Motor(MOTOR.LeftSide, MD.Forward, 60)
14             turtleBit.Motor(MOTOR.RightSide, MD.Forward, 85)
15             strip.showColor(neopixel.colors(NeoPixelColors.Blue))
16         } else if (val2 == 67) {
17             turtleBit.Motor(MOTOR.LeftSide, MD.Forward, 85)
18             turtleBit.Motor(MOTOR.RightSide, MD.Forward, 60)
19             strip.showColor(neopixel.colors(NeoPixelColors.Yellow))
20         } else if (val2 == 21) {
21             turtleBit.run(DIR.Run_back, 90)
22             strip.showColor(neopixel.colors(NeoPixelColors.Purple))
23         } else if (val2 == 64) {
24             turtleBit.state(MotorState.stop)
25             strip.showColor(neopixel.colors(NeoPixelColors.Red))
26         }
27     }
28 })
29

```

### 5. Test Result:

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

Download code to micro:bit board, and dial POWER to ON end.

Make IR remote control point at micro:bit and press the button to control smart car to move.



button makes smart car move forward,



stands for turning left,



implies rightward turning,



indicates moving backward ,





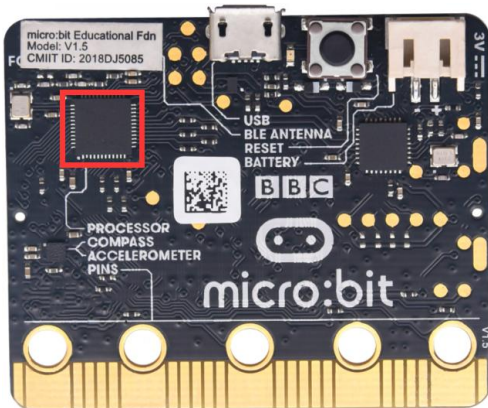
stops car, and 4pcs WS2812RGB light up the corresponding color.

([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.

## 7.17: Bluetooth Multi-purpose Smart Car

### 7.17.1: Read Bluetooth Data



#### 1. Description:

In this lesson, we will control smart car to perform different functions by Bluetooth of micro:bit board. We provide you with an App.

Let' s know its interface and function of every icon first

#### 2. Experimental Preparation:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.

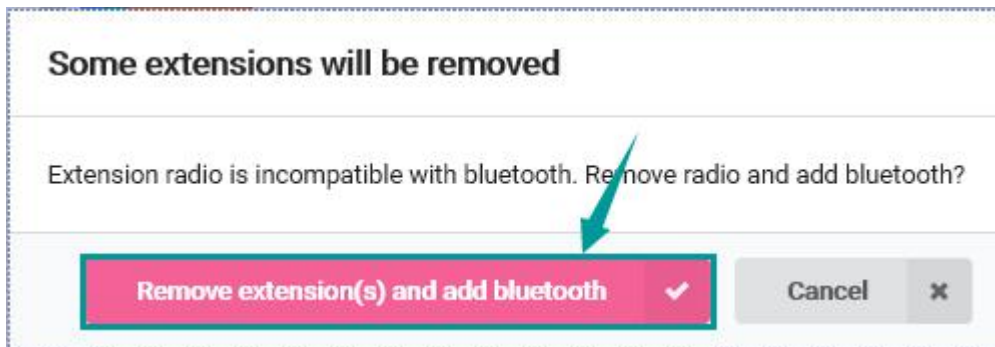


- (3) Dial power switch to ON end
- (4) Connect micro:bit to computer by USB cable and open online Makecode editor.
- (5) Import Hex profile ([How to import?](#)), or click "New Project" and drag blocks step by step(**add turtle-bit extension library first**)

**(How to add turtle-bit 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:**

Type	Route	File Name
Hex	../Makecode Tutorial/Test Code/7.17:	microbit-Read



file	Bluetooth Multi-purpose Smart Car/7.17.1: Read Bluetooth Data	Bluetooth Data.hex
------	---	--------------------

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

(1) Enter "Advanced" → "Serial" → "serial redirect to USB"

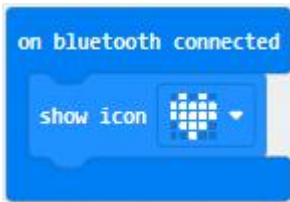
Place it into "on start"



\*\*\*\*\*

(2) Click "Bluetooth" → "on bluetooth connected"

Go to "Basic" to move "show icon" block into "on bluetooth connected" block.

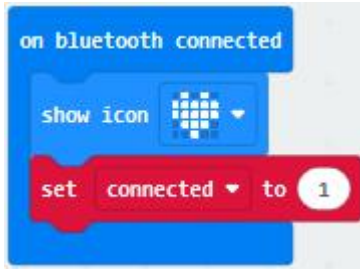


\*\*\*\*\*

(3) Click "Variables" → "Make a Variable..." → "New variable name: " dialog box.

Input "connected" and click "OK" to create variable "connected" .

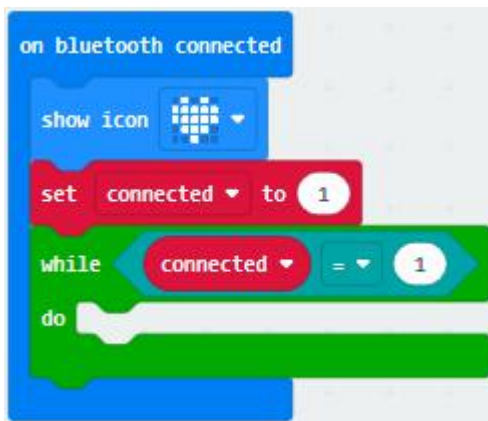
Drag "set connected to 0" under block "show icon" and change 0 into 1.



(4) 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.



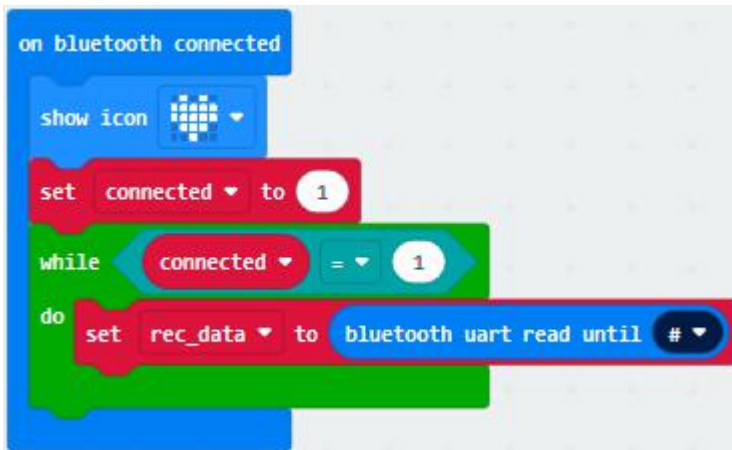
\*\*\*\*\*

(5) 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()"

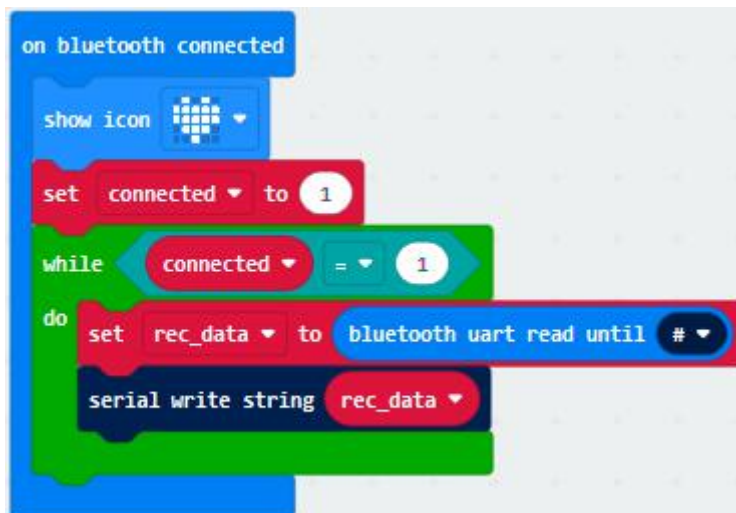
Keep it into 0 box and click triangle button to select #.



(6) Go to "Advanced" → "Serial" → "serial write string"

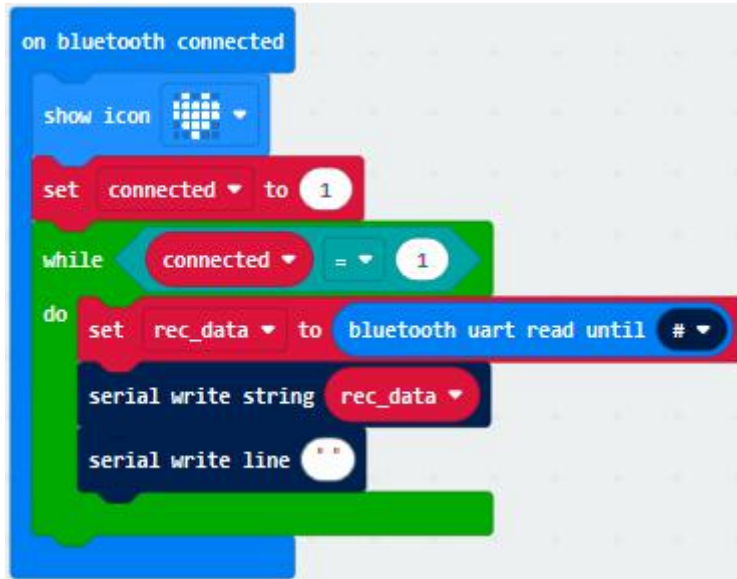
Move it below "set rec\_data...until#" block

And combine variable "rec\_data" with "serial write string" block.



(7) Click "Advanced" → "Serial" → "serial write line" and edit code string as follows:





(8) Click "Bluetooth" to drag out "on bluetooth disconnected" .

(9) Go to "Bluetooth" → "on bluetooth disconnected"

Copy "show icon" block and keep it into block "on bluetooth disconnected"

Click triangle button to select " " pattern.





## Complete Program

```
on start
  serial redirect to USB

on bluetooth connected
  show icon [Bluetooth]
  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 while

on bluetooth disconnected
  show icon [Bluetooth]
```



“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

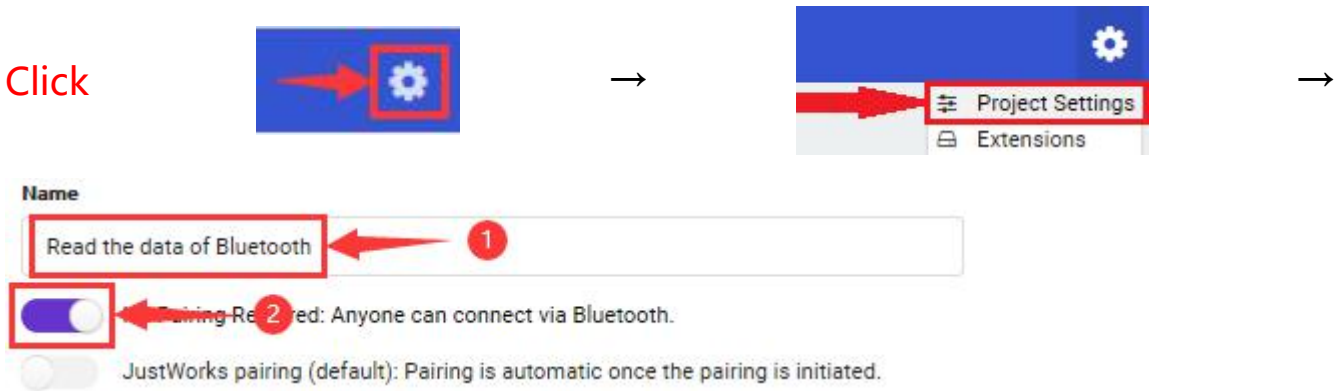
LED dot matrix displays “📶” pattern.

Click “JavaScript” to view the corresponding JavaScript code:



#### 4. Test Result:

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.



## IOS system

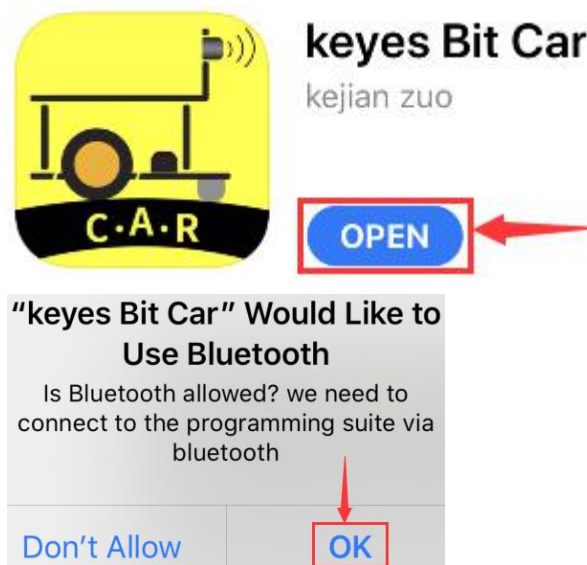
a. Open App Store



b. Search keys Bit Car and click “” icon to download keys Bit Car



c. After the download, tap “OK” when a dialog box appears up.



d. Enable Bluetooth of cellphone or iPad.

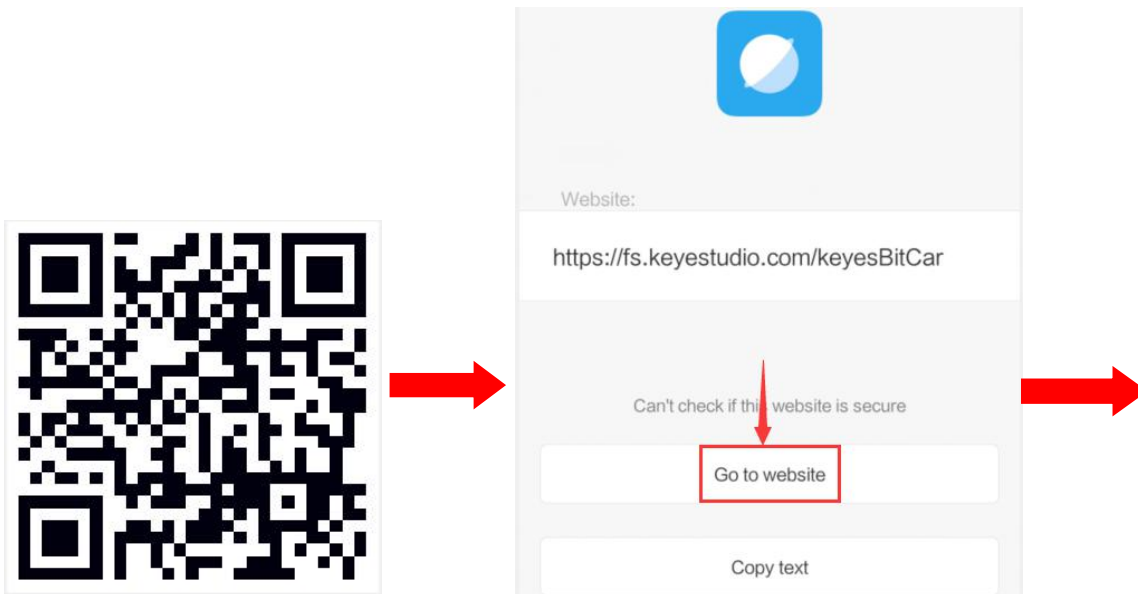


Click "connect" button to search Bluetooth.

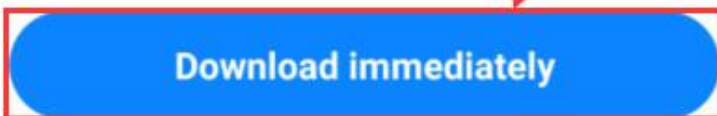
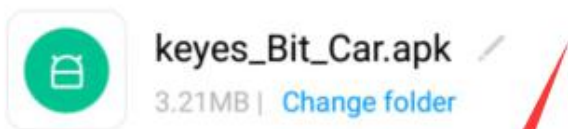
Then select "BCC micro:bit" to connect Bluetooth.

## Android system

Scan the QR code and enter website to download **keyes\_Bit\_Car.apk**

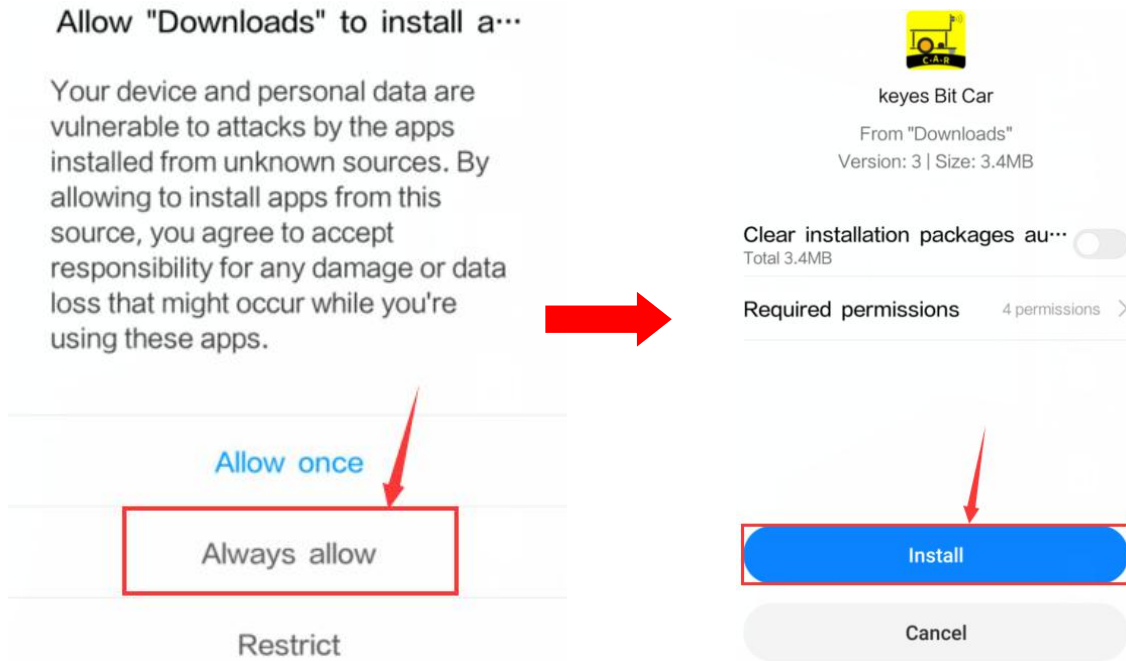


### SaveAPKfile





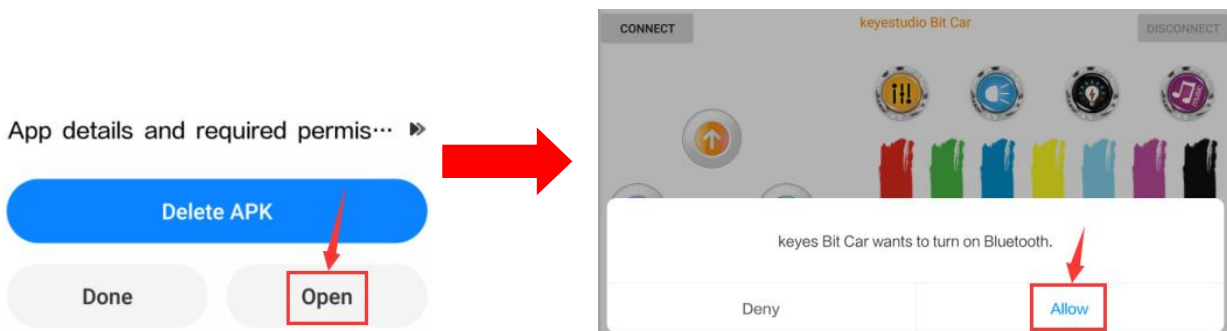
Then click "Always allow" and tap "install"



e. Tap "Open" or click "keyes Bit Car" icon to enter app.

A dialog box appears, then click "Allow" to turn on Bluetooth.

You also enable Bluetooth firstly.



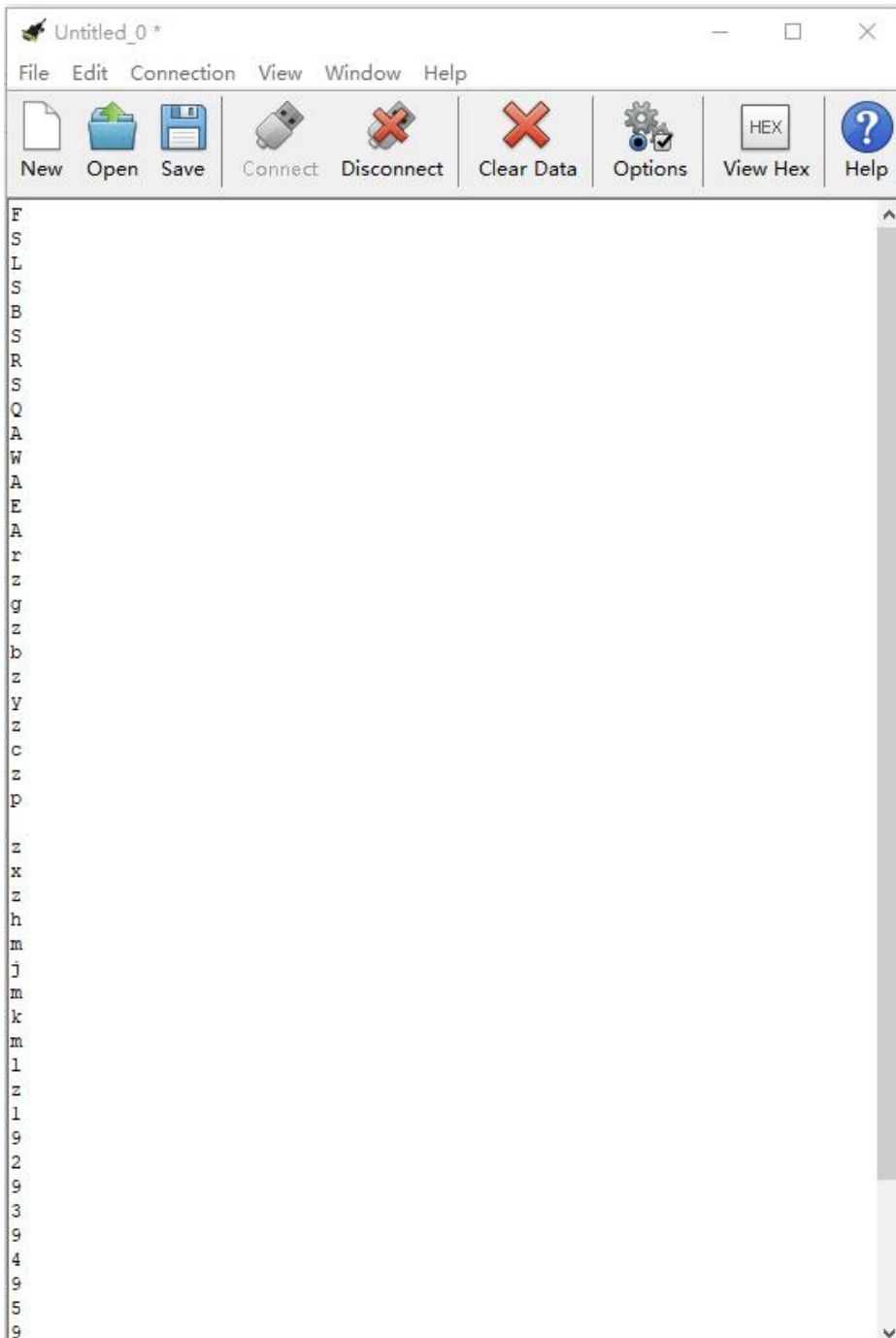
Click "CONNECT" to search and link with Bluetooth.

Open CoolTerm, click Options to select SerialPort. Set COM port and



115200 baud rate. Click "OK" and "Connect" .

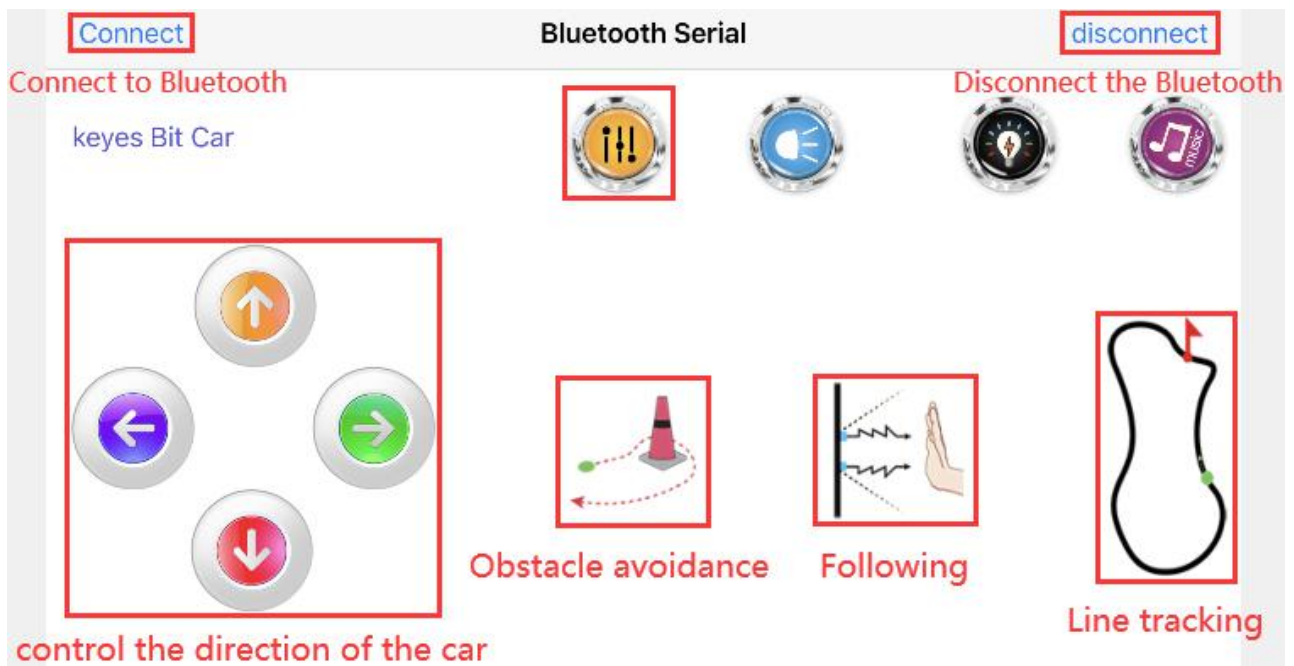
Point at micro:bit board and press the icons on APP, the corresponding characters are shown on CoolTerm monitor.








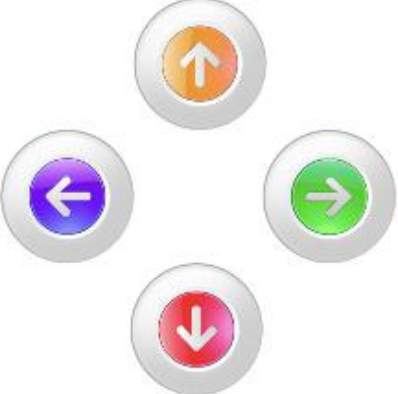
Through the test, we get the function of every icon, as shown below:





Connect Bluetooth Serial disconnect


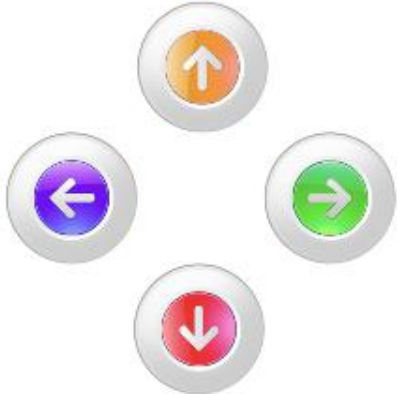

keys Bit Car



Control the two front RGB lights to display distinct color

Connect Bluetooth Serial disconnect

keys Bit Car



control 4 pcs WS2812 RGB to light up



## 7.17.2: Multi-purpose Smart Car



### 1. Description:

In this lesson, we will control the smart car to perform multipurpose function.



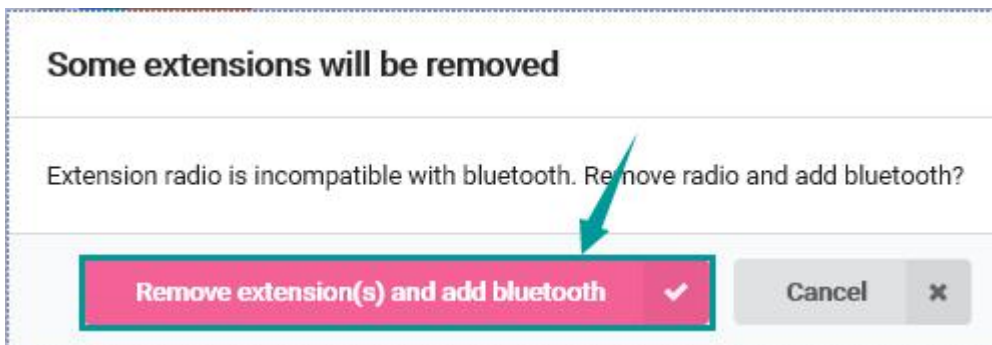
## 2. Experimental Preparation:

- (1) Insert micro:bit board into slot of V2 shield.
- (2) Place batteries into battery holder.
- (3) Dial power switch to ON end
- (4) Connect micro:bit to computer by USB cable and open online Makecode editor.
- (5) Import Hex profile [\(How to import?\)](#) , or click "New Project" and drag blocks step by step(**add turtle-bit extension library first**)

### [\(How to add turtle-bit extension?\)](#)

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:

Type	Route	File Name
Hex file	../Makecode Tutorial/Test Code/7.17 : Bluetooth Multi-purpose Smart Car  /7.17.2: Multi-purpose Smart Car	microbit-Multi-purpose Smart Car.hex

**Complete Code**

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



```
on start
  set strip to NeoPixel at pin PB with 4 leds as RGB (200 forward)
  LED brightness 200

on bluetooth connected
  show icon
  set connected to 1
  while connected == 1
    do
      set rec_data to bluetooth user read until
      call Control_Car
      call Select_Mode
      call control_RGB
      call control_NeoPixel
      call control_music
  on bluetooth disconnected
    show icon
```

```
function Control_Car
  if rec_data == 'P' then
    car Run_forward speed: 100 %
  else if rec_data == 'Q' then
    car Run_back speed: 100 %
  else if rec_data == 'L' then
    car Turn_Left speed: 100 %
  else if rec_data == 'R' then
    car Turn_Right speed: 100 %
  else if rec_data == 'S' then
    car brake
```

```
function Select_Mode
  if rec_data == 'Q' then
    set car_mode to 1
  else if rec_data == 'K' then
    set car_mode to 2
  else if rec_data == 'S' then
    set car_mode to 3
  else if rec_data == 'A' then
    set car_mode to 4
  car brake
```

```
function car_follow
  if distance_val > 9 and distance_val < 20 then
    car Run_forward speed: 80 %
  else if distance_val > 5 and distance_val < 9 then
    car brake
  else if distance_val < 5 then
    car Run_back speed: 80 %
  else if distance_val < 20 then
    car brake
```

```
function car_avoid
  set distance_val to Ultrasonic
  if distance_val < 10 then
    car Turn_Left speed: 80 %
  else
    car Run_forward speed: 80 %
```

```
function control_NeoPixel
  if rec_data == 'h' then
    set Neo_data to 1
  else if rec_data == 'g' then
    set Neo_data to 2
  else if rec_data == 'k' then
    set Neo_data to 3
  else if rec_data == 'i' then
    set Neo_data to 0
    call changeColor
  else if rec_data == 'e' then
    set Neo_data to 4
    strip clear
    strip show
```

```
function car_Tracking
  if LineTracking == 2 or LineTracking == 3 or LineTracking == 5 or LineTracking == 6 or LineTracking == 7 then
    car Run_forward speed: 80 %
  else if LineTracking == 4 then
    LeftSlide motor run Back speed: 80 %
    RightSlide motor run Forward speed: 80 %
  else if LineTracking == 1 then
    LeftSlide motor run Forward speed: 80 %
    RightSlide motor run Back speed: 80 %
  else
    car brake
```



```
function control_RGB
  if rec_data == 'a' then
    set RGBled LeftSide
    R: 255
    G: 0
    B: 0
    set RGBled RightSide
    R: 255
    G: 0
    B: 0
  else if rec_data == '1' then
    set RGBled LeftSide
    R: 0
    G: 255
    B: 0
    set RGBled RightSide
    R: 0
    G: 255
    B: 0
  else if rec_data == 'b' then
    set RGBled LeftSide
    R: 0
    G: 0
    B: 255
    set RGBled RightSide
    R: 0
    G: 0
    B: 255
  end if
end function

function changeColor
  set color_c_flag to color_c_flag + 1
  if color_c_flag == 11 then
    set color_c_flag to 1
  end if
  if color_c_flag == 1 then
    strip >> show color red
  else if color_c_flag == 2 then
    strip >> show color orange
  else if color_c_flag == 3 then
    strip >> show color yellow
  else if color_c_flag == 4 then
    strip >> show color green
  else if color_c_flag == 5 then
    strip >> show color blue
  else if color_c_flag == 6 then
    strip >> show color indigo
  else if color_c_flag == 7 then
    strip >> show color violet
  else if color_c_flag == 8 then
    strip >> show color purple
  else if color_c_flag == 9 then
    strip >> show color white
  else if color_c_flag == 10 then
    strip >> show color black
  end if
end function

function control_music
  if rec_data == '1' then
    ring tone (Hz) Middle C
  else if rec_data == '2' then
    ring tone (Hz) Middle D
  else if rec_data == '3' then
    ring tone (Hz) Middle E
  else if rec_data == '4' then
    ring tone (Hz) Middle F
  else if rec_data == '5' then
    ring tone (Hz) Middle G
  else if rec_data == '6' then
    ring tone (Hz) Middle A
  else if rec_data == '7' then
    ring tone (Hz) Middle B
  else if rec_data == '8' then
    ring tone (Hz) High C
  else if rec_data == '9' then
    rest (ms) 1/8 * beat
  end if
end function
```



```
else if nec_data == 'y' then
  set RGBLed LeftSide
  R: 255
  G: 255
  B: 0
  set RGBLed RightSide
  R: 255
  G: 255
  B: 0
else if nec_data == 'c' then
  set RGBLed LeftSide
  R: 100
  G: 200
  B: 100
  set RGBLed RightSide
  R: 100
  G: 200
  B: 100
else if nec_data == 'p' then
  set RGBLed LeftSide
  R: 255
  G: 100
  B: 255
  set RGBLed RightSide
  R: 255
  G: 100
  B: 255
else if nec_data == 'x' then
  turn off all RGB-led

function nec_leds
  for index from 0 to 4
  do
    strip set pixel color at index to red
    strip show
    wait (us) 100000
  for index2 from 0 to 4
  do
    strip set pixel color at index2 to purple
    strip show
    wait (us) 100000
  for index3 from 0 to 4
  do
    strip set pixel color at index3 to white
    strip show
    wait (us) 100000
```





```
forever
  set distance_val to ultrasonic
  if car_node == 1 then
    call car_void
  else if car_node == 2 then
    call car_follow
  else if car_node == 3 then
    call car_tracking
  +
  if neo_data == 1 then
    strip set pixel color at pick random 0 to 4 to hue red pick random 0 to 255 green pick random 0 to 255 blue pick random 0 to 255 saturation 255 luminosity 50
    strip show
  else if neo_data == 2 then
    for index from 0 to 4
    do
      strip set pixel color at index to red pick random 0 to 255 green pick random 0 to 255 blue pick random 0 to 255
      strip show
      wait (at) 10000
      strip clear
    +
  else if neo_data == 3 then
    call neo_water
  else if neo_data == 4 then
    strip show bar graph of 0 up to 255
  +
```



```
1 function Select_Mode () {
2   if (rec_data == "Q") {
3     car_mode = 1
4   } else if (rec_data == "W") {
5     car_mode = 2
6   } else if (rec_data == "E") {
7     car_mode = 3
8   } else if (rec_data == "A") {
9     car_mode = 0
10    turtleBit.state(MotorState.brake)
11  }
12 }
13 bluetooth.onBluetoothConnected(function () {
14   basic.showIcon(IconNames.Heart)
15   connected = 1
16   while (connected == 1) {
17     rec_data = bluetooth.uartReadUntil(serial.delimiters(Delimiters.Hash))
18     Control_Car()
19     Select_Mode()
20     control_RGB()
21     control_Neopixel()
22     control_music()
23   }
24 })
25 bluetooth.onBluetoothDisconnected(function () {
26   basic.showIcon(IconNames.Sad)
27 })
28 function control_RGB () {
29   if (rec_data == "r") {
30     turtleBit.SetLed(
31       LR.LeftSide,
32       255,
33       0,
34       0
35     )
36     turtleBit.SetLed(
37       LR.RightSide,
38       255,
39       0,
40       0
41     )
42   } else if (rec_data == "g") {
43     turtleBit.SetLed(
44       LR.LeftSide,
```



```
45     0,
46     255,
47     0
48   )
49   turtleBit.SetLed(
50     LR.RightSide,
51     0,
52     255,
53     0
54   )
55 } else if (rec_data == "b") {
56   turtleBit.SetLed(
57     LR.LeftSide,
58     0,
59     0,
60     255
61   )
62   turtleBit.SetLed(
63     LR.RightSide,
64     0,
65     0,
66     255
67   )
68 } else if (rec_data == "y") {
69   turtleBit.SetLed(
70     LR.LeftSide,
71     255,
72     255,
73     0
74   )
75   turtleBit.SetLed(
76     LR.RightSide,
77     255,
78     255,
79     0
80   )
81 } else if (rec_data == "c") {
82   turtleBit.SetLed(
83     LR.LeftSide,
84     100,
85     200,
86     100
87   )
```



```
88     turtleBit.SetLed(  
89     LR.RightSide,  
90     100,  
91     200,  
92     100  
93     )  
94 } else if (rec_data == "p") {  
95     turtleBit.SetLed(  
96     LR.LeftSide,  
97     255,  
98     100,  
99     255  
100    )  
101    turtleBit.SetLed(  
102    LR.RightSide,  
103    255,  
104    100,  
105    255  
106    )  
107 } else if (rec_data == "x") {  
108     turtleBit.OFFLed()  
109 }  
110 }  
111 function car_avoid () {  
112     distance_val = turtleBit.ultra()  
113     if (distance_val <= 10) {  
114         turtleBit.run(DIR.Turn_Left, 80)  
115     } else {  
116         turtleBit.run(DIR.Run_forward, 80)  
117     }  
118 }  
119 function Control_Car () {  
120     if (rec_data == "F") {  
121         turtleBit.run(DIR.Run_forward, 100)  
122     } else if (rec_data == "B") {  
123         turtleBit.run(DIR.Run_back, 100)  
124     } else if (rec_data == "L") {  
125         turtleBit.run(DIR.Turn_Left, 100)  
126     } else if (rec_data == "R") {  
127         turtleBit.run(DIR.Turn_Right, 100)  
128     } else if (rec_data == "S") {  
129         turtleBit.state(MotorState.brake)  
130     }  
131 }
```



```
131 }
132 function control_Neopixel () {
133   if (rec_data == "h") {
134     Neo_data = 1
135   } else if (rec_data == "j") {
136     Neo_data = 2
137   } else if (rec_data == "k") {
138     Neo_data = 3
139   } else if (rec_data == "l") {
140     Neo_data = 0
141     changeColor()
142   } else if (rec_data == "m") {
143     Neo_data = 0
144     strip.clear()
145     strip.show()
146   }
147 }
148 function neo_watar () {
149   for (let index = 0; index <= 4; index++) {
150     strip.setPixelColor(index, neopixel.colors(NeoPixelColors.Red))
151     strip.show()
152     control.waitMicros(100000)
153   }
154   for (let index2 = 0; index2 <= 4; index2++) {
155     strip.setPixelColor(index2, neopixel.colors(NeoPixelColors.Purple))
156     strip.show()
157     control.waitMicros(100000)
158   }
159   for (let index3 = 0; index3 <= 4; index3++) {
160     strip.setPixelColor(index3, neopixel.colors(NeoPixelColors.White))
161     strip.show()
162     control.waitMicros(100000)
163   }
164 }
165 function car_follow () {
166   if (distance_val > 9 && distance_val <= 30) {
167     turtleBit.run(DIR.Run_forward, 80)
168   } else if (distance_val > 6 && distance_val <= 9) {
169     turtleBit.state(MotorState.brake)
170   } else if (distance_val <= 6) {
171     turtleBit.run(DIR.Run_back, 80)
172   } else if (distance_val <= 30) {
173     turtleBit.state(MotorState.brake)
```



```
174     }
175 }
176 function changeColor () {
177     color_c_flag = color_c_flag + 1
178     if (color_c_flag == 11) {
179         color_c_flag = 1
180     }
181     if (color_c_flag == 1) {
182         strip.showColor(neopixel.colors(NeoPixelColors.Red))
183     } else if (color_c_flag == 2) {
184         strip.showColor(neopixel.colors(NeoPixelColors.Orange))
185     } else if (color_c_flag == 3) {
186         strip.showColor(neopixel.colors(NeoPixelColors.Yellow))
187     } else if (color_c_flag == 4) {
188         strip.showColor(neopixel.colors(NeoPixelColors.Green))
189     } else if (color_c_flag == 5) {
190         strip.showColor(neopixel.colors(NeoPixelColors.Blue))
191     } else if (color_c_flag == 6) {
192         strip.showColor(neopixel.colors(NeoPixelColors.Indigo))
193     } else if (color_c_flag == 7) {
194         strip.showColor(neopixel.colors(NeoPixelColors.Violet))
195     } else if (color_c_flag == 8) {
196         strip.showColor(neopixel.colors(NeoPixelColors.Purple))
197     } else if (color_c_flag == 9) {
198         strip.showColor(neopixel.colors(NeoPixelColors.White))
199     } else if (color_c_flag == 10) {
200         strip.showColor(neopixel.colors(NeoPixelColors.Black))
201     }
202 }
203 function control_music () {
204     if (rec_data == "1") {
205         music.ringTone(262)
206     } else if (rec_data == "2") {
207         music.ringTone(294)
208     } else if (rec_data == "3") {
209         music.ringTone(330)
210     } else if (rec_data == "4") {
211         music.ringTone(349)
212     } else if (rec_data == "5") {
213         music.ringTone(392)
214     } else if (rec_data == "6") {
215         music.ringTone(440)
```



```
216 } else if (rec_data == "7") {
217   music.ringTone(494)
218 } else if (rec_data == "8") {
219   music.ringTone(523)
220 } else if (rec_data == "9") {
221   music.rest(music.beat(BeatFraction.Eighth))
222 }
223 }
224 function car_Tracking () {
225   if (turtleBit.LineTracking() == 2 || (turtleBit.LineTracking() == 3 || (turtleBit.LineTracking() == 5 || (turtleBit.LineTracking() == 6 || turtleBit.LineTracking() == 7)))) {
226     turtleBit.run(DIR.Run_forward, 60)
227   } else if (turtleBit.LineTracking() == 4) {
228     turtleBit.Motor(LR.LeftSide, MD.Back, 80)
229     turtleBit.Motor(LR.RightSide, MD.Forward, 80)
230   } else if (turtleBit.LineTracking() == 1) {
231     turtleBit.Motor(LR.LeftSide, MD.Forward, 80)
232     turtleBit.Motor(LR.RightSide, MD.Back, 80)
233   } else {
234     turtleBit.state(MotorState.brake)
235   }
236 }
237 let color_c_flag = 0
238 let Neo_data = 0
239 let distance_val = 0
240 let connected = 0
241 let car_mode = 0
242 let rec_data = ""
243 let strip: neopixel.Strip = null
244 strip = neopixel.create(DigitalPin.P8, 4, NeoPixelMode.RGB)
245 turtleBit.LED_brightness(200)
246 basic.forever(function () {
247   distance_val = turtleBit.ultra()
248   if (car_mode == 1) {
249     car_avoid()
250   } else if (car_mode == 2) {
251     car_follow()
252   } else if (car_mode == 3) {
253     car_Tracking()
254   }
255   if (Neo_data == 1) {
256     strip.setPixelColor(randint(0, 4), neopixel.hsl(neopixel.rgb(randint(0, 255), randint(0, 255), randint(0, 255))), 255, 50)
257     strip.show()
258   } else if (Neo_data == 2) {
```

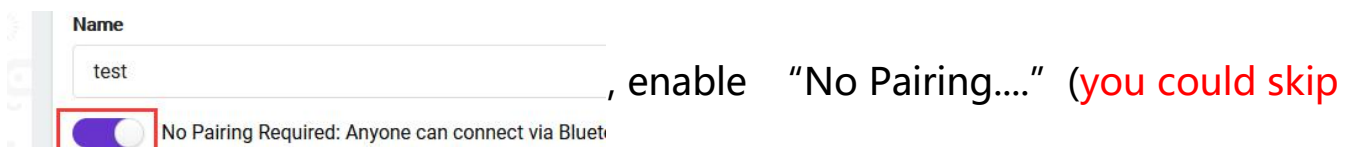
```
259     for (let index4 = 0; index4 <= 4; index4++) {
260       strip.setPixelColor(index4, neopixel.rgb(randint(0, 255), randint(0, 255), randint(0, 255)))
261       strip.show()
262       control.waitMicros(100000)
263       strip.clear()
264     }
265   } else if (Neo_data == 3) {
266     neo_watar()
267   } else if (Neo_data == 4) {
268     strip.showBarGraph(0, 255)
269   }
270 })
271
```



## 4. Test Result:

We will control micro:bit smart car to move via app.

Enter [Makecode online editor](#) → [Projecting Settings](#) →



**this step if you import test code directly)**

Download code and turn on the switch at the back of micro:bit car. Then control smart car via “keyes Bit Car” app

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

## 8. Resource

<https://fs.keyestudio.com/KS4014>

Wiki page: [https://wiki.keyestudio.com/Main\\_Page](https://wiki.keyestudio.com/Main_Page)

Official website: <https://keyestudio.com/>



