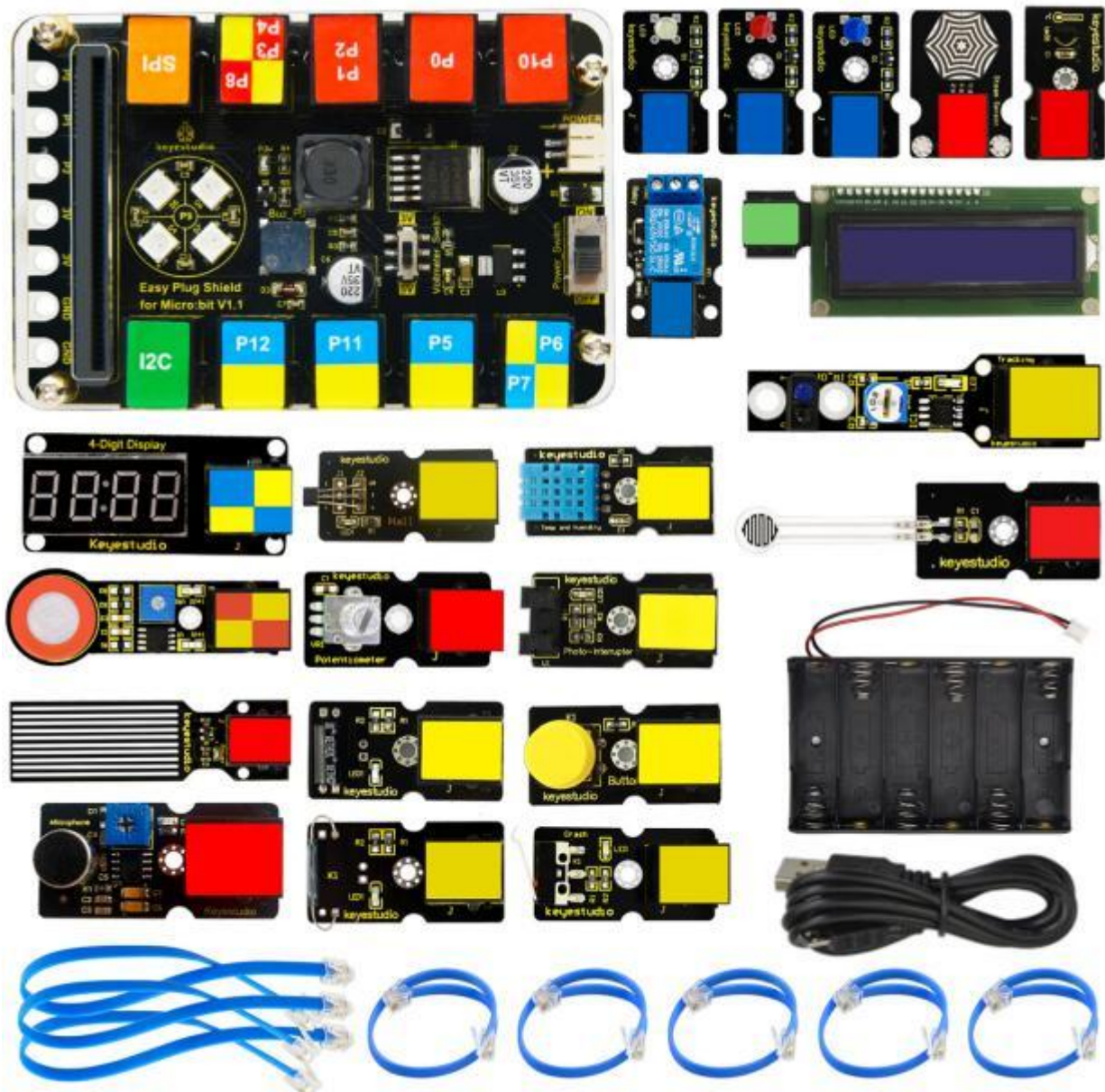


keystudio

EASY PLUG Super Starter Kit for BBC Micro:bit STEM EDU



Contents

1. Description.....	4
2. Kit List.....	4
3. Introduction to Micro:bit V2.0.....	8
(1) What is Micro:bit?.....	8
(2) Comparison between V2.0 & V1.5.....	9
(3) Pinout.....	13
(4) Notes for the application of Micro:bit main board V2.0.....	15
4. Install Micro:bit driver.....	16
5. Getting Started with Micro:bit.....	16
5.1 Write code and program.....	16
5.2 Makecode:	24
5.3. Quick Download.....	25
5.4 Resources and test code.....	29
5.5 Input test code.....	30
5.6 CoolTerm Installation.....	34
6. Projects.....	37
Project 1: Heartbeat.....	38
Project 2: Light Up A Single LED.....	41
Project 3: LED Dot Matrix.....	46

Project 4: Programmable Buttons.....	59
Project 5: Temperature Detection.....	70
Project 6: Geomagnetic Sensor.....	82
Project 7: Accelerometer.....	99
Project 8: Light Detection.....	115
Project 9: Speaker.....	121
Project 10: Touch-sensitive Logo.....	125
Project 11: Microphone.....	130
Project 12: Bluetooth Data Reading.....	142
Project 13: LED Blink.....	162
Project 14: Breathing Light.....	167
Project 15: Blink and Breath.....	171
Project 16: Play Music.....	175
Project 17: RGB.....	181
Project 18: Button Control.....	195
Project 19: Tilt Control.....	200
Project 20: Relay Module.....	204
Project 21: Crash Sensor.....	208
Project 22: Follow Black Line.....	212
Project 23: Magnetic Detection.....	215
Project 24: 4-digit LED Display.....	219
Project 25: Light Interrupter.....	225
Project 26: EASY Plug Reed Switch Module.....	229

Project 27: Hear Footstep.....	234
Project 28: Rotary Potentiometer.....	238
Project 29: Alcohol Content in the Air.....	244
Project 30: Ambient Temperature Detection.....	249
Project 31: Water Level Alarm.....	253
Project 32: 1602 LCD Display.....	259
Project 33: Vapor in the Air.....	264
Project 34: Pressure Detection.....	272
Project 35: Make A Thermo-hygrometer.....	277
7.Resources.....	285

1. Description



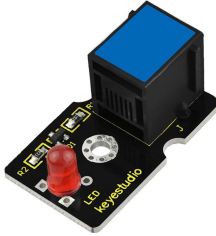
Micro:bit is significantly applied to STEM education for teenagers, as a small microcontroller, which features small in size, easy to carry, and powerful function. At present, innovative technology products, like robots, wearable devices and interactive electronic games can be produced by programming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

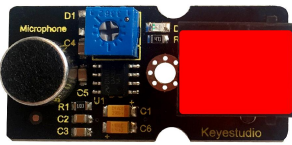

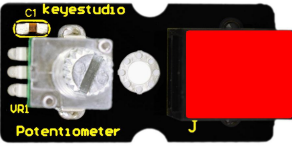

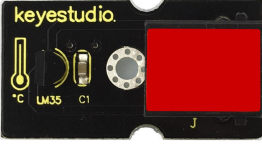
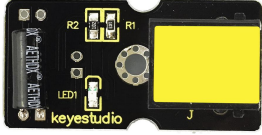
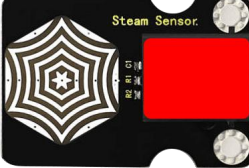

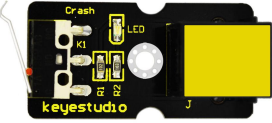
MakeCode is a framework for creating interactive and engaging programming experiences for those new to the world of programming. The platform provides the foundation for a tailored coding experience to create and run user programs on actual hardware or in a simulated target.

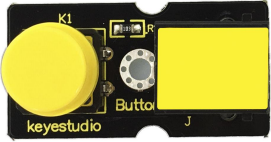
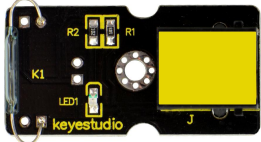

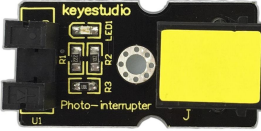
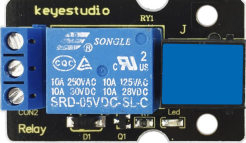
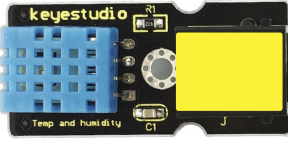



To make you deeply know the micro:bit, we also provide test code and projects.





This super starter kit incorporates different sensors and modules such as passive buzzer, 1602 LCD module, RGB, crash sensor and so on. The detailed projects, from simple to difficult will spur your inspiration and bring in the magical programming world.

2. Kit List

#	Component	Quantity	Picture
0	Micro:bit board is Not Included in KS4020 Kit		
0	Micro:bit board is Included in KS4021 Kit	1	
1	EASY Plug Shield for Micro bit V1.1	1	
2	EASY Plug White LED Module	1	
3	EASYP Plug Blue LED Module	1	
4	EASY Plug Red LED Module	1	
5	EASY Plug thin-film Pressure Sensor	1	

6	EASY Plug Analog Sound Sensor	1	
7	EASY plug Water Level Sensor	1	
8	EASY plug Potentiometer Sensor	1	
9	EASY Plug Analog Alcohol Sensor	1	
10	EASY Plug LM35 Temperature Sensor Module	1	
11	EASY Plug Digital Tilt Sensor Module	1	
12	EASY plug Steam Sensor	1	
13	EASY Plug Hall Magnetic Sensor	1	
14	EASY Plug Crash Sensor	1	

15	EASY Plug Digital Push Button	1	
16	EASY Plug Reed Switch Module	1	
17	EASY Plug Line Tracking Sensor	1	
18	EASY Plug Photo Interrupter Module	1	
19	EASY Plug Single Relay Module	1	
20	EASY plug DHT11 Temperature and Humidity Sensor	1	
21	EASY Plug 1602 LCD Module	1	
22	EASY Plug 4-digit LED Display	1	
23	200mm Blue RJ11 Cable	5	

24	300mm Blue RJ11 Cable	3	
25	Micro USB cable	1	
26	6-Slot AA Battery Holder	1	
27	1.5V AA Battery(not included)	6	

3. Introduction to Micro:bit V2.0

(1) What is Micro:bit?

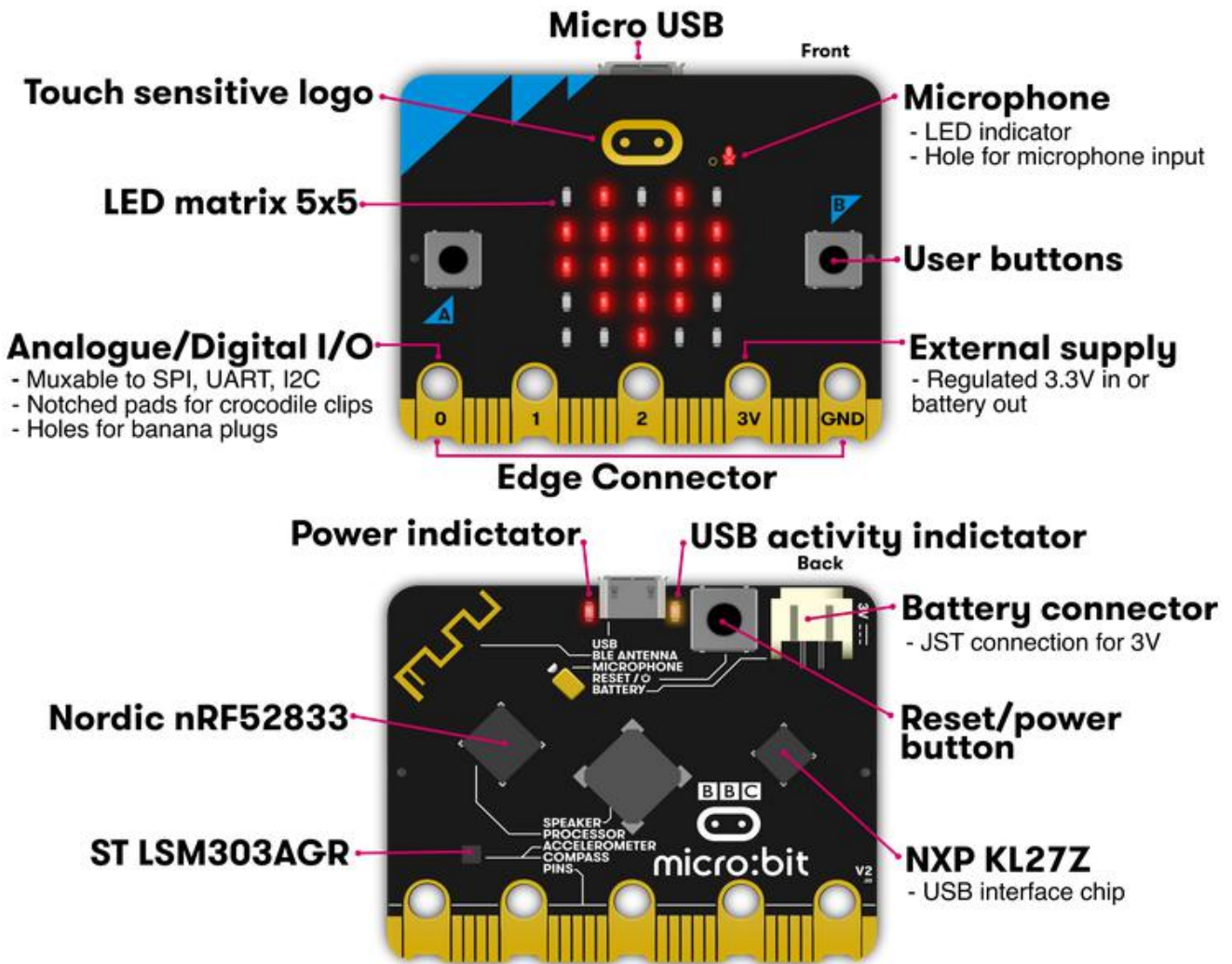
Designed by BBC, Micro:bit main board aims to help children aged above 10 years old to have a better learning of programming.

It is equipped with loads of components, including a 5*5 LED dot matrix, 2 programmable buttons, a compass, a Micro USB interface and a Bluetooth module and others. Though it is just the size of a credit card, it boasts multiple functions. To name just a few, it can be applied in programming video games, making interactions between light and sound, controlling a robot, conducting scientific experiments,

developing wearable devices and make some cool inventions like robots and musical instruments, basically everything imaginable.

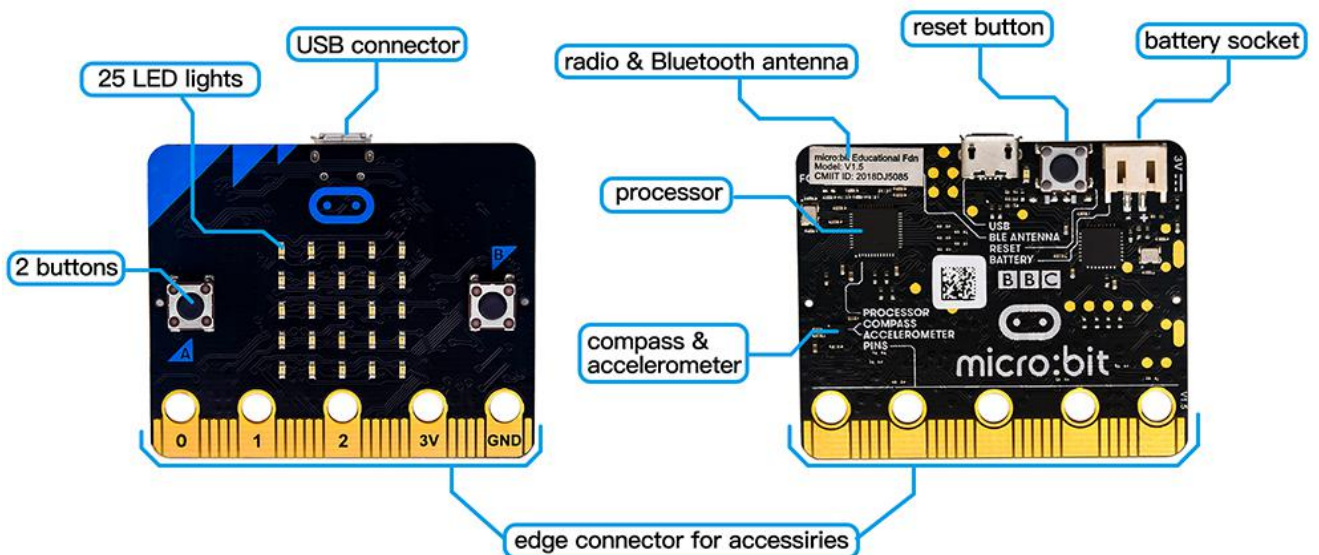
The latest version, that's version 2.0, of Micro:bit main board has a touch-sensitive logo and a MEMS microphone. And there is a buzzer built in the other side of the board which makes playing all kinds of sound possible without any external equipment. The golden fingers and gears added provide a better fixing of crocodile clips. Moreover, this board has a sleeping mode to lower the power consumption of battery and it can be entered if users long press the Reset & Power button on the back of it. More importantly, the CPU capacity of this version is much better than that of the V1.5 and the V2 has more RAM. In final analysis, the Micro:bit main board V2 can allow customers to explore more functions so as to make more innovative products.

(2) Comparison between V2.0 & V1.5



Micro:bit main board

V2.0



Micro:bit main board V1.5

More details:



	V1.5	V2
PROCESSOR	Nordic Semiconductor nRF51822	Nordic Semiconductor nRF52833
MEMORY	256KB Flash, 16KB RAM	512KB Flash, 128KB RAM
INTERFACECHIP	NXP KL26Z, 16KB RAM	NXP KL27Z, 32KB RAM
MICROPHONE	N/A	MEMS microphone and LED indicator
SPEAKER	N/A	On board speaker
TOUCH	N/A	Touch sensitive logo
EDGE CONNECTOR	25pins,PWM,I2C,SPI and Extension interface. 3 ring pins for connectin crocodile clips/banana plugs.	
	3 dedicated GPIO	4 dedicated GPIO Notched for easier connection
I2C	Shared (mux) I2C bus	Dedicated I2C bus
WIRELESS	2.4GHz Radio/BLE Bluetooth 4.0	2.4GHz Radio/BLE Bluetooth 5.0
POWER	Micro USB 5V power supply, 3V port or battery power supply	Micro USB 5V power supply, 3V port or battery power supply LED Indicator, Power off (push and hold power button)
CURRENT AVAILABLE	90mA	200mA
MOTION SENSOR	ST LSM 303	
PROGRAMMING SOFTWARE	C++, Makecode, Python, Scratch	
SIZE	5cm(W) x 4cm(H)	

For the Micro: Bit main board V2, pressing the Reset & Power button , it will reset the Micro: Bit and rerun the program.If you hold it tight, the red LED will slowly get darker.When the power indicator becomes

darker, releasing the button and your Micro: Bit board will enter sleep mode for power saving .This will make your battery more durable. And you could press this button again to 'wake up' your Micro:bit.

For more information,please resort to following links:

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

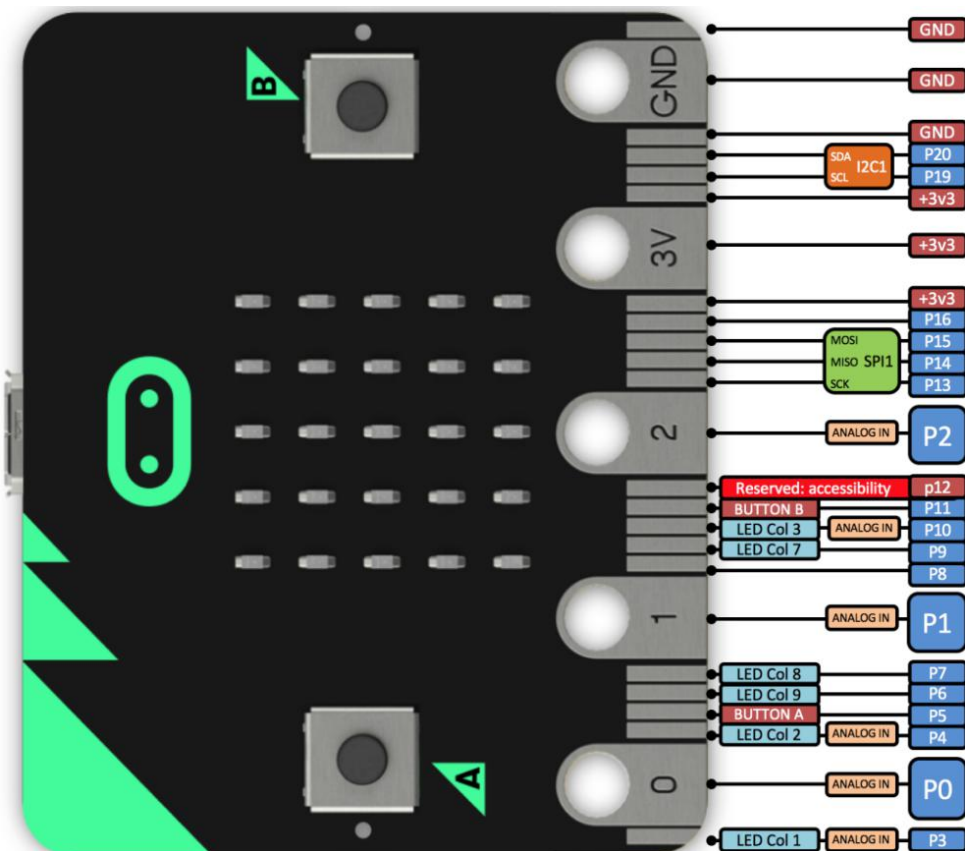
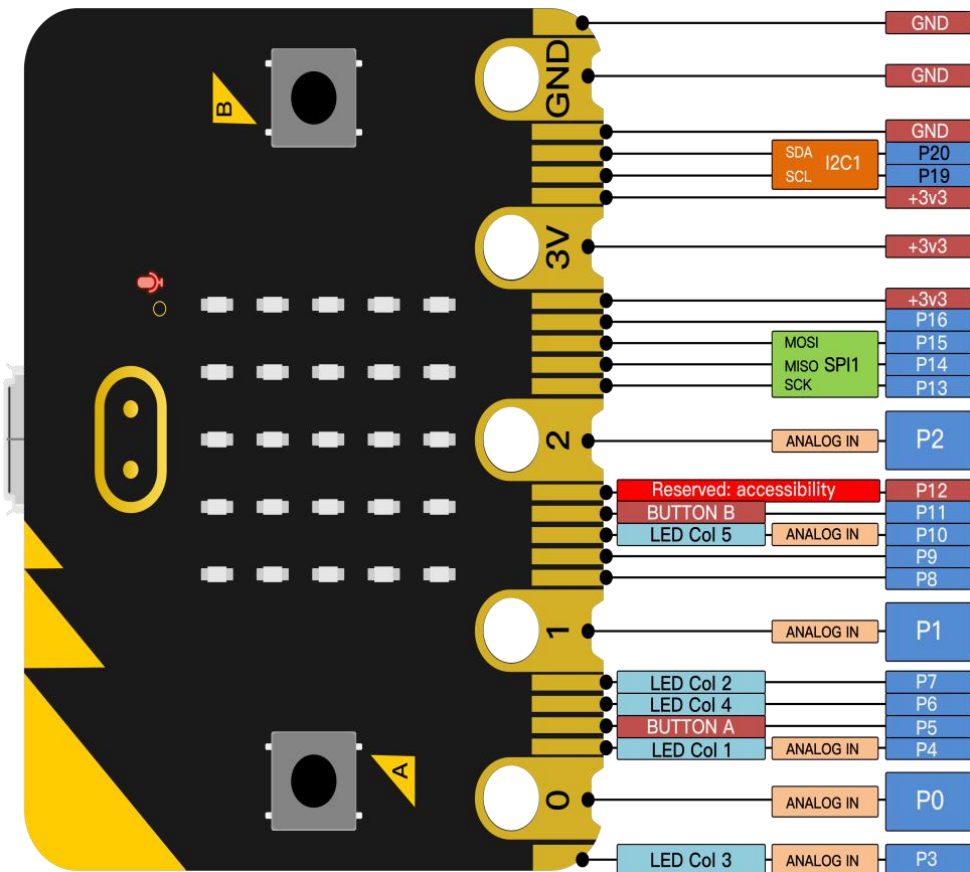
<https://microbit.org/new-microbit/>

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

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

(3) Pinout

Micro:bit main board V2.0 VS V1.5



Browse the official website for more details:

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

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

(4) Notes for the application of Micro:bit main board V2.0

- a. it is recommended to cover it with a silicone protector to prevent short circuit for it has a lot of sophisticated electronic components.
- b. its IO port is very weak in driving since it can merely handle current less than 300mA. Therefore, do not connect it with devices operating in large current, such as servo MG995 and DC motor or it will get burnt. Furthermore, you must figure out the current requirements of the devices before you use them and it is generally recommended to use the board together with a Micro:bit shield.
- c. It is recommended to power the main board via the USB interface or via the battery of 3V. The IO port of this board is 3V, so it does not support sensors of 5V. If you need to connect sensors of 5 V, a Micro:Bit expansion board is required.
- d. When using pins (P3, P4, P6, P7, P10) shared with the LED dot matrix, blocking them from the matrix or the LEDs may display randomly and the data about sensors maybe wrong.
- e. The battery port of 3V cannot be connected with battery more than 3.3V or the main board will be damaged.
- f. Forbid to use it on metal products to avoid short circuit.

To put it simple, Micro:bit V2 main board is like a micro computer which has made programming at our fingertips and enhanced digital innovation. And about programming environment, BBC provides a website: <https://microbit.org/code/>, which has a graphical MakeCode program easy for use.

4. Install Micro:bit driver

If you have downloaded micro:bit driver, then no need to download it again.

If it is you first time to use micro:bit main board, then you will have to download the driver.

First of all, connect the micro:bit to your computer using a USB cable. And enter the link <https://fs.keyestudio.com/KS4021-KS4021>

to download the driver file of micro:bit,

mbed_usb_202
0_x64_1212.exe ·

5. Getting Started with Micro:bit

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

5.1 Write code and program

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

You are recommended to browse the official website of Micro:bit for

more details, and the link is attached below:

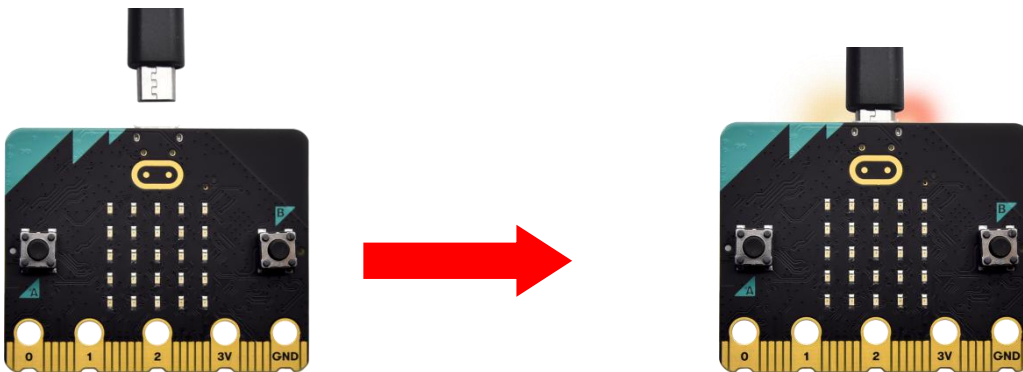
<https://microbit.org/guide/quick/>

Step 1: connect the Micro: Bit main board V2 with your computer

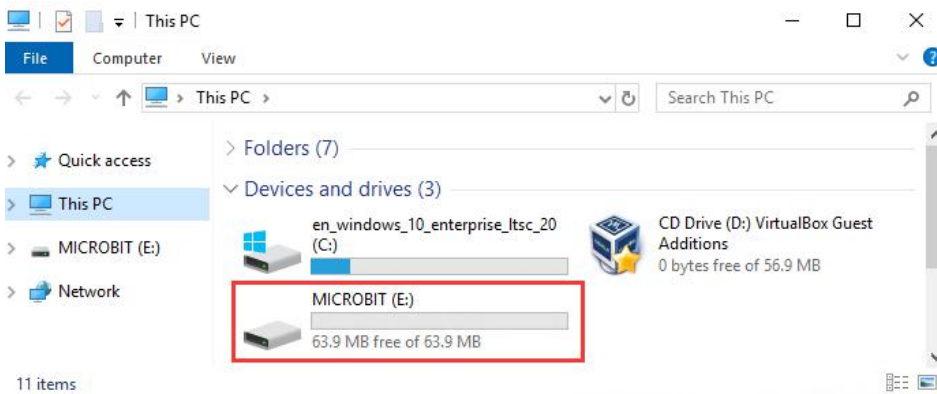
Firstly, link the Micro: Bit main board V2 with your computer via the USB cable. Macs, PCs, Chromebooks and Linux (including Raspberry Pi) systems are all compatible with the Micro: Bit main board V2.

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

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



Secondly, if the red LED on the back of the board is on, that means the board is powered. Then Micro: Bit main board V2 will appear on your computer as a driver named 'MICROBIT'. Please note that it is not an ordinary USB disk as shown below.



Step 2: writing programs

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

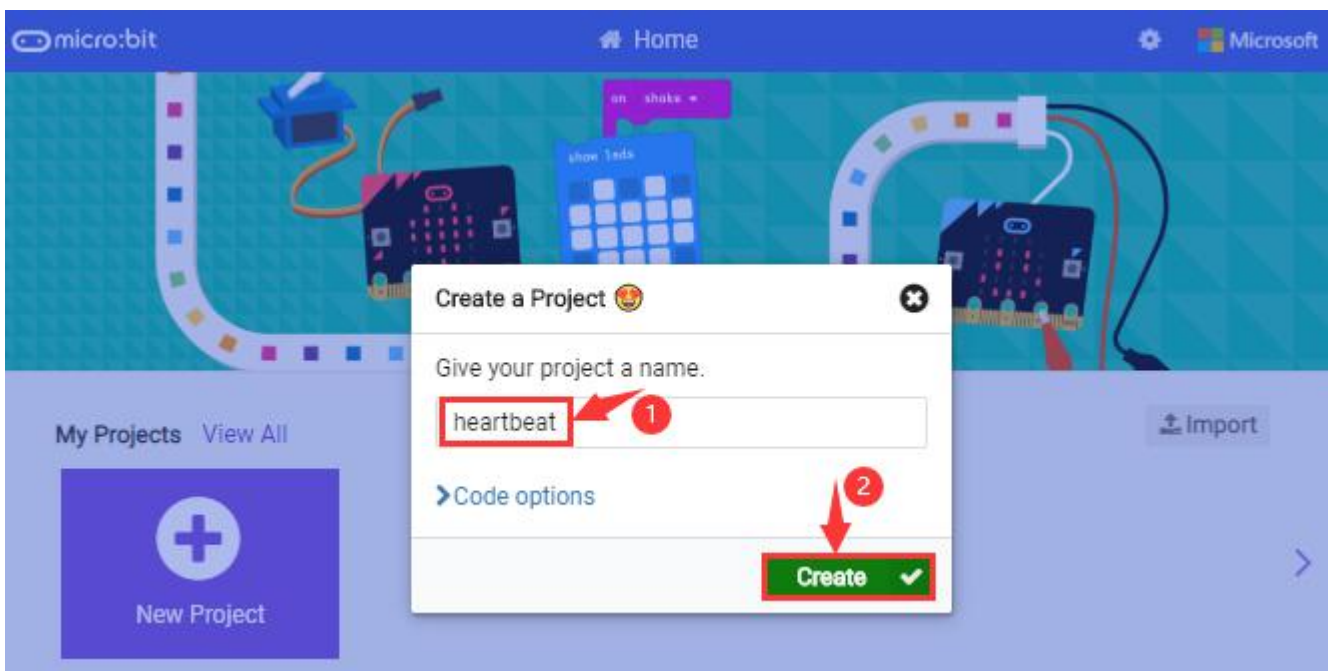
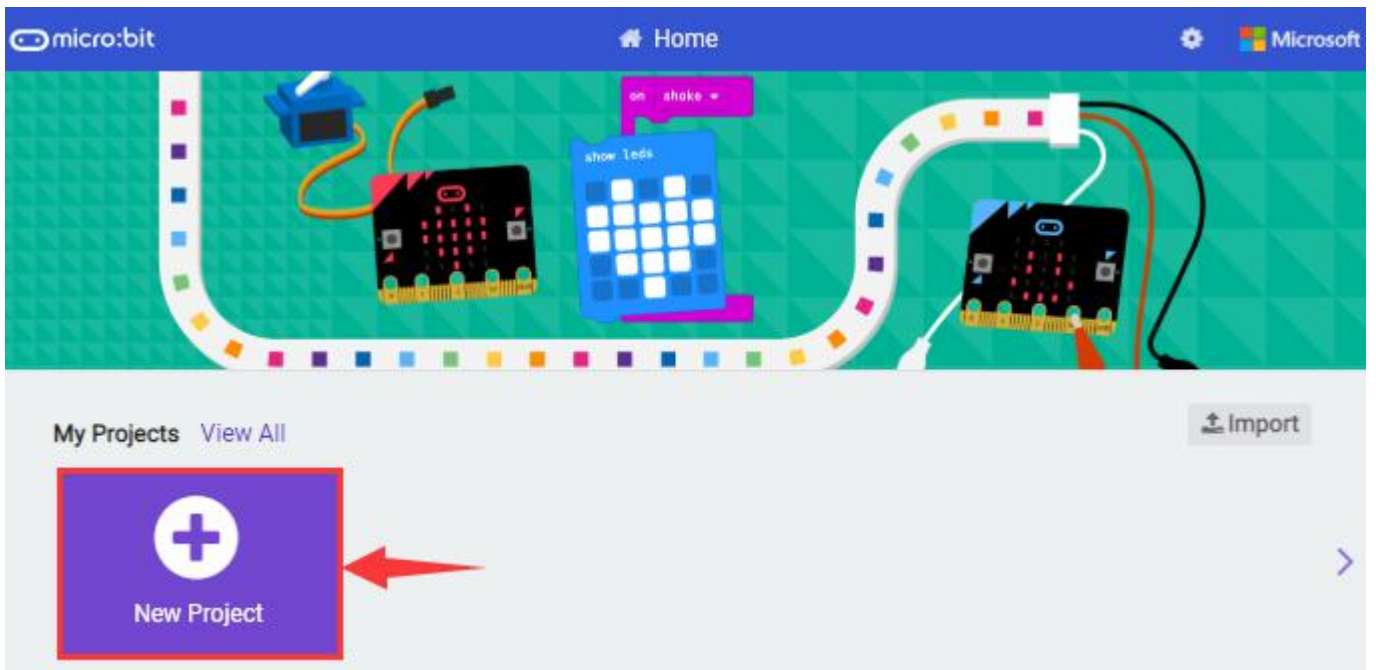
Click 'New Project' ;

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

(If you are running Windows 10 system, it is also viable to edit on the APP MakeCode for micro:bit , which is exactly like editing in the website.

And the link to the APP is

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

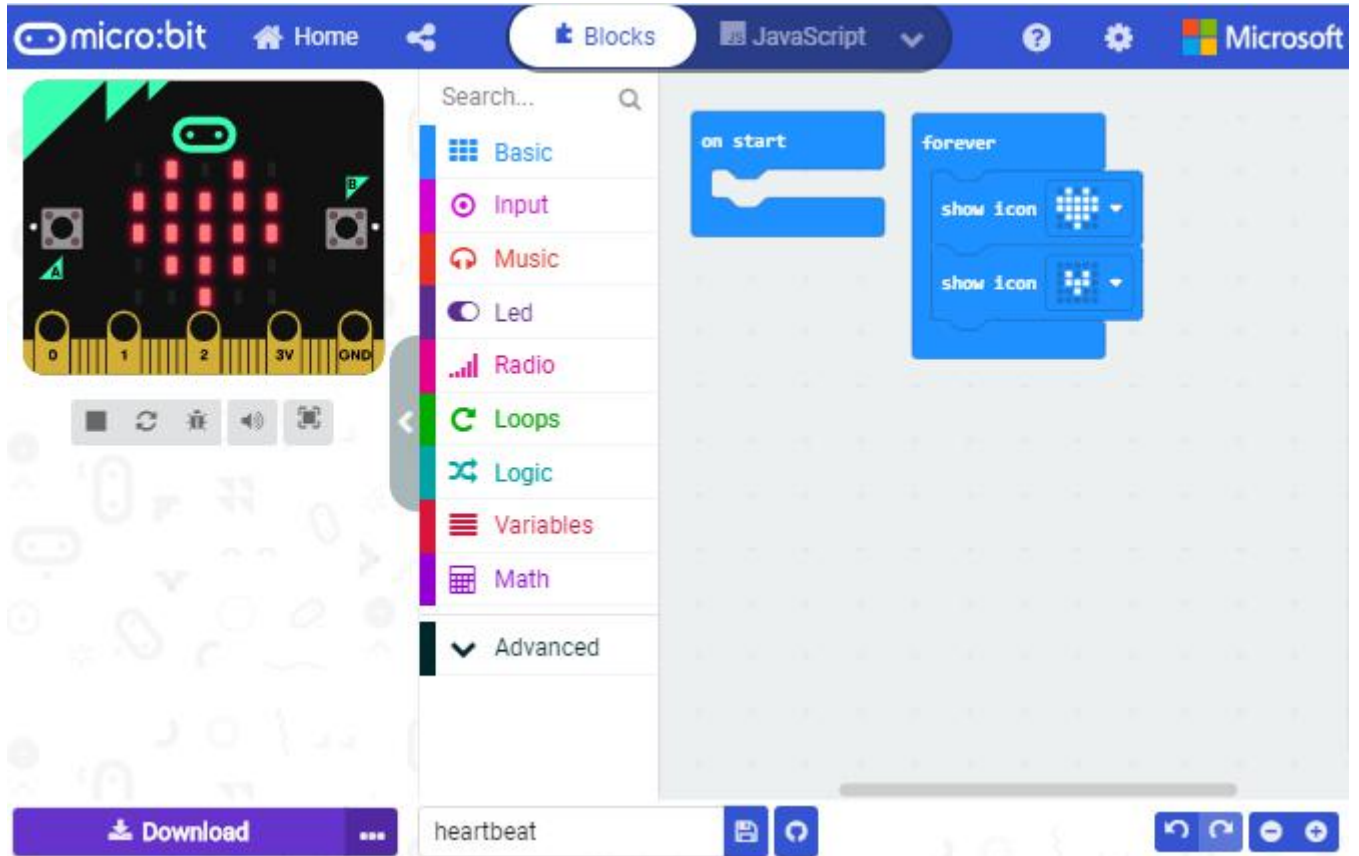


Write a set of micro:bit code. You can drag some modules in the Blocks to the editing area and then run your program in Simulator of MakeCode editor as shown in the picture below which demonstrates

how to edit 'heartbeat' program .

As for loading test code , please turn to Chapter 5.5.

And introduction of Makecode is on the next chapter 5.2.

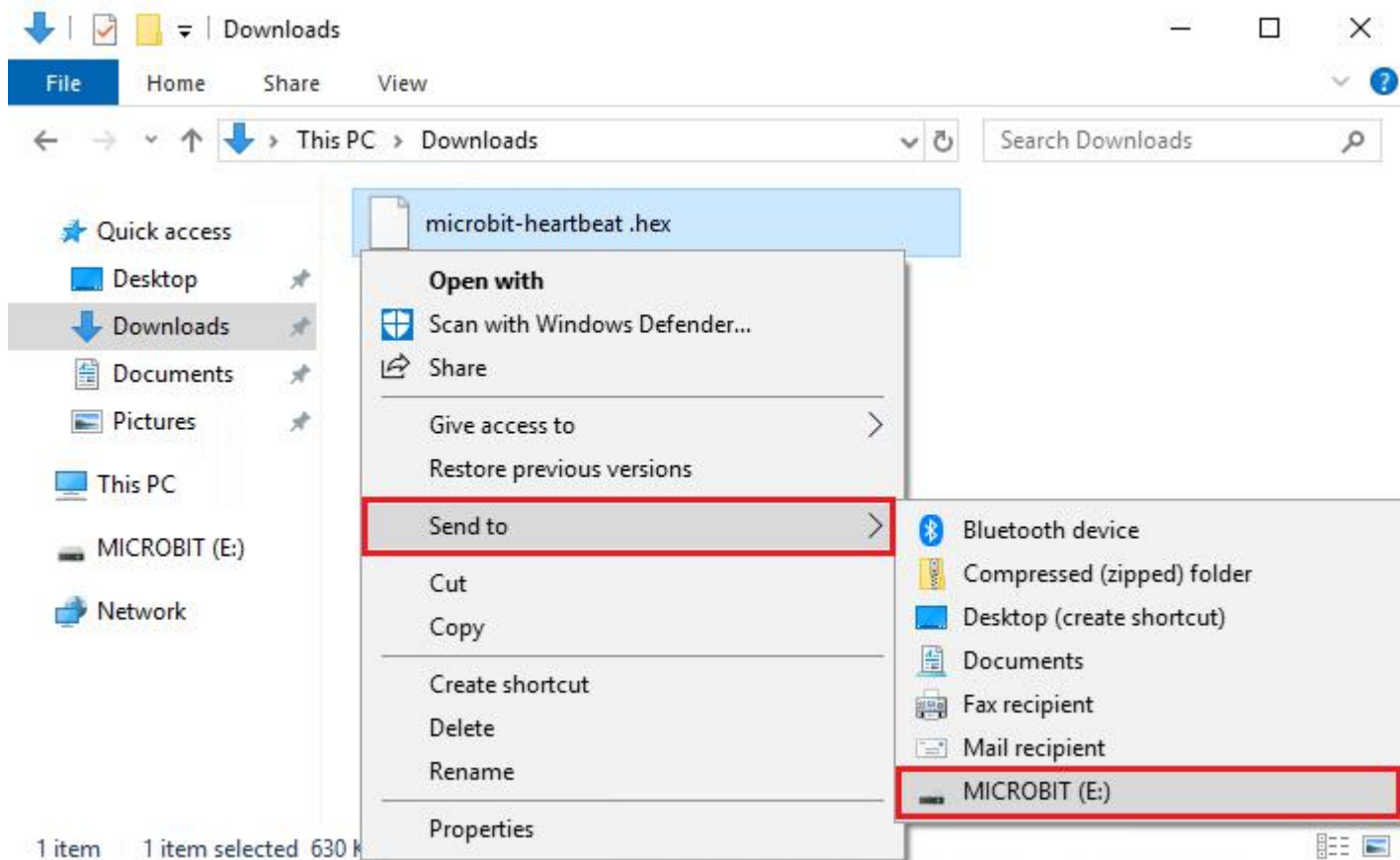


Step 3: download test code

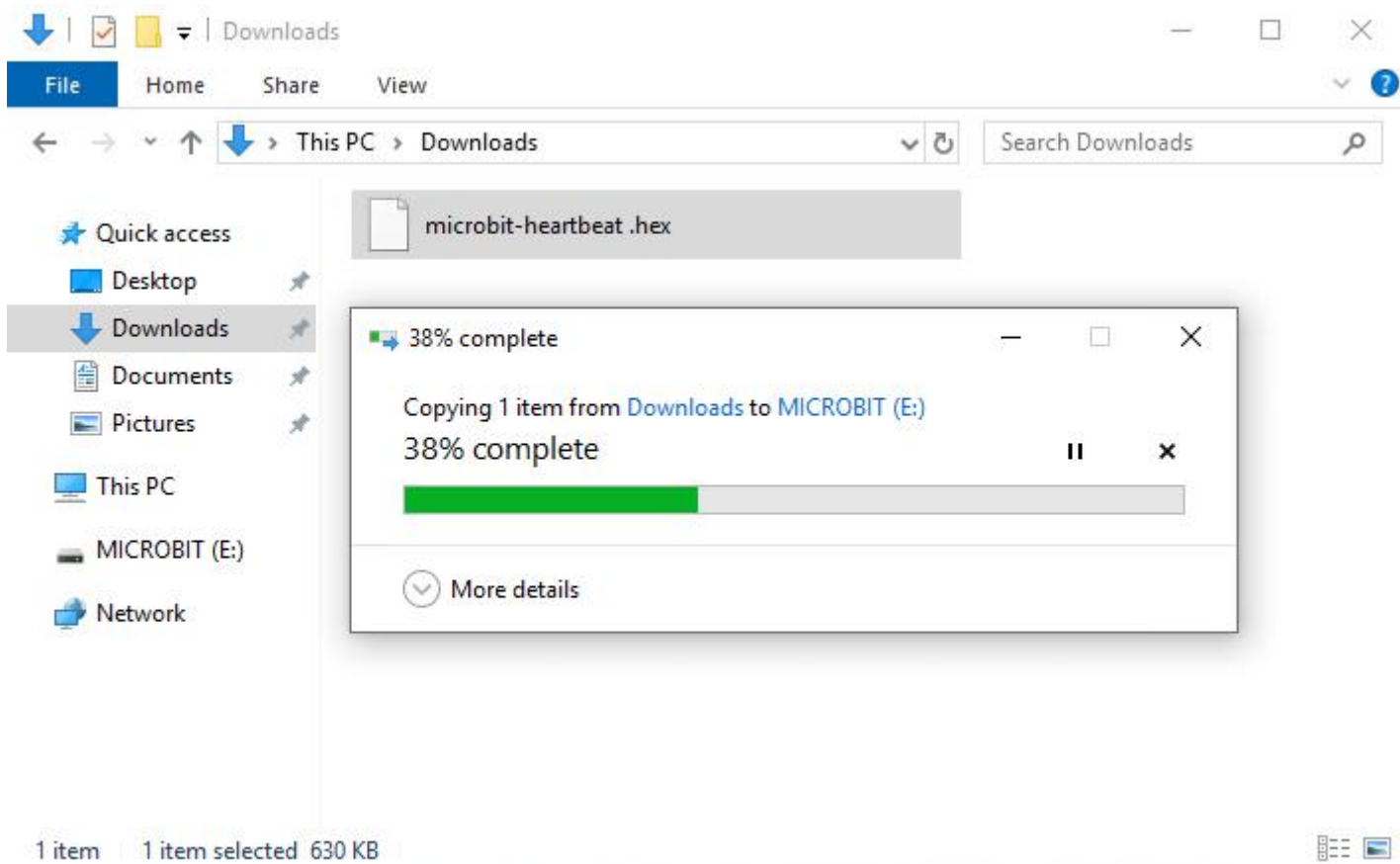
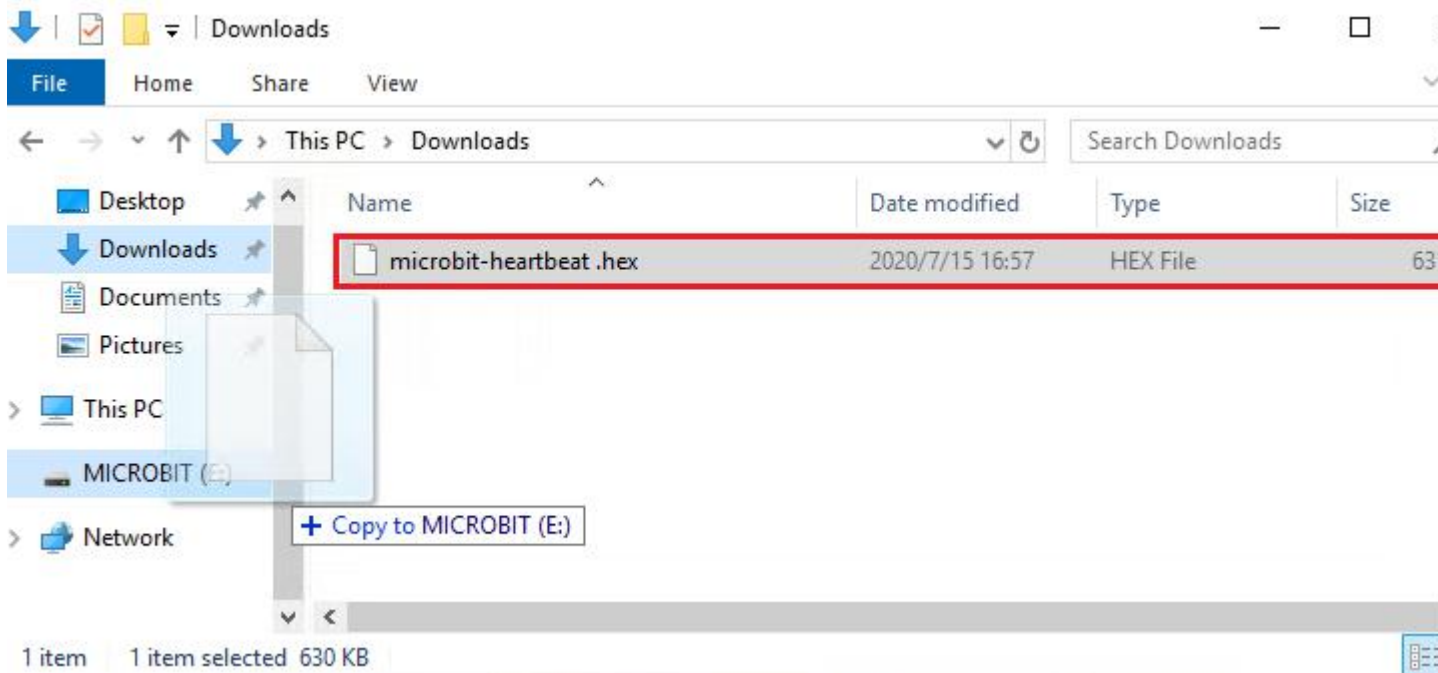
If your computer is Windows 10 and you have downloaded the APP MakeCode for micro:bit to write program, what you will have to do to download the program to your Micro: Bit main board V2 is merely clicking the 'Download' button, then all is done.

If you are writing programs through the website, following these steps: Click the 'Download' in the editor to download a "hex" file, which is a compact program format that the Micro: Bit main board can read. Once the hexadecimal file is downloaded, copy it to your board

V2 just like the process that you copy the file to the USB driver. If you are running Windows system, you can also right-click and select 'Send to → Microbit (E)' to copy the hex file to the Micro: Bit main board V2



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

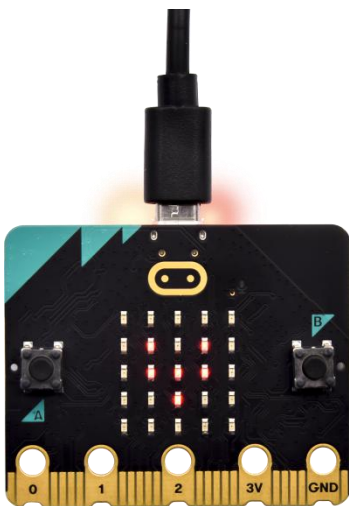


During the process of copying the downloaded hex file to the Micro: Bit main board V2, the yellow signal light on the back side of the board

flashes. When the copy is completed, the yellow signal light will stop flashing and remain on.

Step 4: run the program

After the program is uploaded to the Micro: Bit main board V2, you could still power it via the USB cable or change to via an external power. The 5 x 5 LED dot matrix on the board displays the heartbeat pattern.



micro USB cable

external power (3V)

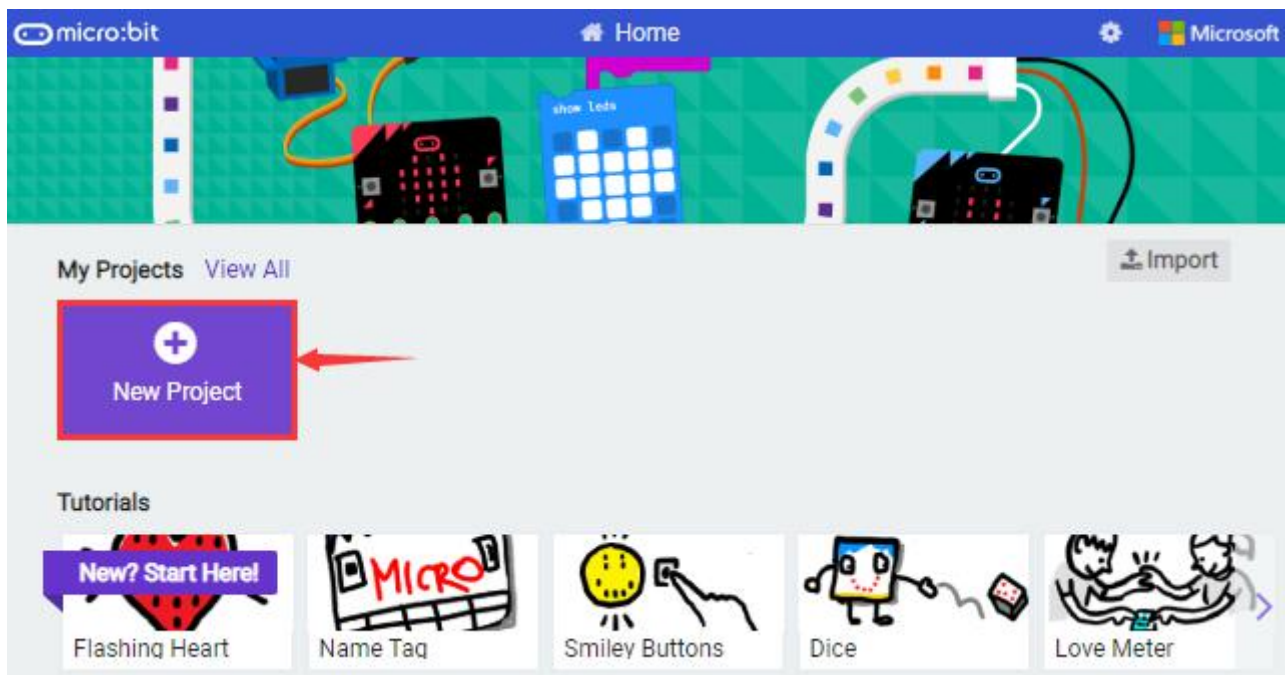
Step 5: other programming languages

This chapter has described how to use the Micro: Bit main board V2.

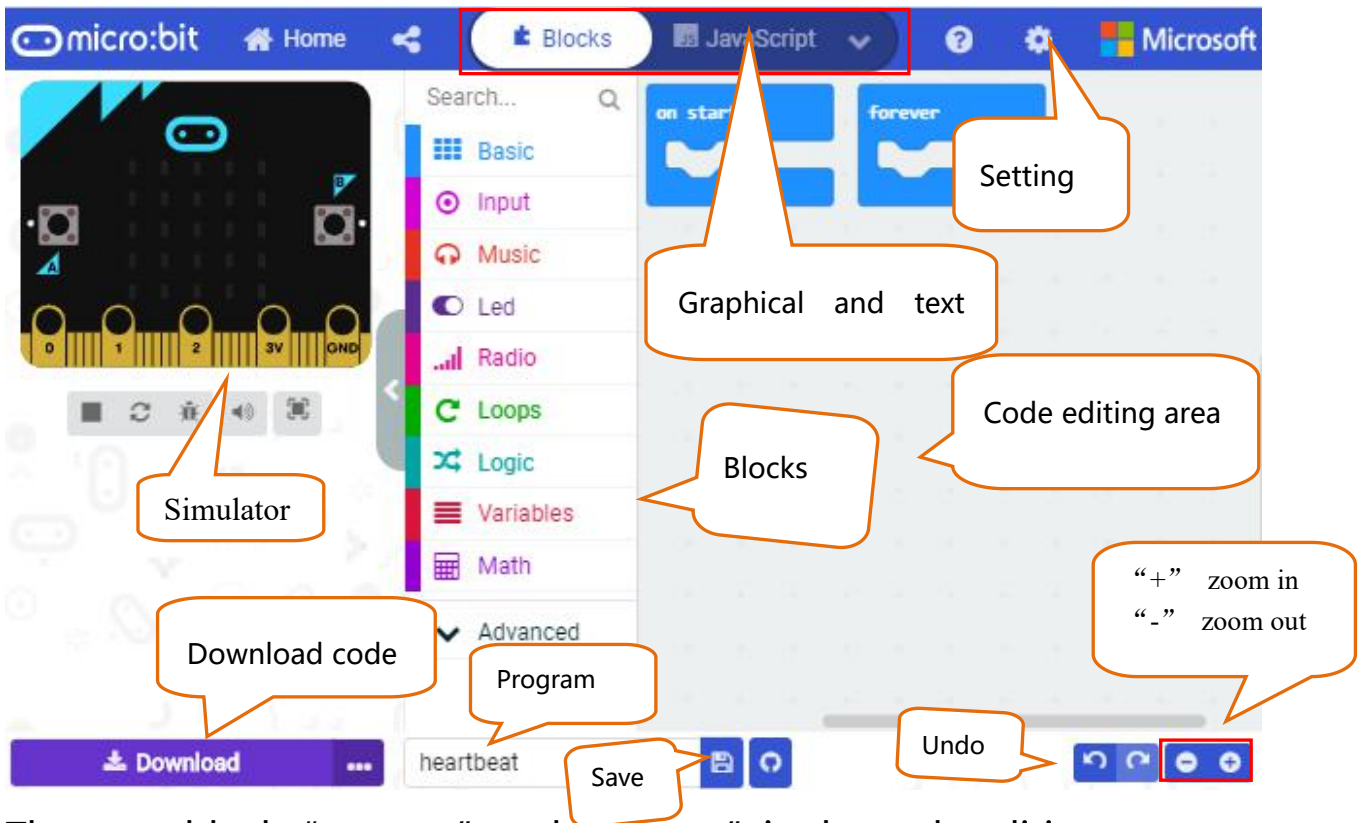
But except for the Makecode graphical programming introduced you can also write Micro: Bit programs in other languages. Go to the link: <https://microbit.org/code/> to know about other programming languages , or view the link: <https://microbit.org/projects/>, to find something you want to have a go.

5.2 Makecode:

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



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



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

When the power is plugged or reset, "on start" means that blocks in the code are only executed once, "forever" implies that code will run cyclically.

5.3.Quick Download

As mentioned before, if your computer is Windows 10 and you have downloaded the APP MakeCode for micro:bit to write programs, the program written can be quickly downloaded to the Micro: Bit main board V2 by selecting 'Download' .

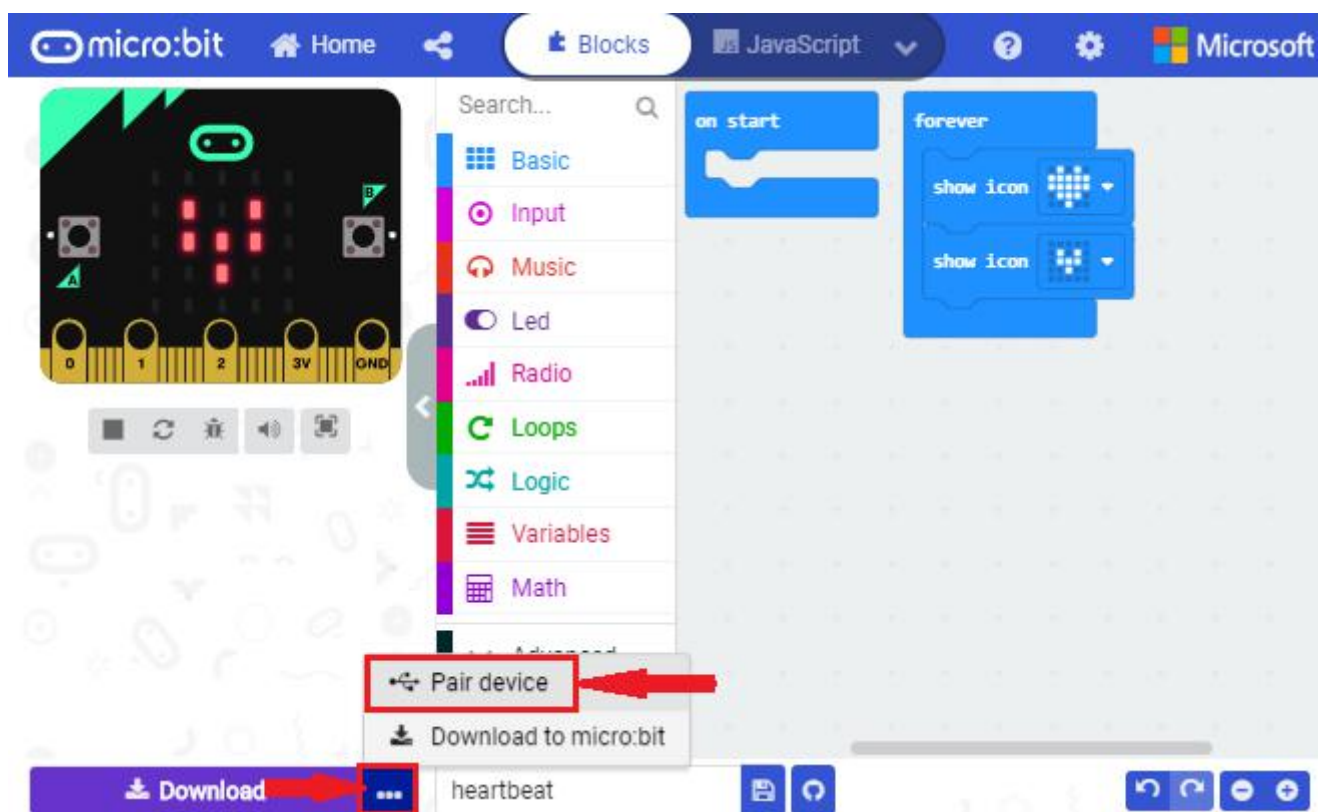
While it is a little more trickier if you are using a browser to enter makecode. However, if you use Google Chrome, suitable for Linux , macOS and Windows 10, the process can be quicker too.

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

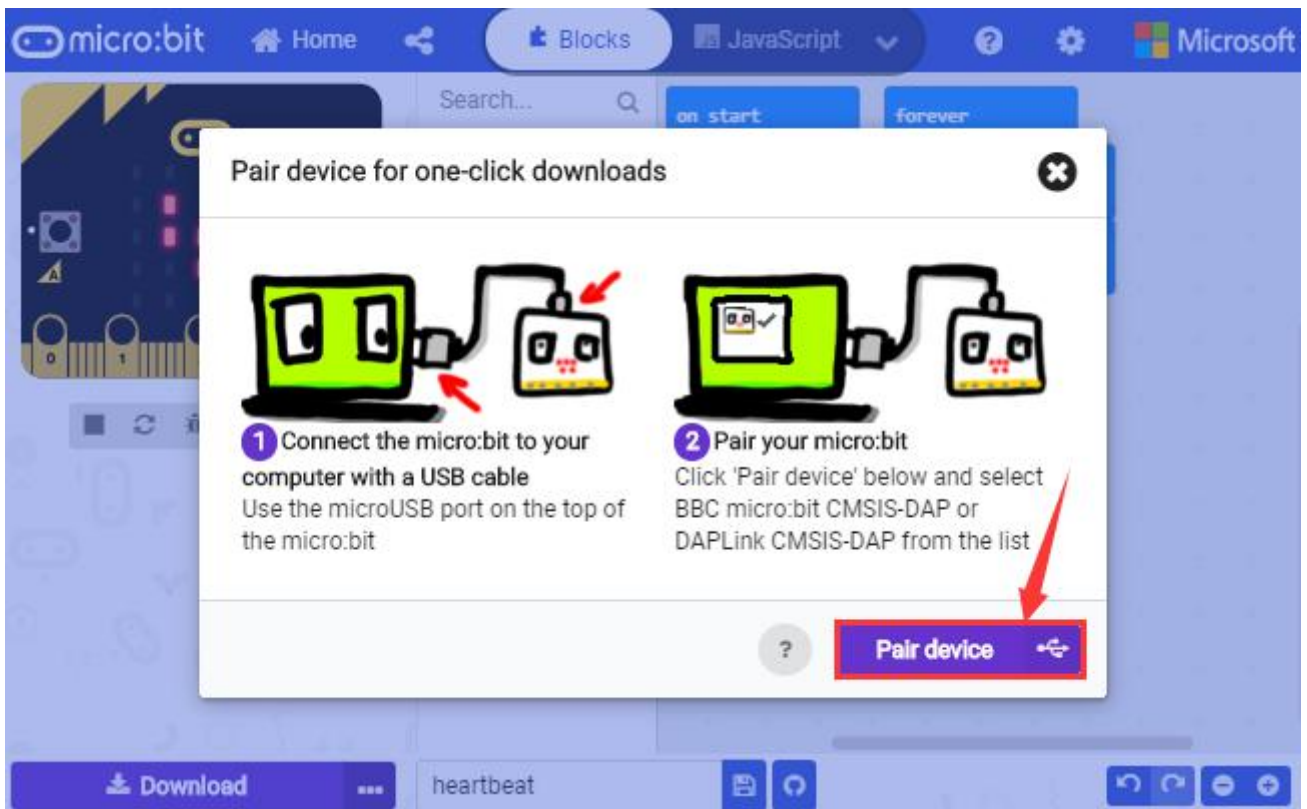
You could refer to the following steps to connect and pair devices.

Pairing device

Connect micro:bit to your computer by USB cable. Click “...” beside “Download” and click “Pair device” .



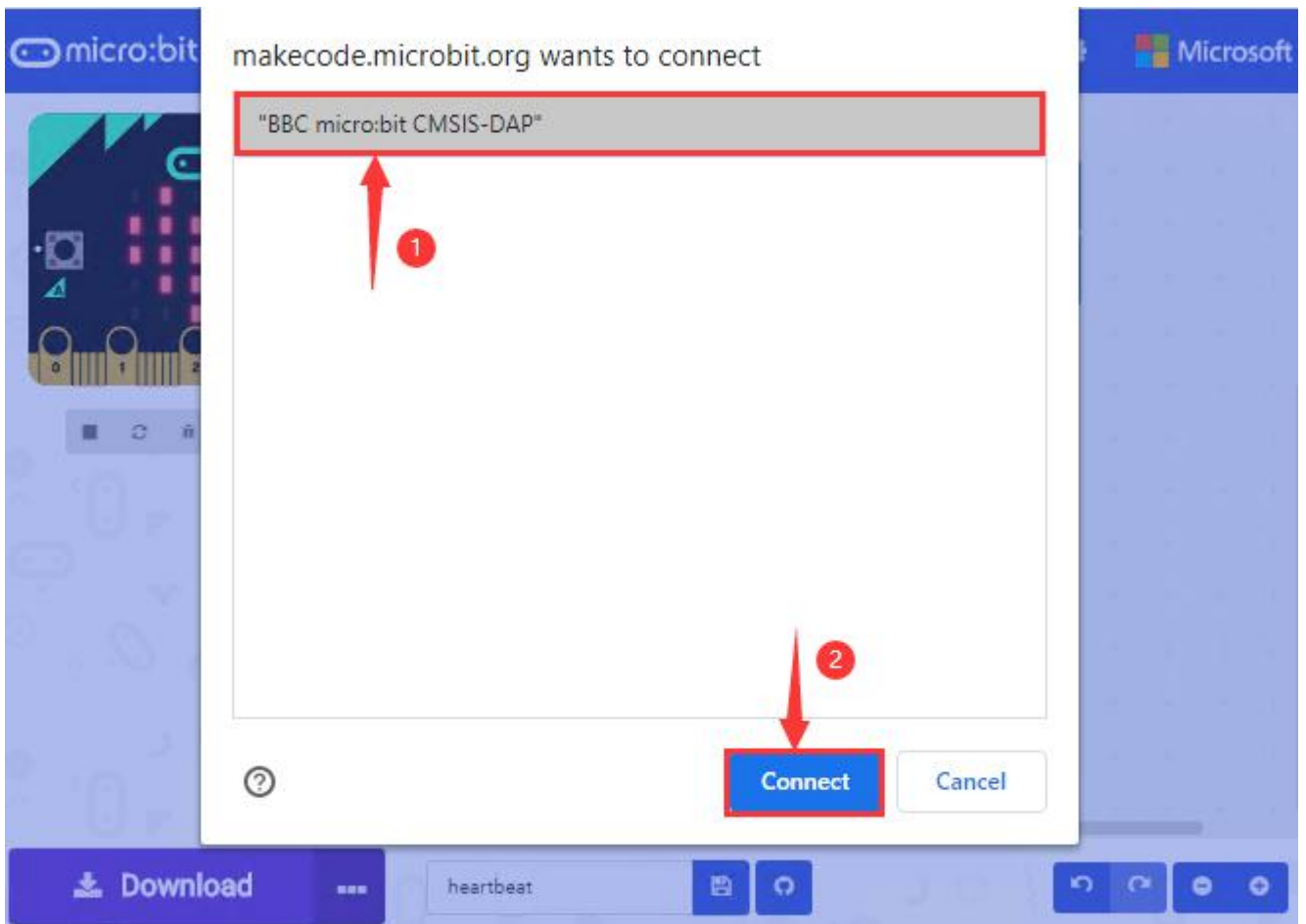
Then click another “Pair device” as shown below.



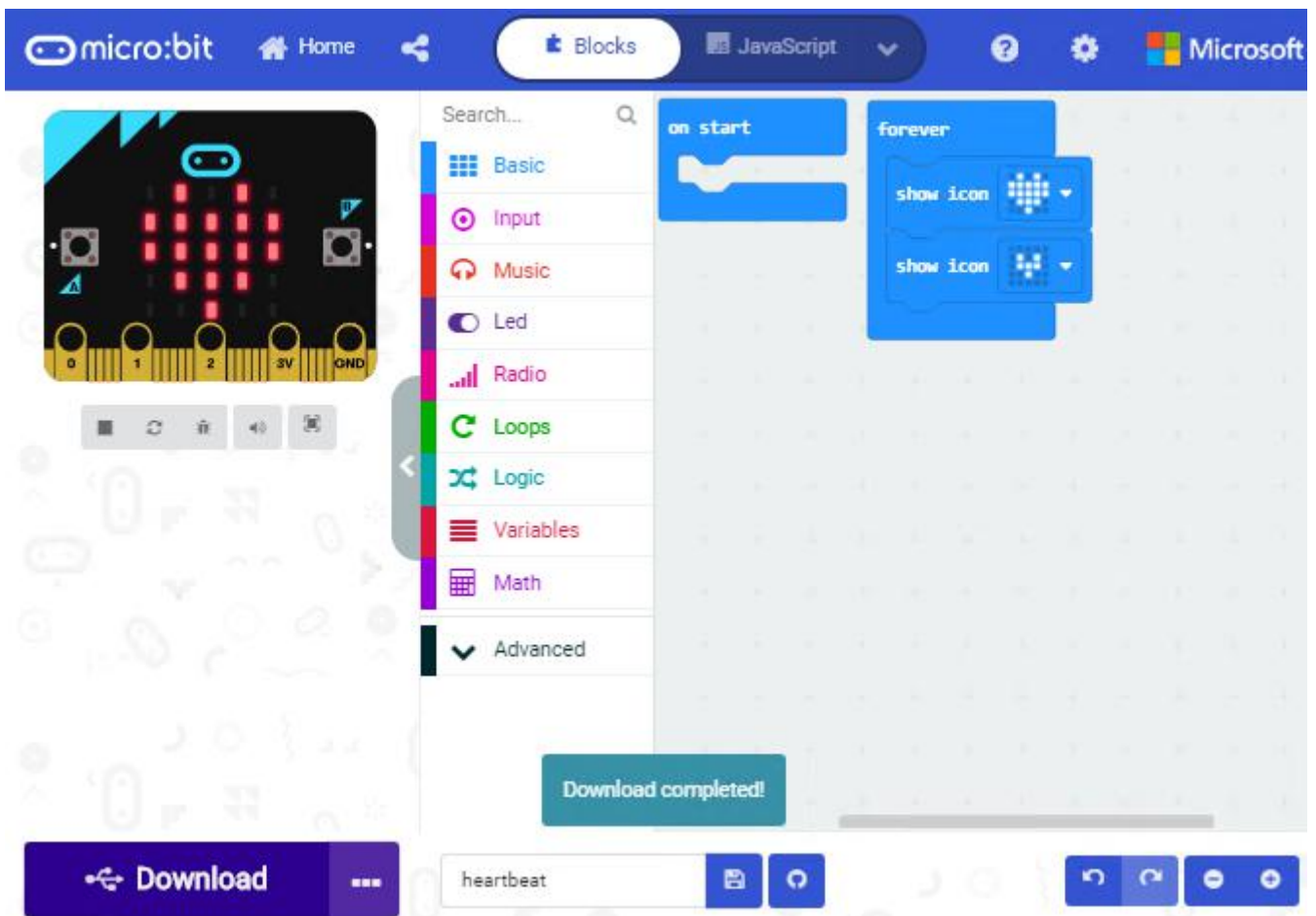
Then select "BBC micro:bit CMSIS-DPA" and click "Connect". If "BBC micro:bit CMSIS-DPA" does not show up for selection, please refer to <https://makecode.microbit.org/device/usb/webusb/troubleshoot>

We also provide [7. 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. How to Update the Firmware](#) we provide.



Then click " Download" . The program is directly downloaded to Micro: Bit main board V2 and the sentence "Download completed!" appears.



5.4 Resources and test code

Tools ,test code and other resources can be downloaded via the link <https://fs.keyestudio.com/KS4020-4021>

Download and unzip the file, you will see a file clip named KS4020 (KS4021) Keyestudio EASY PLUG Super Starter Kit For BBC micro : bit STEM EDU-, and it contains following files:

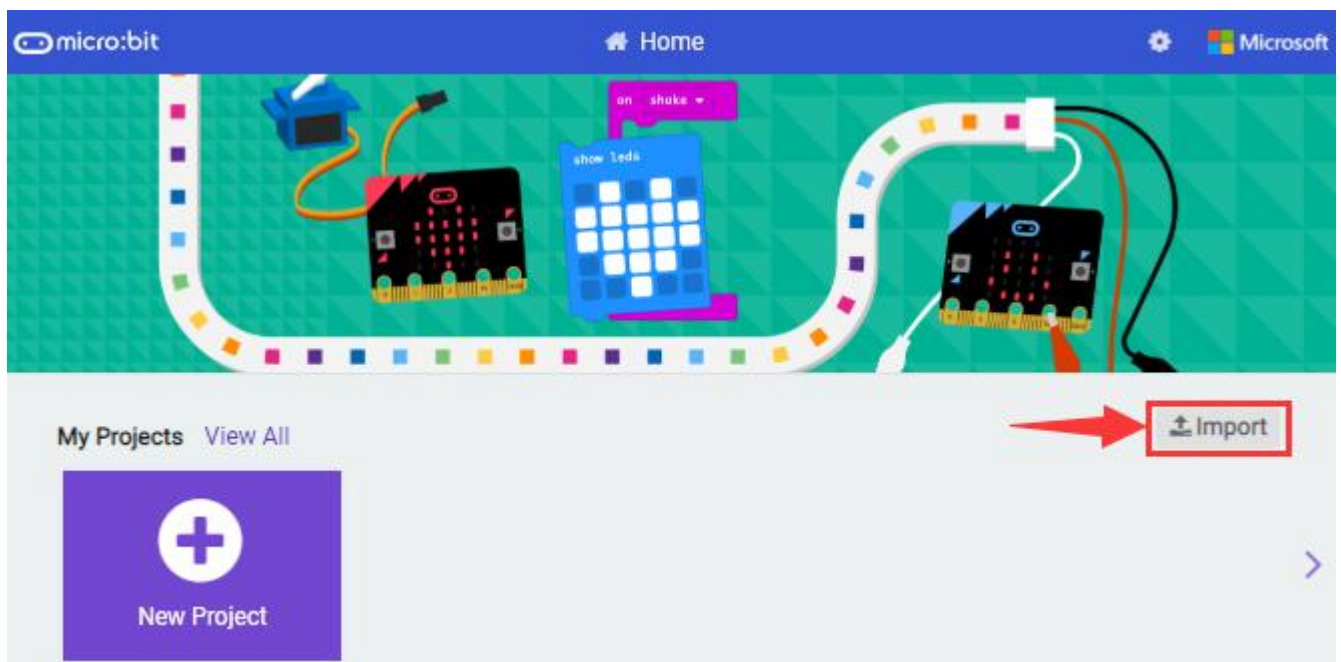
1. About Keyestudio
2. Project Code
3. Tutorial
4. How to Update the Firmware
5. Microbit Driver Installation
6. Troubleshooting-MAINTENANCE Mode
7. Troubleshooting-WebUSB
8. Cool Term Download

5.5 Input test code

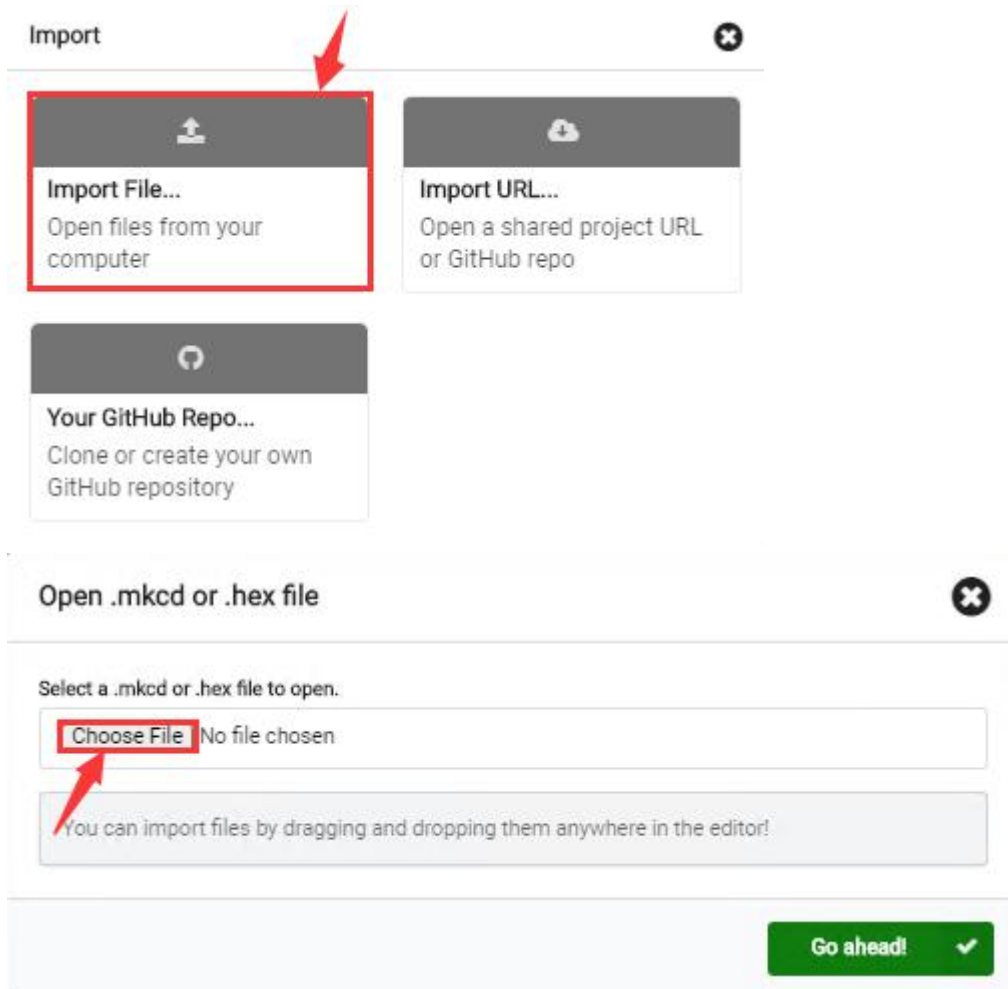
We provide hexadecimal code files (project files) for each project. The file contains all the contents of the project and can be imported directly, or you can manually drag the code blocks to complete the program for each project. For simple projects, dragging a block of code to complete the program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

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

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

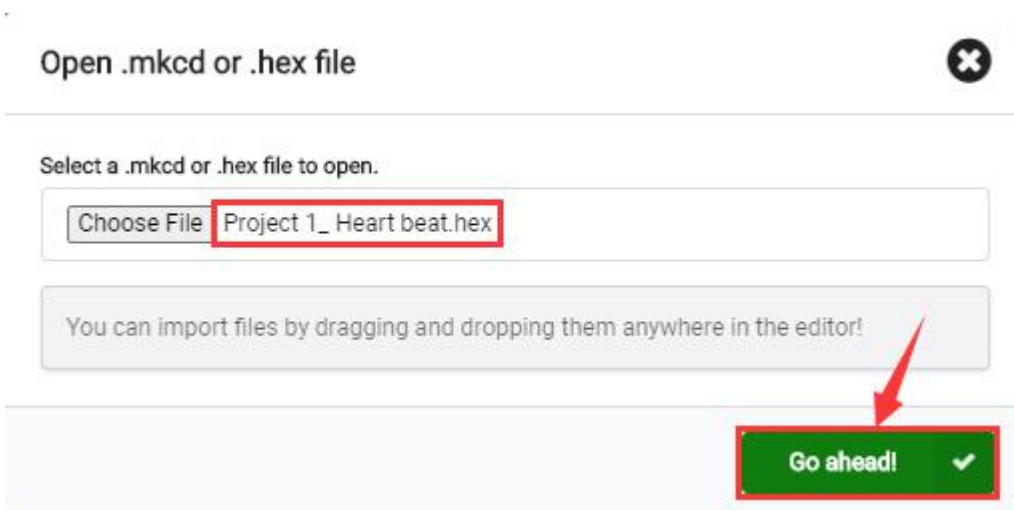
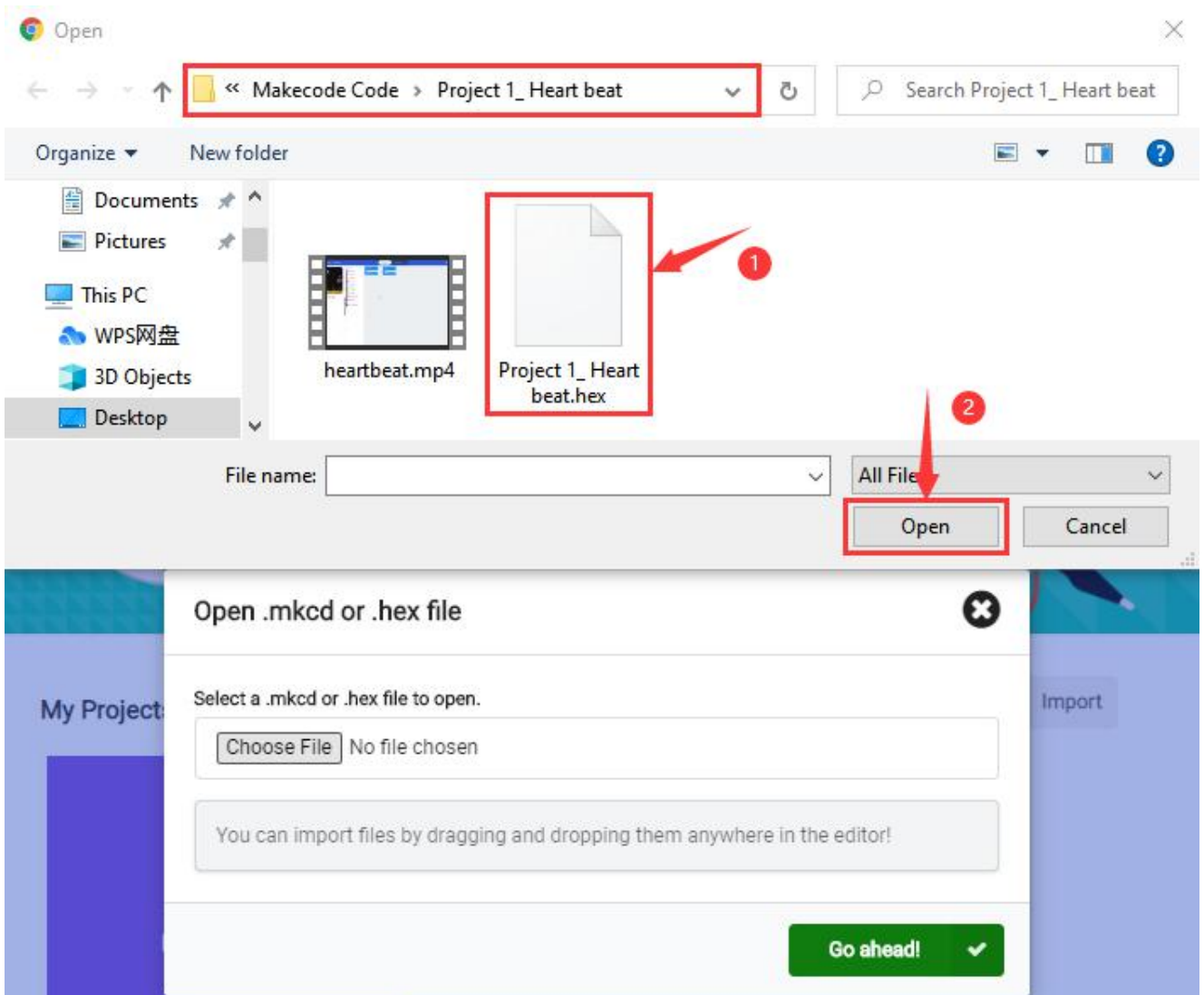


Click "Import File" ;



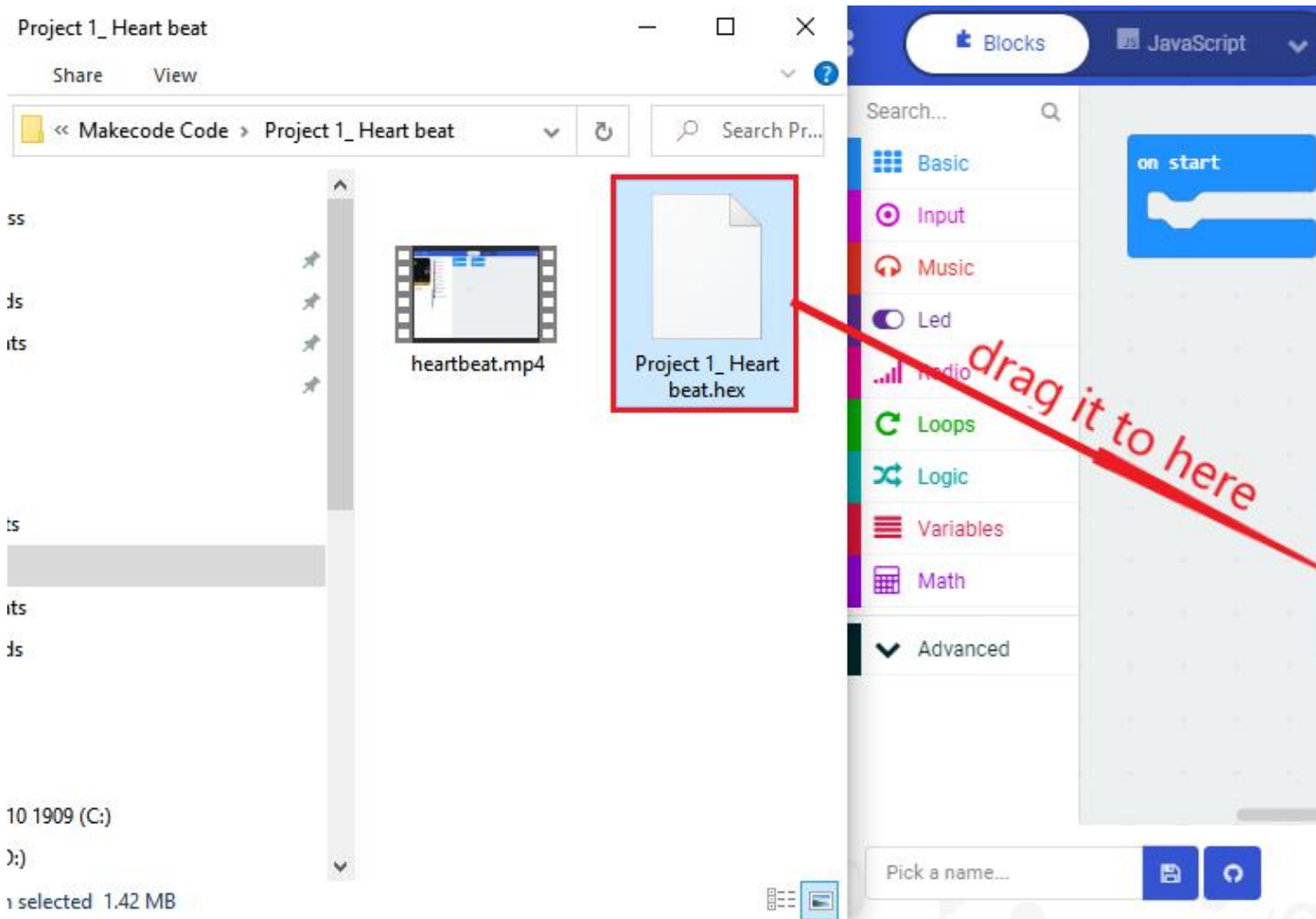
Select " ../Makecode Code/Project 1_ Heart beat/Project 1_ Heart beat.hex" ;

Then click "Go ahead" .



In addition to importing the test code file provided into the Makecode compiler above, you can also drag the the test code file provided into

the code editing area of the Makecode compiler, as shown in the figure below:



After a few seconds, it is done.



Note: if your computer system is Windows7 or 8 instead of Windows 10, the pairing cannot be done via Google Chrome. Therefore, digital signal or analog signal of sensors and modules cannot be shown on the serial port simulator. However, you need to read the corresponding digital signal or analog signal. So what can we do? You can use the CoolTerm software to read the serial port data of the micro:bit. Next chapter is about how to install CoolTerm.

5.6 CoolTerm Installation

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

Download CoolTerm program:

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

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

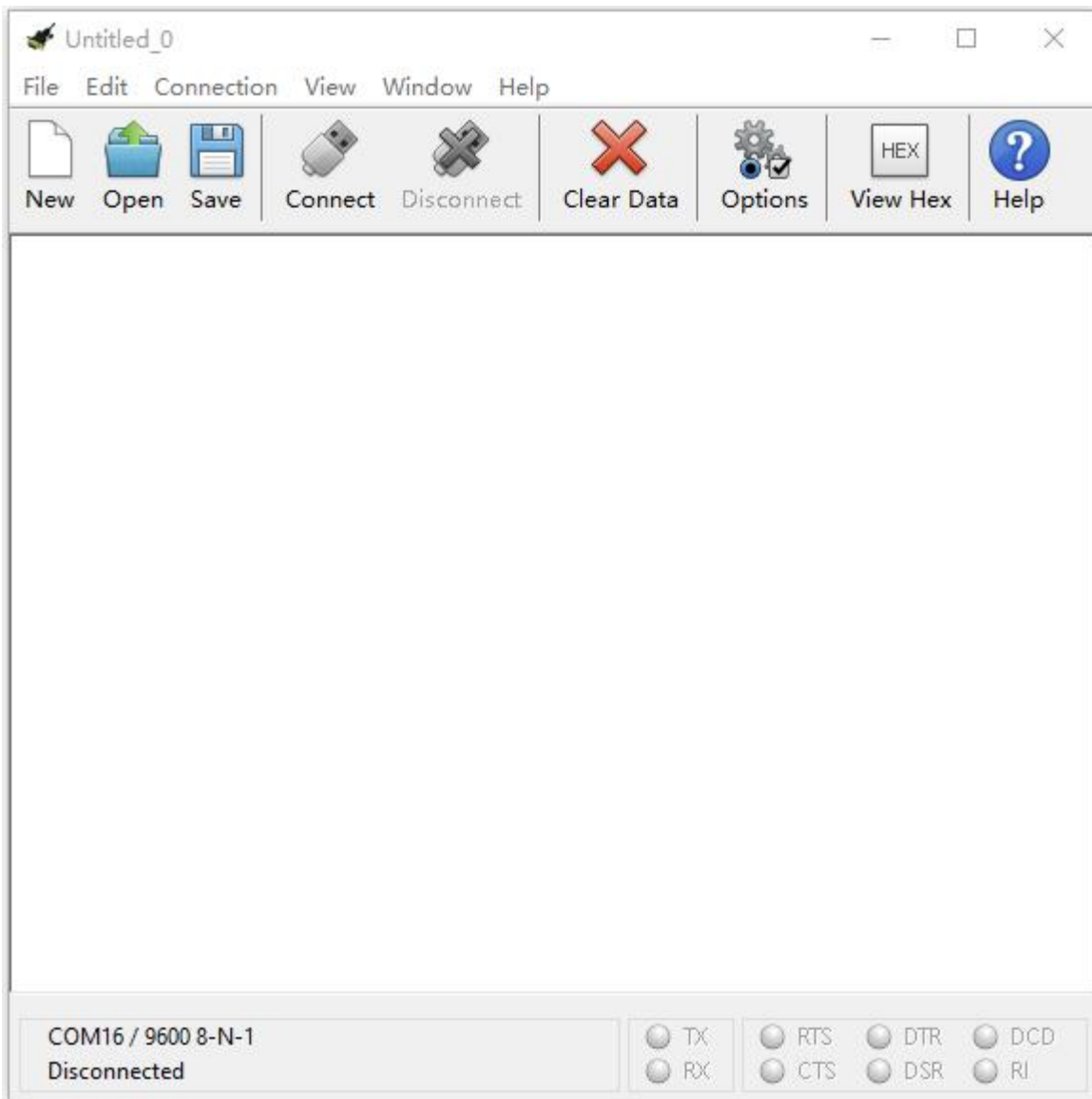
(2) Choose "win" to download the zip file of CoolTerm

(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
msvcr120.dll	2019/4/3 14:33	Application extension	941 KB
ReadMe.txt	2019/5/18 20:35	Text Document	31 KB
vccorlib140.dll	2019/4/3 14:33	Application extension	387 KB
vcruntime140.dll	2019/4/3 14:33	Application extension	88 KB
Windows System Requirements.txt	2018/1/7 14:29	Text Document	1 KB
XojoGUIFramework64.dll	2019/4/3 14:33	Application extension	30,801 KB

(2) Double-click  CoolTerm.exe .



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

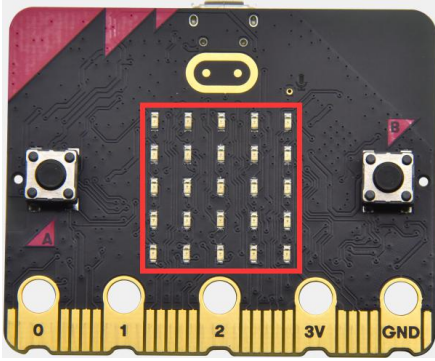


 New	Opens up a new Terminal	
 Open	Opens a saved Connection	
 Save	Saves the current Connection to disk	
 Connect	Opens the Serial Connection	
 Disconnect	Closes the Serial Connection	
 Clear Data	Clears the Received Data	
 Options	Opens the Connection Options Dialog	
 View Hex	Displays the Terminal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

6. Projects

(Note: project 1 to 12 will be conducted with the built-in sensors and LED dot matrix of the Micro:bit main board V2)

Project 1: Heartbeat



(1) Project Description

This experiment can be conducted with a micro:bit V2 main board, a micro USB cable and a computer.

The micro:bit will display a big heart-shaped image then a smaller one. That is so-called a heartbeat image.

(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code:



Attach the Micro:bit main board V2 to your computer via the Micro USB cable and begin editing.

Firstly, click" basic" module and find and drag the block "show icon



" to module "forever" ;



Secondly, click "basic" module again and find and drag the block "show icon  " to module "forever" and click the little triangle to select "show icon  " ;



Thirdly, click "basic" module and find and drag the block "pause (ms) 100" to the code block and click the little triangle to select 500;




Complete Program:



①In "on start" the program only runs once;

②In "forever" the program runs cyclically;

③The LED dot matrix displays pattern  ;

④The LED dot matrix displays pattern 

Note:the "on start" means that blocks in the code are only executed

once, “forever” implies that code will run cyclically.



Click “JS JavaScript” , you will find the corresponding programming languages.



Click the little triangle “of JS JavaScript” to choose “Python” , you will find the corresponding Python programming languages.



(4) Test Results:

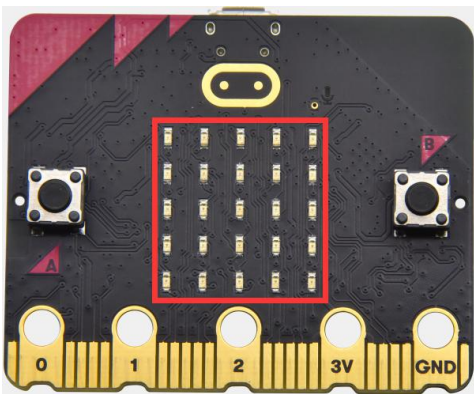
Upload test code to micro:bit V2 and keep the computer and micro:bit board connected, Then the LED dot matrix shows pattern “” and “” alternatively.

(Please refer to chapter 5.3 to know how to download test code

quickly.)

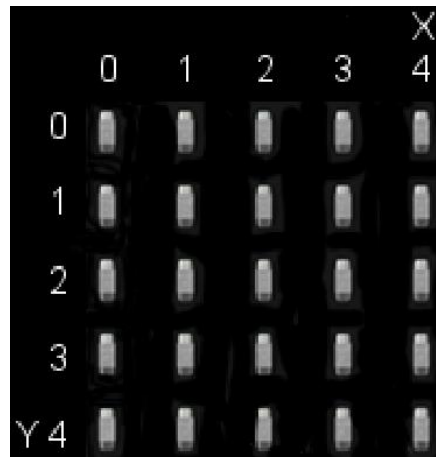
If the downloading is not smooth, please remove the micro USB from the main board and then reconnect them and reopen Makecode to try again.

Project 2: Light Up A Single LED



(1) Project Description:

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



(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code:

Attach the Micro:bit main board V2 to your computer via the Micro USB cable and begin editing.

Firstly, click "Led" module and then the "more" module to find and drag the block "led enable false " to block "on start"; click the little triangle of "led enable false " to select " true" ;



Secondly, click " Led" module and to find and drag the block "toggle

x 0 y 0 " to block "forever" and alter "x0" to " x1" ;



Thirdly, click" Basic" module to find and drag the block" pause(ms)100" to "forever" block and set pause to 500;

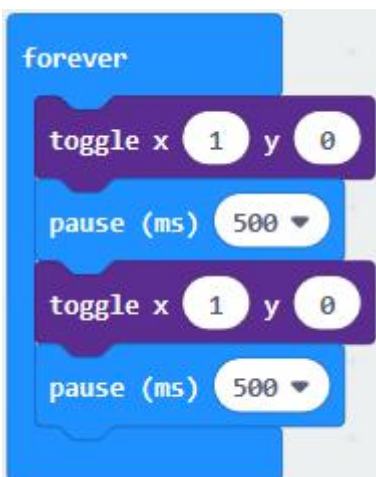


Fourthly, copy the block

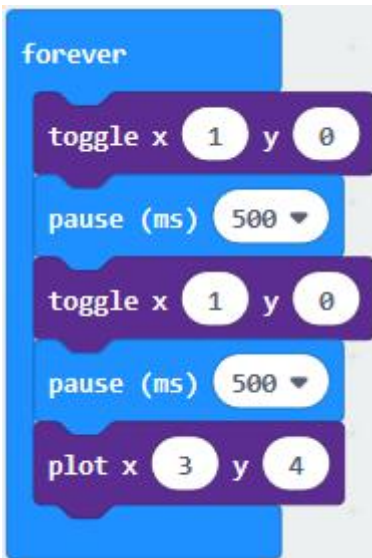


and place it into forever"

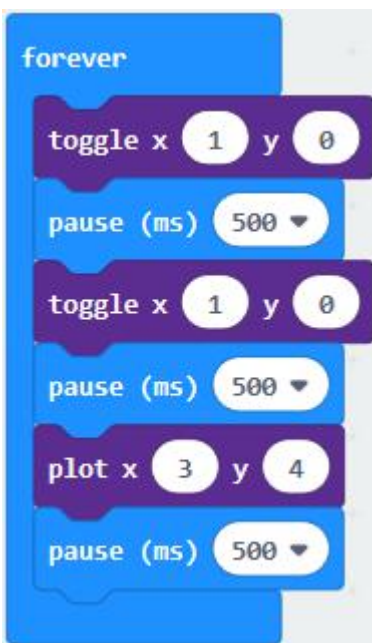
block;



Fifthly, click" Led" module to find and drag the block" plot x 0 y 0" to "forever" block and change the "x 0 y 0" to "x 3 y 4" ;



Sixthly, copy the block "pause(ms)500" and place it into forever" block;



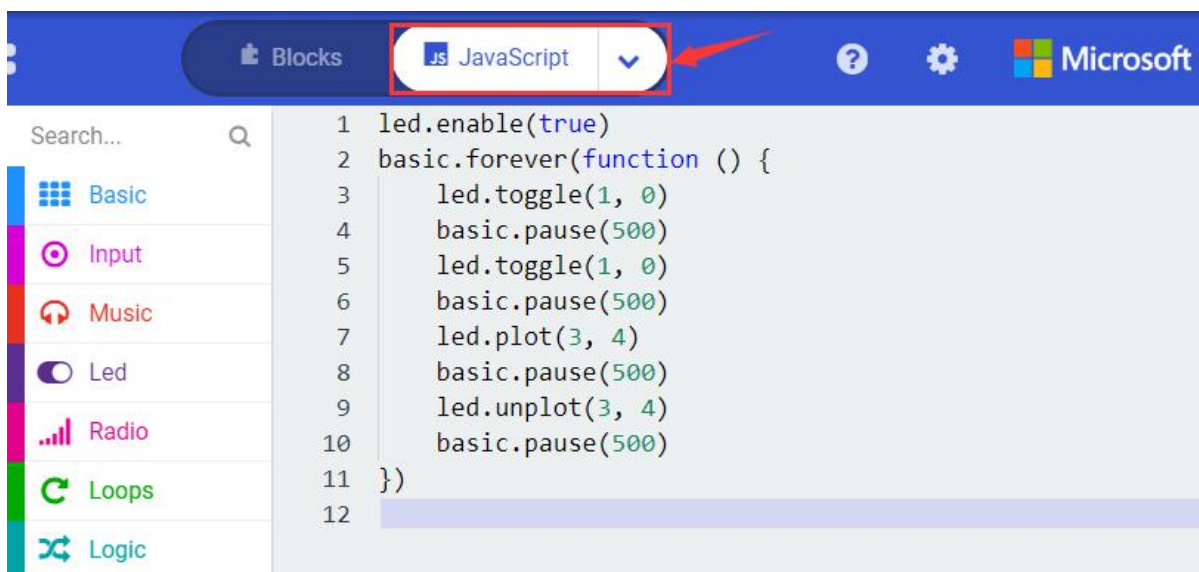
Lastly, click "Led" module to find and drag the block "unplot x 0 y 0" to "forever" block and change "x 0 y 0" to "x 3 y 4"; and copy and place the block "pause(ms)500" to block "forever" ;

Complete Program:

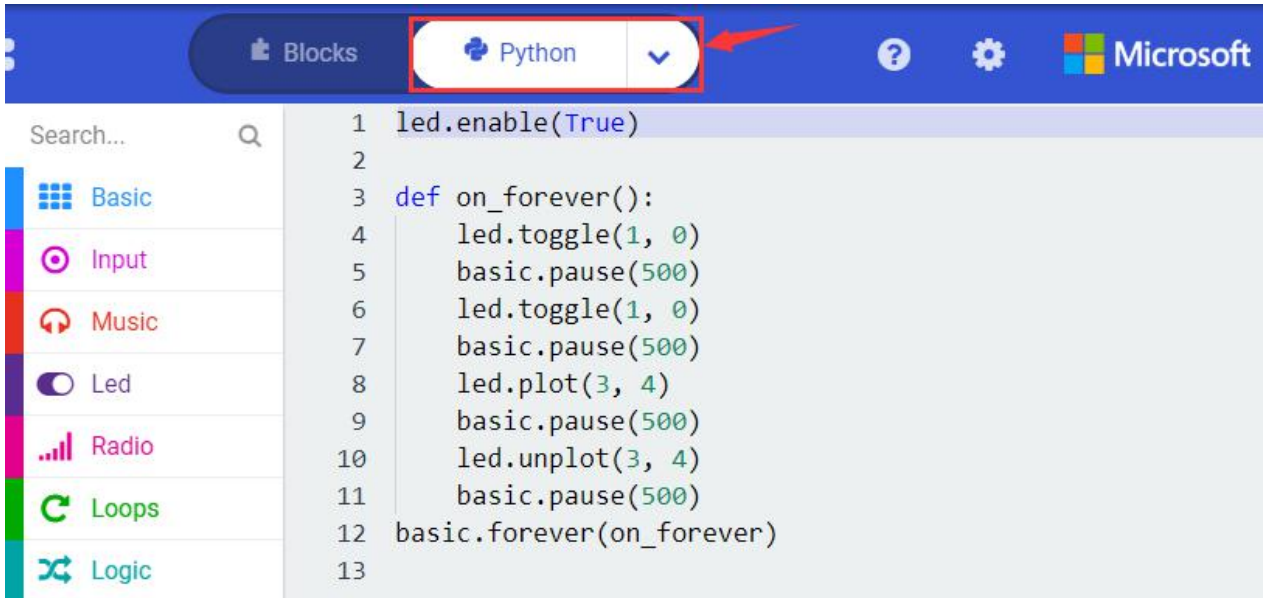


- ① In "on start" the program only runs once;
- ② Open the LED dot matrix on the microbit board;
- ③ In "forever" the program runs cyclically;
- ④ Change the brightness of the LED positioned in (1,0)
- ⑤ Delay in 500ms;
- ⑥ Change the brightness of the LED positioned in (1,0)
- ⑦ Delay in 500ms;
- ⑧ Change the brightness of the LED positioned in (3,4)
- ⑨ Delay in 500ms;
- ⑩ Turn off the LED in(3,4);
- ⑪ Delay in 500ms;

Click "JS JavaScript" , you will find the corresponding programming languages.



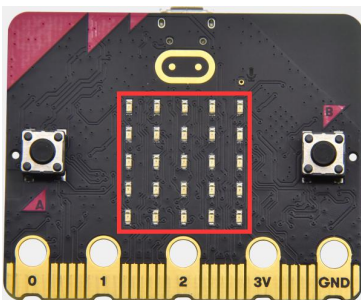
Click the little triangle of JS JavaScript to choose Python, you will find the corresponding Python programming languages.



(4) Test Results

Uploading test code to micro:bit V2 and power on it via the USB cable. The LED at (1,0) flashes for 0.5s and the one at (3,4) flashes for 0.5s

Project 3: LED Dot Matrix



(1) Project Description:

Dot matrices are very commonplace in daily life. They are founded in

LED advertisement screens, elevator floor displays, bus stops and so on.

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

(3) Components Needed:

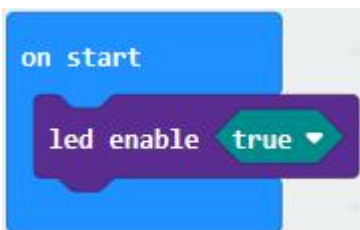
- Micro:bit main board V2 *1
- Micro USB cable*1

(4) Test Code 1:

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

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

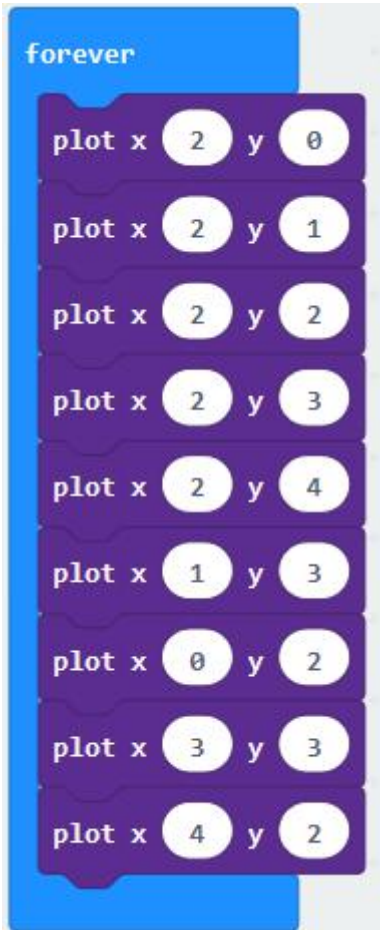
B. Click the drop-down triangle button to select " true "



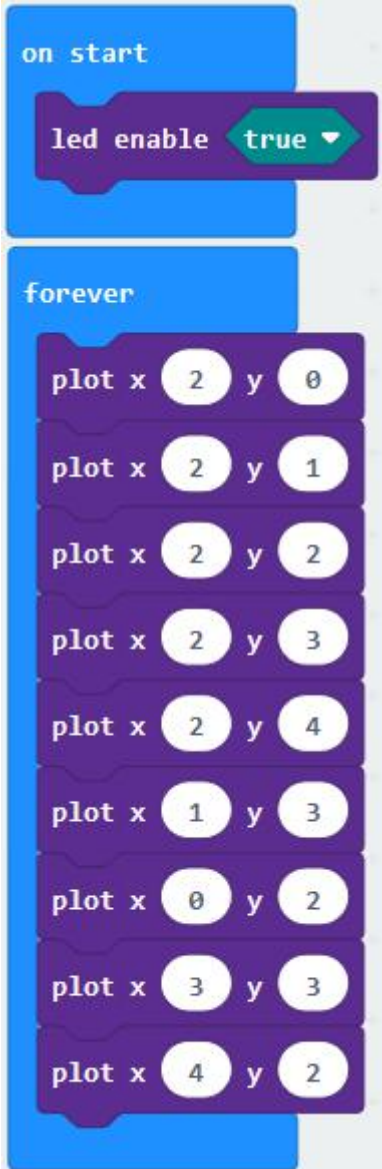
C. Combine it with "on start" block

Click "Led" to move "plot x 0 y 0" into "forever" , then replicate "plot

$x_0 y_0$ for 8 times, respectively set to $x_2 y_0$, $x_2 y_1$, $x_2 y_2$, $x_2 y_3$, $x_2 y_4$, $x_1 y_3$, $x_0 y_2$, $x_3 y_3$, $x_4 y_2$.



Complete Program:



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

Turn on LED dot matrix.

The program under the block “forever” runs cyclically.

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

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

Blocks JS JavaScript ? ⚙️ Microsoft

```

1 led.enable(true)
2 basic.forever(function () {
3     led.plot(2, 0)
4     led.plot(2, 1)
5     led.plot(2, 2)
6     led.plot(2, 3)
7     led.plot(2, 4)
8     led.plot(1, 3)
9     led.plot(0, 2)
10    led.plot(3, 3)
11    led.plot(4, 2)
12 })
13

```

Blocks Python ? ⚙️ Microsoft

```

1 led.enable(True)
2
3 def on_forever():
4     led.plot(2, 0)
5     led.plot(2, 1)
6     led.plot(2, 2)
7     led.plot(2, 3)
8     led.plot(2, 4)
9     led.plot(1, 3)
10    led.plot(0, 2)
11    led.plot(3, 3)
12    led.plot(4, 2)
13 basic.forever(on_forever)
14

```

(4) Test Results 1:

Upload code 1 , we will see the



icon

(5) Test Code 2:

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

A. Enter "Basic" → "show number 0" block,

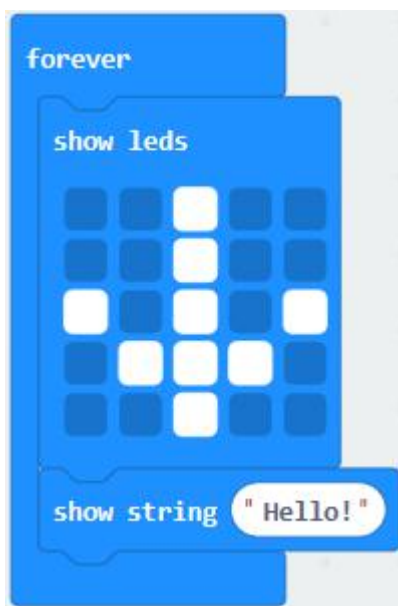
B. Duplicate it for 4 times, then separately set to "show number 1" ,
"show number 2" , "show number 3" , "show number 4" , "show number
5" .



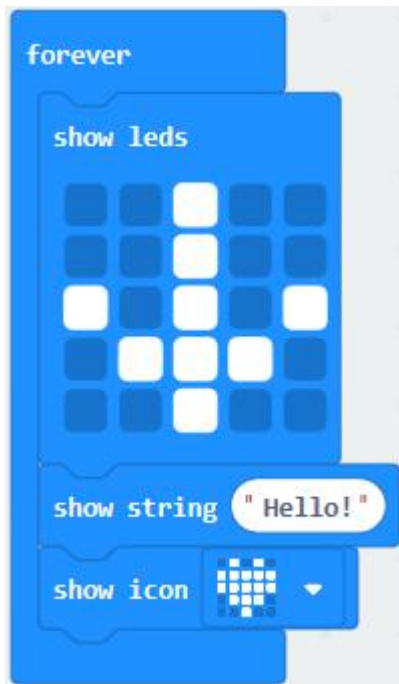
Click "Basic" → "show leds" , then put it into "forever" block, tick blue boxes to light LED and generate "↓" pattern.



Move out the block "show string" from "Basic" block, and leave it beneath the "show leds" block



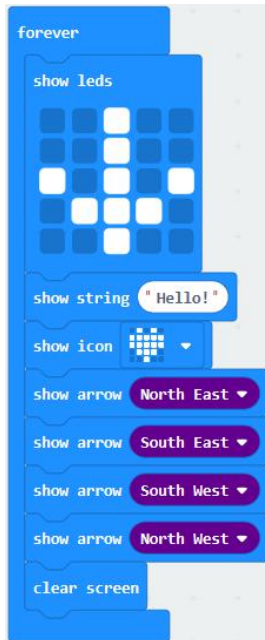
Choose "show icon" from "Basic" block, and leave it beneath the block "show string "Hello!" block



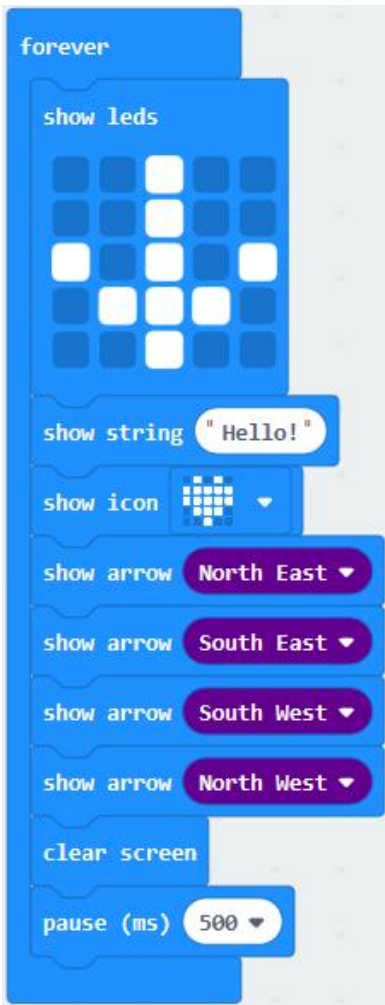
- A. Enter "Basic" → "show arrow North" ;
- B. Leave it into "forever" block, replicate "show arrow North" for 3 times, respectively set to "North East" , "South East" , "South West" , "North West" .



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



Drag "pause (ms) 100" block from "Basic" block and set to 500ms, then leave it below "clear screen" block.



Complete Program:



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

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

Under the block “forever”, program runs cyclically.

Dot matrix shows the “↓” pattern

Dot matrix scrolls to show “Hello!”

“♥” is shown on dot matrix

LED dot matrix displays “North East” arrow.

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

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

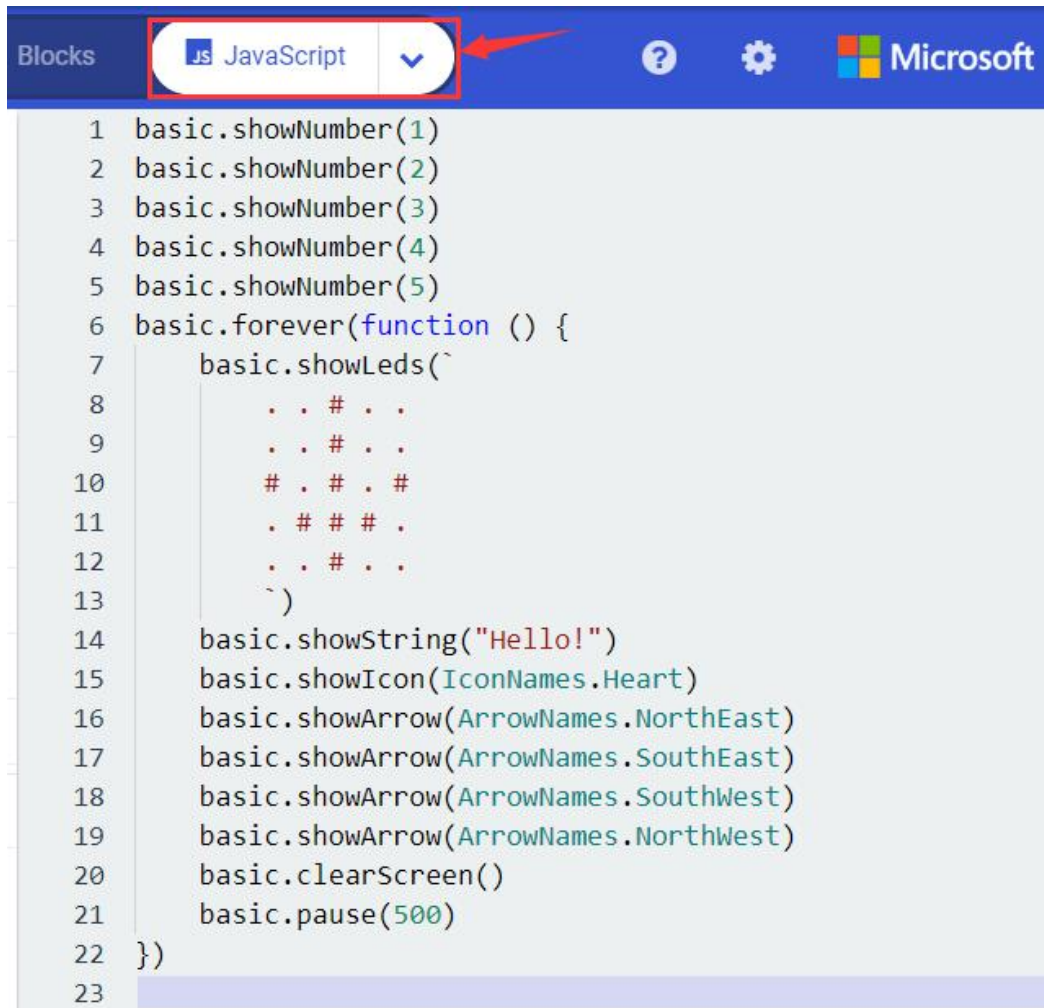
The “North West” arrow is displayed on LED dot matrix

Clear the screen

Delay in 500ms

Select “JavaScript” and “Python” to switch into JavaScript and Python

language code:



```
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
```

```

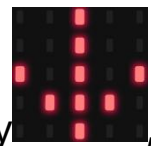
Blocks Python
1 basic.show_number(1)
2 basic.show_number(2)
3 basic.show_number(3)
4 basic.show_number(4)
5 basic.show_number(5)
6
7 def on_forever():
8     basic.show_leds("""
9         . . # . .
10        . . # . .
11        # . # . #
12        . # # # .
13        . . # . .
14        """)
15     basic.show_string("Hello!")
16     basic.show_icon(IconNames.HEART)
17     basic.show_arrow(ArrowNames.NORTH_EAST)
18     basic.show_arrow(ArrowNames.SOUTH_EAST)
19     basic.show_arrow(ArrowNames.SOUTH_WEST)
20     basic.show_arrow(ArrowNames.NORTH_WEST)
21     basic.clear_screen()
22     basic.pause(500)
23 basic.forever(on_forever)
24

```

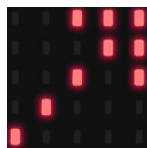
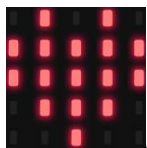
(6) Test Results 2:

Upload code 2 and plug the micro:bit to a computer. The micro: bit

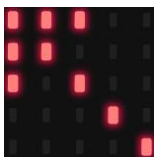
starts showing number 1, 2, 3, 4, and 5, then cyclically display



"Hello!" ,

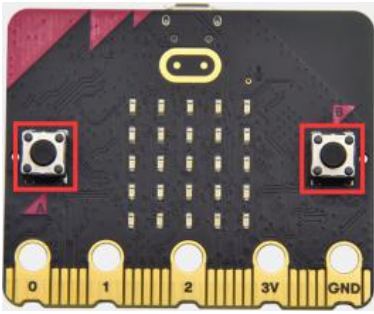


and



patterns.

Project 4: Programmable Buttons



(1) Project Description:

Buttons can be used to control circuits. In an integrated circuit with a button, the circuit is controlled by the button.

Micro:bit main board V2 boasts three buttons, two programmable buttons(A and B), and the one on the other side is a reset button. By pressing the two buttons, three different signals can be output.

The micro:bit will show A, B and AB if you press button A, B and AB respectively.

Let' s get started.

(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code 1:

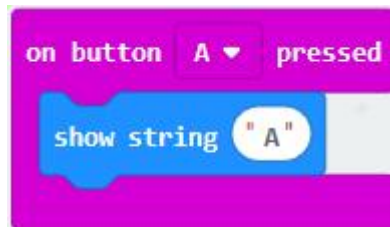
Link computer with micro:bit board by micro USB cable, and program in MakeCode editor,

Delete "on start" and "forever" firstly, then click "Input" → "on button

A pressed"

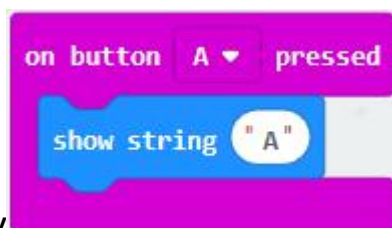
A. Click "Basic" → "show string" ;

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



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





Copy once, and set to "on button A+B pressed"

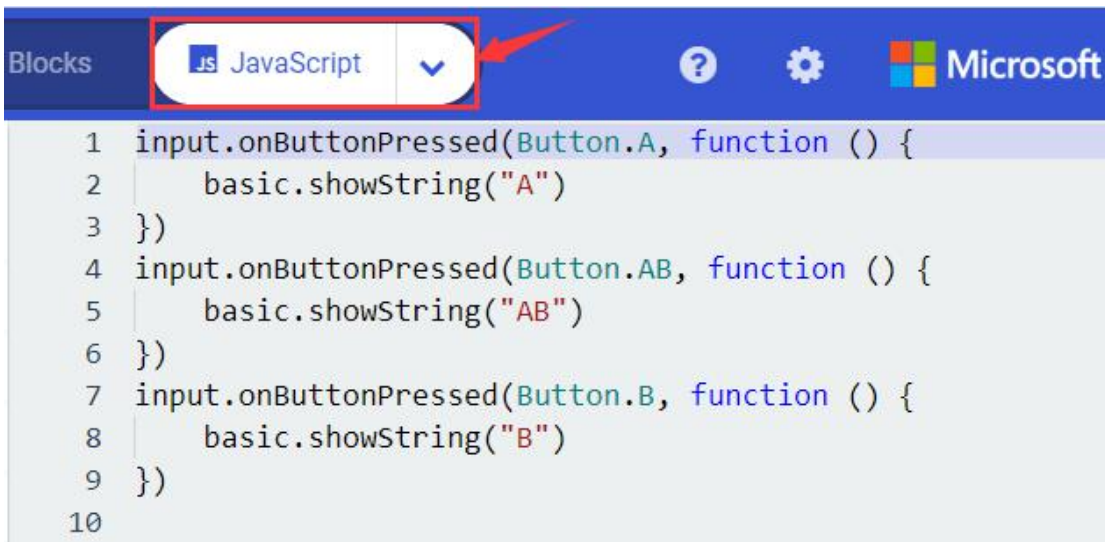
and "show string "AB"



Complete Code:

	<p>Press button A on Micro: bit main board Show the character "A"</p> <p>Press button B on Micro: bit main board Show the character "B"</p> <p>Press button A and B at same time Display the character "AB"</p>

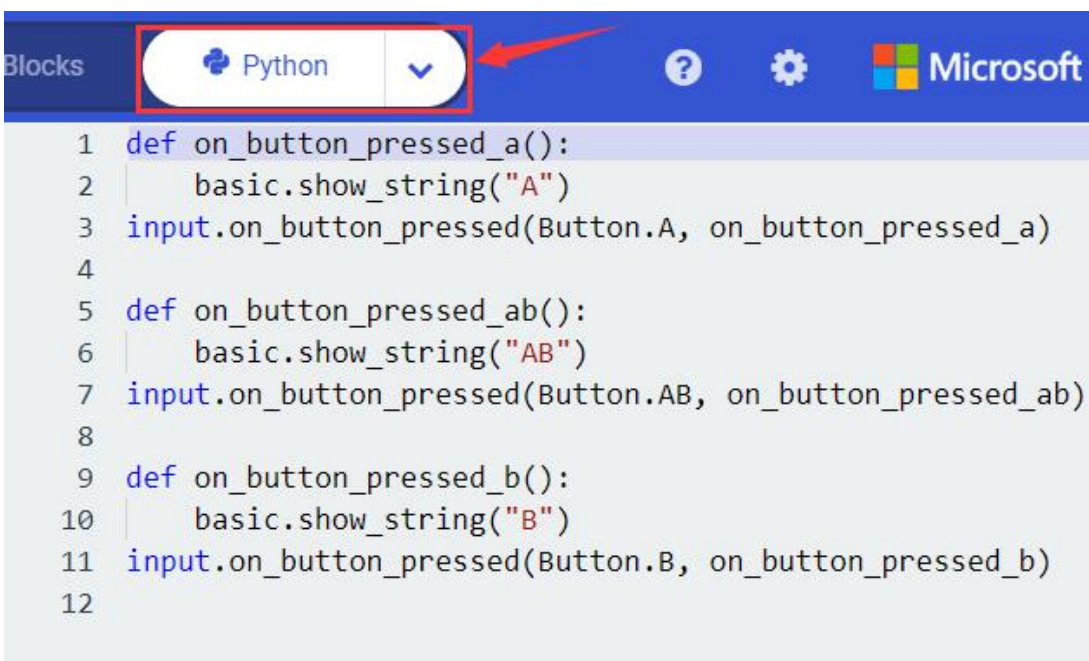
Select "JavaScript" and "Python" to switch into JavaScript and Python language code:



```

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

```



```

1 def on_button_pressed_a():
2   basic.show_string("A")
3 input.on_button_pressed(Button.A, on_button_pressed_a)
4
5 def on_button_pressed_ab():
6   basic.show_string("AB")
7 input.on_button_pressed(Button.AB, on_button_pressed_ab)
8
9 def on_button_pressed_b():
10  basic.show_string("B")
11 input.on_button_pressed(Button.B, on_button_pressed_b)
12

```

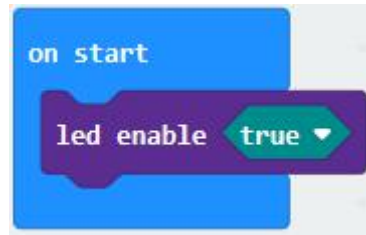
(4) Test Results 1:

Upload the test code 1 to micro:bit main board V2. Then the 5*5 LED dot matrix will show A, B and AB if button A and B pressed together.

(5) Test Code 2:

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

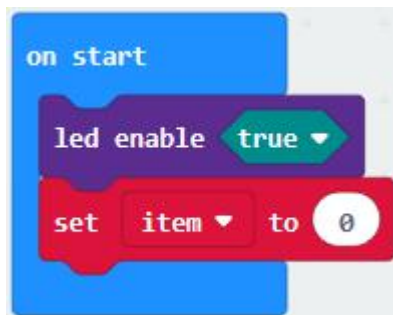
B. Put it into the block "on start" , click drop-down triangle button to



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" .

(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.

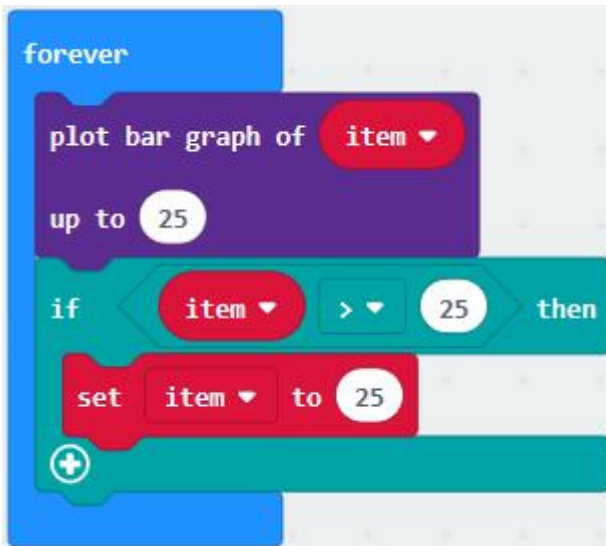


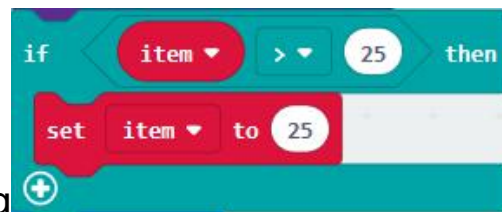
(6)A. Go to "Logic" to move out "if...true...then..." and "=" blocks,

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.



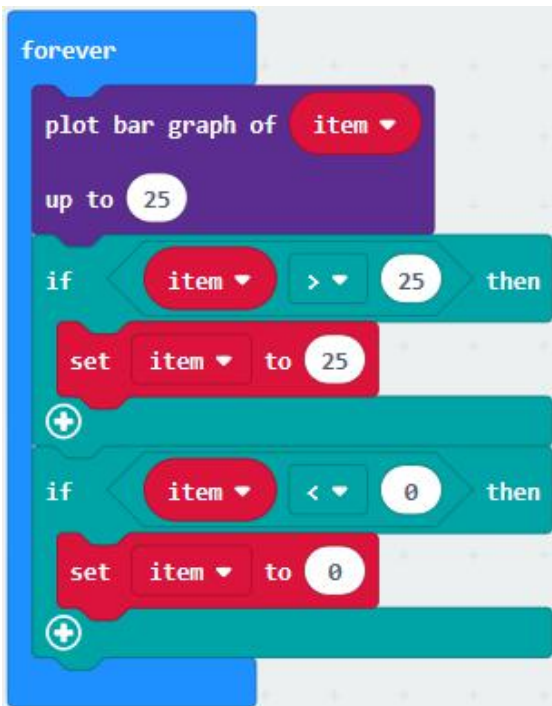


(7) A. Replicate code string  once

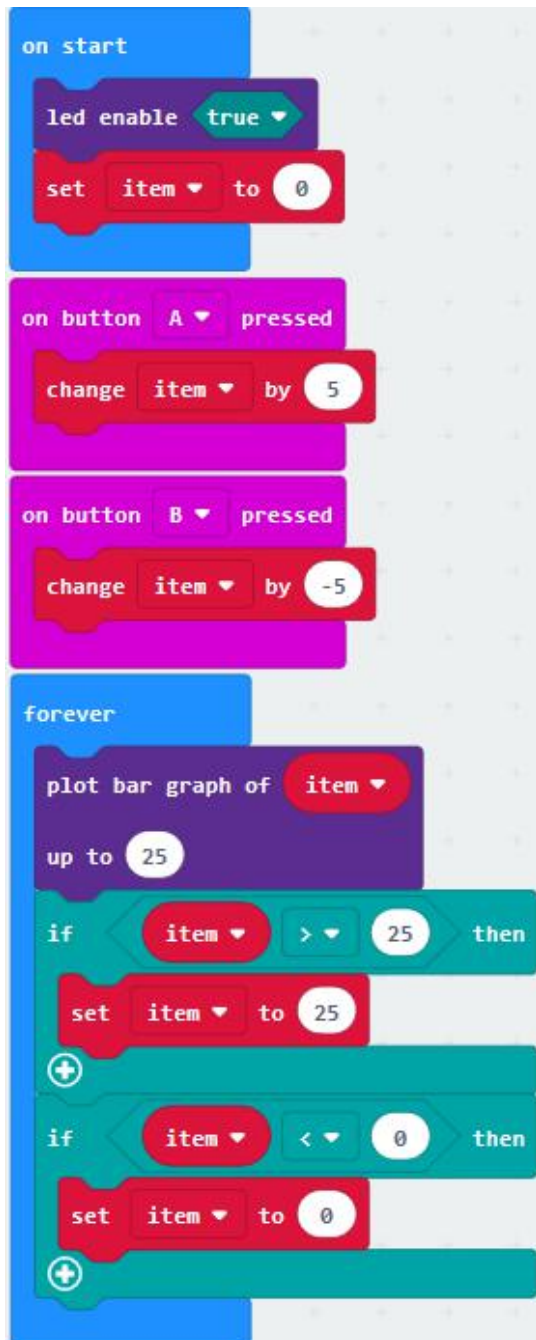
B. ">" is modified into "<" and 25 is changed into 0,



C. Leave it beneath  code string.



Complete Program:



“on start”: command block runs once

to start program.

Turn on LED dot matrix

Set the initial value of item to 0

Press button A on Micro:bit board

Change item by 5

Press button B on Micro:bit board

Change item by -5

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

LEDs

If item is greater than 25

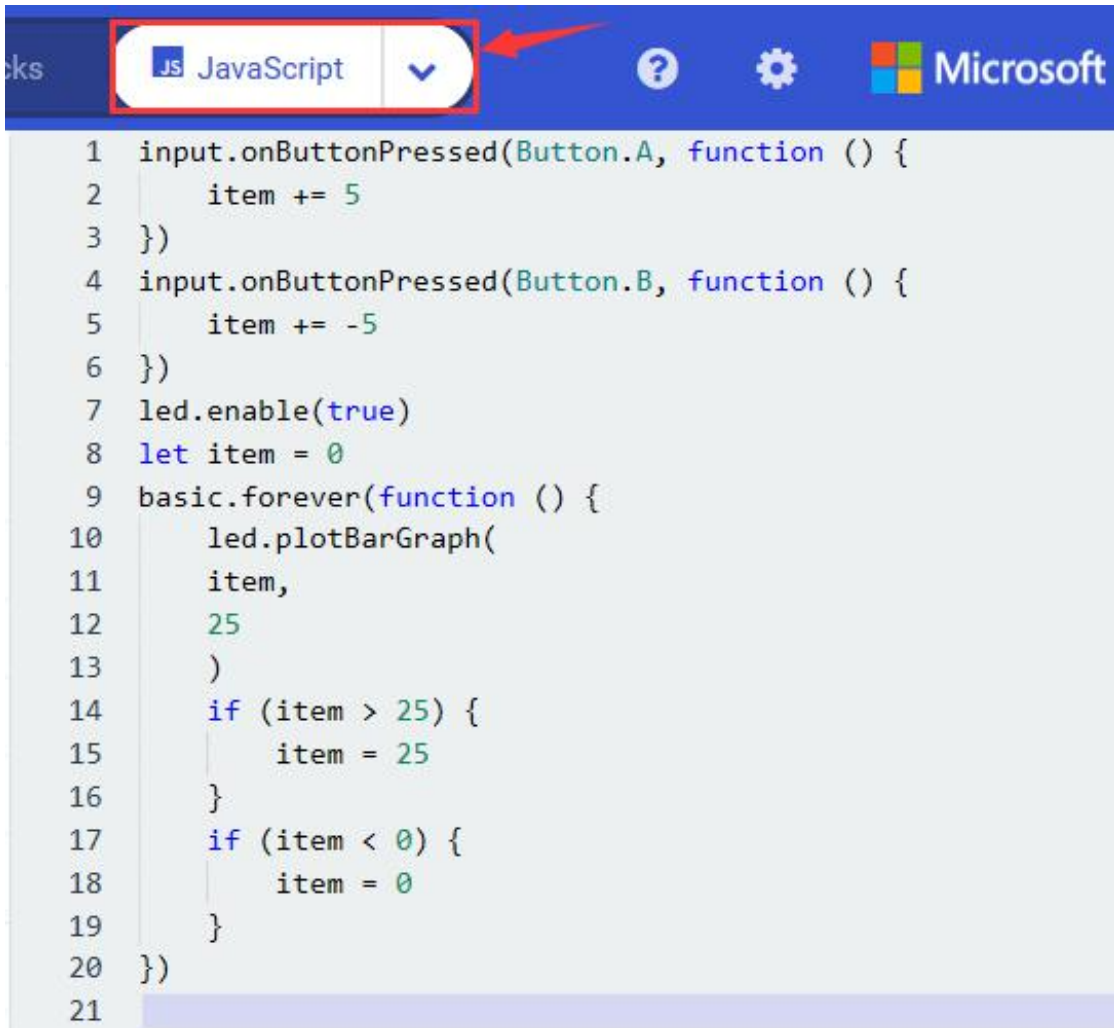
Then set item to 25

If item is less than 0

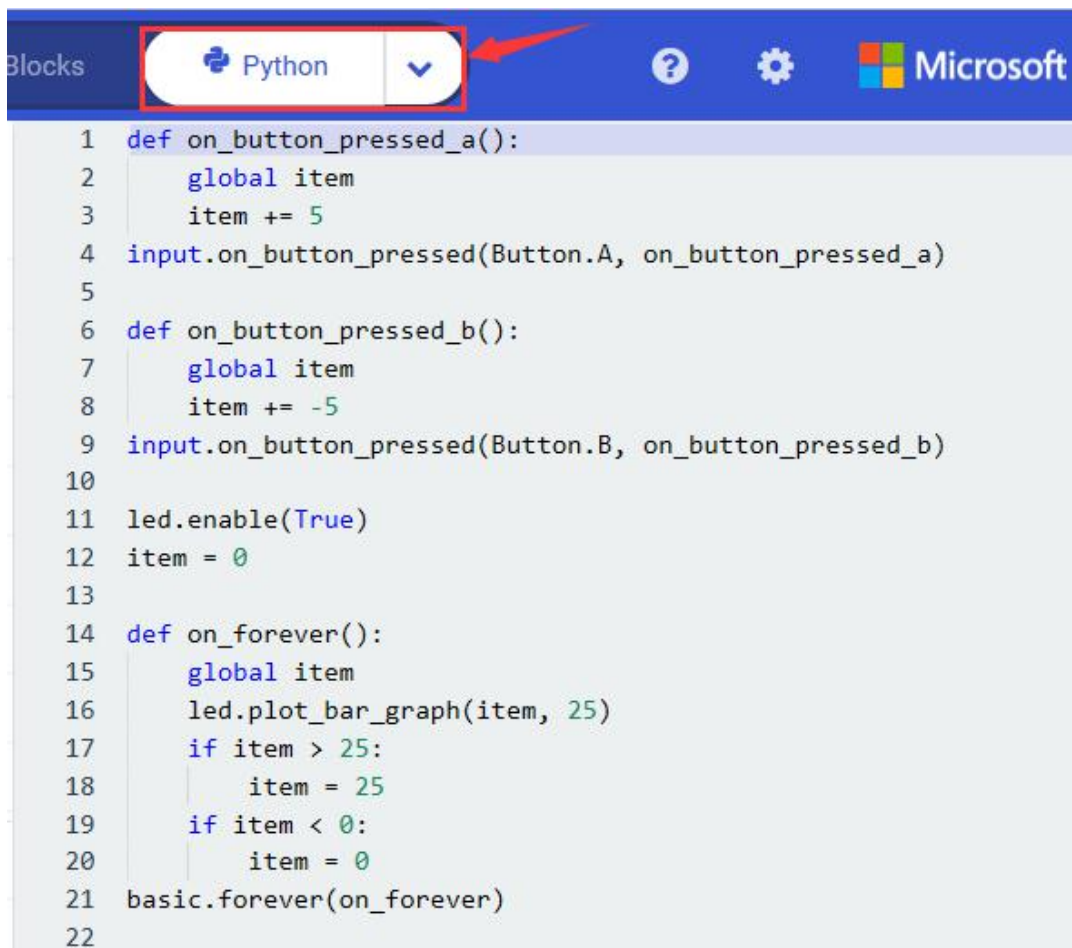
Then Set item to 0

Select “JavaScript” and “Python” to switch into JavaScript and Python

language code:



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



```
1 def on_button_pressed_a():
2     global item
3     item += 5
4 input.on_button_pressed(Button.A, on_button_pressed_a)
5
6 def on_button_pressed_b():
7     global item
8     item += -5
9 input.on_button_pressed(Button.B, on_button_pressed_b)
10
11 led.enable(True)
12 item = 0
13
14 def on_forever():
15     global item
16     led.plot_bar_graph(item, 25)
17     if item > 25:
18         item = 25
19     if item < 0:
20         item = 0
21 basic.forever(on_forever)
22
```

(6) Test Results 2:

Uploading test code 2 to micro:bit main board V2. A row of lights are turned on when you press button A, B the LEDs turning red reduce.

Project 5: Temperature Detection



(1) Project Description:

Actually, the micro:bit V2 is not equipped with a temperature sensor, but uses the built-in NFR52833 chip for temperature detection. Therefore, the detected temperature should be closer to the chip.

(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*

(3) Test Code 1:

Click "Advanced" → "Serial" → "serial redirect to USB" into "on start"



A. Go to "Serial" → "serial write value "x" =0" into "forever"



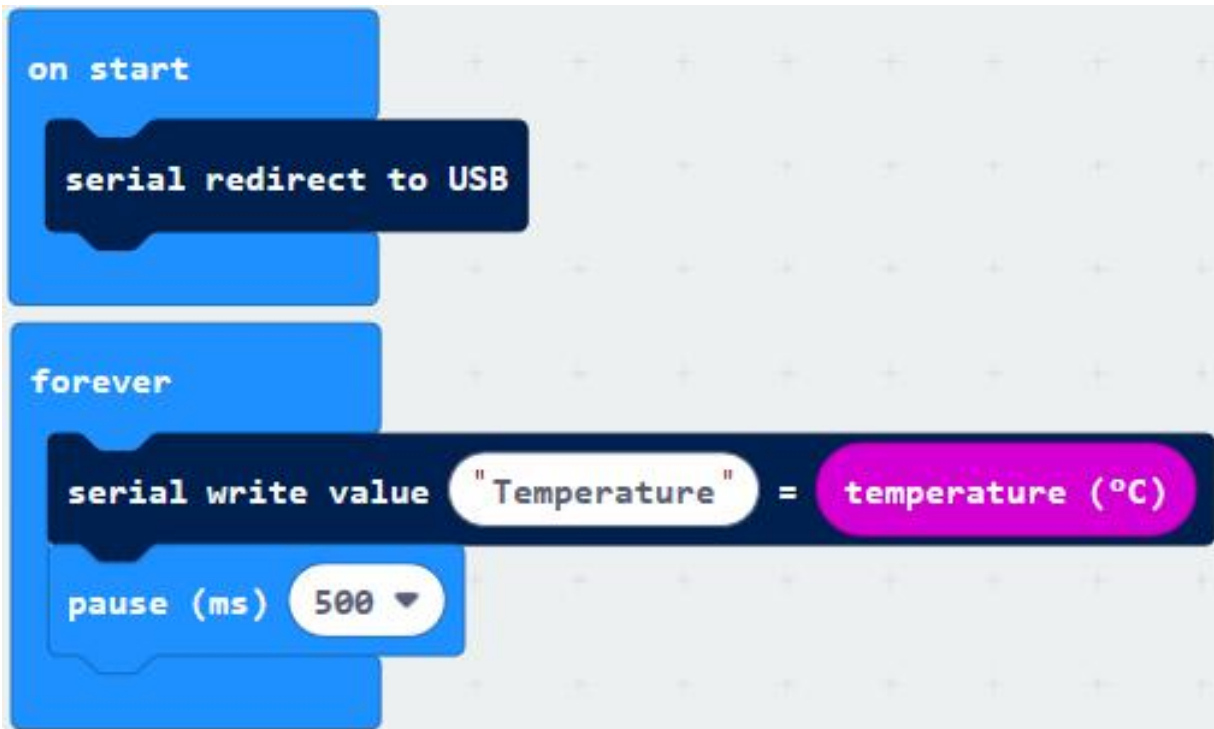
Click "Input" → "temperature(°C)" into "into serial write value "x"
=0 and change" 0" into "temperature"



Go to "Basic" → "pause (ms) 100" into "forever" and set pause to 500

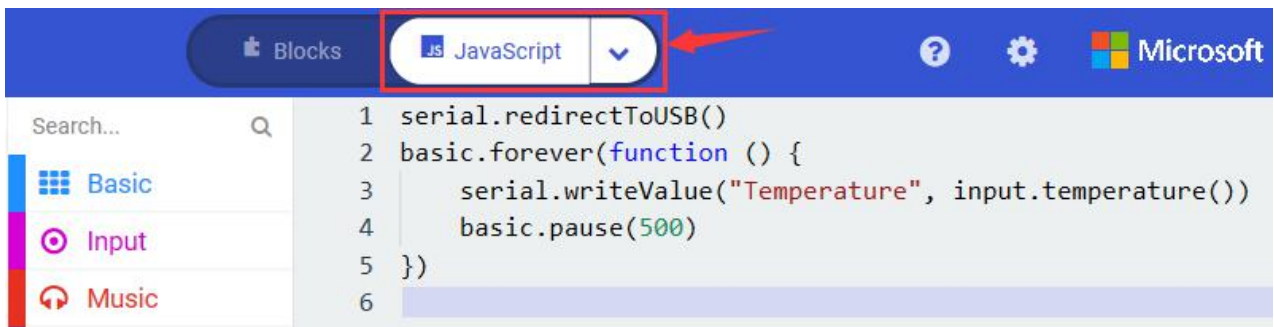


Complete Program:



- ① In "on start" the program only runs once;
- ② Redirect serial to USB;
- ③ In "forever" the program runs cyclically;
- ④ The serial writes the temperature value detected by the sensor;
- ⑤ Delay in 50ms.

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



The screenshot shows the Microsoft MakeCode IDE interface. At the top, the 'Blocks' menu is open, and the 'JavaScript' option is selected, highlighted with a red box and a red arrow. The main editor area displays the following JavaScript code:

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

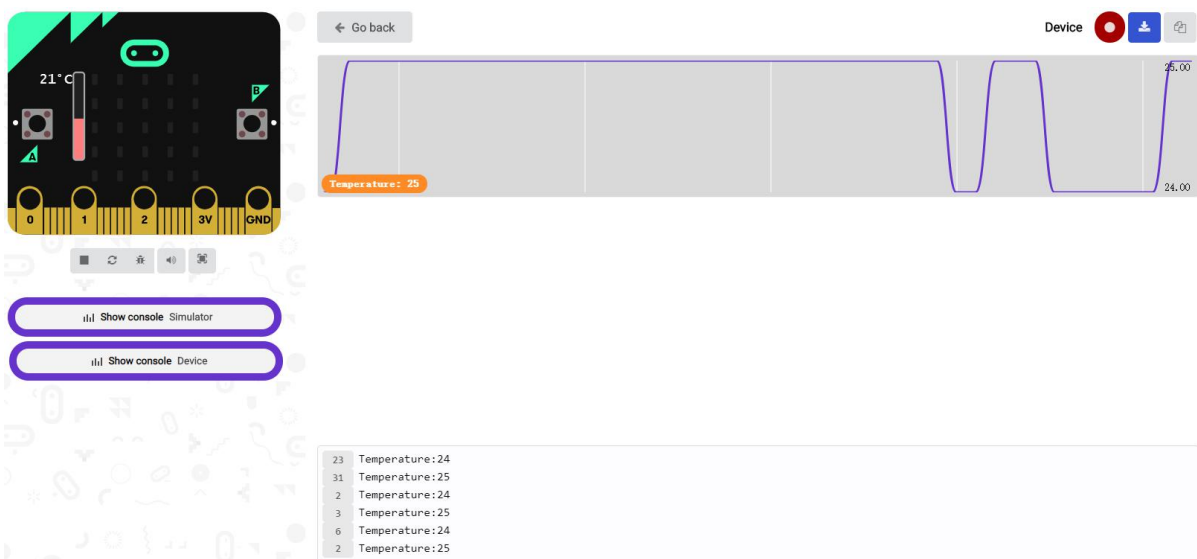


The screenshot shows the Microsoft MakeCode IDE interface. At the top, the 'Blocks' menu is open, and the 'Python' option is selected, highlighted with a red box and a red arrow. The main editor area displays the following Python code:

```
1 serial.redirect_to_usb()  
2  
3 def on_forever():  
4     serial.write_value("Temperature", input.temperature())  
5     basic.pause(500)  
6 basic.forever(on_forever)  
7
```

(4) Test Results 1:

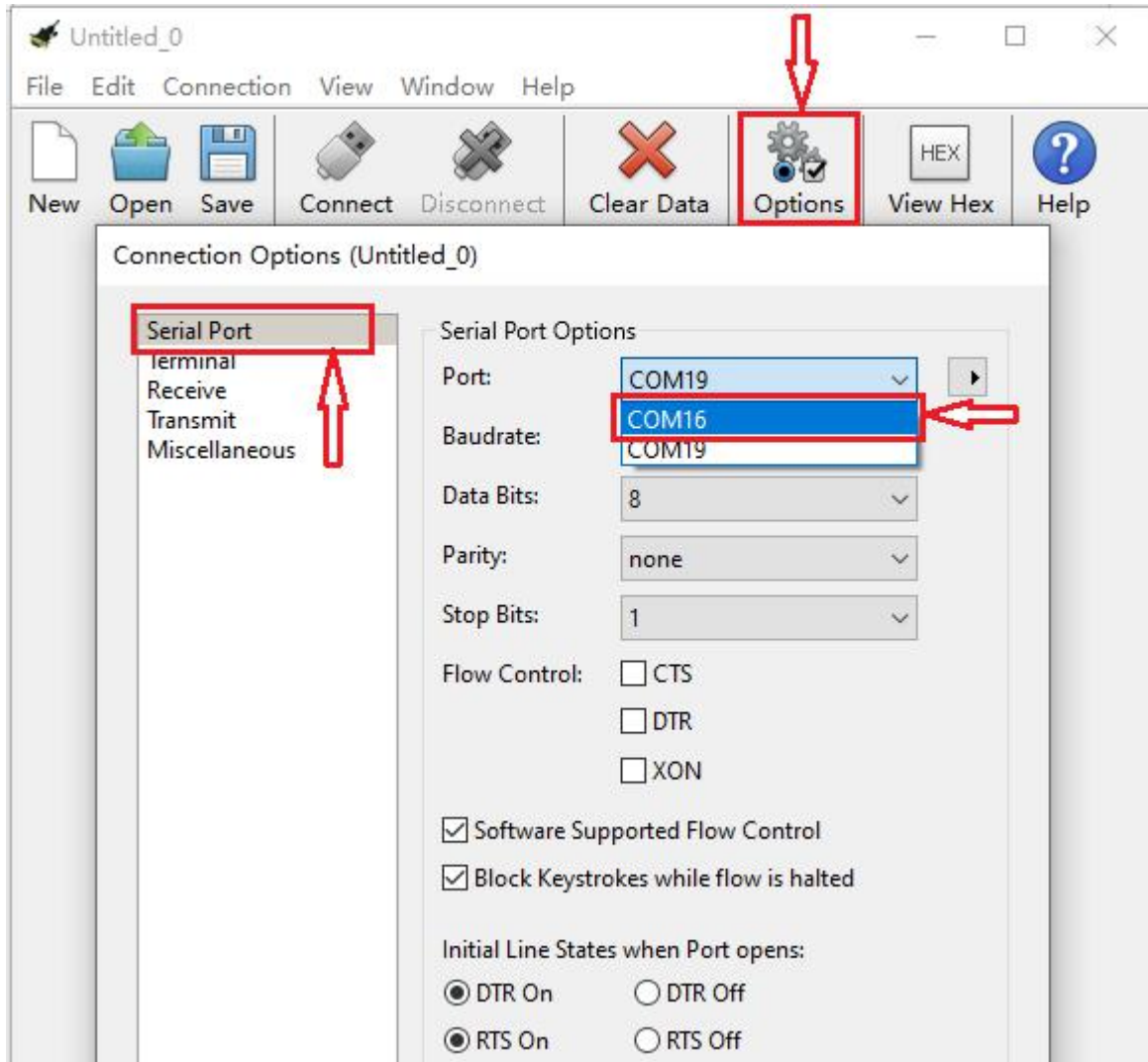
Upload test code 1 to micro:bit V2, attach it to a computer and click "Show console Device". Then the data of temperature is displayed in the serial monitor page as shown below.

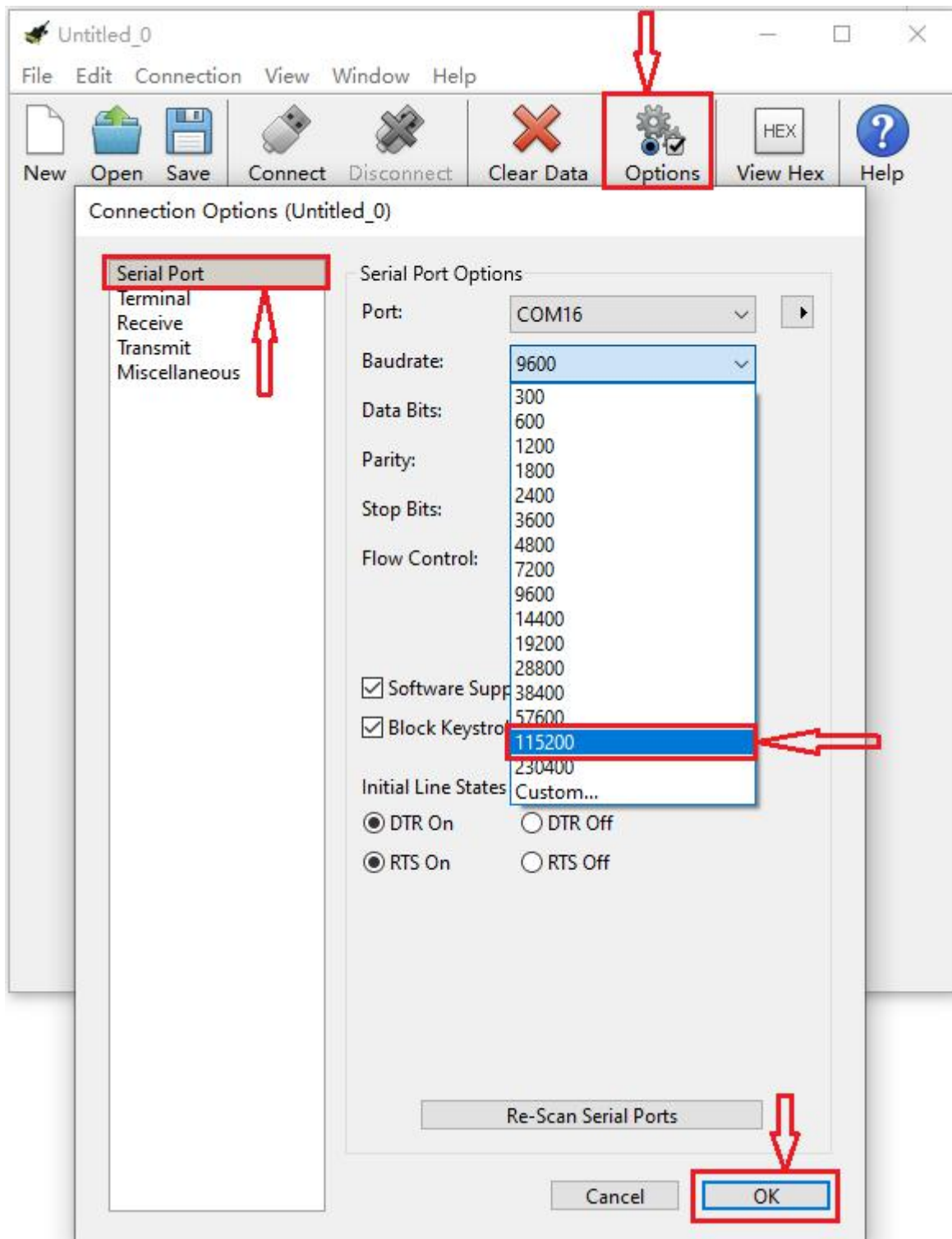


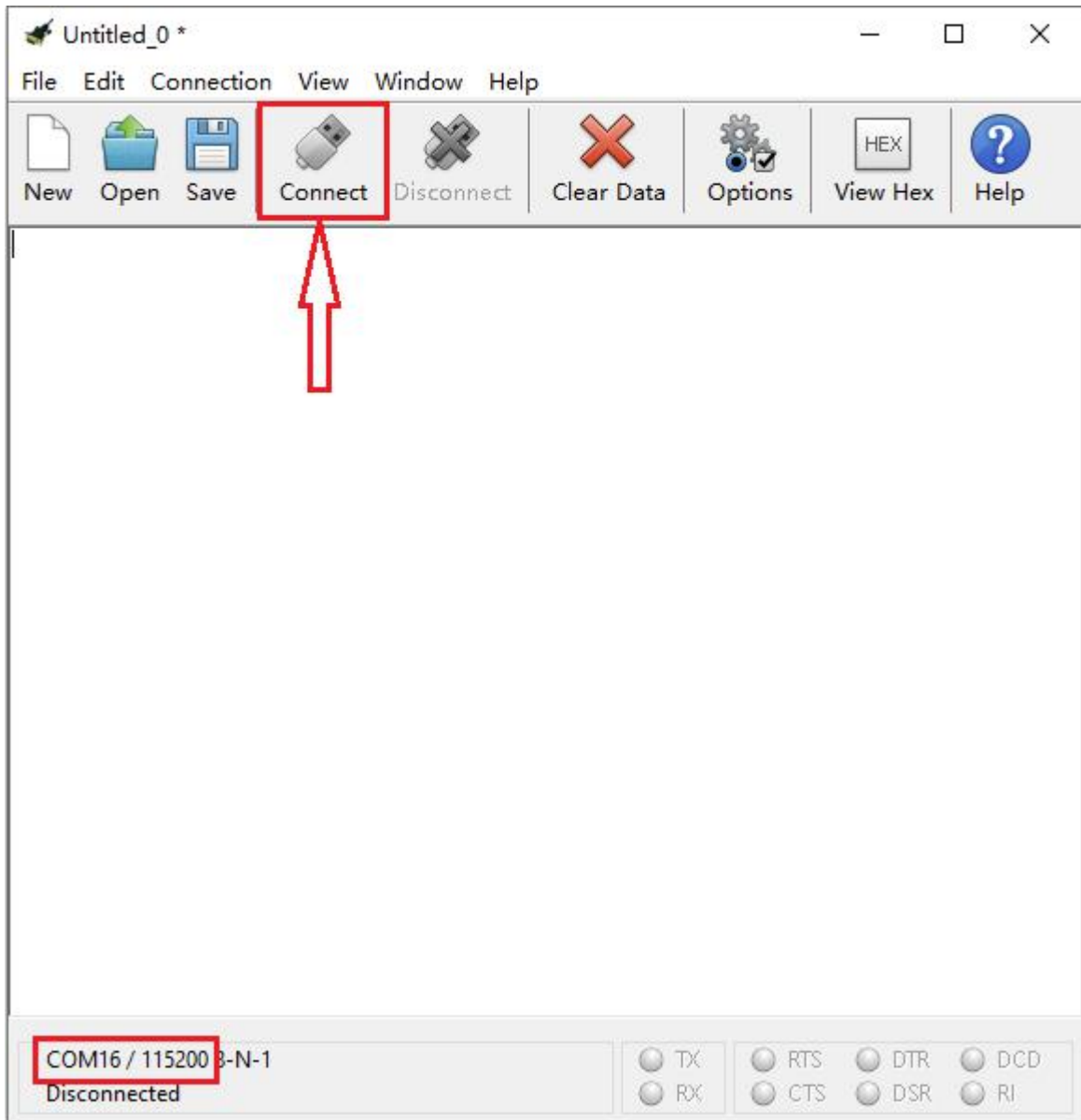
Only on Windows 10 can Google Chrome match with devices. If you use other Windows systems, the CoolTerm serial monitor is best choice for reading data.

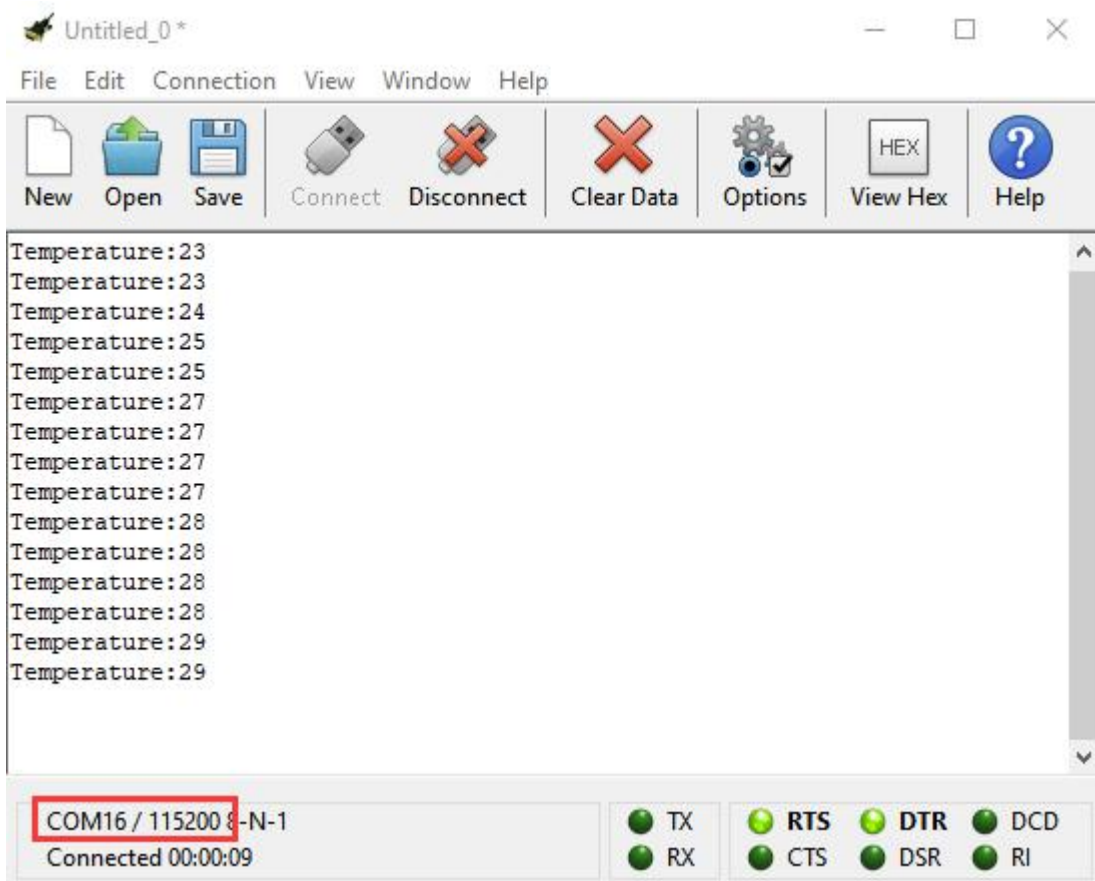
Open CoolTerm software, click Options, select SerialPort, set COM port and baud rate to 115200 (after testing, the baud rate of USB SerialPort communication on Micro:bit V2 is 115200), click OK, and Connect.

Then the CoolTerm serial monitor shows the change of temperature value in the current environment, as shown in the figures below :









Test Code 2:

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

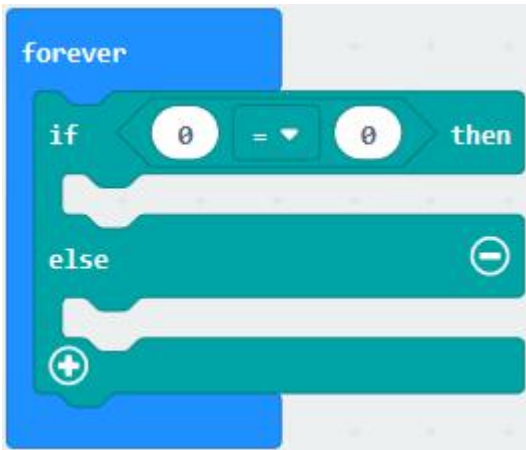
(1) A. Go to “Led” → “more” → “led enable false” block,

B. Keep it into the “on start” block, tap the triangle button to select

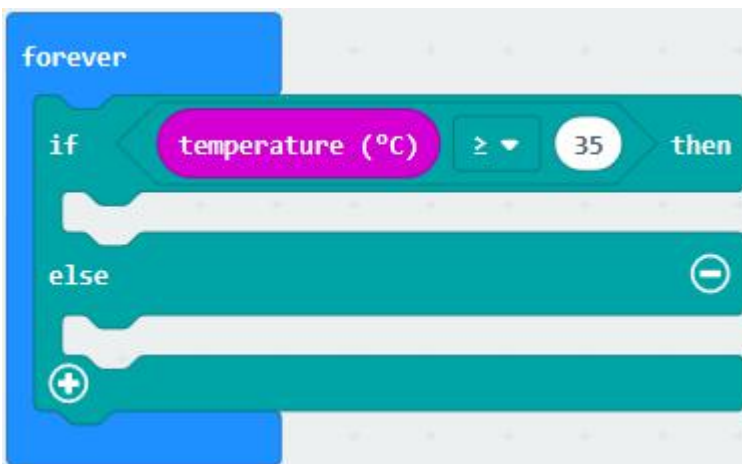




(2) Tap “Logic” and drag “if...then...else” into “forever” block; and then

drag "=" into "true"



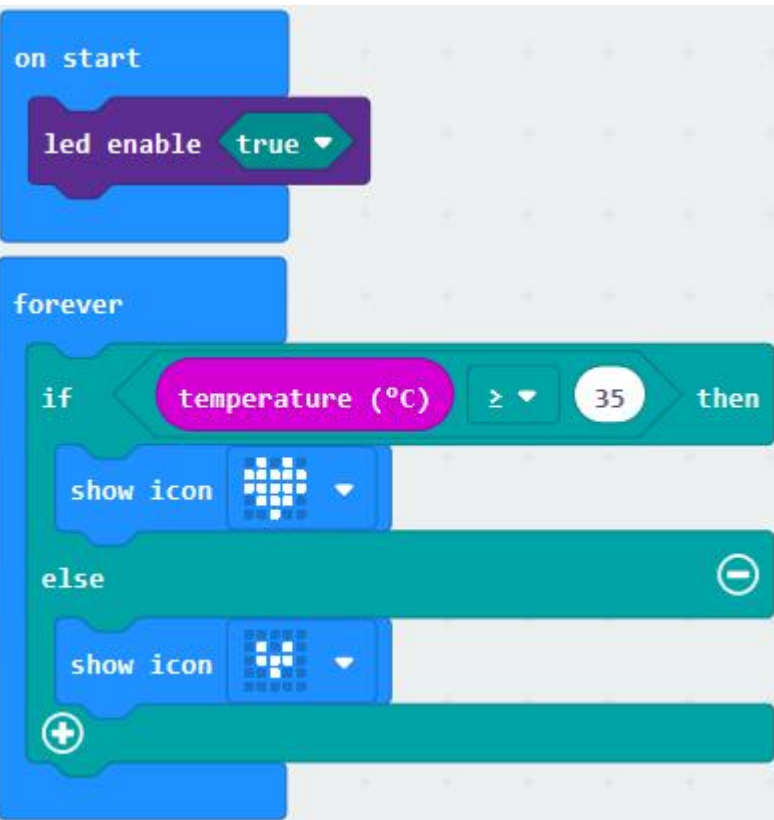
(3) Enter "Input" to move "temperature(°C)" into the left side of "="; click the little triangle of "=" to choose "≥", and change the "0" to "35"





(4) Click "Basic" to find out block "show icon" and move it into "then"; copy and place the block "show icon" to "else" and click the little triangle of  to select 

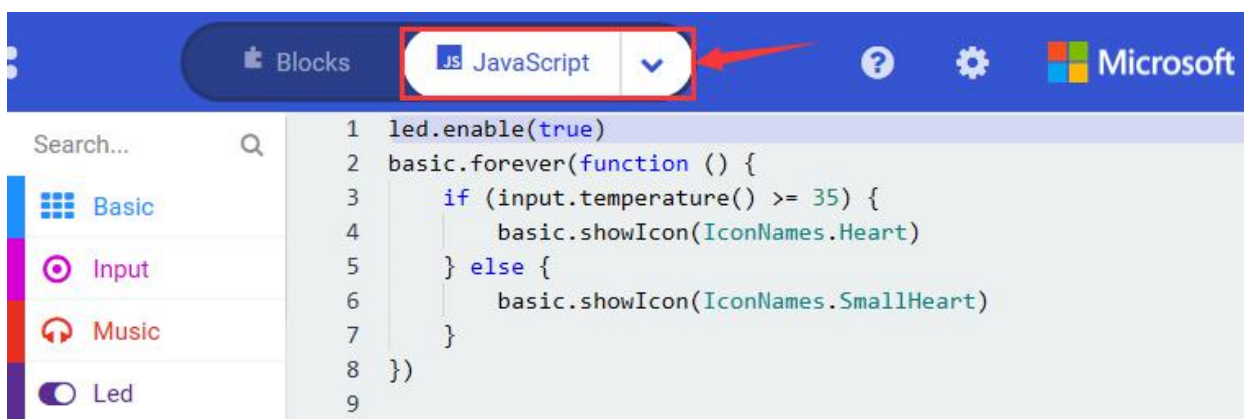


Complete Program:



- ① In "on start" the program only runs once;
- ② Open the LED dot matrix on the microbit board;
- ③ In "forever" the program runs cyclically;
- ④ When the condition that the temperature detected by the sensor is $\geq 35^{\circ}\text{C}$ is true, the program in then runs;
- ⑤ LED dot matrix displays pattern "";
- ⑥ When the condition that the temperature detected by the sensor is $\geq 35^{\circ}\text{C}$ is not true, the program in else runs;
- ⑦ LED dot matrix displays pattern "".

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



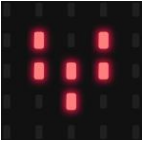
```

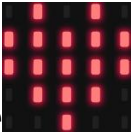
1 led.enable(True)
2
3 def on_forever():
4     if input.temperature() >= 35:
5         basic.show_icon(IconNames.HEART)
6     else:
7         basic.show_icon(IconNames.SMALL_HEART)
8 basic.forever(on_forever)
9

```

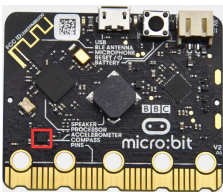
(6) Test Results 2:

Upload the code 2. When the ambient temperature is less than 35°C,

5*5 LED will show . When the temperature is equivalent to or

greater than 35°C, the image  will appear.

Project 6: Geomagnetic Sensor



(1) Project Description:

This project aims to explain the use of the micro: bit geomagnetic

sensor, which can not only detect the strength of the geomagnetic field, but also be used as a compass to find bearings. It is also an important part of the navigation attitude reference system (AHRS). Micro: Bit main board V2 uses LSM303AGR geomagnetic sensor, and the dynamic range of magnetic field is ± 50 gauss. In the board, the magnetometer module is used in both magnetic detection and compass. In this experiment, the compass will be introduced first, and then the original data of the magnetometer will be checked. The main component of a common compass is a magnetic needle, which can be rotated by the geomagnetic field and point toward the geomagnetic North Pole (which is near the geographic South Pole) to determine direction.

(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code 1:

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

(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"

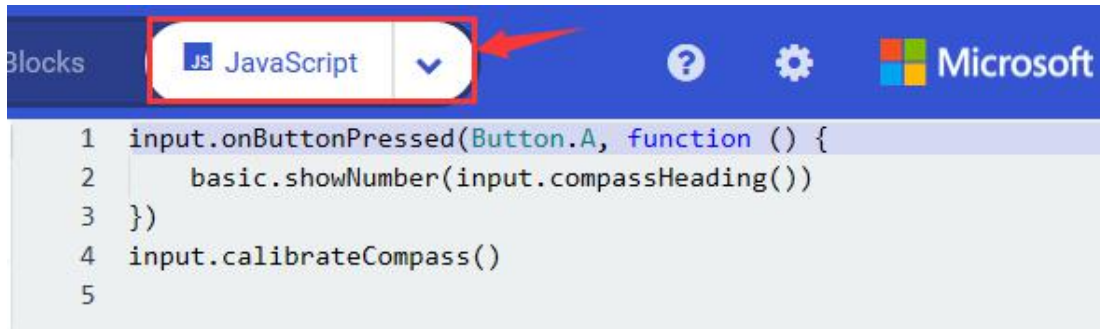


Complete Program:

	<ol style="list-style-type: none"> ① "on start": command block only runs once to start program. ② Calibrate compass ③ Press button A on Micro:bit main board ④ Dot matrix shows the direction of compass heading
--	--

Select "JavaScript" and "Python" to switch into JavaScript and Python

language code:



```

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

```



```

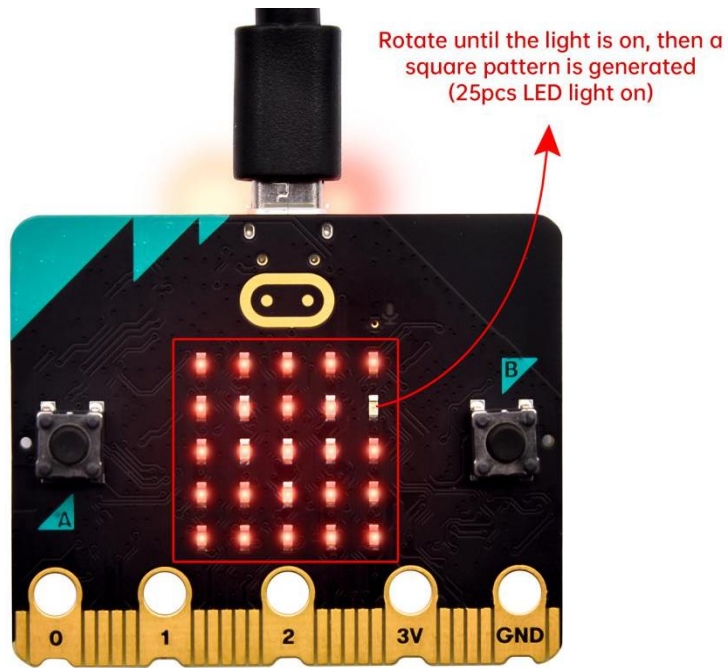
1 def on_button_pressed_a():
2     basic.show_number(input.compass_heading())
3 input.on_button_pressed(Button.A, on_button_pressed_a)
4
5 input.calibrate_compass()


```

(4) Test Results 1:

Upload test code to micro:bit V2, plug it to a power, and press the button A. The board will send request to calibrate compass and show "TILT TO FILL SCREEN" ; then enter the calibration page.

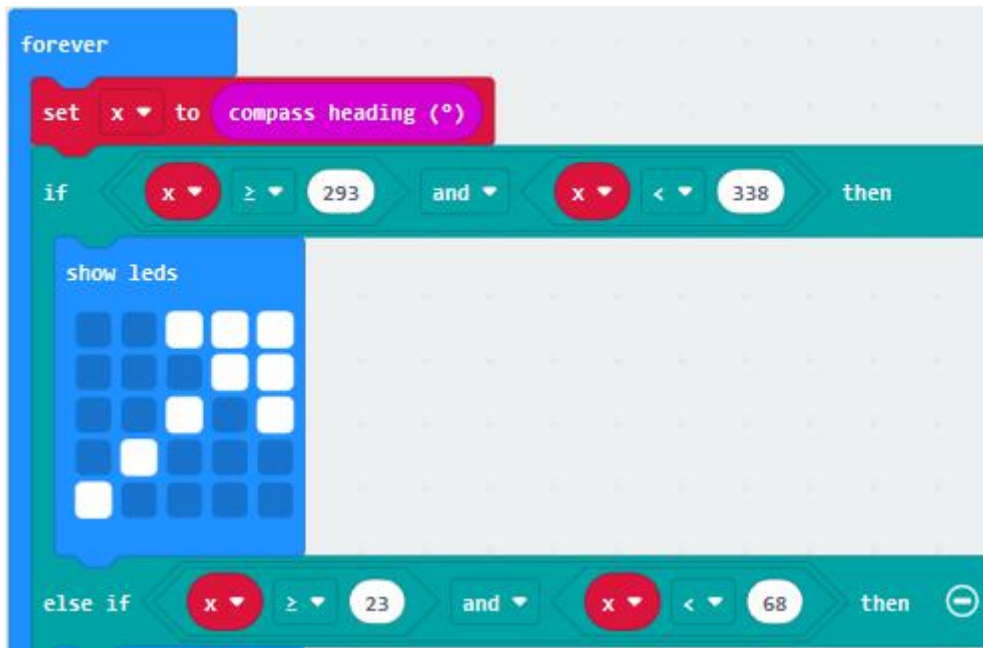
Rotate the board until all 25 LEDs are fully on, as shown below.



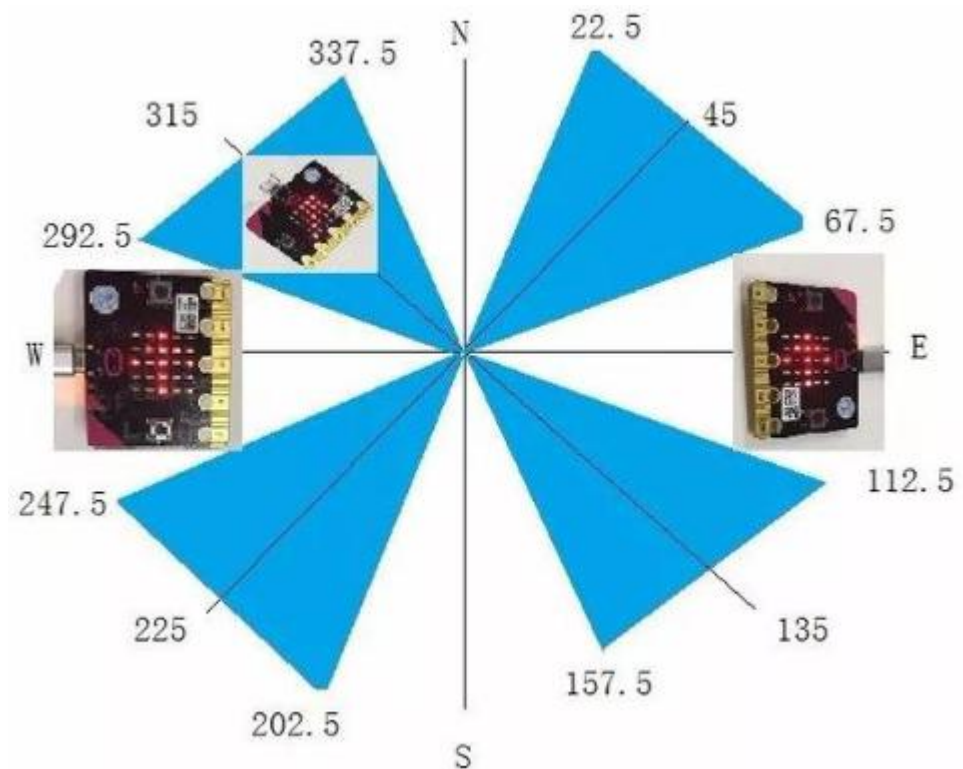
After that, a smile pattern  appears, which implies that the calibration is done. When the calibration process is completed, the reading value will be displayed if you press the button A.

And the direction north, east, south and west correspond to 0° , 90° , 180° and 270° .

(5) Test Code 2:



This module can keep readings and determine direction



For the above picture, the arrow pointing to the upper left when the value ranges from 292.5 to 337.5.

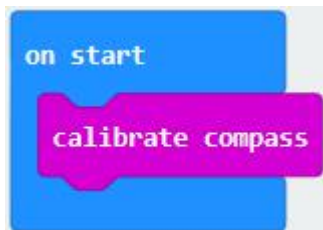
0.5 can't be input in the code, thereby, the values we get are 293 and 338.

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

(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



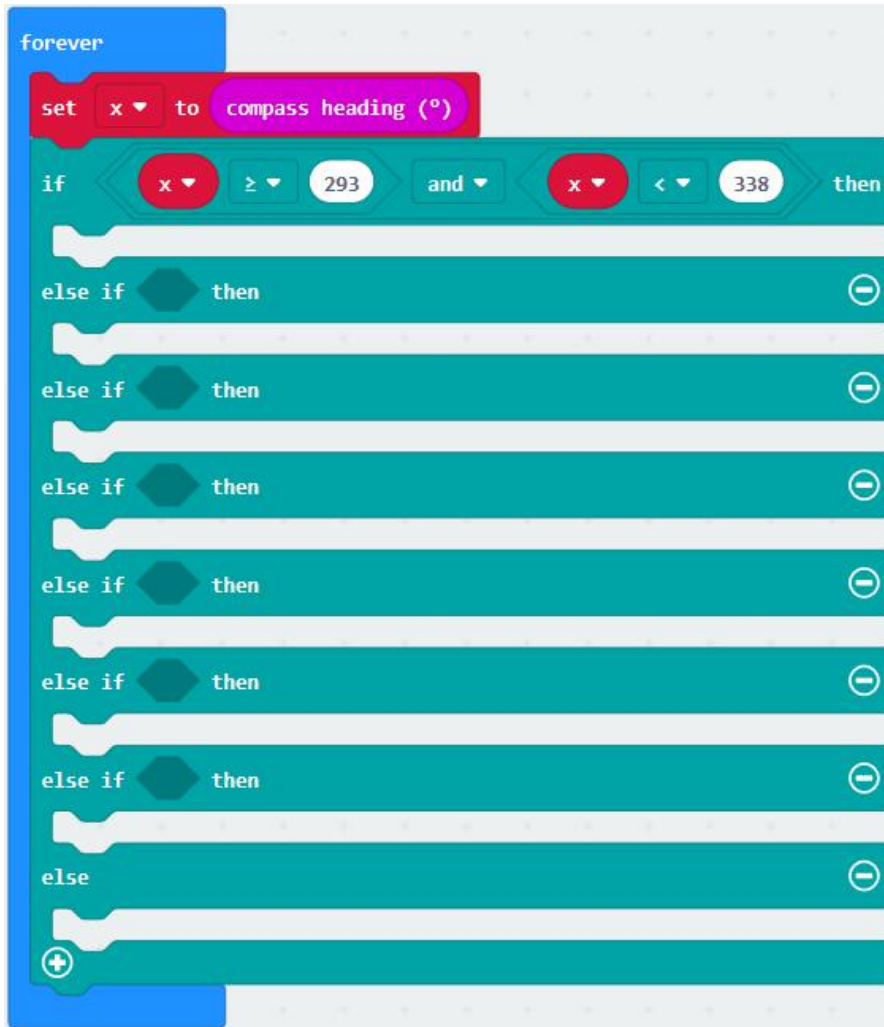
C. Tap "Logic" → "if...then...else" , leave it below block "set x to compass heading" , then click  icon for 6 times.

(4) A. Place "and" into "true" block

B. Then move "=" block to the left box of "and"

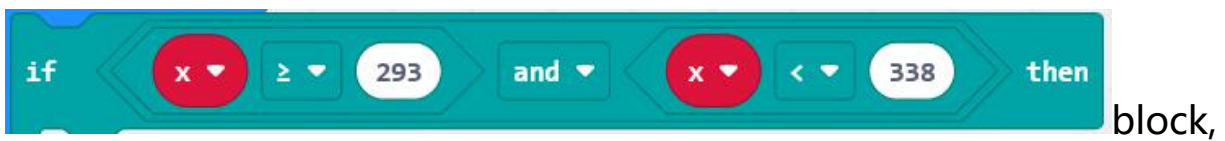
C. Click "Variables" to drag "x" to the left "0" box, change 0 into 293 and set to "≥" ;


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

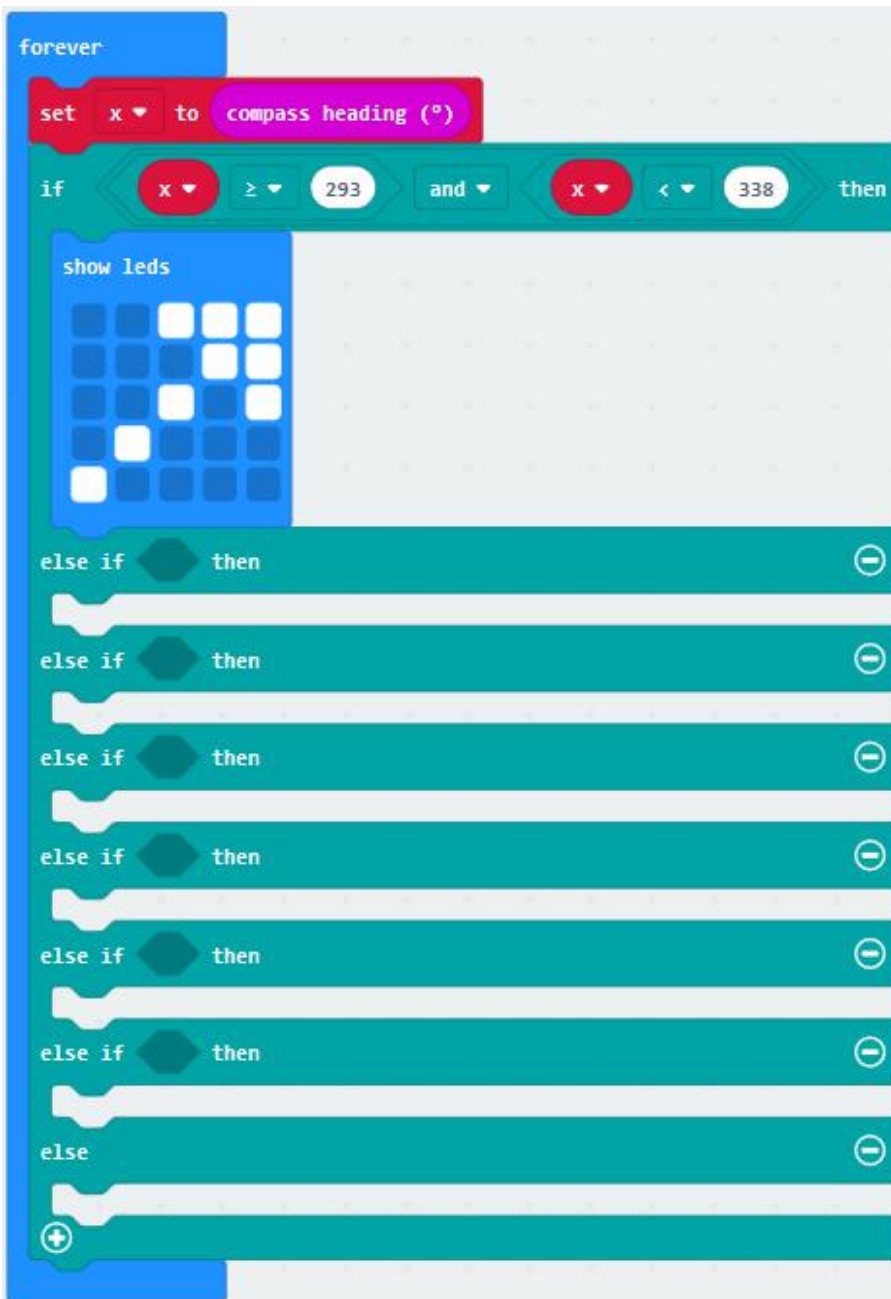



(5) A. Go to "Basic" → "show leds"








B. Lay it down beneath



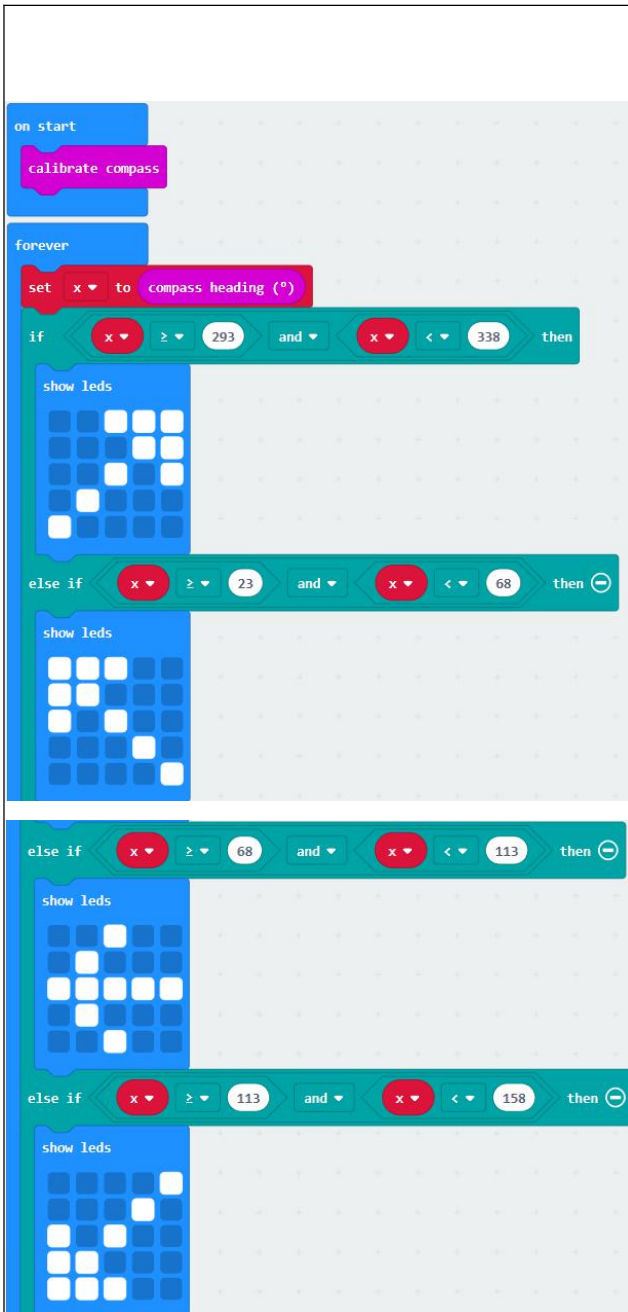
then click "show leds" and the pattern  appears.



- (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”: 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

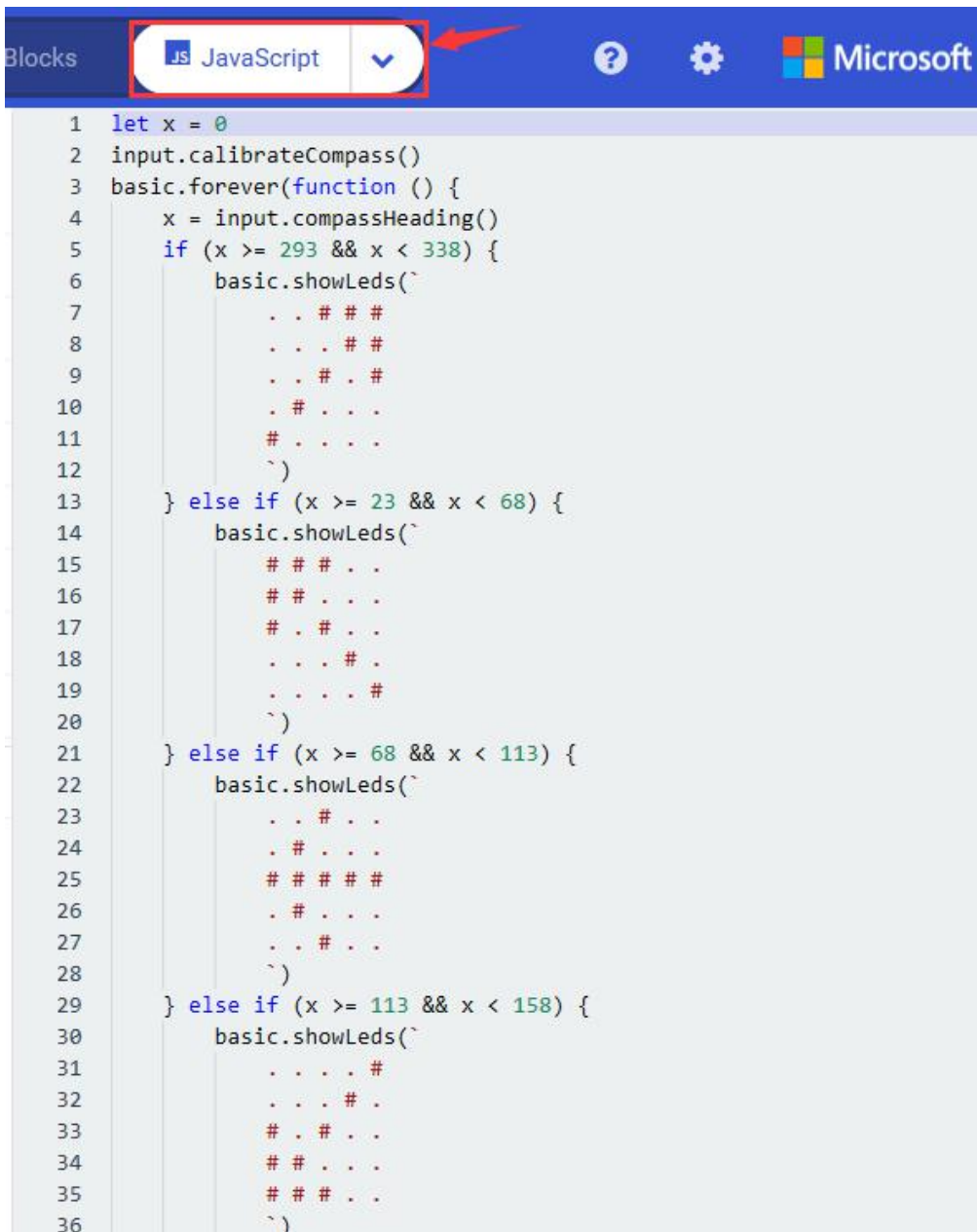
When $203 \leq x < 248$, the next program will be executed.
Dot matrix displays

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

Select "JavaScript" and "Python" to switch into JavaScript and Python

language code:



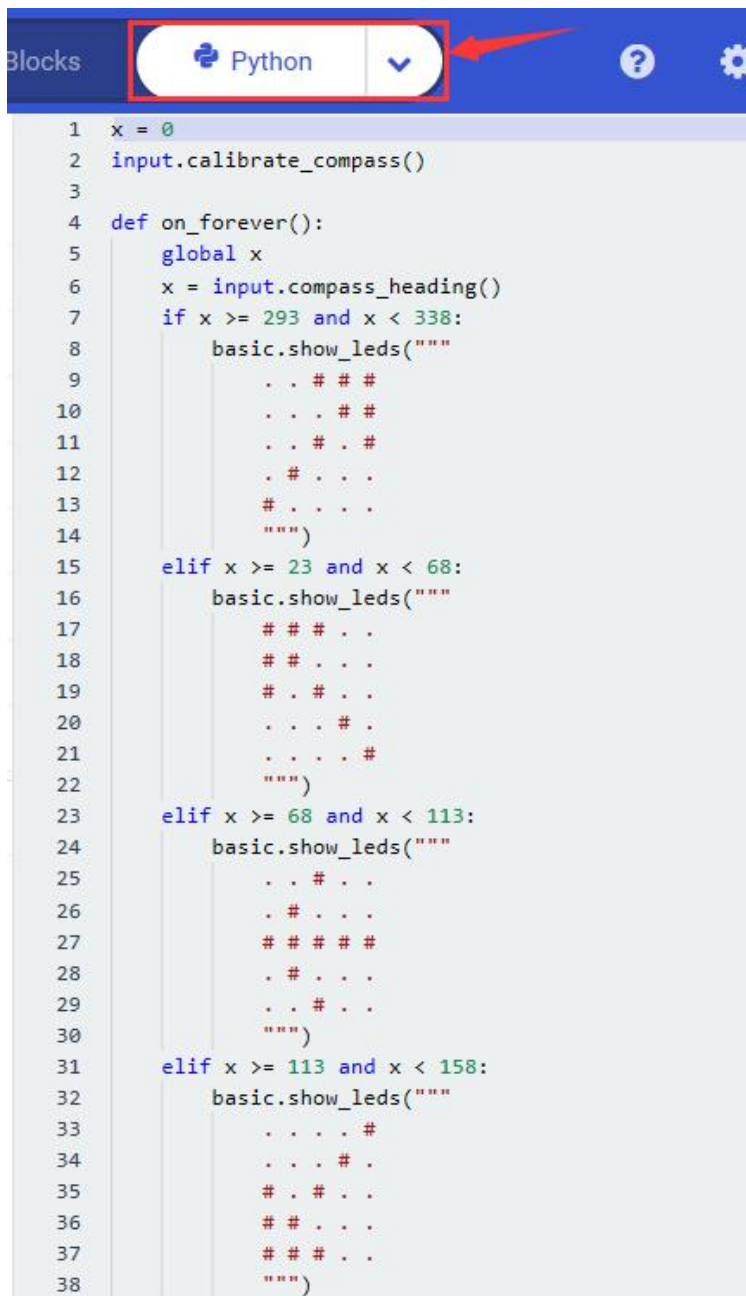
```

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
```



```
1 x = 0
2 input.calibrate_compass()
3
4 def on_forever():
5     global x
6     x = input.compass_heading()
7     if x >= 293 and x < 338:
8         basic.show_leds("""
9             . . # # #
10            . . . # #
11            . . # . #
12            . # . . .
13            # . . . .
14            """)
15     elif x >= 23 and x < 68:
16         basic.show_leds("""
17             # # # . .
18             # # . . .
19             # . # . .
20             . . . # .
21             . . . . #
22             """)
23     elif x >= 68 and x < 113:
24         basic.show_leds("""
25             . . # . .
26             . # . . .
27             # # # # #
28             . # . . .
29             . . # . .
30             """)
31     elif x >= 113 and x < 158:
32         basic.show_leds("""
33             . . . . #
34             . . . # .
35             # . # . .
36             # # . . .
37             # # # . .
38             """)
```

```

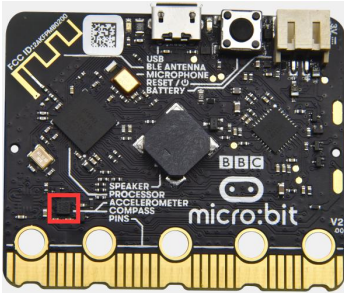
39 elif x >= 158 and x < 203:
40     basic.show_leds("""
41         . . # . .
42         . . # . .
43         # . # . #
44         . # # # .
45         . . # . .
46         """)
47 elif x >= 203 and x < 248:
48     basic.show_leds("""
49         # . . . .
50         . # . . .
51         . . # . #
52         . . . # #
53         . . # # #
54         """)
55 elif x >= 248 and x < 293:
56     basic.show_leds("""
57         . . # . .
58         . . . # .
59         # # # # #
60         . . . # .
61         . . # . .
62         """)
63 else:
64     basic.show_leds("""
65         . . # . .
66         . # # # .
67         # . # . #
68         . . # . .
69         . . # . .
70         """)
71 basic.forever(on_forever)
72

```

(6) Test Results 2

Upload code 2 and plug the micro:bit to a power. After calibrating and tilting the micro:bit V2, the LED dot matrix will display the direction signs.

Project 7: Accelerometer



(1) Project Description:

The Micro: bit V2 has a built-in LSM303AGR gravity acceleration sensor, also known as accelerometer, with a resolution of 8/10/12 bits. The code section sets the range to 1g, 2g, 4g, and 8g.

We often use accelerometer to detect the status of machines.

In this project, we will introduce how to measure the position of the board with the accelerometer. And then have a look at the original three-axis data output by the accelerometer.

(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code 1:

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

(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 logo 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 image shows four Scratch code blocks stacked vertically. Each block is a purple 'on' event block with a blue 'show number' block attached. The events are 'screen down', 'tilt left', 'tilt right', and 'free fall'. The numbers shown are 5, 6, 7, and 8 respectively.

Event	Number Shown
on screen down	5
on tilt left	6
on tilt right	7
on free fall	8

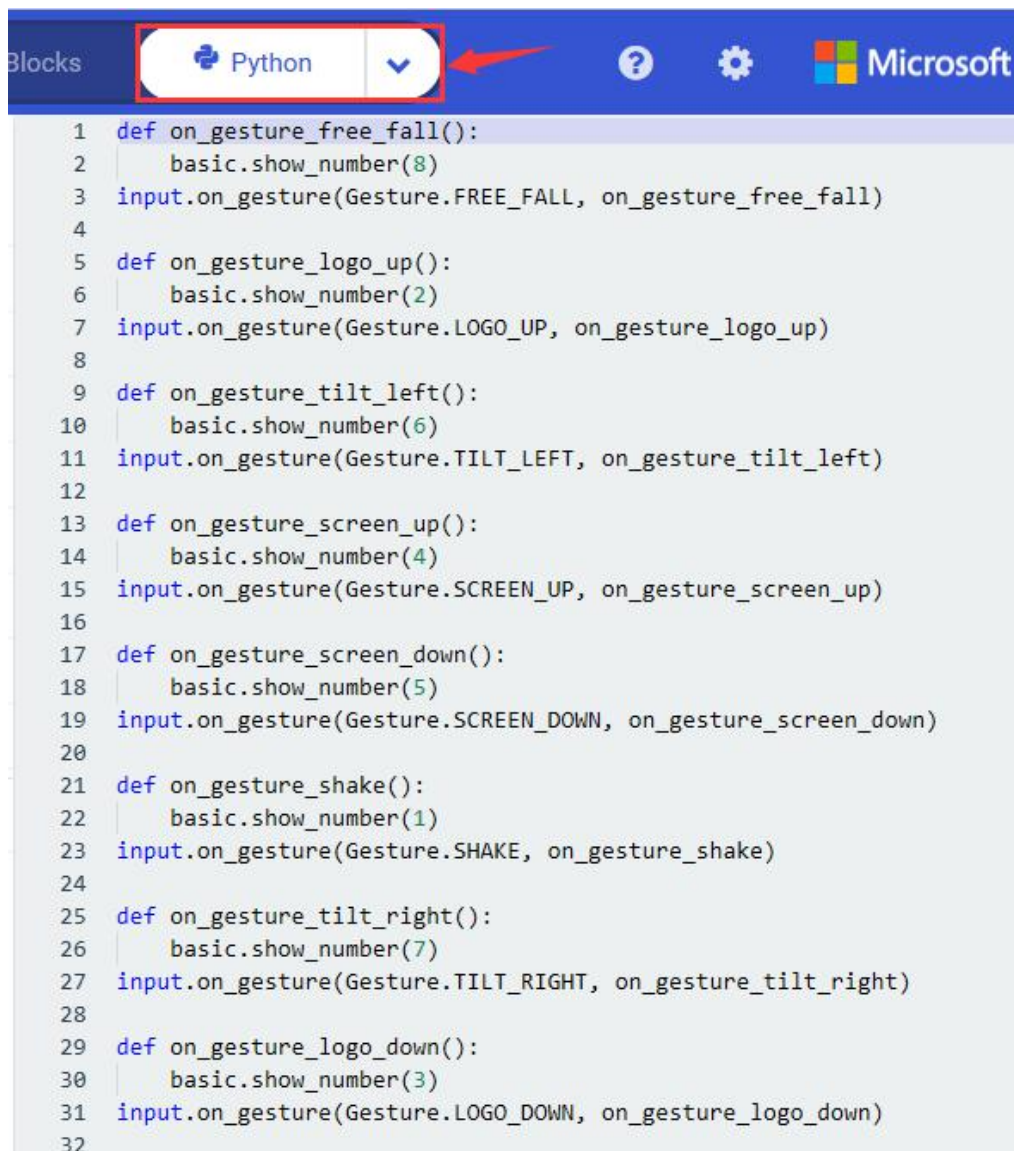
The screen is down
Number 5 is shown
The Micro:bit board is tilt to the left
Number 6 is displayed
The Micro:bit board is tilt to the right
Number7 is displayed
When the Micro:bit board is free fall
LED dot matrix shows 8

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



The image shows a screenshot of an IDE interface. At the top, there is a blue header bar. On the left side of the header, the word "locks" is partially visible. In the center of the header, there is a language selector dropdown menu. The dropdown is currently set to "JavaScript" with a small blue icon to its left and a downward-pointing chevron to its right. A red rectangular box highlights the entire dropdown menu, and a red arrow points from the right towards the chevron. To the right of the dropdown are two icons: a question mark and a gear. Below the header, the main area of the IDE contains a code editor with the following JavaScript code:

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

```
1 def on_gesture_free_fall():
2     basic.show_number(8)
3 input.on_gesture(Gesture.FREE_FALL, on_gesture_free_fall)
4
5 def on_gesture_logo_up():
6     basic.show_number(2)
7 input.on_gesture(Gesture.LOGO_UP, on_gesture_logo_up)
8
9 def on_gesture_tilt_left():
10    basic.show_number(6)
11 input.on_gesture(Gesture.TILT_LEFT, on_gesture_tilt_left)
12
13 def on_gesture_screen_up():
14    basic.show_number(4)
15 input.on_gesture(Gesture.SCREEN_UP, on_gesture_screen_up)
16
17 def on_gesture_screen_down():
18    basic.show_number(5)
19 input.on_gesture(Gesture.SCREEN_DOWN, on_gesture_screen_down)
20
21 def on_gesture_shake():
22    basic.show_number(1)
23 input.on_gesture(Gesture.SHAKE, on_gesture_shake)
24
25 def on_gesture_tilt_right():
26    basic.show_number(7)
27 input.on_gesture(Gesture.TILT_RIGHT, on_gesture_tilt_right)
28
29 def on_gesture_logo_down():
30    basic.show_number(3)
31 input.on_gesture(Gesture.LOGO_DOWN, on_gesture_logo_down)
32
```

(4) Test Results 1:

Uploading the test code 1 to micro:bit V2 and powering it on via the USB cable. If we shake the micro: bit V2. the LED dot matrix will display "1" .

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



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

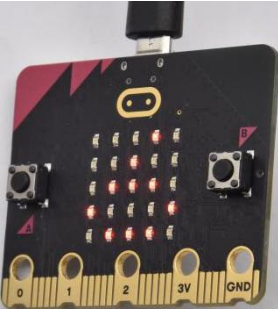


When it is placed still on the desk, showing its front side, the number 4 appears.



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

When the board is tilted to the left , the LED dot matrix shows the number 6 as shown below.



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



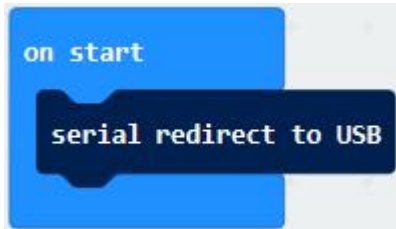
When the board is knocked to the floor, this process can be considered as a free fall and the LED dot matrix shows the number 8. (please note that this test is not recommended for it may damage the main board.)

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

(5) Test Code 2:

(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

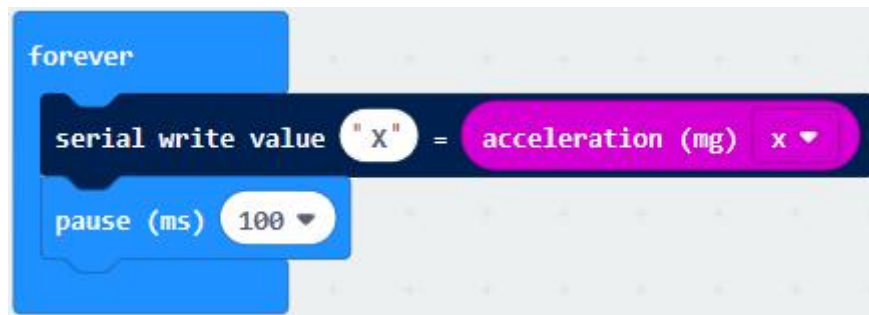


(3) A. Click "Input" → "acceleration(mg) x" ;

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.





(5) Replicate code string

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



Complete Program:



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

Serial redirects to USB

The program under the block “forever” runs cyclically.

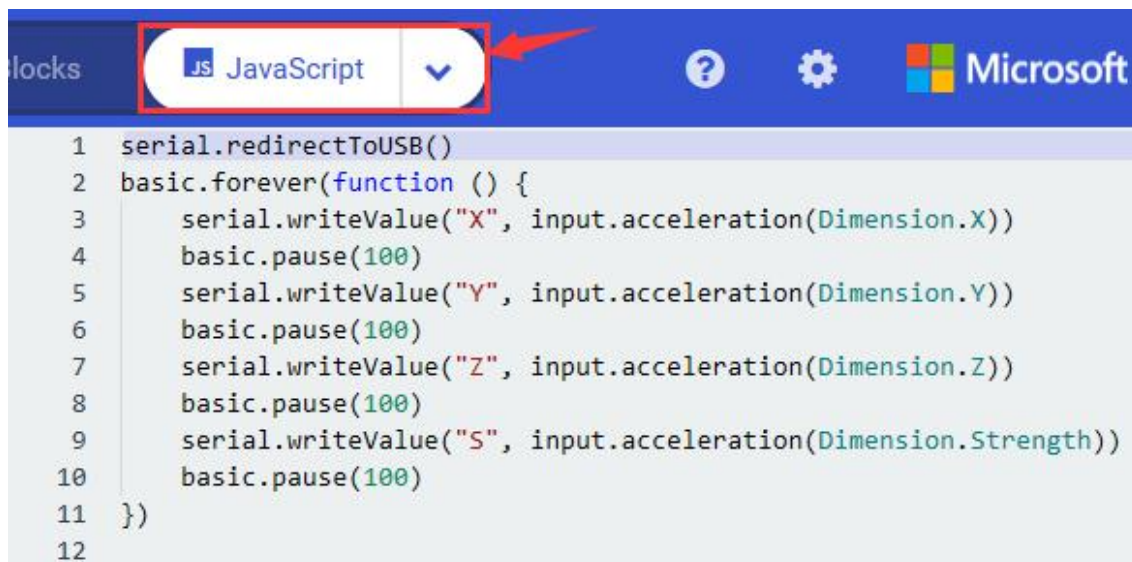
Serial write value “X”=acceleration value on x axis

Serial write value “Y”=acceleration value on y axis

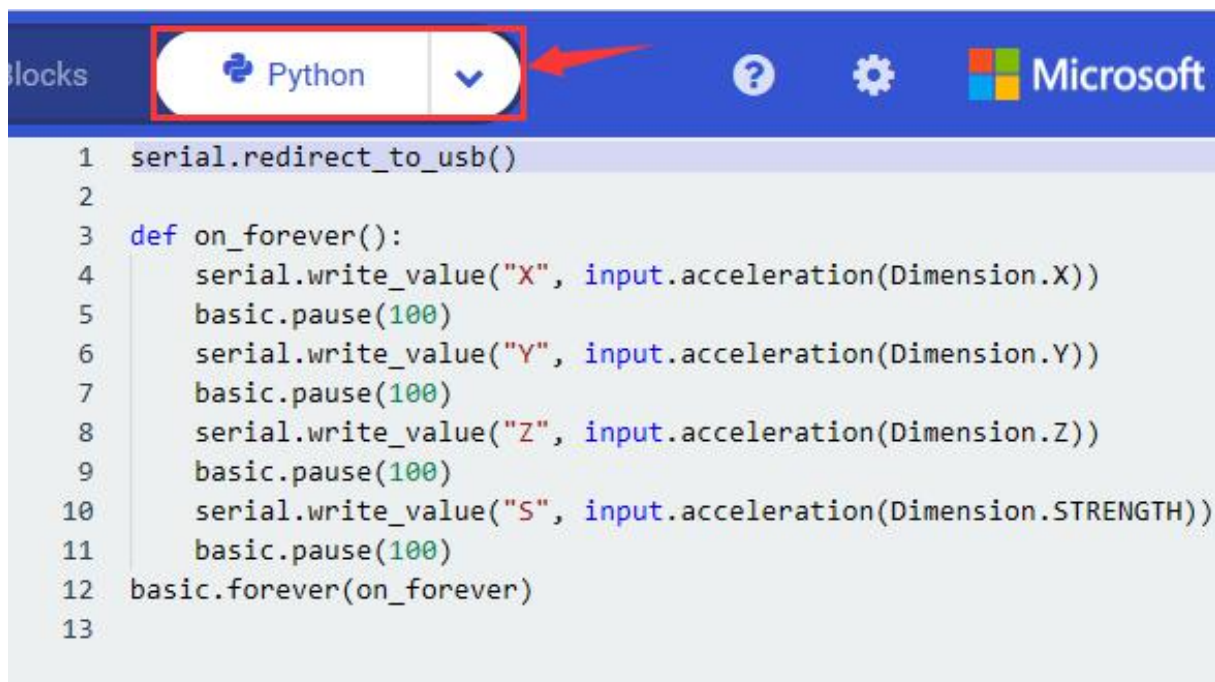
Serial write value “Z”=acceleration value on z axis

Serial write value “S”=acceleration value on s axis

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



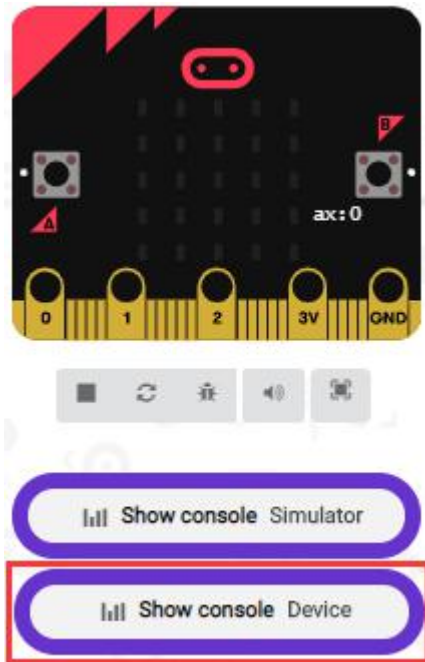
```
1 serial.redirectToUSB()  
2 basic.forever(function () {  
3     serial.writeValue("X", input.acceleration(Dimension.X))  
4     basic.pause(100)  
5     serial.writeValue("Y", input.acceleration(Dimension.Y))  
6     basic.pause(100)  
7     serial.writeValue("Z", input.acceleration(Dimension.Z))  
8     basic.pause(100)  
9     serial.writeValue("S", input.acceleration(Dimension.Strength))  
10    basic.pause(100)  
11 })  
12
```



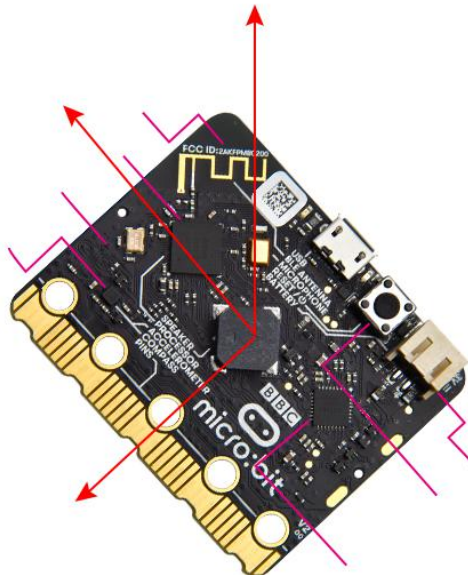
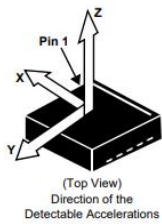
```
1 serial.redirect_to_usb()  
2  
3 def on_forever():  
4     serial.write_value("X", input.acceleration(Dimension.X))  
5     basic.pause(100)  
6     serial.write_value("Y", input.acceleration(Dimension.Y))  
7     basic.pause(100)  
8     serial.write_value("Z", input.acceleration(Dimension.Z))  
9     basic.pause(100)  
10    serial.write_value("S", input.acceleration(Dimension.STRENGTH))  
11    basic.pause(100)  
12 basic.forever(on_forever)  
13
```

(6) Test Results 2

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



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



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



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

You could open CoolTerm software, click Options, select SerialPort, set COM port and baud rate to 115200 (after testing, the baud rate of USB SerialPort communication on Micro: Bit main board V2 is 115200), click OK, and Connect. The CoolTerm serial monitor shows the data of X axis, Y axis and Z axis , as shown in the figures below :

Untitled_0*

File Edit Connection View Window Help

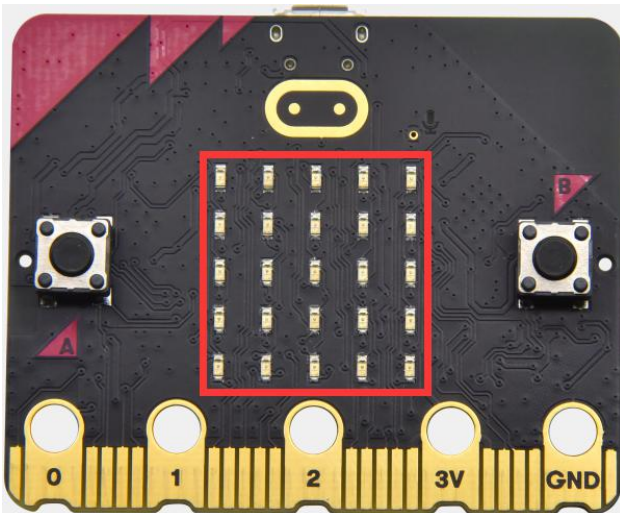
New Open Save Connect Disconnect Clear Data Options View Hex Help

S:922
X:-912
Y:864
Z:-620
S:1320
X:-280
Y:-676
Z:-296
S:1364
X:-180
Y:-836
Z:-4
S:878
X:-812
Y:-268
Z:-300
S:518
X:140
Y:-372
Z:1004
S:1108
X:-656
Y:-268
Z:-992
S:740
X:84
Y:-40

COM16 / 115200 -N-1
Connected 00:00:05

TX RX RTS CTS DTR DSR DCD RI

Project 8: Light Detection



(1) Project Description:

In this project, we focus on the light detection function of the Micro:bit main board V2. It is achieved by the LED dot matrix. And it can be viewed as a photosensor.

(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code:

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

(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"



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

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

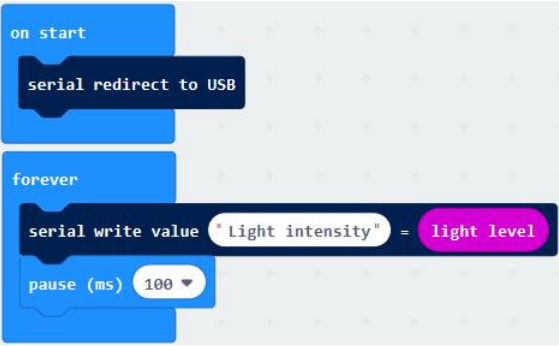


(5) A. Click "Basic" → "pause (ms) 100" ;

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



Complete Program:



The image shows a Scratch code editor with the following blocks:

- on start** block containing a **serial redirect to USB** block.
- forever** loop block containing:
 - serial write value** block with "Light intensity" in the text field and "light level" in the value field.
 - pause (ms)** block with "100" in the value field.

“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

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



The screenshot shows the Microsoft MakeCode interface with the language dropdown set to JavaScript. A red box highlights the dropdown menu, and a red arrow points to it. The code in the editor is:

```

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

```



The screenshot shows the Microsoft MakeCode interface with the language dropdown set to Python. A red box highlights the dropdown menu, and a red arrow points to it. The code in the editor is:

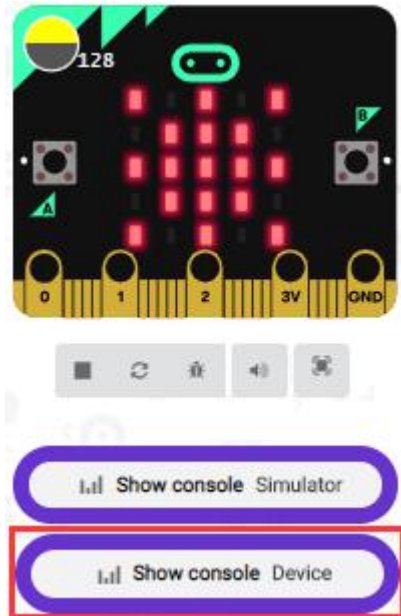
```

1 serial.redirect_to_usb()
2
3 def on_forever():
4     serial.write_value("Light intensity", input.light_level())
5     basic.pause(100)
6 basic.forever(on_forever)
7

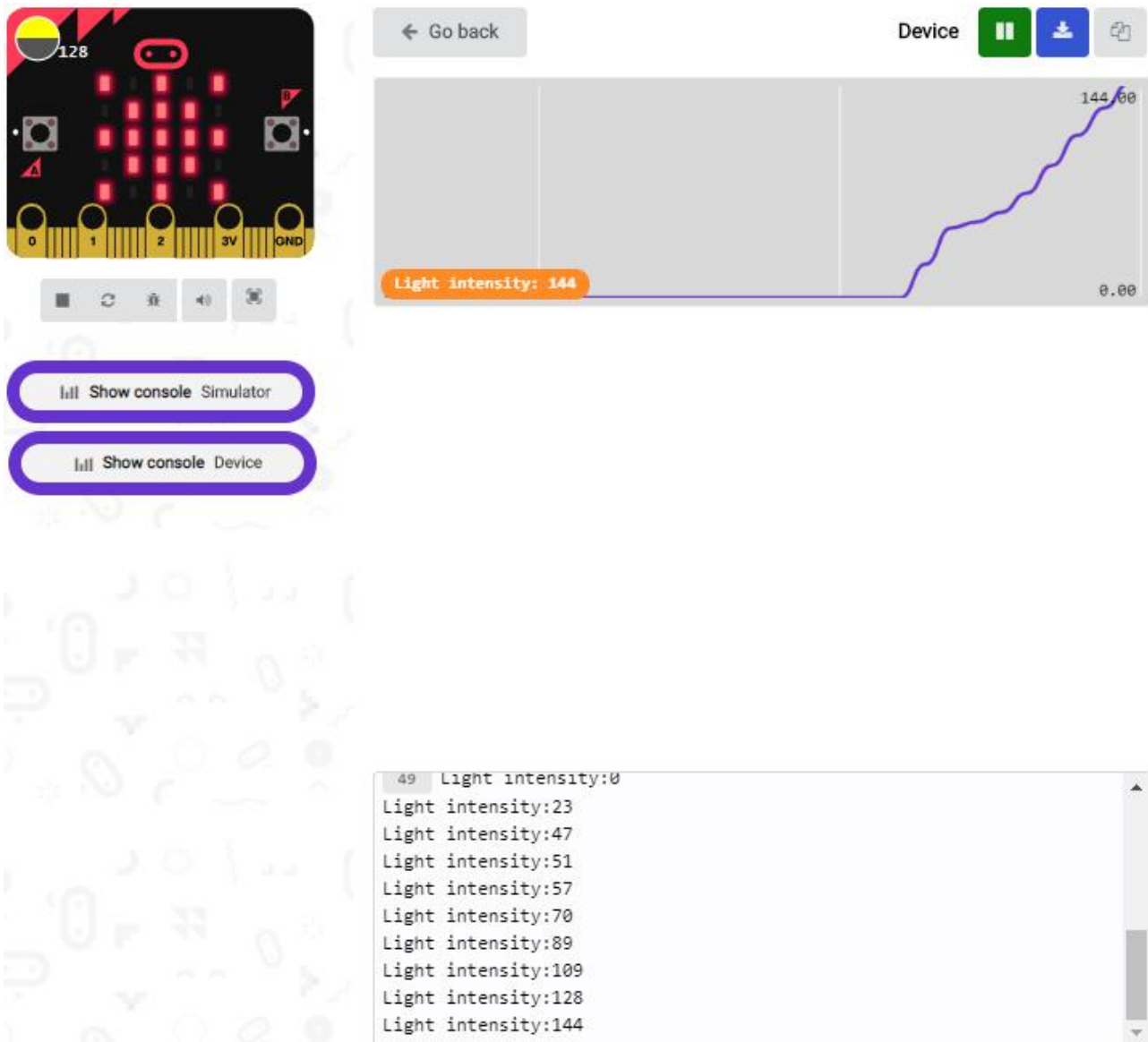
```

(4) Test Results:

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



When the LED dot matrix is covered by hand, the light intensity showed is approximately 0; when the LED dot matrix is exposed to light, the light intensity displayed gets stronger with the light as shown below.

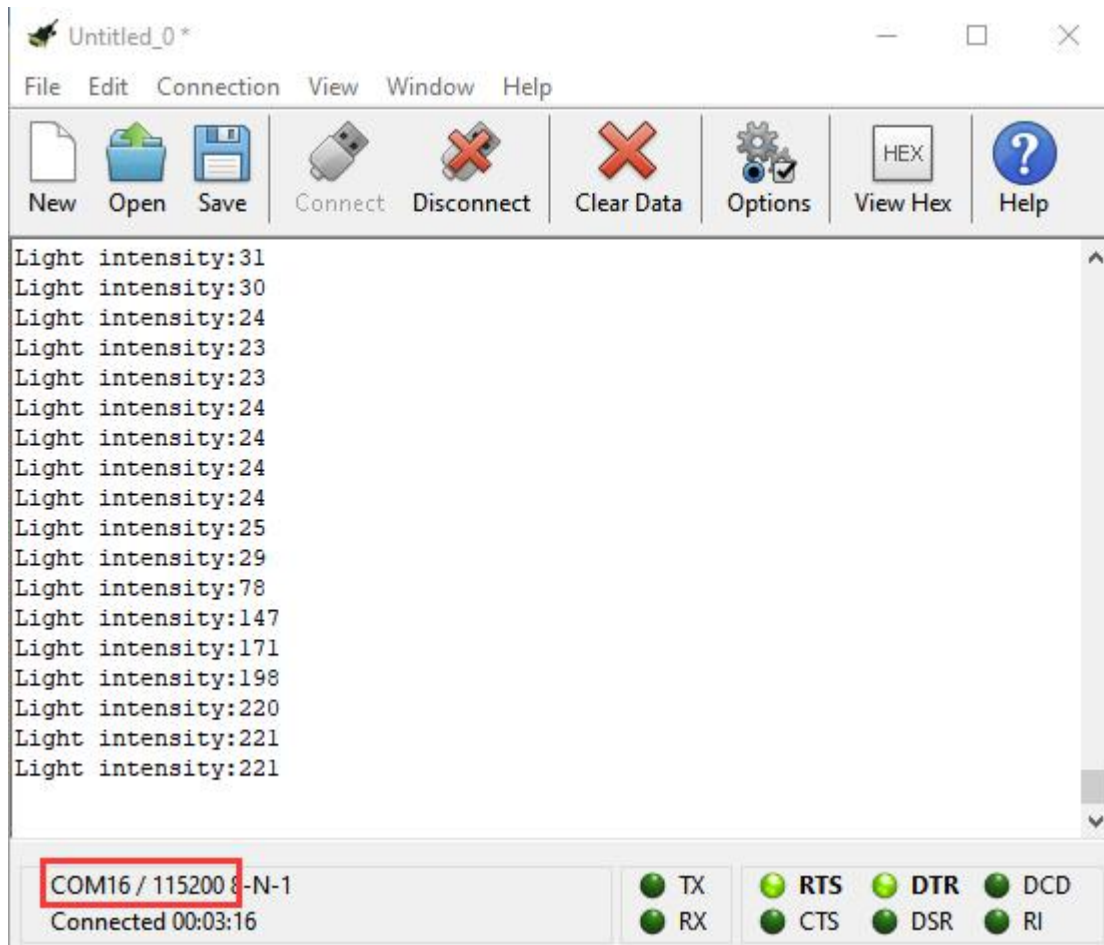


The 20 in the code is an arbitrary value of light intensity. If the current light level is less than or equal to 20, the moon will appear on the LED dot matrix. If it's bigger than 20, the sun will appear.

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

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

SerialPort communication on Micro: Bit main board V2 is 115200), click OK, and Connect. The CoolTerm serial monitor shows the value of light intensity , as shown in the figures below :



Project 9: Speaker



(1) Project Description:

The Micro: bit main board V2 has an built-in speaker, which makes adding sound to the programs easier. We can program the speaker to air all kinds of tones .

(2) Components Needed:

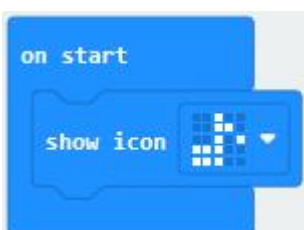
- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code:

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

(1) Enter "Basic" module to find "show icon" and drag it into "on start" block;

Click the little triangle to find 



(2) Enter "Music" module to find and drag "play sound giggle until done" into "forever" block;

Enter "Basic" module to find and drag "pause(ms) 100" into "forever" block ;

Change 100 into 1000;



(3) Copy



three times and place it into


"forever" block ;

Click the little triangle to select "happy" ," hello" ," yawn" ;



Complete Program:



- ① In "on start" the program only runs once;
- ② LED dot matrix displays pattern "  " ;
- ③ In "forever" the program runs cyclically;
- ④ The buzzer makes the sound "giggle" ;
- ⑤ Delay in 1000ms;
- ⑥ The buzzer makes the sound "happy" ;
- ⑦ Delay in 1000ms;
- ⑧ The buzzer makes the sound "hello" ;
- ⑨ Delay in 1000ms;
- ⑩ The buzzer makes the sound "yawn" ;
- ⑪ Delay in 1000ms;

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

```

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

```

```

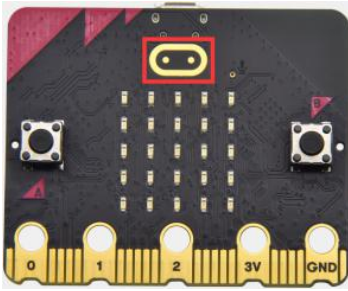
1 basic.show_icon(IconNames.EIGHTH_NOTE)
2
3 def on_forever():
4     soundExpression.giggle.play_until_done()
5     basic.pause(1000)
6     soundExpression.happy.play_until_done()
7     basic.pause(1000)
8     soundExpression.hello.play_until_done()
9     basic.pause(1000)
10    soundExpression.yawn.play_until_done()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

```

(4) Test Results:

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

Project 10: Touch-sensitive Logo



(1) Project Description:

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

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

(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code:

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

Delete block "on start" and "forever" ;

Enter "Input" module to find and drag "on logo pressed" ;
 Click the little triangle to find "touched" ' ;



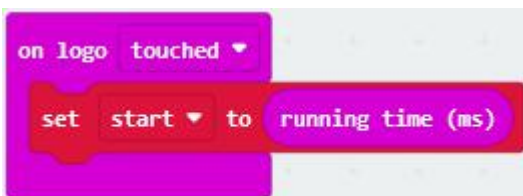
Enter module "Variables" →choose "Make a Variable" →input "start"
 →click "OK"

The variable "start" is established;

Enter "Variables" module to find and drag "set start to 0" into "on logo touched" block;

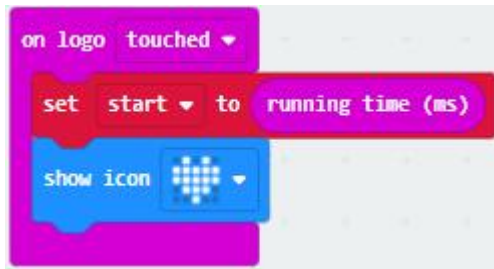


Enter "Input" module →click "more" → find and drag "running time(ms)" into the "0" of "set start to 0" block;



Enter "Basic" module to find and drag "show icon  into "on logo

touched" block;



Enter "Input" module to find and drag "on logo pressed" → choose "released" → establish variable "time" ;

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

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



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

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



Enter "Basic" module to find and drag "show number" into "on logo released" block;

Enter "Math" module to find and drag "square root 0" into "0" ;

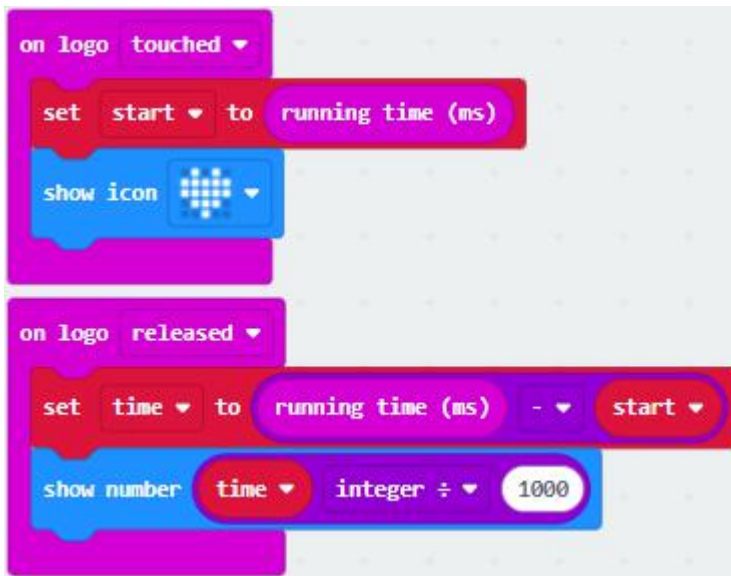
Click the little triangle to find "integer ÷" ;




Enter "Variables" module to find and drag "time" into "0" on the left side of "0-0" and change the "0" on the right side to "1000" ;

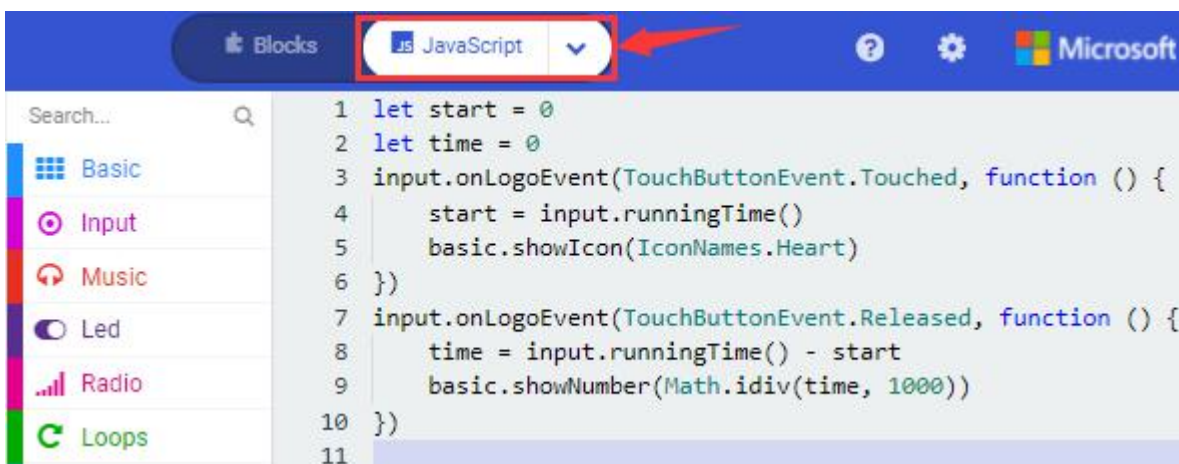


Complete Program:



- ① Touch the logo on the micro:bit with hand;
- ② Assign "running time" to variable "start";
- ③ LED dot matrix displays pattern ;
- ④ Put your hand away from the logo;
- ⑤ Assign "running time" to variable "time"
- ⑥ LED dot matrix displays the integer of variable "time" divided by 1000.

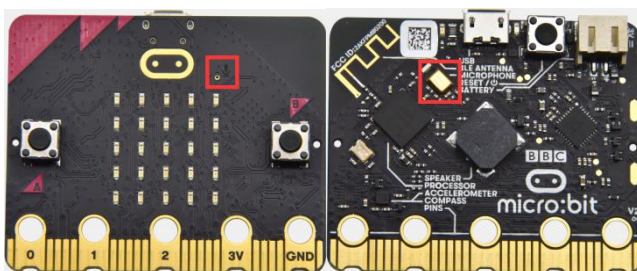
Select "JavaScript" and "Python" to switch into JavaScript and Python language code:



(4) Test Results:

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

Project 11: Microphone



(1) Project Description:

The built-in microphone of micro:bit V2 can test the sound volume in the ambient environment. When you clap, the microphone LED

indicator will be on.

Since it can measure the intensity of sound, you can make a noise scale or disco lighting changing with music. The microphone is placed on the opposite side of the microphone LED indicator and in proximity with holes that lets sound pass. When the board detects sound, the LED indicator will light up.

(2) Components Needed:

- Micro:bit main board V2 *1
- Micro USB cable*1

(3) Test Code 1:

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

Delete block "on start" and "forever" ;


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

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



Copy  once;

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

Click the little triangle of  to choose "  " ;



Complete Program:



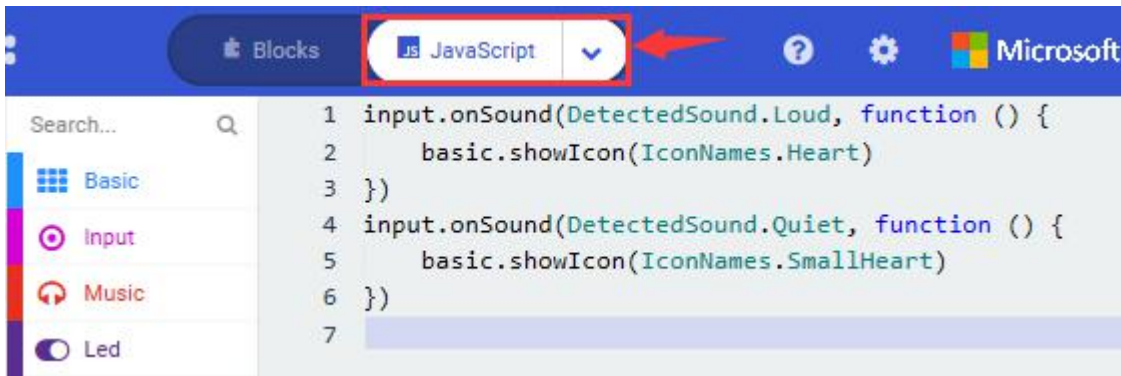
①The microphone on the micro:bit detects sound;

②LED dot matrix displays pattern "  " ;

③ The microphone on the micro:bit detects no sound;

④LED dot matrix displays pattern "  " .

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

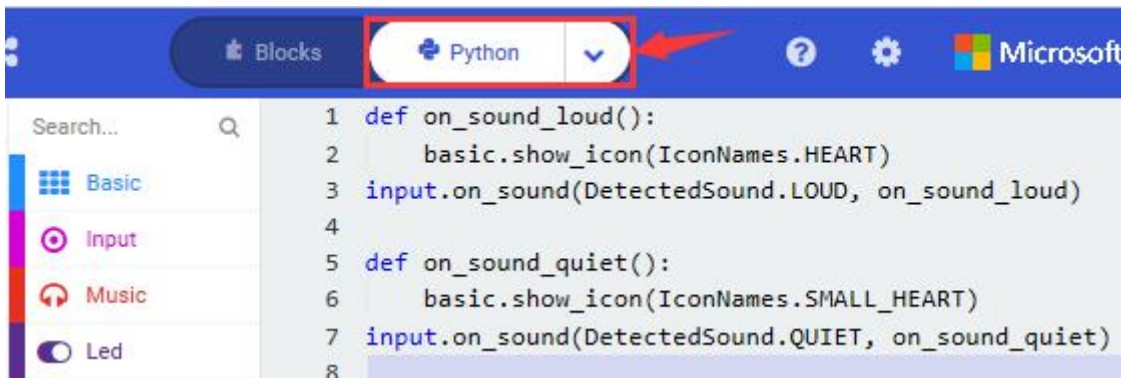


The screenshot shows the MakeCode editor interface. The top bar has a 'Blocks' button and a language dropdown menu currently set to 'JavaScript'. A red box highlights the dropdown menu, and a red arrow points to it. The left sidebar shows a search bar and a list of categories: Basic, Input, Music, and Led. The main editor area displays JavaScript code for sound detection:

```

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

```





The screenshot shows the MakeCode editor interface with the language dropdown menu set to 'Python'. A red box highlights the dropdown menu, and a red arrow points to it. The left sidebar is the same as in the previous screenshot. The main editor area displays Python code for sound detection:

```

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

```

(4) Test Results 1:

Uploading test code to micro:bit main board V2 and powering the board via the USB cable. The LED dot matrix displays pattern  when you clap; however, the image  will appear when it is quiet around.

(5) Test Code 2:

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

Enter "Advanced" module → choose "Serial" to find and drag "serial

redirect to USB" into "on start" block ;



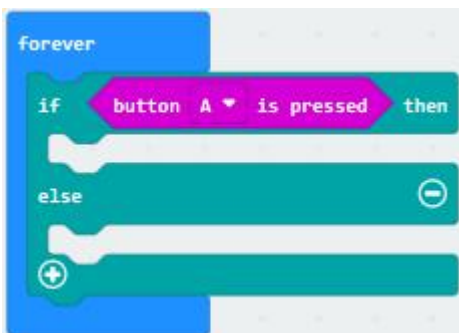
Enter "Variables" module → choose "Make a Variable" → input "maxSound" →click "OK" ,variable " maxSound" is established;

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



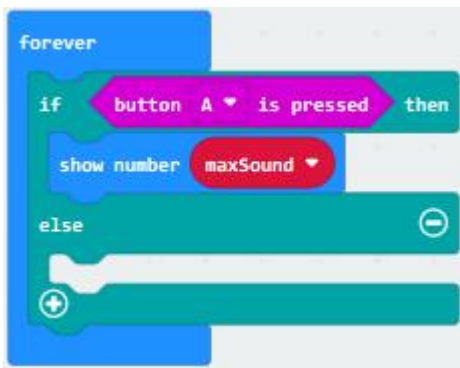
Enter "Logic" module to find and drag "if true then...else" into "forever" block ;

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



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

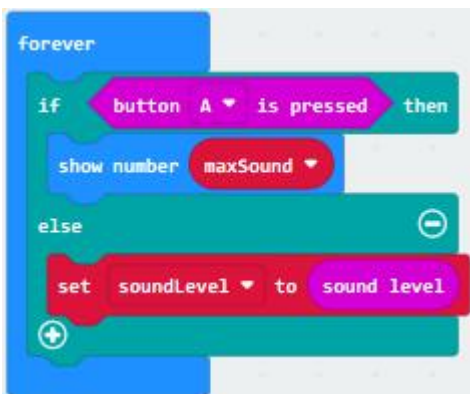
Enter "Variables" module to find and drag "maxSound" into "0" ;



Establish variable "soundLevel" ;

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

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



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

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

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

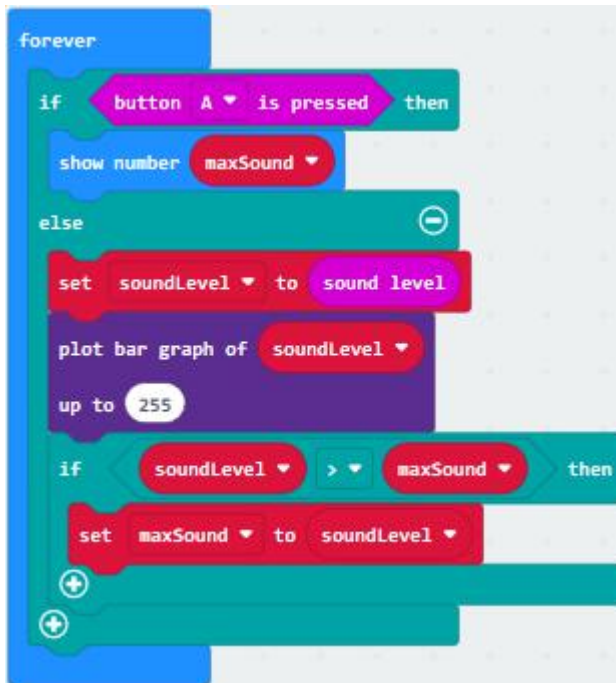


Enter "Logic" module to find and drag "if true then" into "else" block ;
 Enter "Logic" module to find and drag " $0 > 0$ " into "then" ;
 Enter "Variables" module to find and drag "soundLevel" into "0" on
 the left side of "0-0" ;
 Enter "Variables" module to find and drag "maxSound" into "0"
 on the right side;

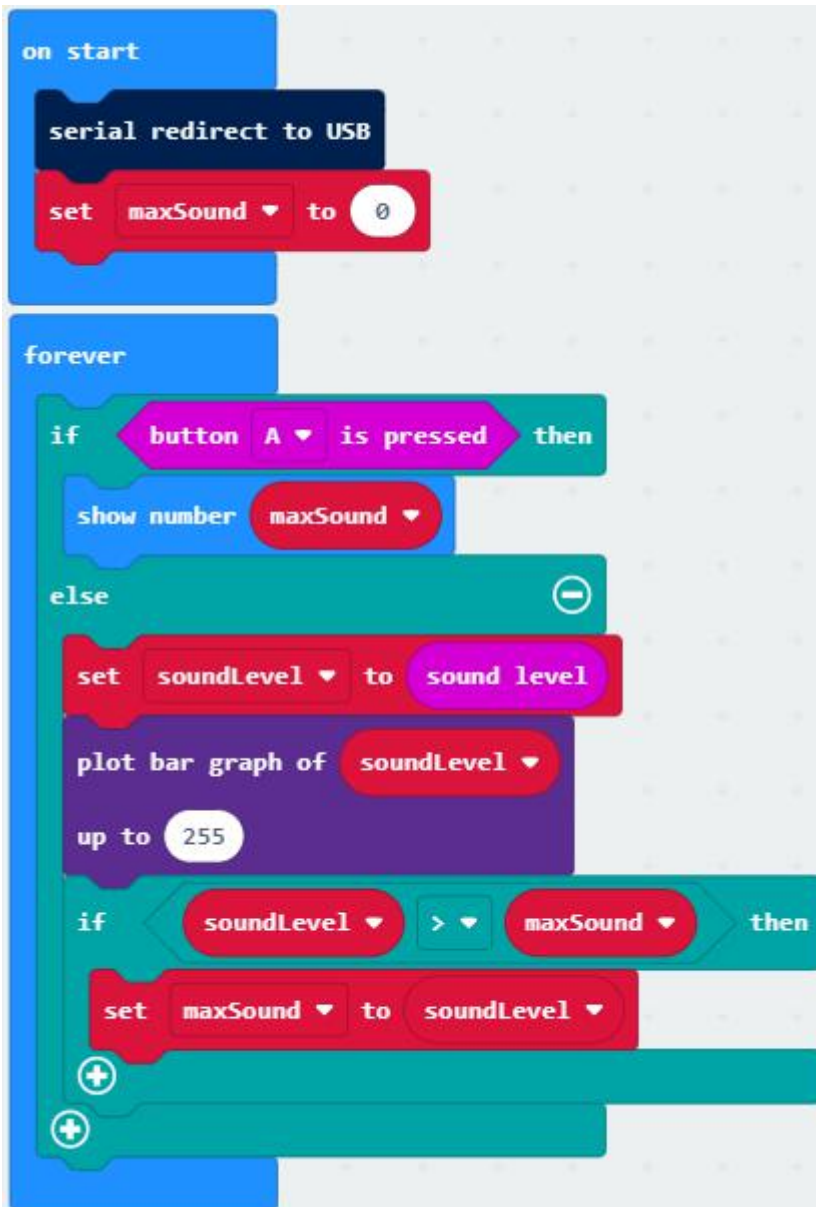


Enter "Variables" module to find and drag "set maxSound to 0" into the
 second "then" ;

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



Complete Program :

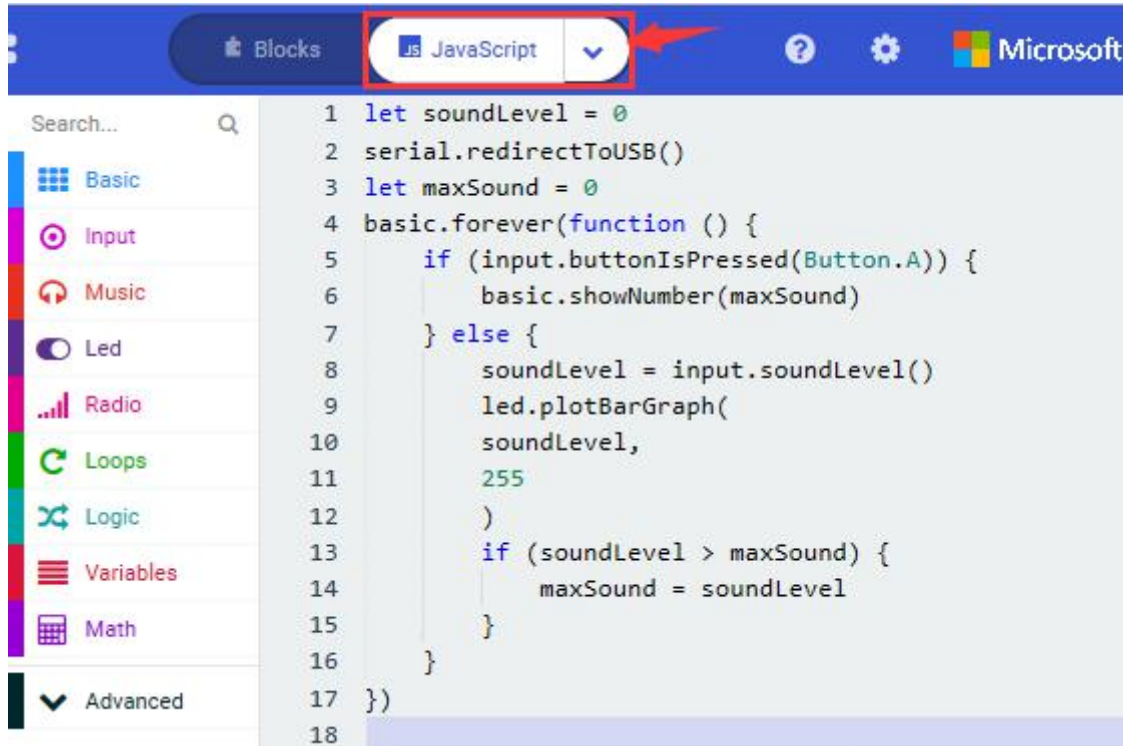


- ① In "on start" the program only runs once;
- ② Redirect serial to USB;
- ③ Set the initial value of variable "maxSound" to 0;
- ④ In "forever" the program runs cyclically;

- ⑤ When button A is pressed, the program in "then" runs;
- ⑥ LED dot matrix displays the loudest sound value detected by the microphone sensor;
- ⑦ When above conditions are not true, the program in "else" runs;
- ⑧ Assign sound value to variable "soundLevel";
- ⑨ LED dot matrix displays the brightness of LED; and the brightest value is 25;

- ⑩ If the sound value detected is bigger than the loudest one in ambient environment
- ⑪ Then assign the sound value detected to variable "soundLevel".

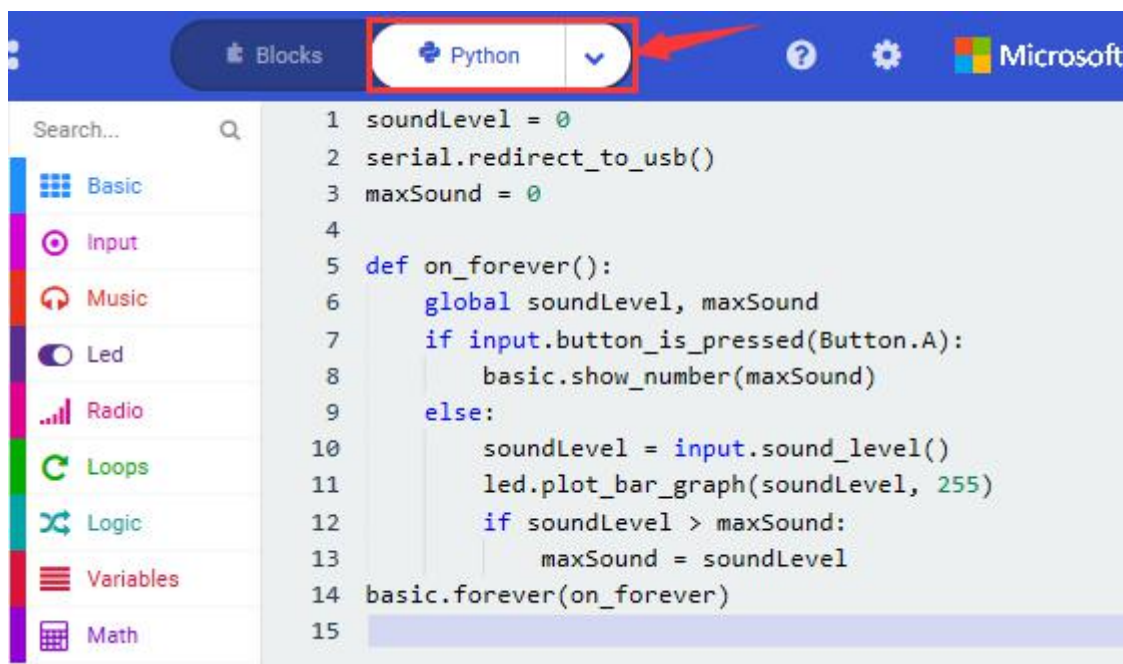
Select "JavaScript" and "Python" to switch into JavaScript and Python language code:



```

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

```



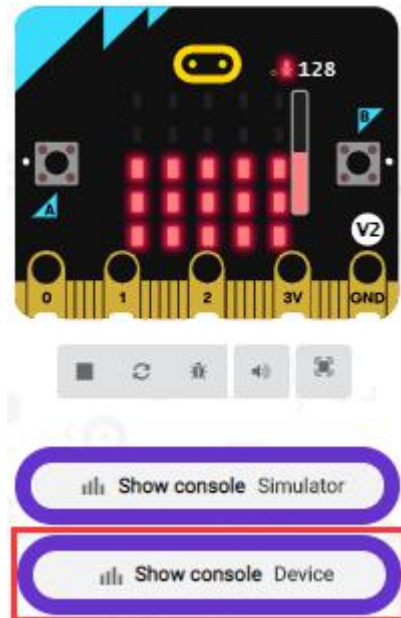
```

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

```

(6) Test Results 2:

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



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

The image displays a digital simulator interface. On the left is a virtual hardware board with an LED dot matrix and a volume meter. The volume meter shows a value of 128. Below the board are two buttons: "Show console Simulator" and "Show console Device". On the right is a waveform graph showing a signal with a peak value of 195.00. Below the graph is a console window displaying a list of values:

```

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

```

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

Project 12: Bluetooth Data Reading



(1) Project Description:

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

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

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

(2) Preparation

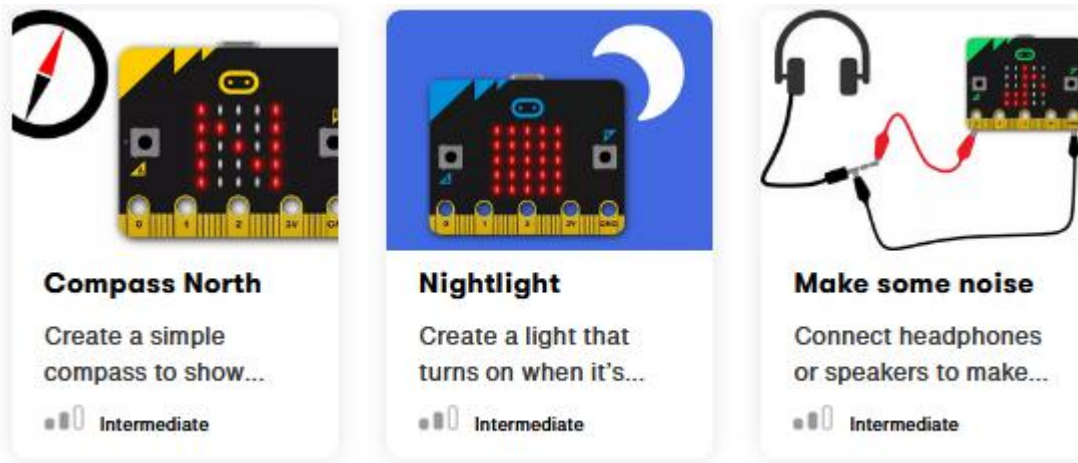
*Attach the Micro:bit main board V2 to your computer via the Micro USB cable.

*An Apple device (a phone or an iPad) or an Android device;

(3) Procedures:

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





If you need help

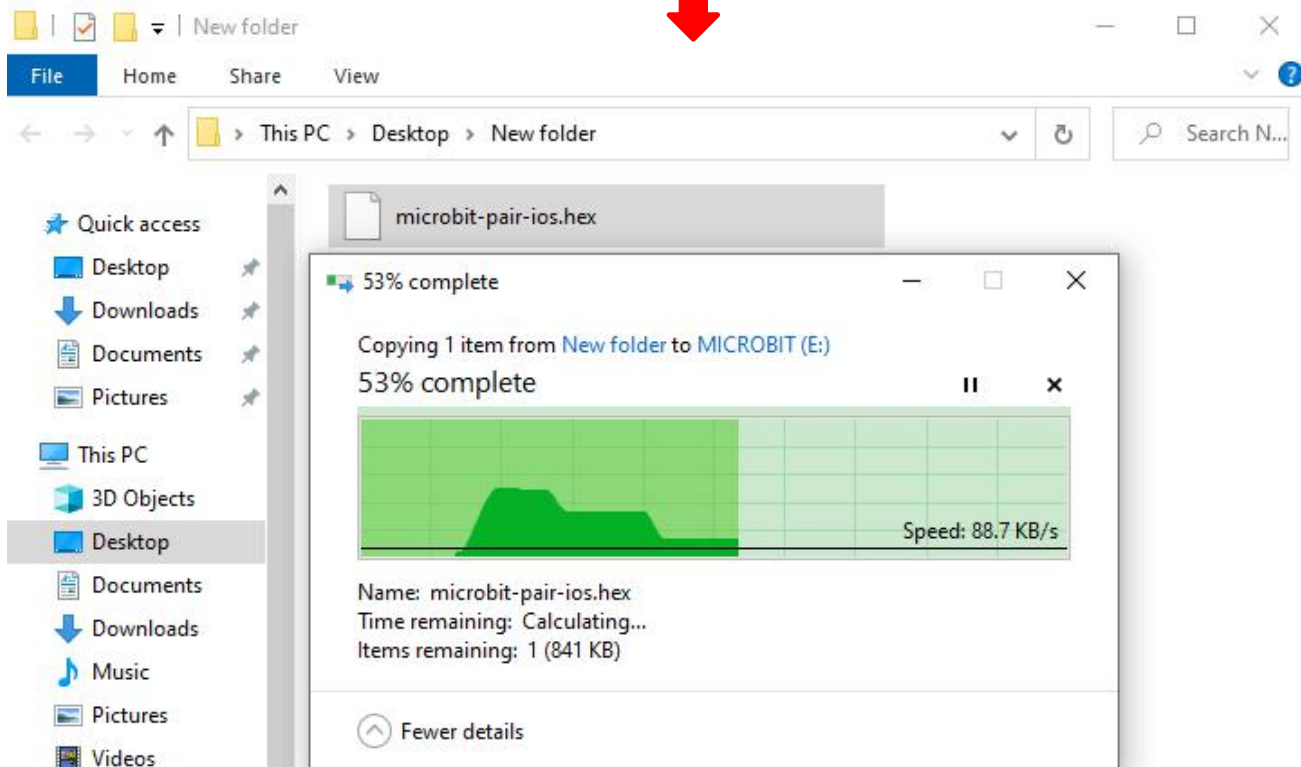
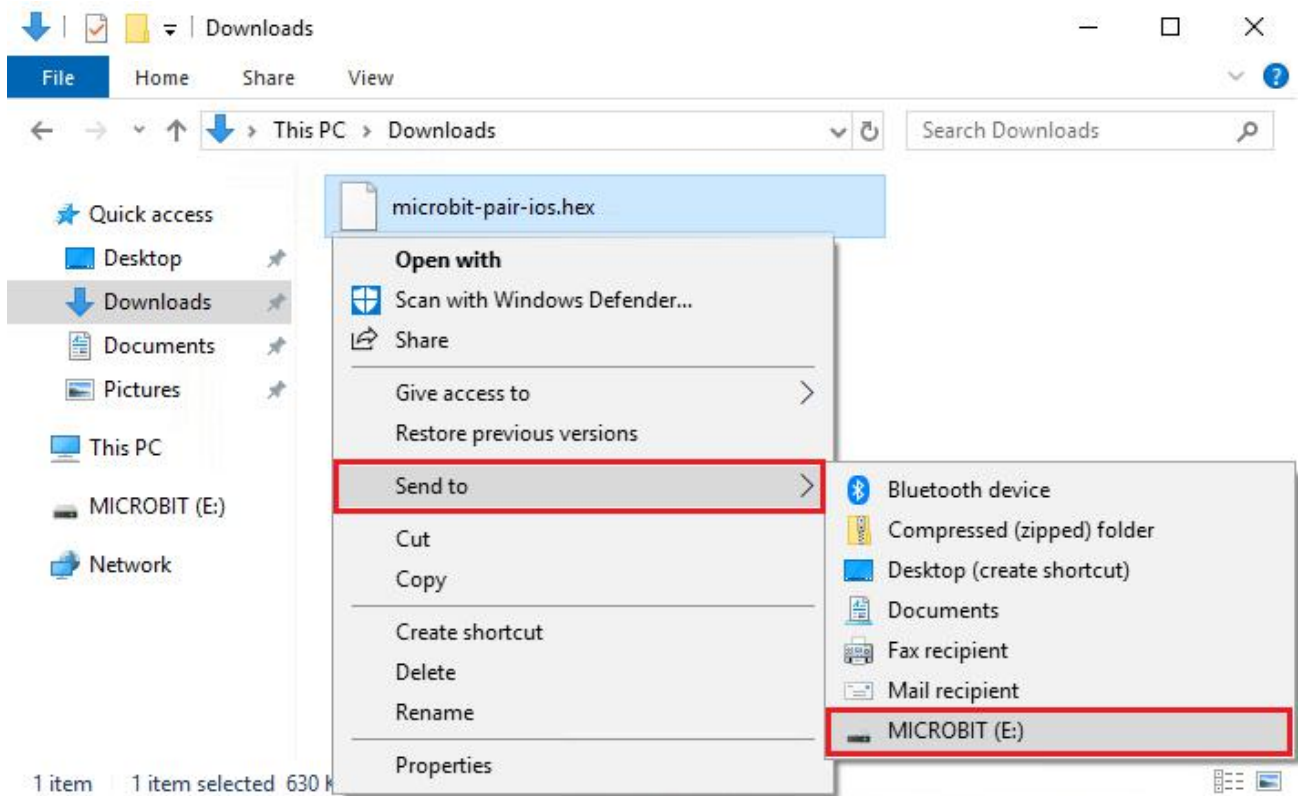
If you're having problems flashing code from your iOS device to your micro:bit, download this HEX file and transfer it to your micro:bit from a computer, or visit our support site.

[Download pairing HEX file](#)

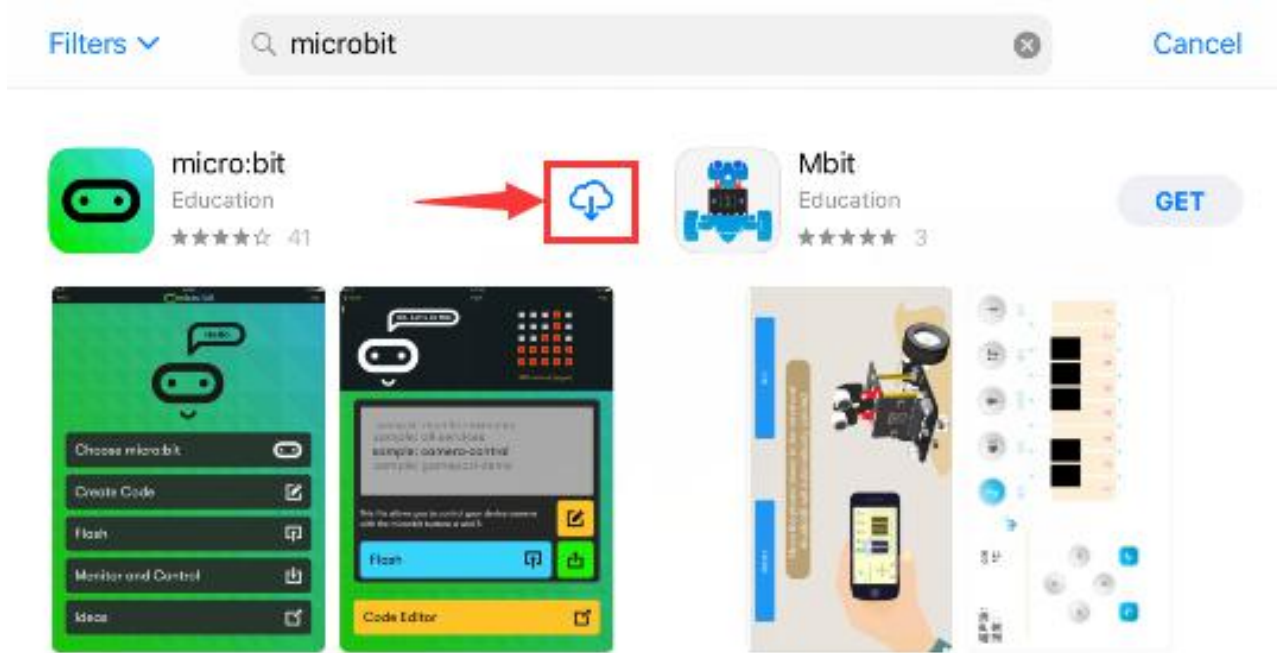
[iOS app support](#)

Monitor and control

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

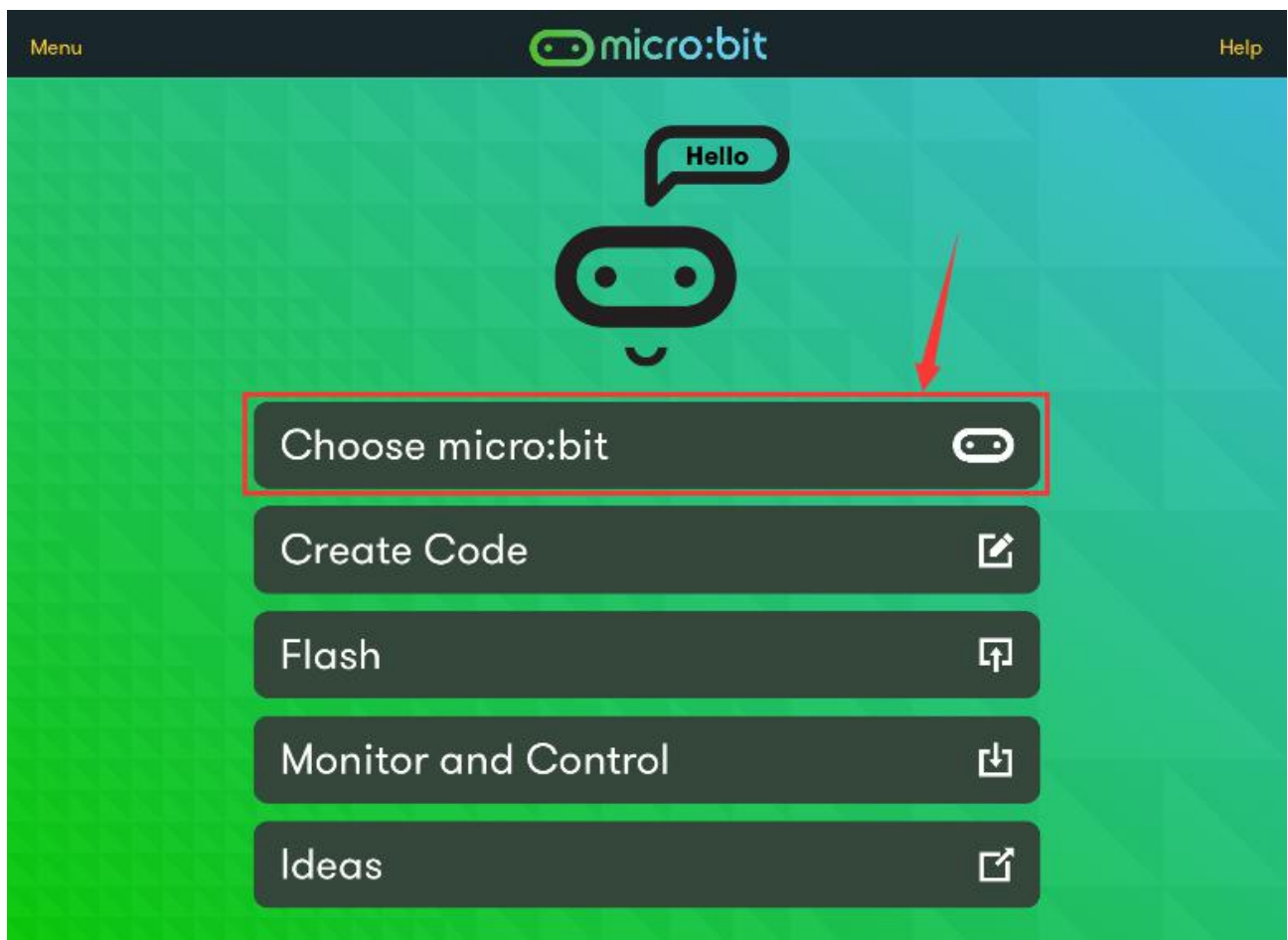


Search "micro bit" in your App Store to download the APP micro:bit.

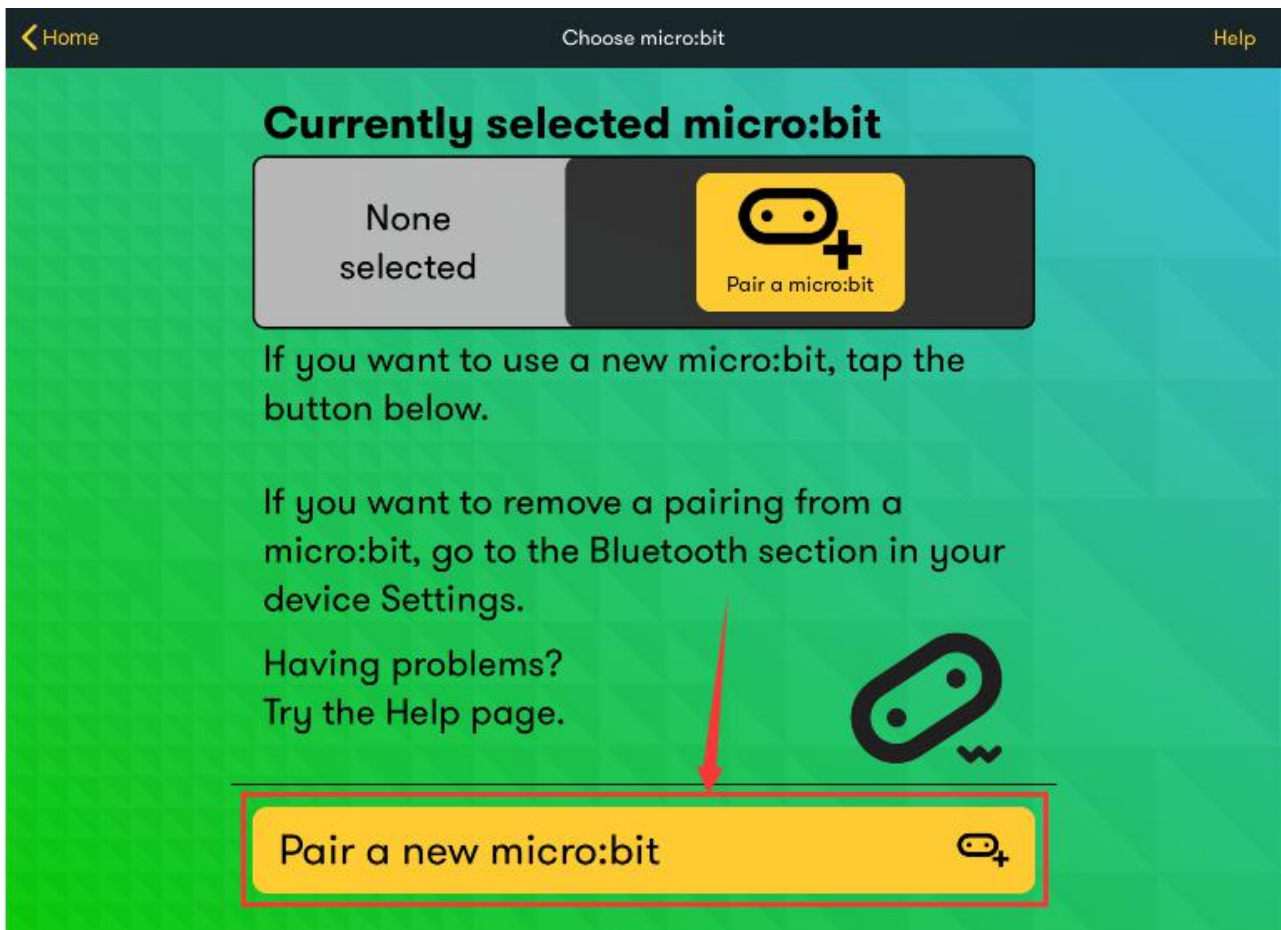


Connect your Apple device with Micro: Bit main board V2:

Firstly, turn on the Bluetooth of your Apple device and open the APP micro:bit to select item "Choose micro:bit" to start pairing Bluetooth. Please make sure that the Micro: Bit main board V2 and your computer are still linked via the USB cable.



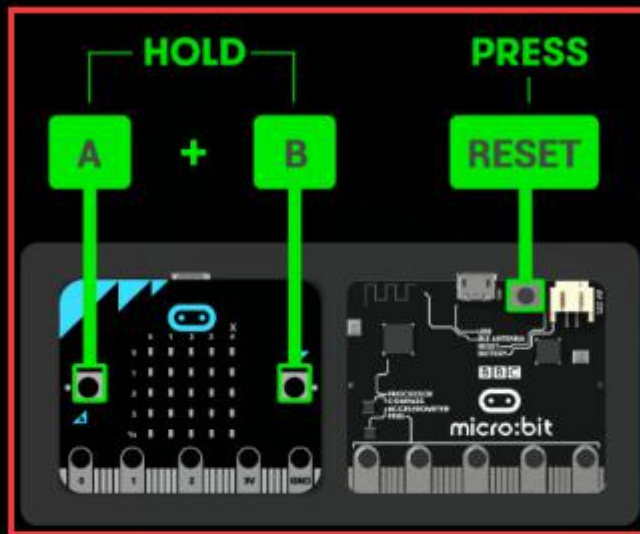
Secondly, click "Pair a new micro:bit" ;



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

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

How to pair your micro:bit



Let's do this

Step 1

HOLD the A and B buttons and
PRESS and RELEASE RESET



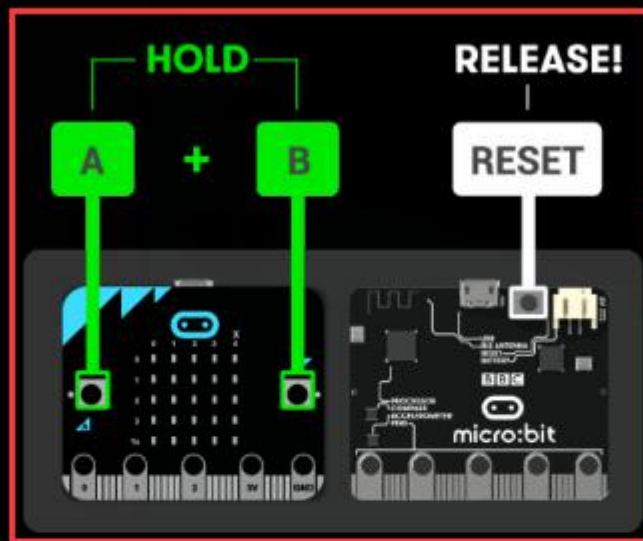
Cancel



Next



How to pair your micro:bit



Step 1

HOLD the A and B buttons and
PRESS and RELEASE RESET

Let's do this



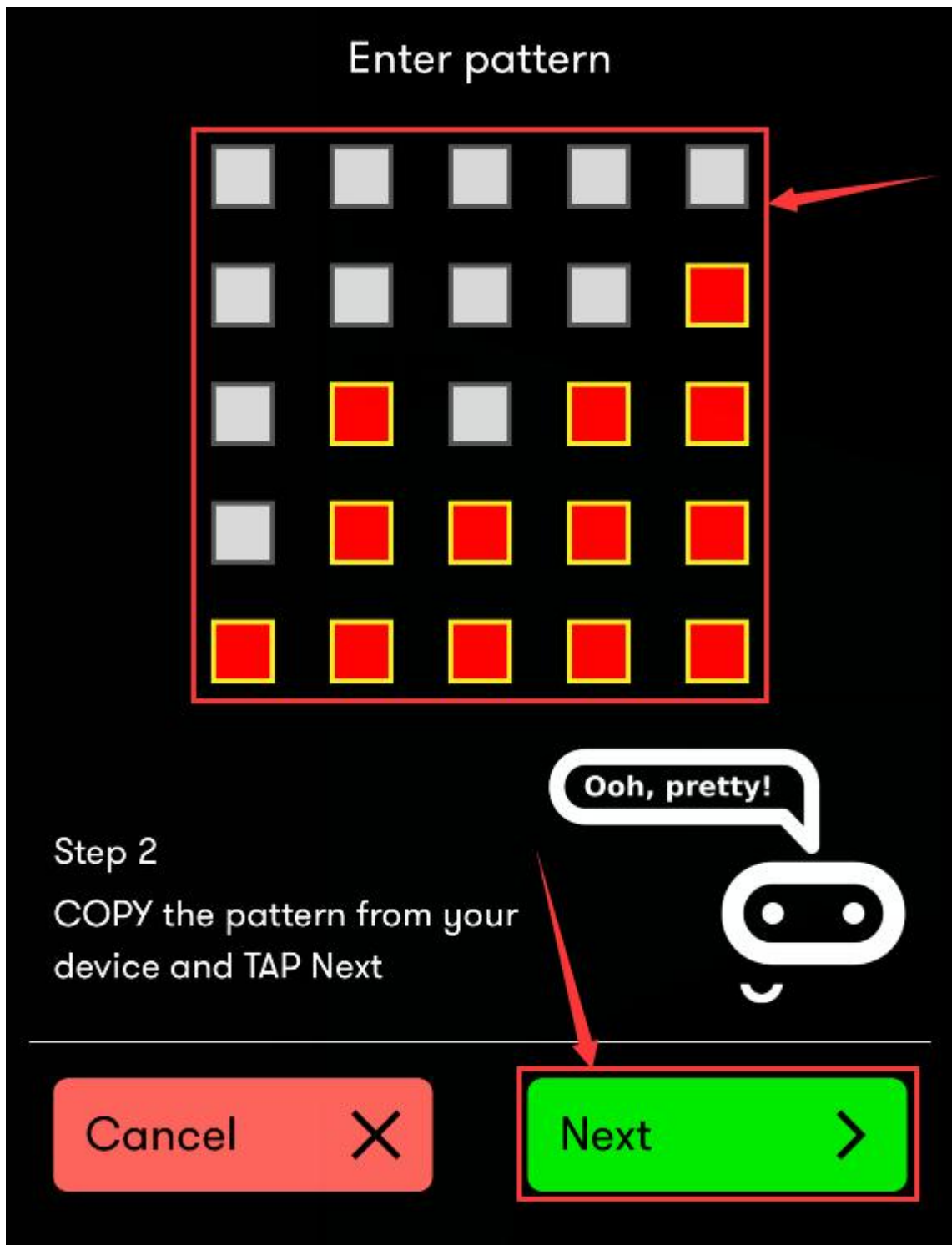
Cancel



Next

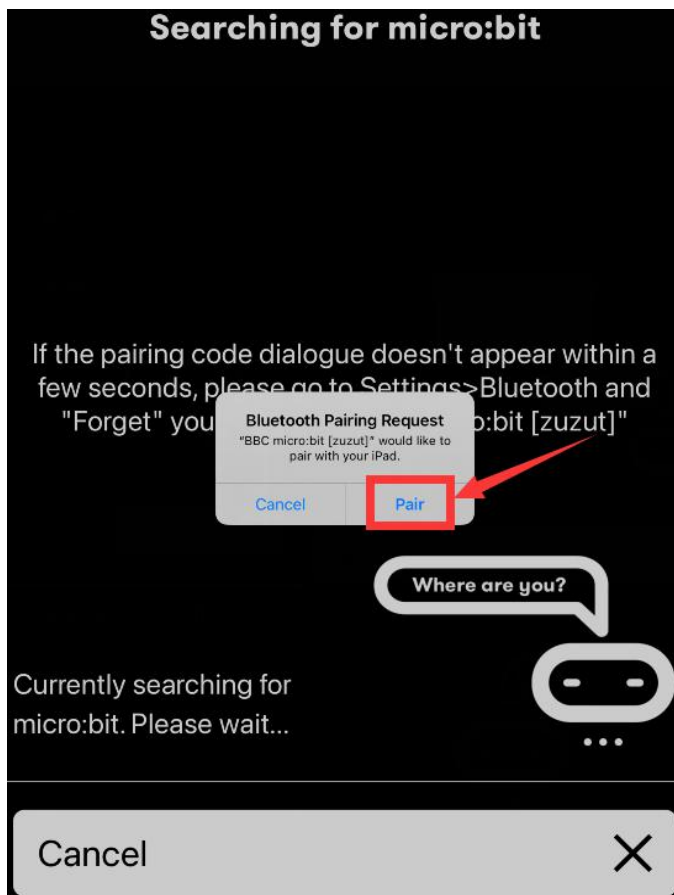
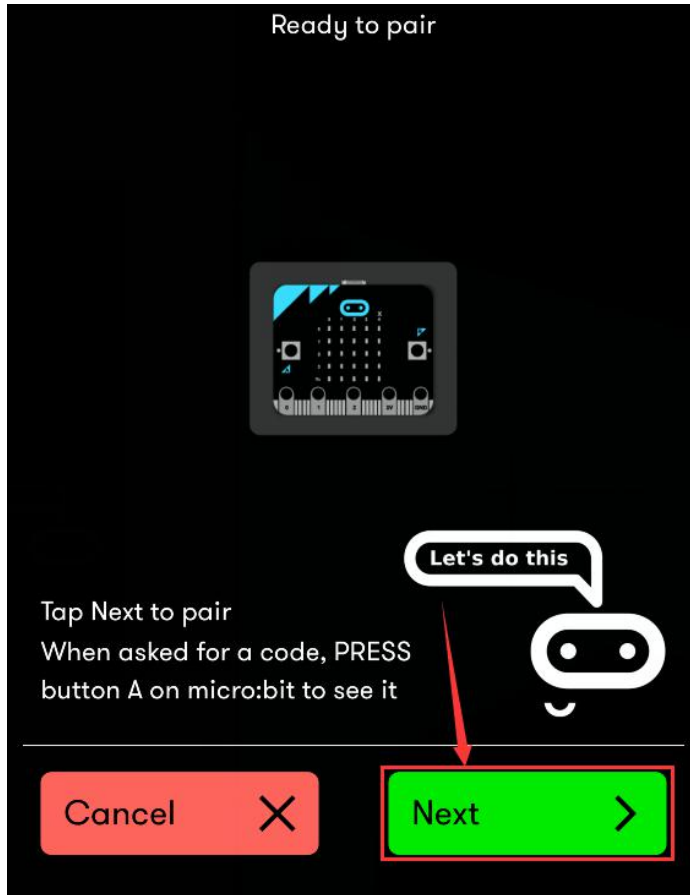


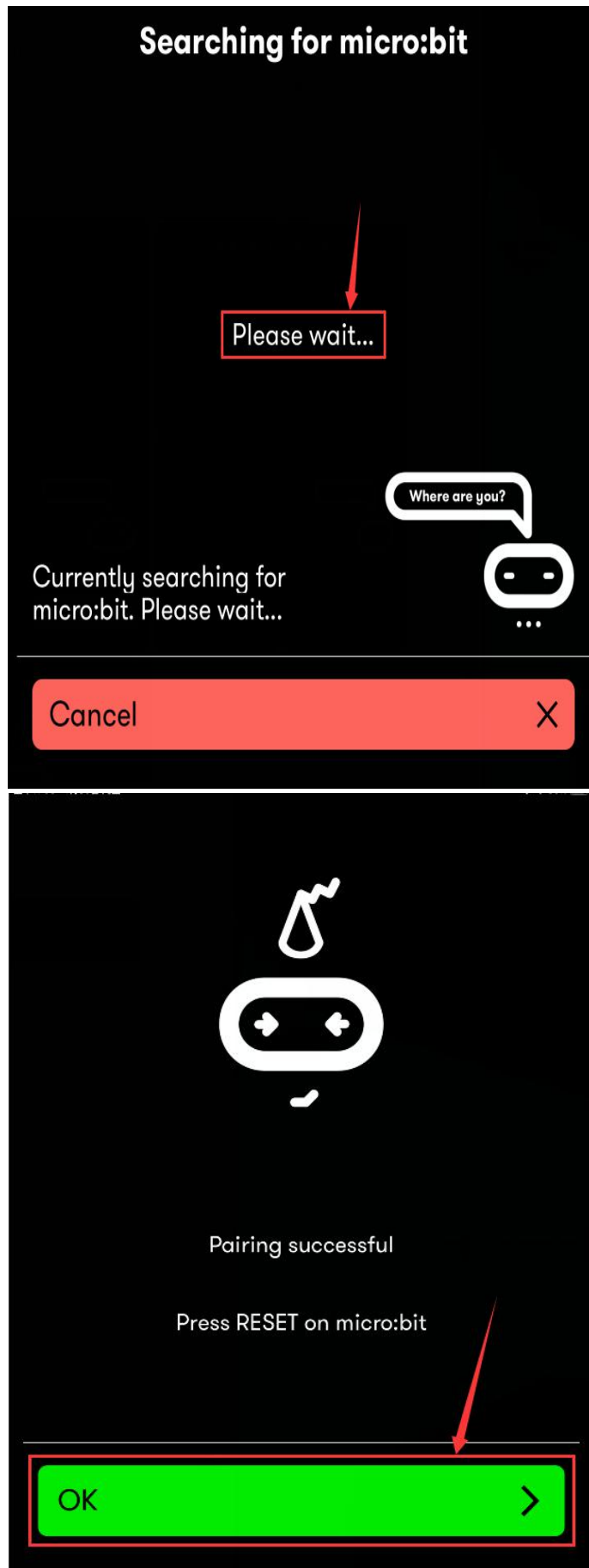
Set the password pattern on your Apple device as the same pattern showed on the matrix and click Next.



Still click Next and a dialog box pops up as shown below. Then click "Pair". A few seconds later, the match is done and the LED dot matrix


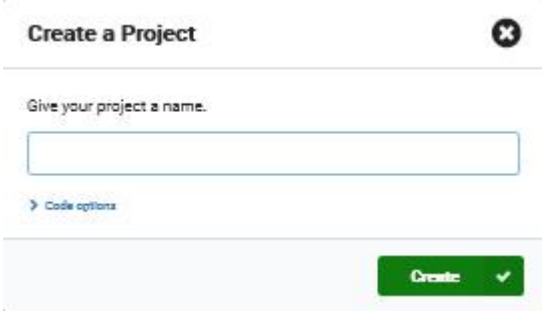
displays the "v" pattern.

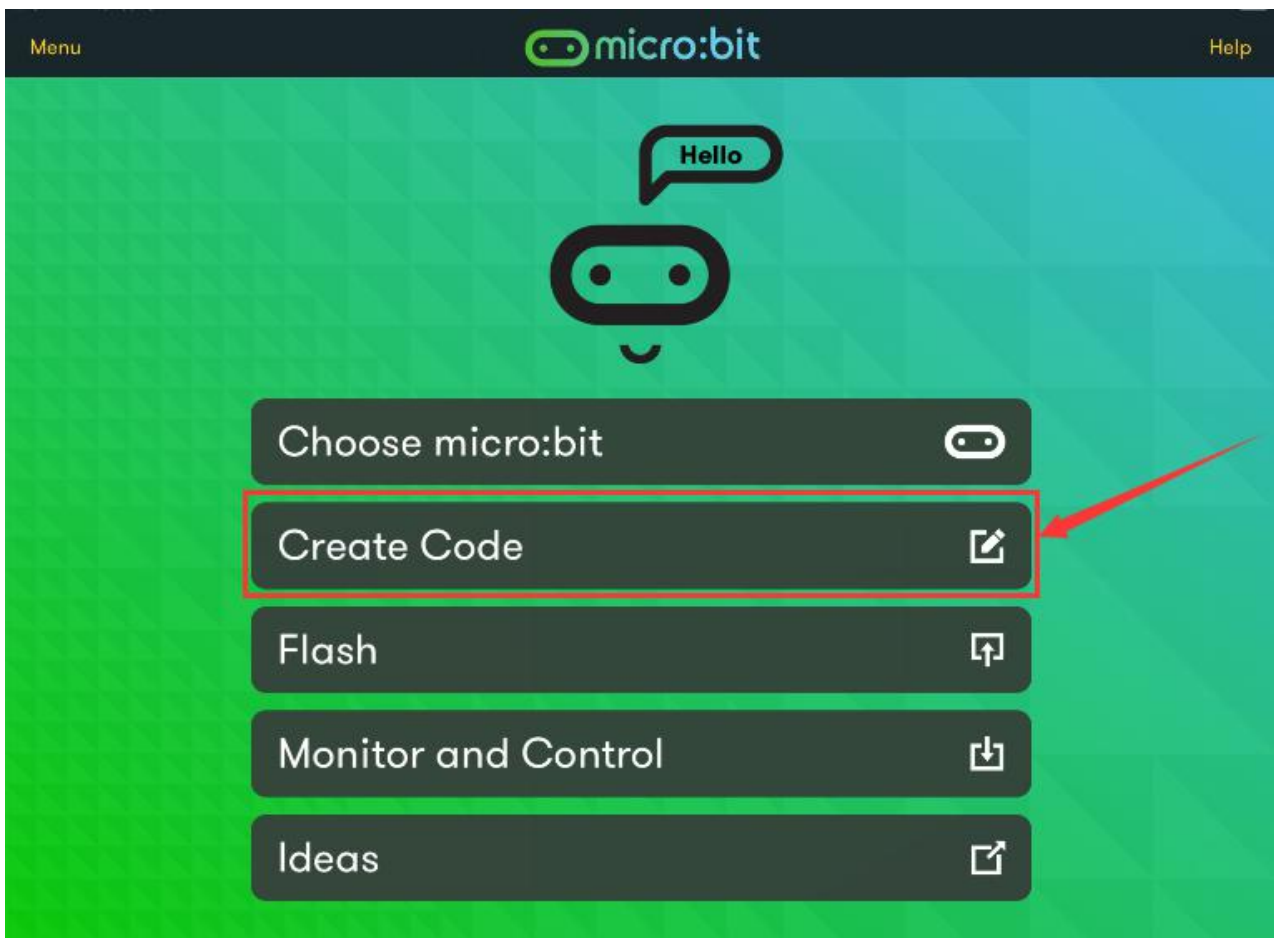


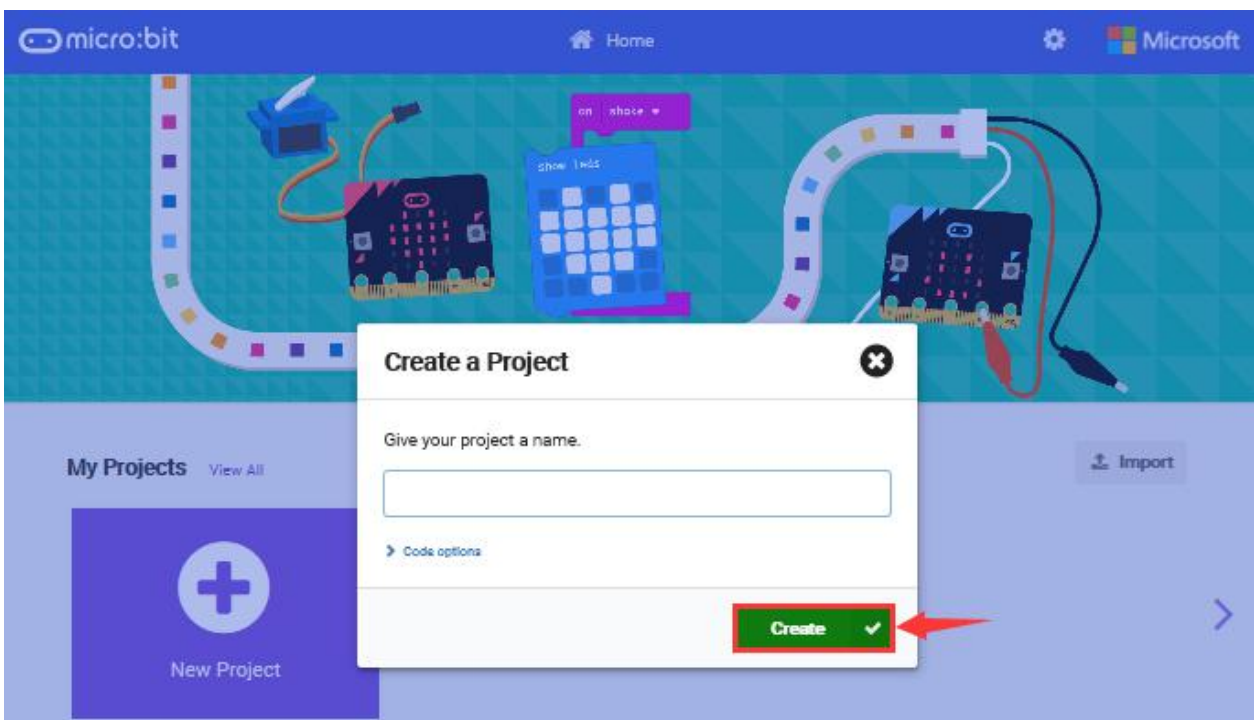
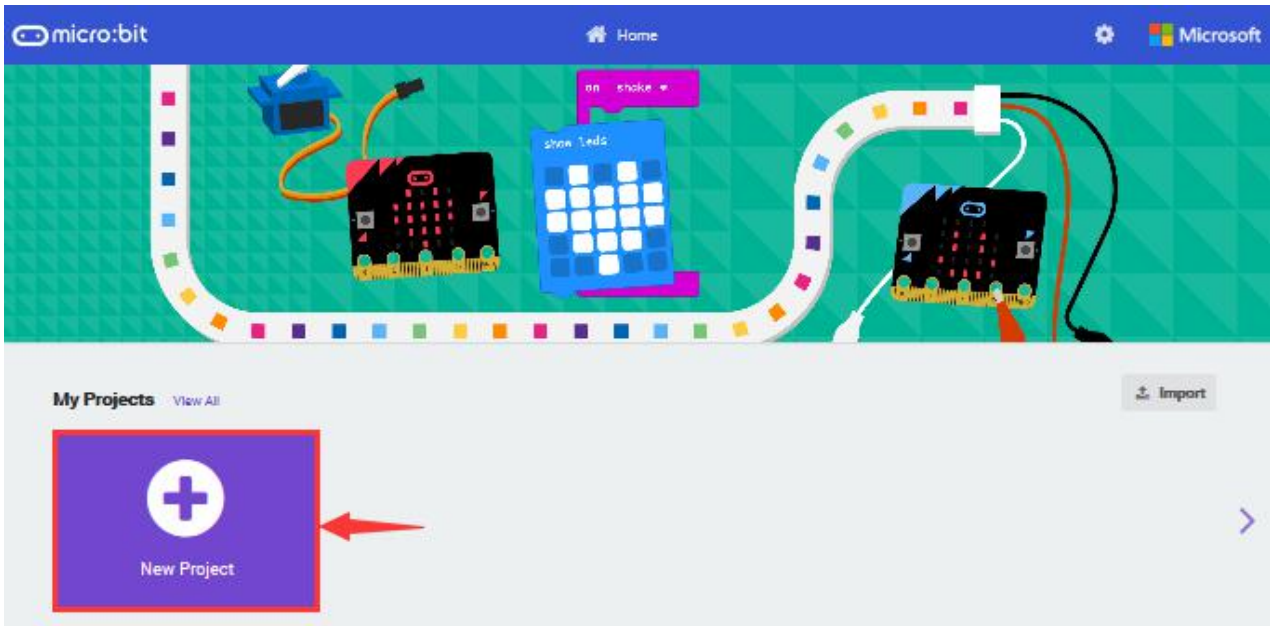


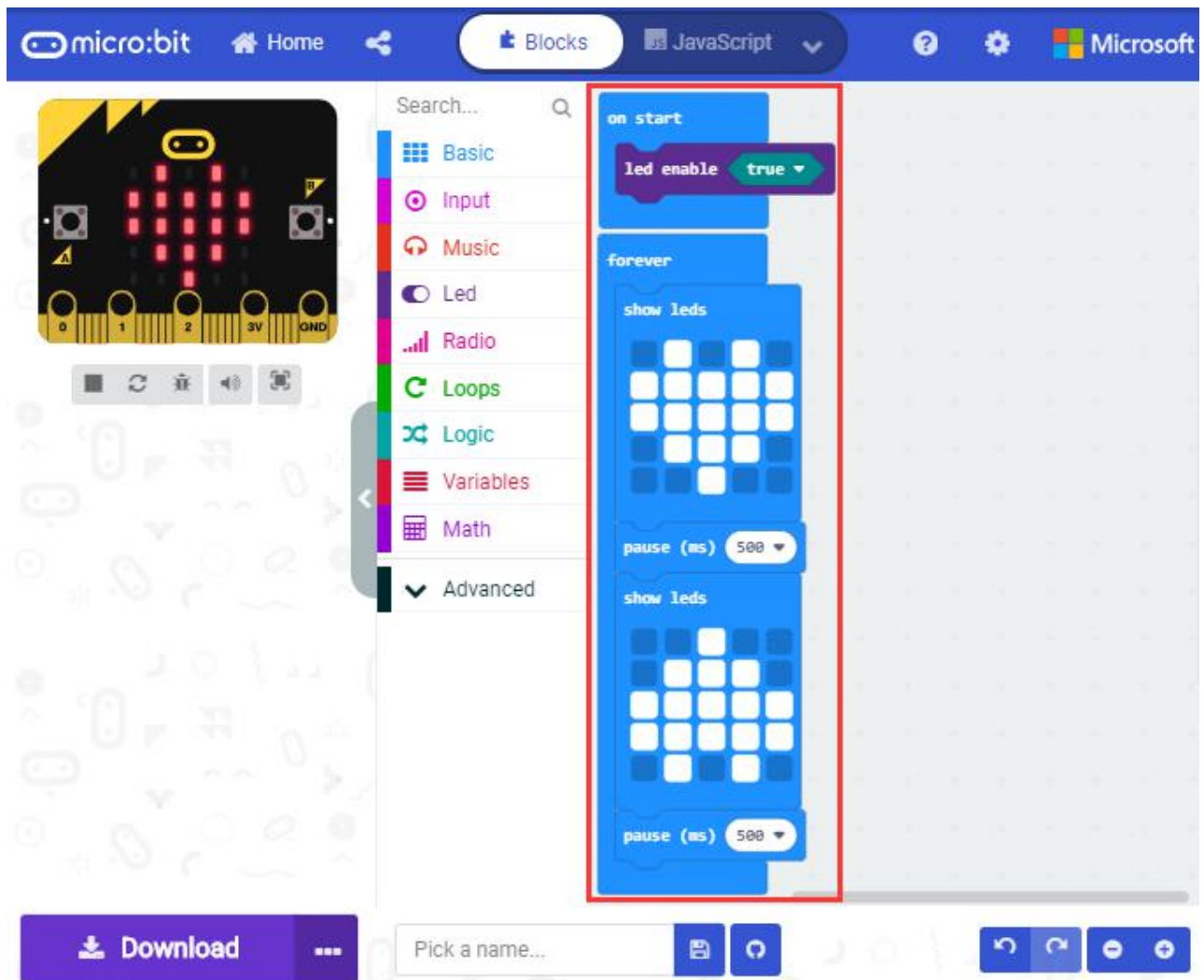
After the match with Bluetooth, write and upload code with the App.


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

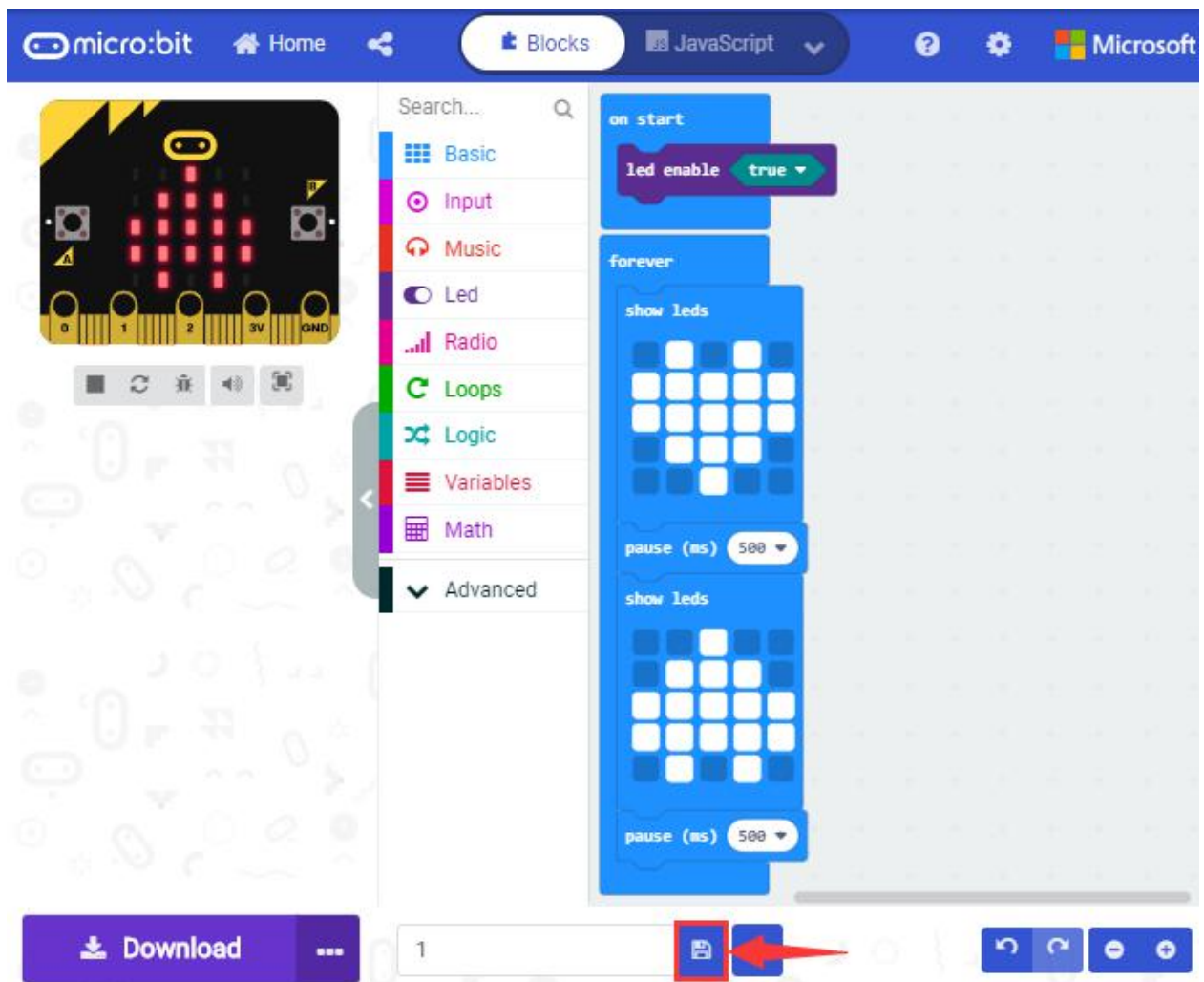
Click  and the box  appears, and then select "Create ✓".



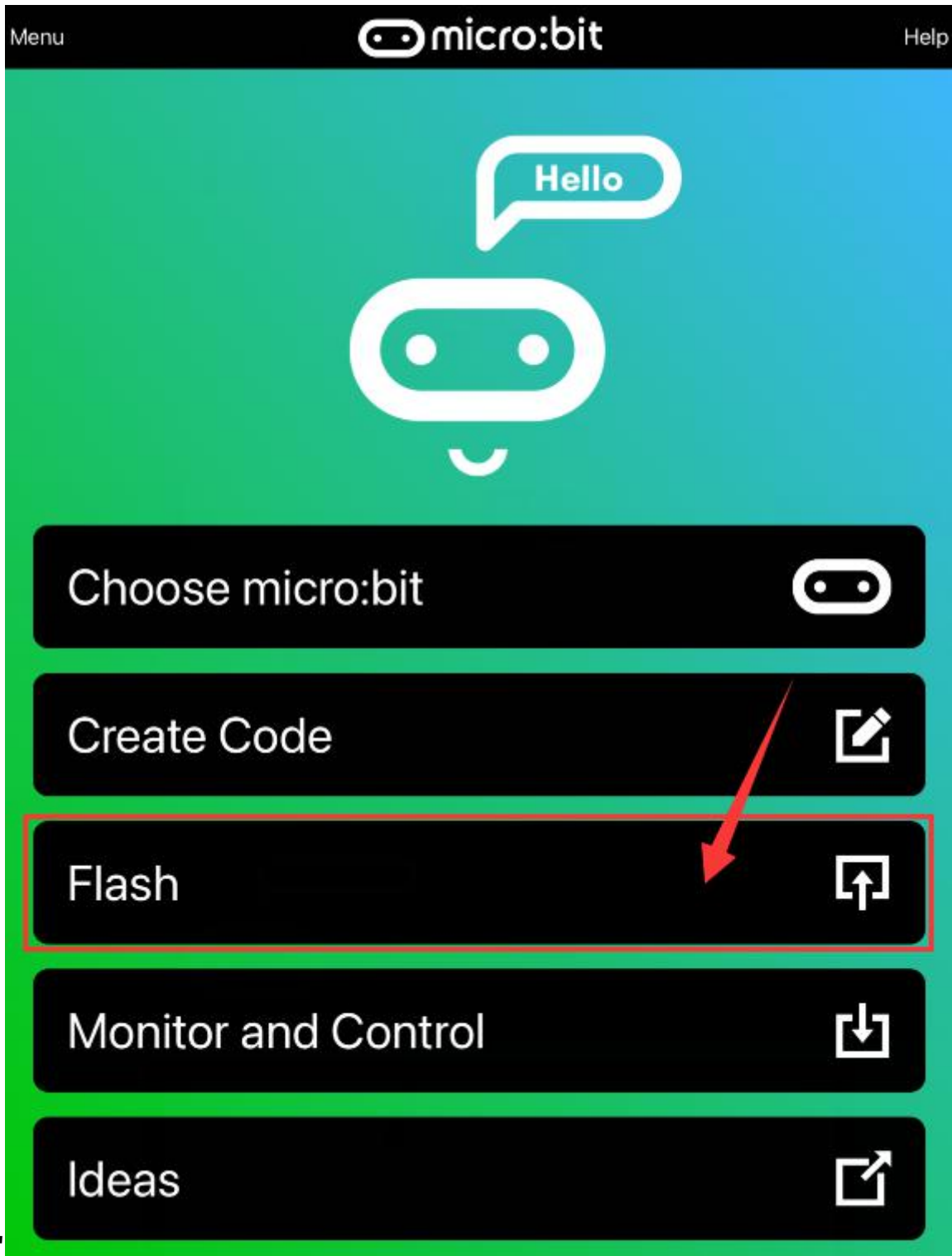




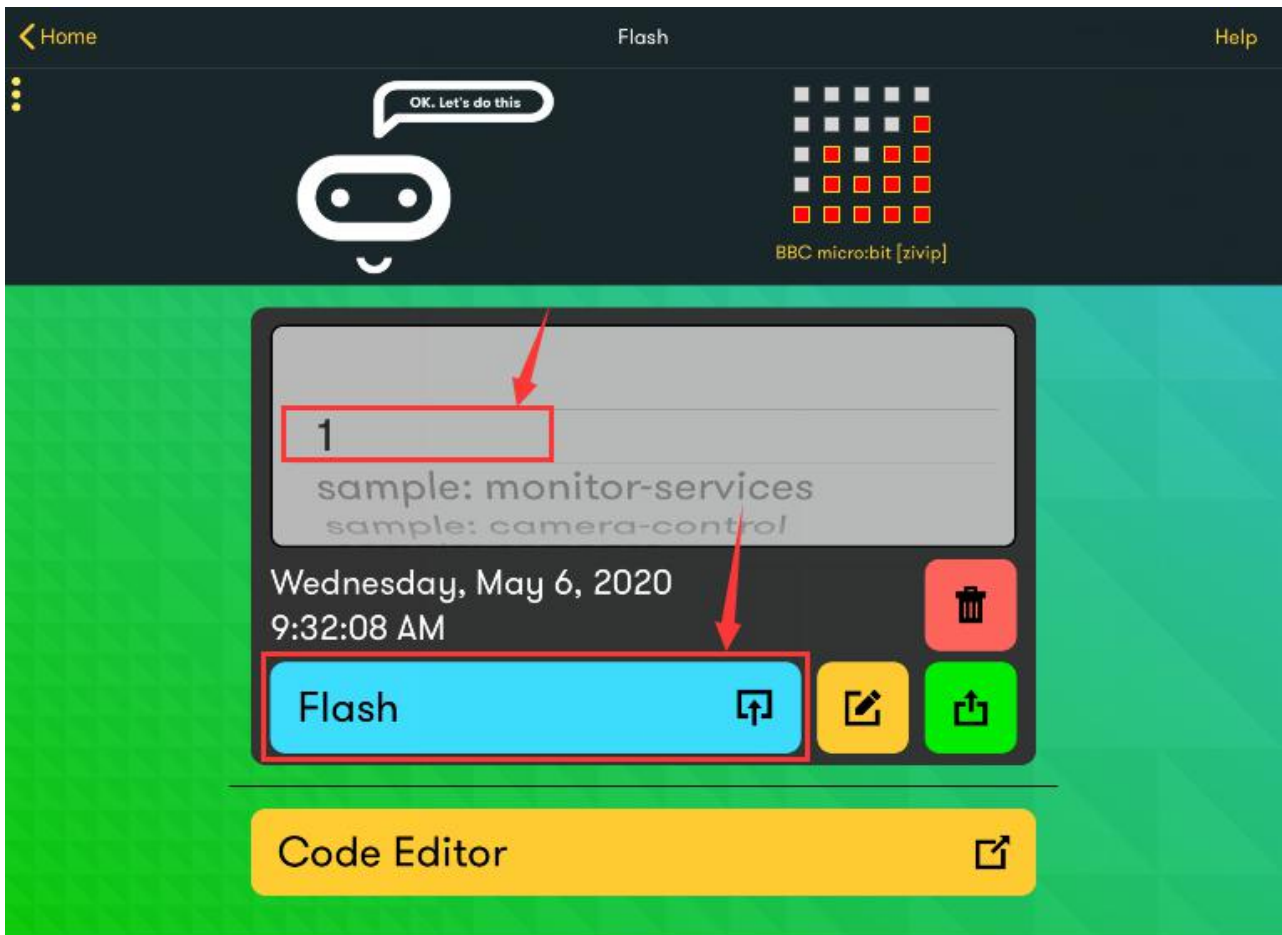
Name the code as "1" and click  to save it.

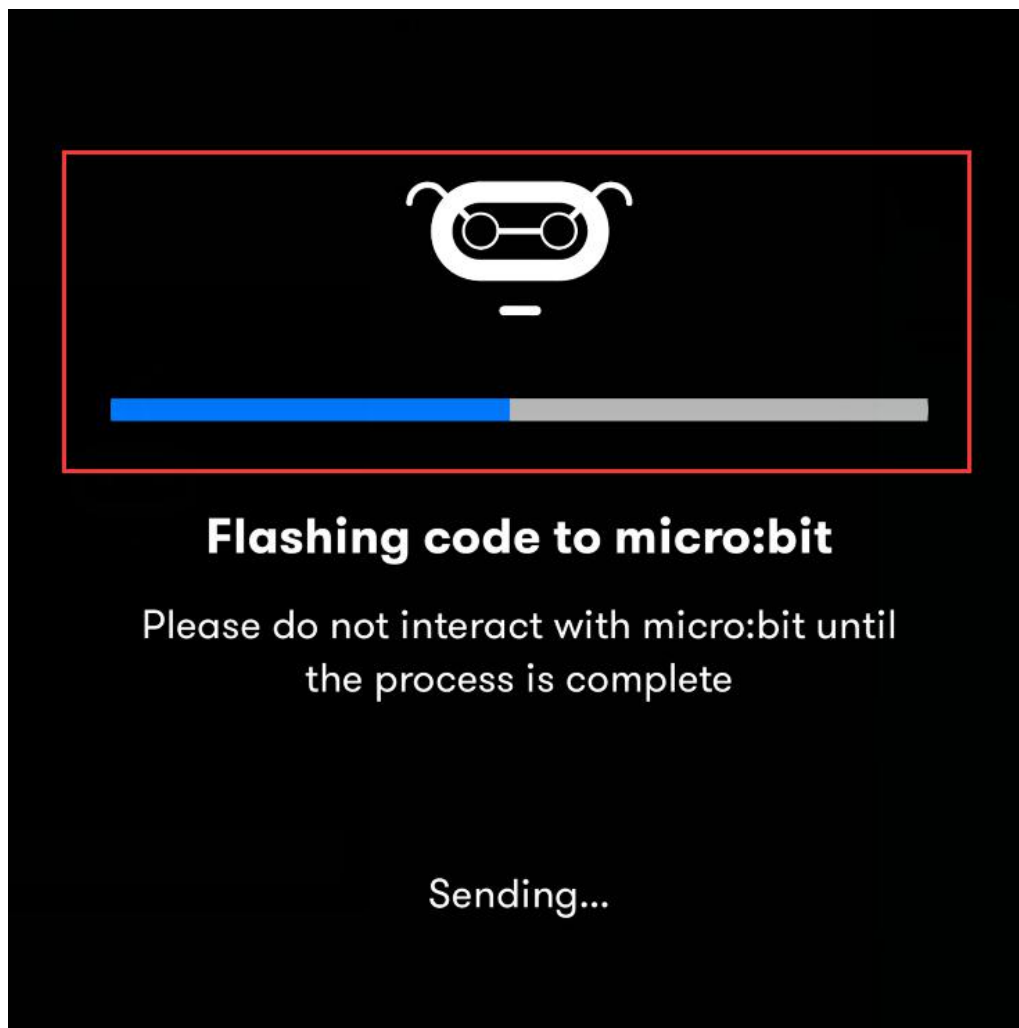


Click the third item "Flash" to enter the uploading page. The default code program for uploading is the one saved just now and named "1" and then click the other "Flash" to upload the code program

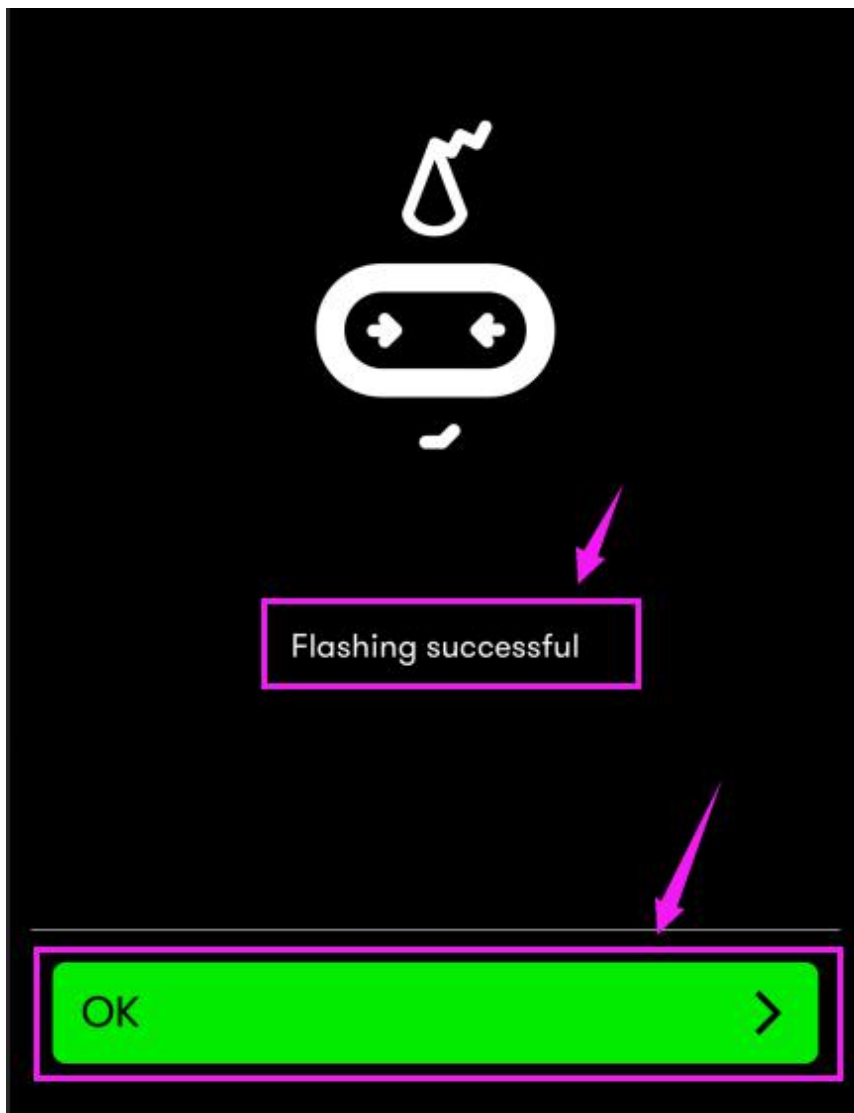


"1"





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



Projects below all conduct with the built-in sensors and the LED dot matrix while the following ones will carry out with the help of external sensors/modules of EASY Plug Shield for micro bit V1.1.

Project 13: LED Blink

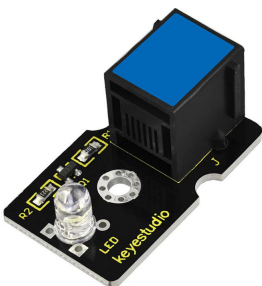
1. Description:

LED blink is a basic experiment. You will learn how to make white LED blink through code. Please turn off dot matrix on micro:bit before testing.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug White LED Module*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug White LED Module



The LED light modules have shiny colors, ideal for Arduino starters. It can be easily connected to IO/Sensor shield.

Note: this module needs to be used together with EASY Plug Shield for micro bit V1.1. You can also choose other LED to emit different color of light like white, blue, green, yellow and red.

3. Specification:

- Interface: Easy plug
- Sensor type: Digital
- Working voltage: 5V
- Easy to use
- Useful for light projects

EASY Plug Shield for micro:bit V2

Micro:bit is a basic development board designed by the British Broadcasting Corporation for youth programming education. It supports the PXT graphical programming interface developed by Microsoft, without the need to download an additional compiler, and can be used under Windows, macOS, IOS, Android and other operating systems.

We combine the EASY Plug shield with the micro:bit due to the inconvenience of wiring up micro:bit .

The golden finger interfaces, as well as 10 pcs easy plug ports (RJ11 6P6C interfaces) could be connected to other modules and sensors,

therefore, you don' t need to worry about wiring up components incorrectly.

The shield comes with 4 pcs WS2812 LEDs controlled by P9, P0 controls passive buzzer; and two dial switches--Power_Switch and Voltmeter_Switch(3.3V, 5V).

The voltage of power supply is DC 6-10V

The Easy Plug port only supports the sensors and modules with RJ11 6P6C port.

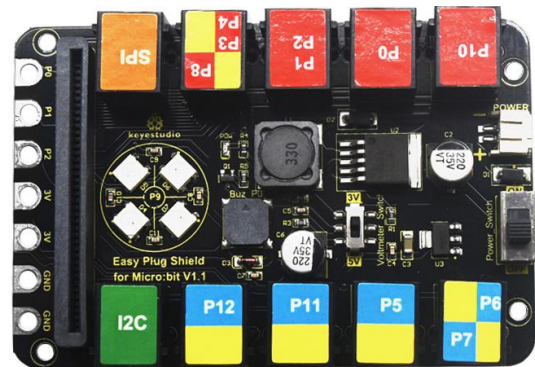
4. Specification:

Power supply: DC 6-10V

Output current: 1.5A

Interface: RJ11 6P6C interface and golden finger interface

Size: 98*65*17mm



5. Interface Description:

G: GND

V: Voltmeter_Switch control, dial to 5V end, 5V; dial to 3V end, 3.3V

I2C Communication Port

G	V	SDA
	SCL	

SDA: P20

SCL: P19

SPI Communication Port

G	V	P16
SCK	MISO	MOSI

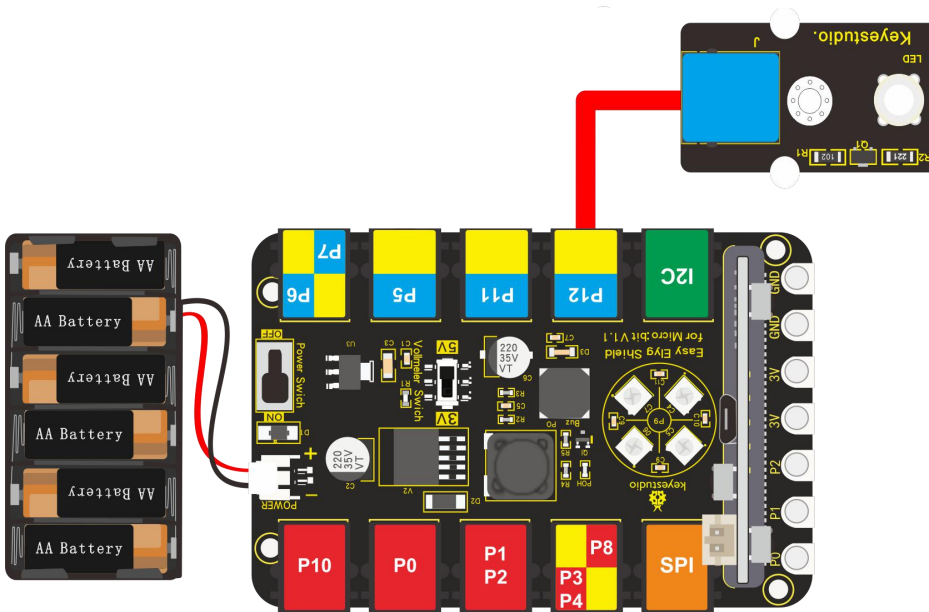
MOSI: P15

MISO: P14

SCK: P13

6. Wiring Up:

Insert the micro:bit onto EASY Plug shield, link the white LED module with P12 port of shield and plug in power.



Note: Dial Voltmeter_Switch to 3V end

7. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.

```

on start
  led enable false

forever
  digital write pin P12 to 1
  pause (ms) 1000
  digital write pin P12 to 0
  pause (ms) 1000

```

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

Turn off dot matrix on micro:bit

The program under the block "forever" runs cyclically.

Set P12 to high level(1), turn on LED

Delay in 1000ms

Set P12 to low level (0) , turn off LED

Delay in 1000ms

8. Test Results:

Wire up, dial Voltmeter_Switch to 5V end, plug in external power and dial Power_Switch to ON end. Upload code to the micro:bit and you will view LED flashing, with interval of 1s.

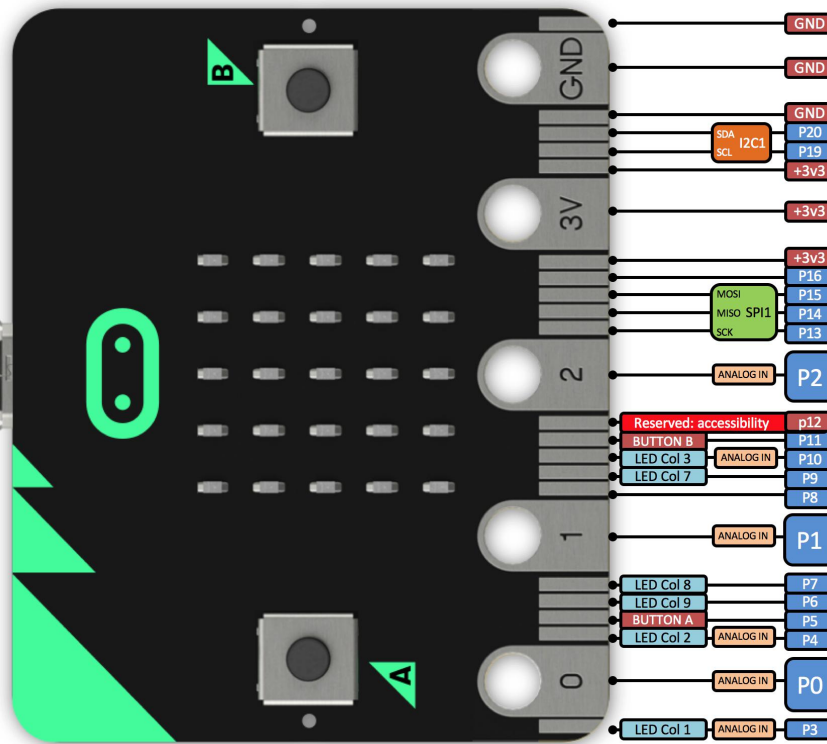
Project 14: Breathing Light

1.Description:

The light breath experiment is a little bit similar to the previous project. This time we connect the EASY Plug Red Led module to the EASY Plug Shield for micro bit V1.1. Connect the pin of LED module to P10 of micro:bit. From the Pinout diagram of micro:bit, you can get the P10 can be used as Analog IN.

This lesson you will learn how to control the brightness of LED on the

module, gradually becoming brighter and dimming, just like the LED is breathing.

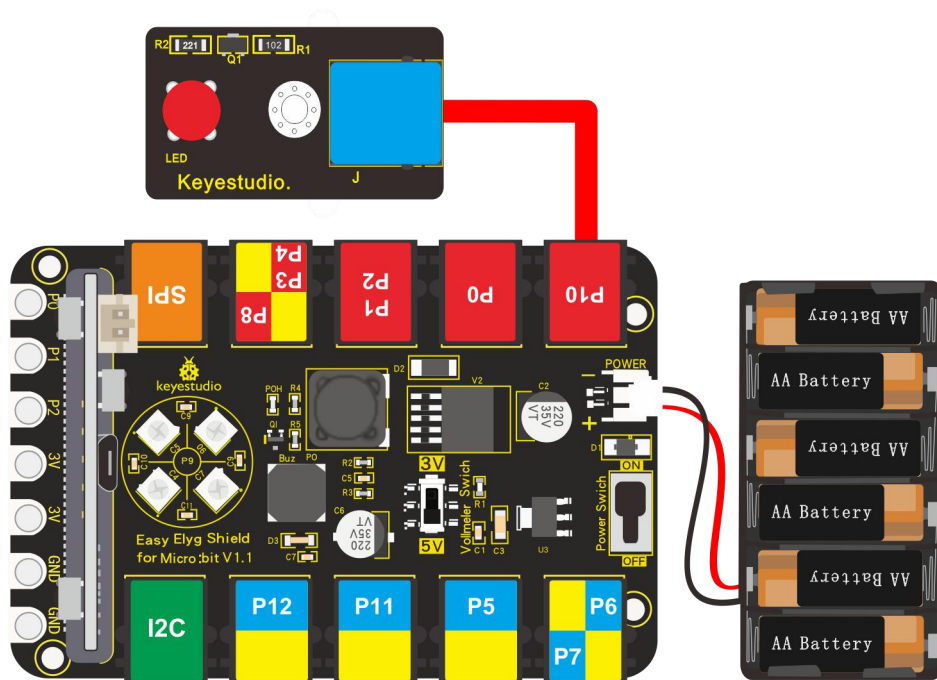


2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Red LED Module*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

3. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect red LED module to P10 of shield with a RJ11 cable, and plug in external power.

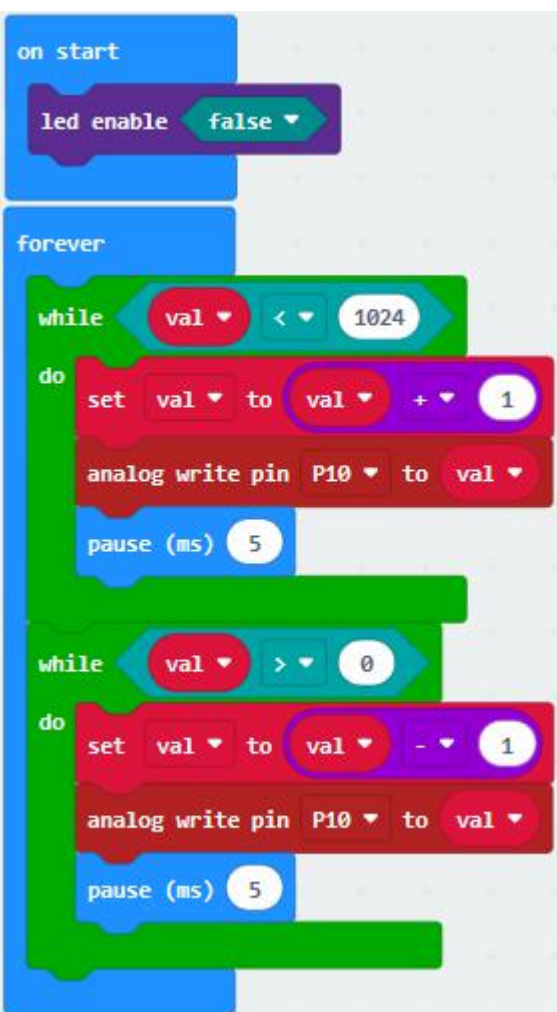


Note: Dial Voltmeter_Switch to 3V end

4. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.



“on start” : command block only runs once to start program.
 Turn off dot matrix on micro:bit
 The program under the block “forever” runs cyclically.
 When val<1024, run the program in the do block
 Set val to val+1
 Set analog value of P10 to val
 Delay in 5ms
 When val>0, run the program in the do block
 Set val to val-1
 Set the analog value of P10 to val.
 Delay in 5ms

5. Test Results:

Wire up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

You will find LED of module get brighter then darker, like human breath.

Project 15: Blink and Breath

1. Description:

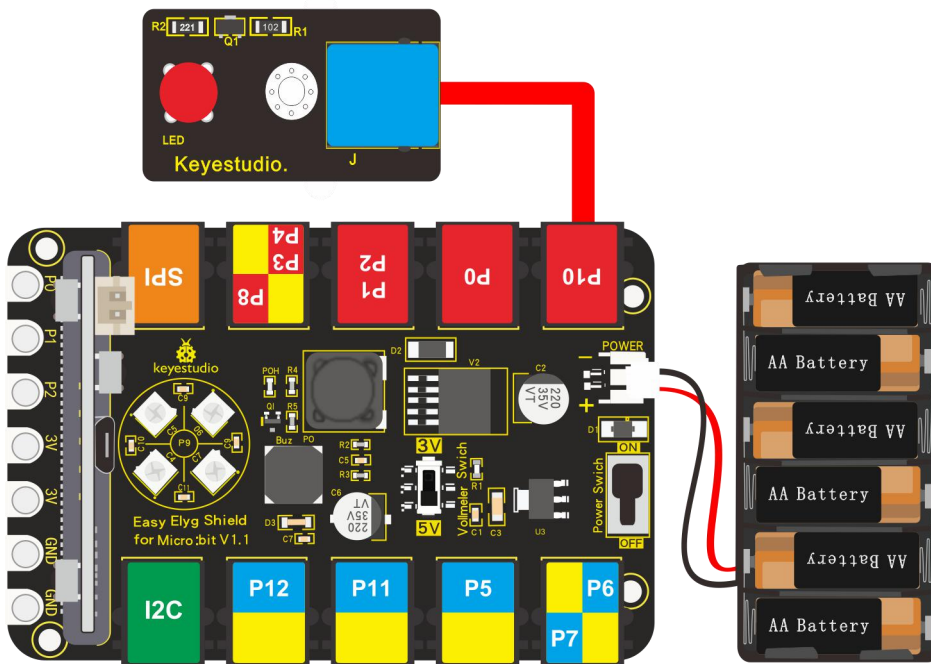
In this project, we will combine LED flash and breathing effect together.

2. What You Need:

- Micro:bit Board*1 EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Red LED Module*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

3. Wiring Up:

Insert the micro:bit onto EASY Plug shield, connect the red LED module to P10 of shield with a RJ11 cable and connect external power.



Note: Dial Voltmeter_Switch to 3V end

4.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.

```
on start
  led enable false

forever
  repeat 2 times
    do
      digital write pin P10 to 1
      pause (ms) 1000
      digital write pin P10 to 0
      pause (ms) 1000

  repeat 2 times
    do
      while val < 1024
        do
          set val to val + 1
          analog write pin P10 to val
          pause (ms) 5

      while val > 0
        do
          set val to val - 1
          analog write pin P10 to val
          pause (ms) 5
```

The image shows a block-based programming script. It starts with an 'on start' block containing an 'led enable' block set to 'false'. This is followed by a 'forever' loop. Inside the loop, there are two 'repeat 2 times' blocks. The first 'repeat' block contains a 'do' loop with two steps: 'digital write pin P10 to 1' followed by a 'pause (ms) 1000', and 'digital write pin P10 to 0' followed by another 'pause (ms) 1000'. The second 'repeat' block also contains a 'do' loop with two 'while' loops. The first 'while' loop is 'while val < 1024', and its 'do' block contains 'set val to val + 1', 'analog write pin P10 to val', and 'pause (ms) 5'. The second 'while' loop is 'while val > 0', and its 'do' block contains 'set val to val - 1', 'analog write pin P10 to val', and 'pause (ms) 5'.

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

Turn off dot matrix on micro:bit

The program under the block "forever" runs cyclically.

Repeat the program in the do block twice

Set P10 to high level (1) , turn on LED

Delay in 1000ms

Set P10 to low level (0) , turn off LED

Delay in 1000ms

Repeat the program in the do block twice

When val < 1024, run the program in the do block

Set val to val+1

Set the analog value of P10 to val

Delay in 5ms

5. Test Results:

Wire up, dial Voltmeter_Switch to 3V end, plug in power and dial Power_Switch to ON end. Upload program to micro:bit, LED flashes twice and shows breathing effect twice ceaselessly.

Project 16: Play Music

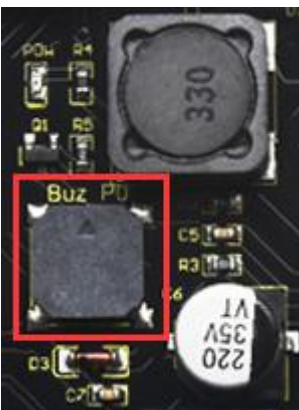
1.Description:

In this project, we will demonstrate how to play music with passive buzzer. Easy Plug shield comes with one. Let' s get started. (Passive buzzer is connected to P0 on Easy Plug shield)

2.What You Need:

- Micro:bit Board*1EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

Passive Buzzer Module:



Buzzers are divided into active buzzers and passive buzzers. The difference between them is a built-in vibration source.

We need 2K-5K square wave to drive passive buzzers because the buzzer on EASY Plug Shield doesn' t come with this kind of source.

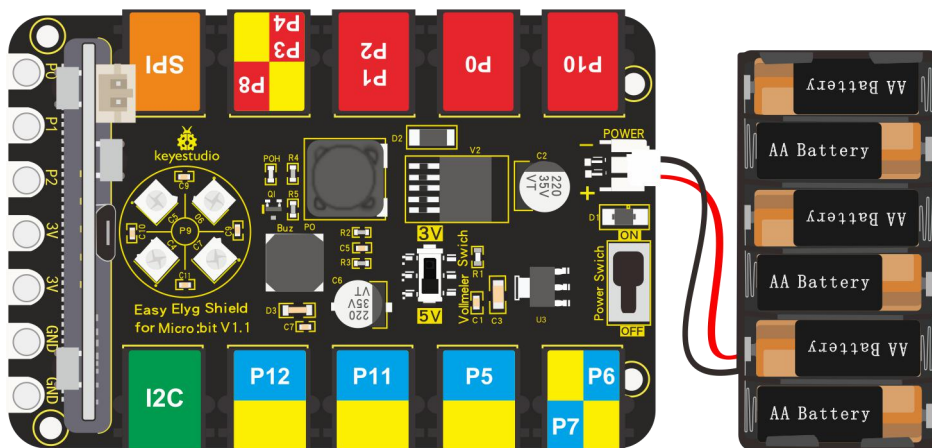
Different frequencies produce different sounds. You can use the micro:bit to produce a simple, interesting and melodic song.

3. Specification:

Working voltage: 3.3-5V

Interface type: Digital

4. Wiring Up:



Note: Dial Voltmeter_Switch to 3V end.

5. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.

Code 1:

```

on start
  led enable false

forever
  while i < 80
    do
      digital write pin P0 to 0
      pause (ms) 1
      digital write pin P0 to 1
      pause (ms) 1
      set i to i + 1
    end do
  end while
  set i to 0
  pause (ms) 100
  while i < 100
    do
      digital write pin P0 to 1
      pause (ms) 2
      digital write pin P0 to 0
      pause (ms) 2
      set i to i + 1
    end do
  end while
end forever
  
```

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

Turn off dot matrix on micro:bit

The program under the block “forever” runs cyclically.

When $i < 80$, run the program in the block

Set P0 to low level(0) to make passive buzzer silent

Delay in 1ms

Set P0 to high level(1) to make passive buzzer emit sound

Delay in 1ms

Set variable i to i+1

Set variable i to 0

Delay in 100ms

When $i < 100$, run the program in the do block

Set P0 to high level(1) to make passive buzzer emit sound

Delay in 2ms

Set P0 to low level(0) to make passive buzzer silent

Delay in 2ms

Set variable i to i+1

Code 2:

```

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

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

Turn off dot matrix on micro:bit

The program under the block “forever” runs cyclically.

Play tone High E for 2 beats

Play tone High E for 1 beat

Play tone high G for 2 beats

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 2 beats

Play tone high C for 2 beats

Play tone high D for 1 beat

Play tone high E for 2 beats

Play tone high E for 1/2 beat

Play tone high D for 1/2 beat

Play tone high D for 2 beats

Play tone high E for 2 beats

Play tone high F for 1 beat

Play tone high G for 2 beats

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 2 beats

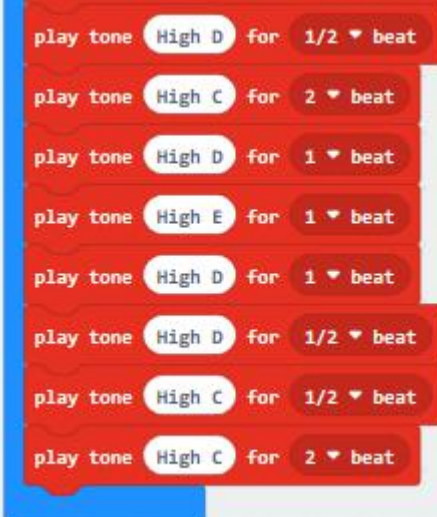
Play tone high D for 1 beat

Play tone high E for 1 beat

Play tone high D for 1 beat

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


Play tone high D for 1/2 beat
 Play tone high D for 1/2 beat
 Play tone high C for 1 /2 beat
 Play tone high C for 1 beat
 Play tone high D for 2 beats
 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
 Play tone high F for 1 /2 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
 Play tone high F for 1 /2 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 D for 1 beat
 Play tone high G for 1 beat
 Play tone high E for 1 beat
 Play tone high E for 2 beats
 Play tone high F for 1 beat
 Play tone high G for 2 beats
 Play tone high F for 1 beat
 Play tone high E for 1 beat
 Play tone high F for 1 /2 beat
 Play tone high E for 2 beats

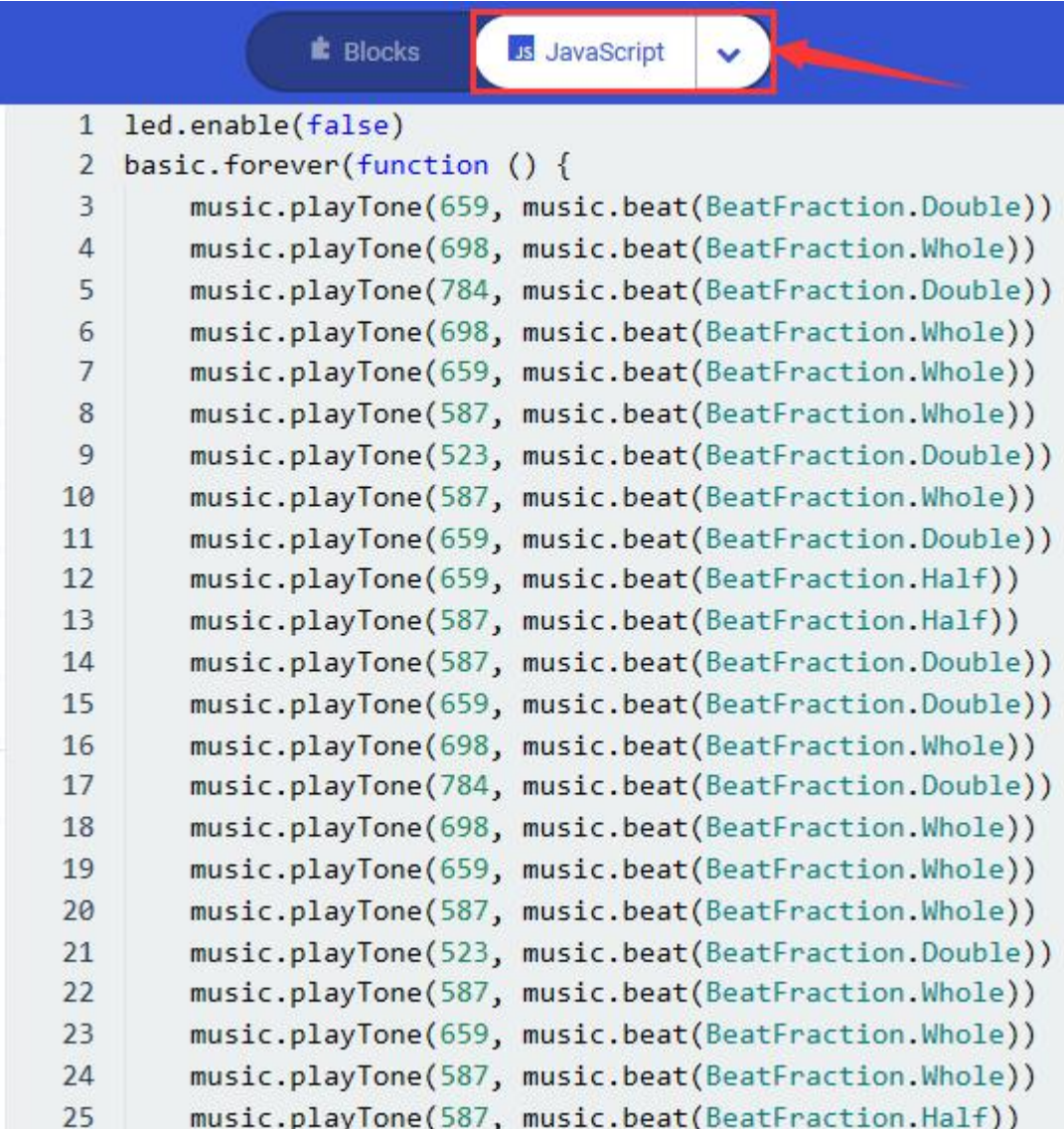


The image shows a vertical stack of Scratch 'play tone' blocks. Each block has a 'play tone' block, a frequency value (High D, High C, High E), a 'for' block, and a beat value (1/2, 1, 2). The blocks are arranged in a sequence that corresponds to the text on the right.

Block	Frequency	Beat
1	High D	1/2
2	High C	2
3	High D	1
4	High E	1
5	High D	1
6	High D	1/2
7	High C	1/2
8	High C	2

Play tone high D for 1/2 beat
 Play tone high C for 2 beats
 Play tone high D for 1 beat
 Play tone high E for 1 beat
 Play tone high D for 1 beat
 Play tone high D for 1/2 beat
 Play tone high C for 1/2 beat
 Play tone high C for 2 beats

Note: Click  to switch into JavaScript code, each frequency and beat of tone is shown below:



The image shows the JavaScript code generated from the Scratch blocks. The code is enclosed in a 'basic.forever' loop. Each line of code calls 'music.playTone' with a frequency and a beat value. The beat values are represented as 'BeatFraction.Double', 'BeatFraction.Whole', and 'BeatFraction.Half'.

```

1 led.enable(false)
2 basic.forever(function () {
3     music.playTone(659, music.beat(BeatFraction.Double))
4     music.playTone(698, music.beat(BeatFraction.Whole))
5     music.playTone(784, music.beat(BeatFraction.Double))
6     music.playTone(698, music.beat(BeatFraction.Whole))
7     music.playTone(659, music.beat(BeatFraction.Whole))
8     music.playTone(587, music.beat(BeatFraction.Whole))
9     music.playTone(523, music.beat(BeatFraction.Double))
10    music.playTone(587, music.beat(BeatFraction.Whole))
11    music.playTone(659, music.beat(BeatFraction.Double))
12    music.playTone(659, music.beat(BeatFraction.Half))
13    music.playTone(587, music.beat(BeatFraction.Half))
14    music.playTone(587, music.beat(BeatFraction.Double))
15    music.playTone(659, music.beat(BeatFraction.Double))
16    music.playTone(698, music.beat(BeatFraction.Whole))
17    music.playTone(784, music.beat(BeatFraction.Double))
18    music.playTone(698, music.beat(BeatFraction.Whole))
19    music.playTone(659, music.beat(BeatFraction.Whole))
20    music.playTone(587, music.beat(BeatFraction.Whole))
21    music.playTone(523, music.beat(BeatFraction.Double))
22    music.playTone(587, music.beat(BeatFraction.Whole))
23    music.playTone(659, music.beat(BeatFraction.Whole))
24    music.playTone(587, music.beat(BeatFraction.Whole))
25    music.playTone(587, music.beat(BeatFraction.Half))
  })
  
```

6. Test Results:

Wire up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code 1 to micro:bit. Then you will hear the buzzer emit two kind of sounds; if you download code 2 to the micro:bit, the song "Ode-to- Joy" will be played.

Project 17: RGB

1. Description:

EASY Plug shield comes with 2812 2x2 full color RGB lights, we will finish three experiments with 2812 2x2 RGB lights

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

2812 2x2 full color RGB:



2812 2x2 full color RGB module is a smart external

control LED light source that integrates control circuit and lighting circuit.

Each LED has the same appearance as a 5050 LED bead, and each component is a pixel point.

The pixel point includes an intelligent digital interface data latch signal shaping and amplifying driving circuit, as well as a high-precision internal oscillator and a 12V high-voltage programmable constant current control part, which effectively ensures that the color of the pixel point light is highly uniform.

The data protocol adopts the single-line return-to-zero code communication mode. After power-on and reset the pixel point, the S pin receives the data transmitted from the controller. And the 24-bit data are extracted by the first pixel and then sent to the data latch inside the pixel point.

LED has advantages of low voltage drive, environmental protection and energy saving, high brightness, wide scattering angle, good consistency, ultra low power, long life and so on.

3. Specification:

Working voltage: DC 5V

Power: 0.1W

Light source: SMD 5050 RGB

IC model: 4pcs/WS2811

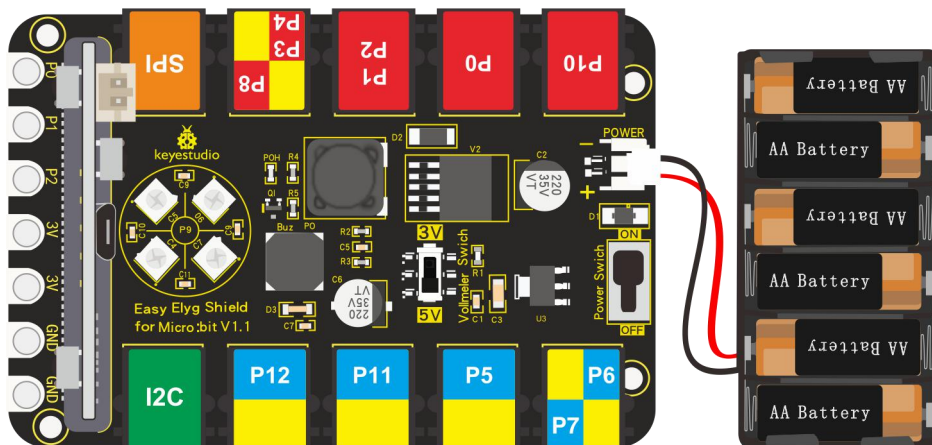
Gray level: 256 levels

Beam angle: 180°

Luminous color: can be adjusted to white, red, yellow, blue, green, etc.

by the controller

4.Wiring Up:



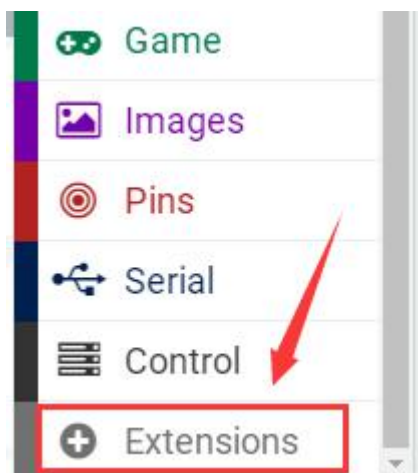
Note: Dial Voltmeter_Switch to 3V end

5. Test Code:

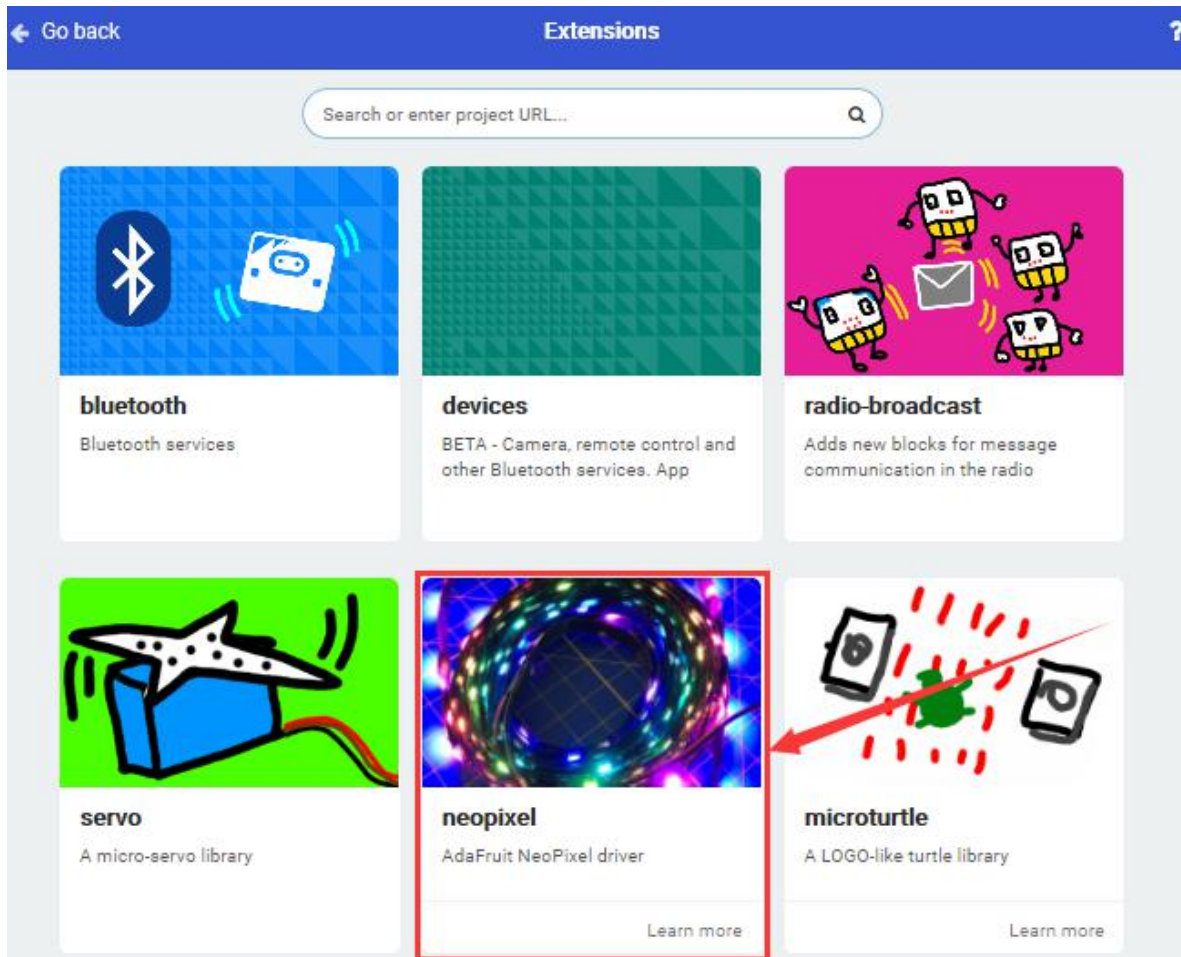
You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.

We need to set test code in library file, and add the library of "neopixe" .



Click "Extensions" → "neopixel" and click to download



You will view library "neopixel" in the editing blocks, as shown below:

The image shows a screenshot of a block-based programming environment's library panel. On the left, a sidebar lists various categories: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, Neopixel, and more. A red arrow points to the 'Neopixel' category. The main area displays the 'Neopixel' library with several blocks:

- set strip to NeoPixel at pin P0 with 24 leds as RGB (GRB format)
- set range to strip range from 0 with 4 leds
- strip show rainbow from 1 to 360
- strip show color red
- strip show bar graph of 0 up to 255
- strip show
- strip clear
- hue 0 saturation 0 luminosity 0
- strip shift pixels by 1
- strip rotate pixels by 1

Code 1:



The image shows a Scratch script for controlling an LED strip. It starts with an 'on start' block containing three steps: 'led enable' set to 'false', 'set strip' to 'NeoPixel at pin P9 with 4 leds as RGB (GRB format)', and 'strip clear'. This is followed by a 'forever' loop with 12 steps, each consisting of a 'strip show color' block and a 'pause (ms) 1000' block. The colors shown in sequence are red, orange, yellow, green, blue, indigo, violet, purple, and white.

```
on start
  led enable false
  set strip to NeoPixel at pin P9 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 only runs once to start program.

Turn off dot matrix on micro:bit

Set strip to NeoPixel at pin P9 with 4 leds as RGB

Turn off 4 pcs WS2812 RGB lights

The program under the block “forever” runs cyclically.

make all RGB show red color

make all RGB show red color

Delay in 1000ms

make all RGB show orange color

Delay in 1000ms

make all RGB show yellow color

Delay in 1000ms

make all RGB show green color

Delay in 1000ms

All of RGB show blue color

Delay in 1000ms

make all RGB show indigo color

Delay in 1000ms

make all RGB show violet color

Delay in 1000ms

make all RGB show purple color

Delay in 1000ms

make all RGB show white color

Delay in 1000ms

Code 2:

```
on start
  led enable false
  set strip to NeoPixel at pin P9 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 only runs once to start program.

Turn off dot matrix on micro:bit

Set strip to NeoPixel at pin P9 with 4 leds 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 pixel color of 4 pcs WS2812 RGB lights to red color

Strip show

Delay in 100ms

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

Turn off 4 pcs WS2812 RGB lights

Set pixel color of 4 pcs WS2812 RGB lights to orange color

Strip show

Delay in 100ms

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

Turn off 4pcs WS2812 RGB lights

Set pixel color of 4 pcs WS2812 RGB lights to yellow color

Strip show

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
do
  strip clear
  strip set pixel color at index to violet
  strip show
  pause (ms) 100
```

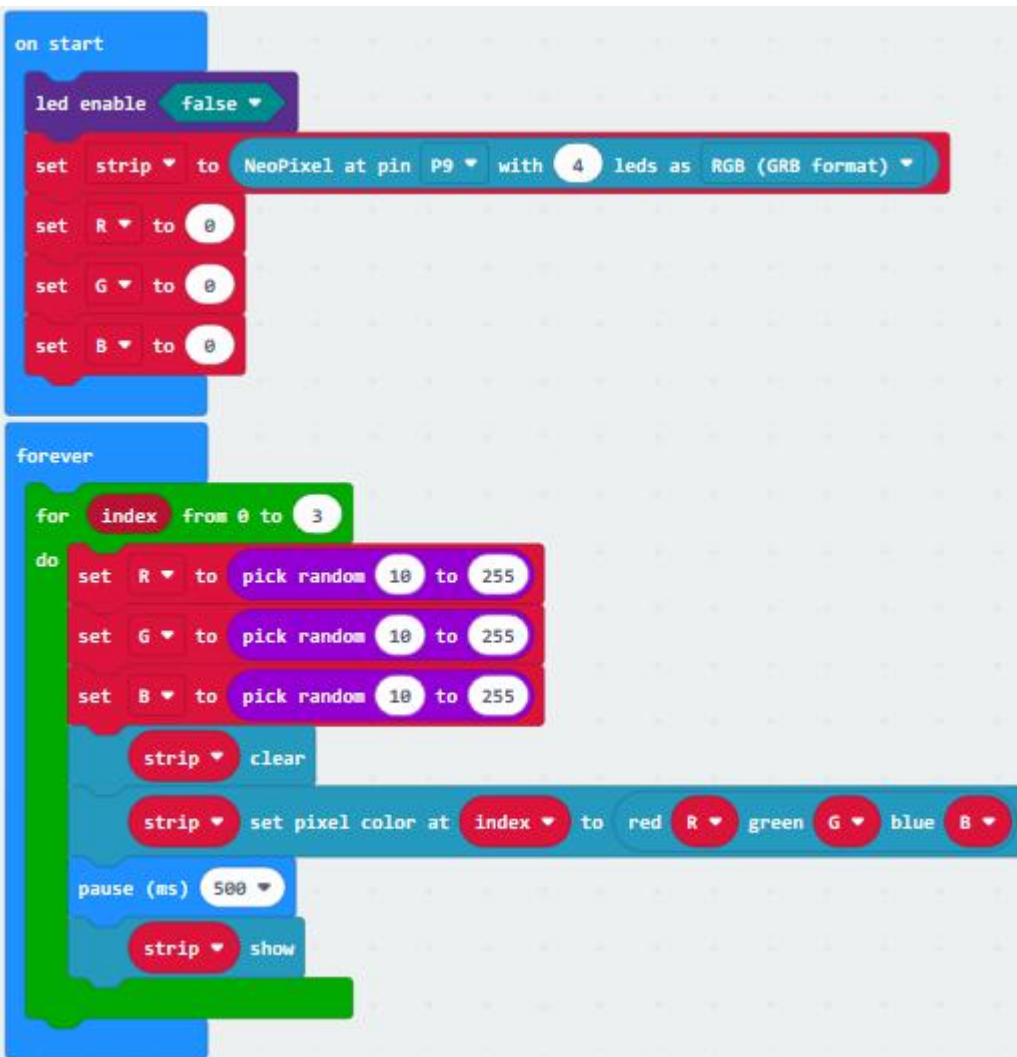
```
for index from 0 to 3
do
  strip clear
  strip set pixel color at index to purple
  strip show
  pause (ms) 100
```

```
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 4pcs WS2812 RGB lights
Set pixel color of 4 pcs WS2812 RGB lights to green color
Strip show
Delay in 100ms
For index from 0 to 3, execute the program under do block
Turn off 4 pcs WS2812 RGB lights
Set pixel color of 4 pcs WS2812 RGB lights to blue color
Strip show
Delay in 100ms
For index from 0 to 3, execute the program under do block
Turn off 4 pcs WS2812 RGB
Set pixel color of 4 pcs WS2812 RGB lights to indigo color
Strip show
Delay in 100ms
For index from 0 to 3, execute the program under do block
Turn off 4 pcs WS2812 RGB lights
Set pixel color of 4 pcs WS2812 RGB lights to violet color
Strip show
Delay in 100ms
For index from 0 to 3, execute the program under do block
Turn off 4 pcs WS2812 RGB
Set pixel color of 4 pcs WS2812 RGB lights to violet color
Strip show
Strip refreshes to display
Delay in 100ms
For index from 0 to 3, execute the program under do block

Turn off 4 pcs WS2812 RGB
Turn off 4 pcs
Set pixel color of 4 pcs WS2812 RGB lights to white color

Strip show
Delay in 100ms

Code 3:

```
on start
  led enable false
  set strip to NeoPixel at pin P9 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
```

The image shows a Scratch script for controlling a NeoPixel LED strip. The script is divided into two main sections: 'on start' and 'forever'. In the 'on start' section, the LED strip is initialized by setting 'led enable' to 'false', and the 'strip' variable is set to 'NeoPixel at pin P9 with 4 leds as RGB (GRB format)'. The red, green, and blue color variables (R, G, B) are all set to 0. The 'forever' loop contains a 'for' loop that iterates over the index of the LEDs from 0 to 3. Inside this loop, the red, green, and blue values are randomly selected between 10 and 255. The LED strip is then cleared, and the selected colors are set for the current index. A 500ms pause is added after each iteration, and the strip is shown.

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

Turn off micro:bit LED dot matrix

Set strip to NeoPixel at pin P9 with 4 leds as RGB

Set strip to initialization o

Set variable R to 0

Set variable G to 0

Set variable B to 0

The program under the block “forever” runs cyclically.

When value of variable index is in 0-3, execute the program in the 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 lights on the strip

Set pixel color of 4 pcs WS2812 RGB lights to RGB

Set 4pcs WS2812 RGB to

Delay in 500ms

strip refreshes to display

6. Test Results:

Wiring up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and download code 1 to micro:bit, WS2812RGB lights display different color.

Download code 2 to micro:bit, WS2812RGB show same color like flow light.

Download code 3 to micro:bit, each WS2812RGB shows random color like flow light.

Project 18: Button Control

1. Description:

Button sensor is commonly used component. In this chapter, we will show you how to control an LED with a button sensor.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Digital Push Module*1
- EASY Plug White LED Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Button



The EASY Plug digital push module is a tidy little design that lets you control a DC power source using an everyday tactile

button.

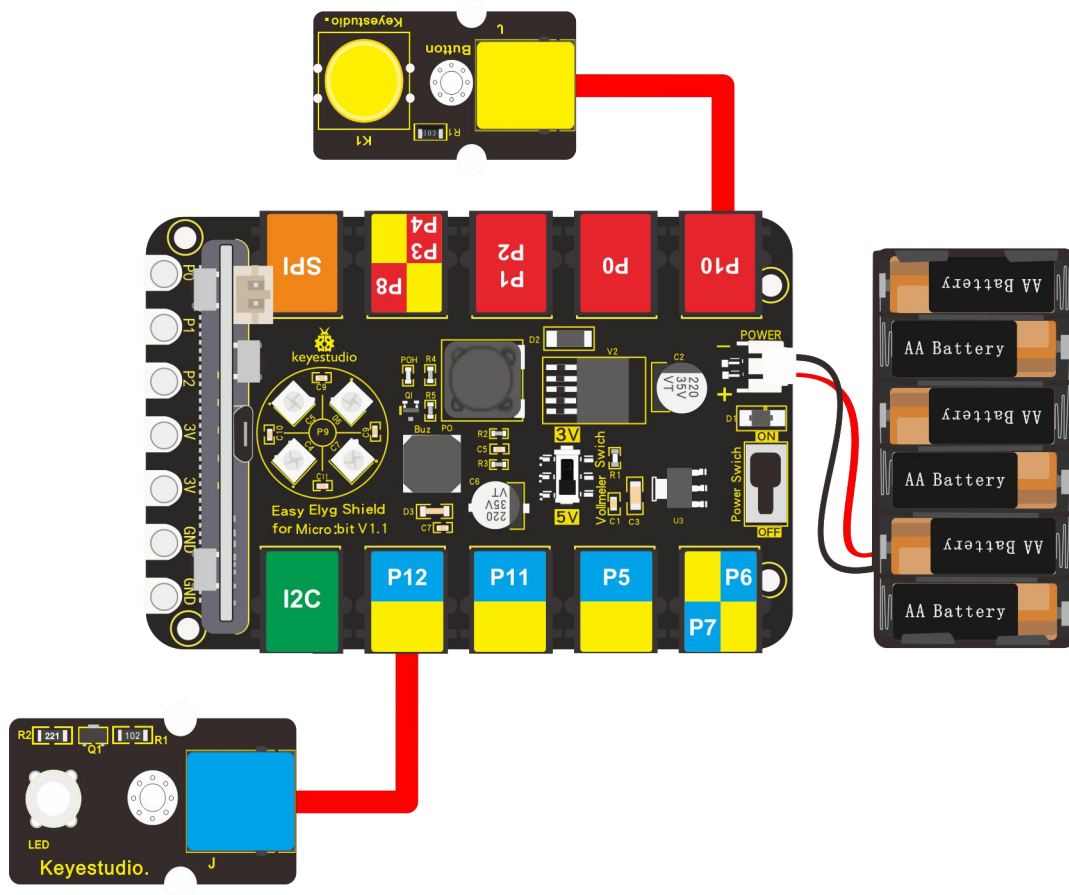
It can be connected to circuit. When it is pressed, the circuit is connected, when released, the circuit is cut.

3. Specification:

- Connector: Easy plug
- Supply Voltage: 3.3V to 5V
- Large button and high-quality top cap
- Sensor type: Digital
- Weight: 5.6g

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, wire up digital push module and LED module to P10 and P12 port of shield and RJ11 cables. Don't forget to connect battery holder.



Note: Dial Voltmeter_Switch to 3V end

5.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.

```
on start
  led enable false
  set PushCounter to 0
  set State to 0
  set Laststate to 0

forever
  serial write value "x" = PushCounter
  set State to digital read pin P10
  if State != Laststate then
    if State = 1 then
      set PushCounter to PushCounter + 1
    +
    +
  pause (ms) 100
  set Laststate to State
  if remainder of PushCounter + 2 = 0 then
    digital write pin P12 to 1
  else
    digital write pin P12 to 0
  +
```

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

Turn off dot matrix on micro:bit

Set variable PushCounter to 0

Set variable State to 0

Set variable Laststate to 0

The program under the block "forever" runs cyclically.

Serial writes "x" =the value of PushCounter

Set the digital signal read by P0 to state

If State≠Laststate, execute the program under then block

If State=1, execute the program under then block

Set PushCounter to PushCounter+1

Delay in 100ms

Set the value of State to variable4 Laststate

If the remainder of PushCounter divided by 2 is 0, execute the program under then block

Set P12 to high level(1), LED turns on

If the remainder of PushCounter divided by 2 is not 0, execute the program under else block

Set P12 to low level(1), LED turns off

6. Test Results:

Wiring up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Press button sensor, LED will be on; press button again, LED off.

Project 19: Tilt Control

1. Description:

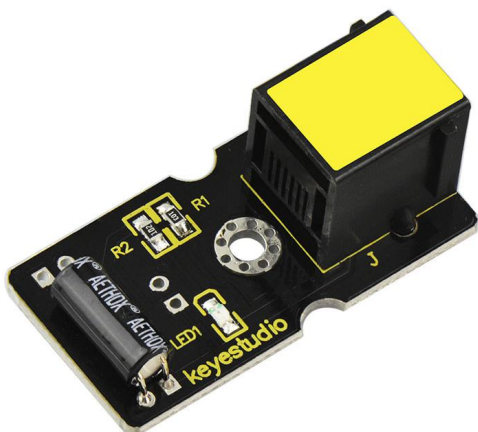
Tilt sensor (tilt ball switch) allows you to detect orientation or inclination. They are small, inexpensive, low-power and easy-to-use.

We will try to control an LED with a tilt sensor.

2. What You Need:

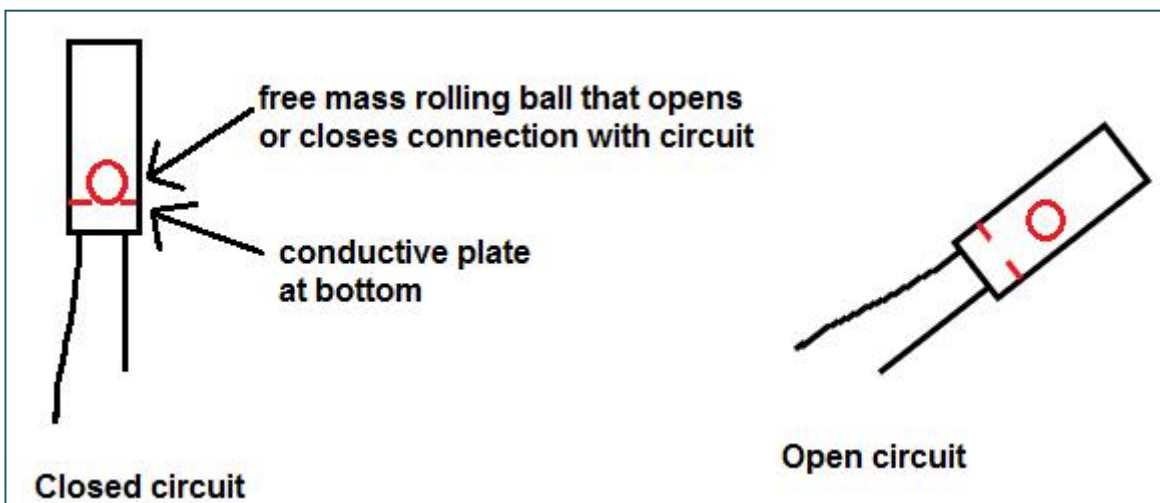
- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Digital Tilt Sensor*1
- EASY plug Red LED Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Digital Tilt Sensor:



This EASY Plug digital tilt sensor module mainly integrates a tilt sensor. The tilt sensor is a component that can detect the tilt of an object. It uses the ball in the switch to change different inclination angles to trigger the circuit. When the ball in the tilt switch runs from one end to the other due to the vibration of an external force, the tilt switch will be turned on, otherwise it will be turned off. The tilt sensor can be applied in orientation detection and alarm.

Schematic Diagram



4. Specification:

Connector: Easy plug

Supply Voltage: 3.3V to 5V

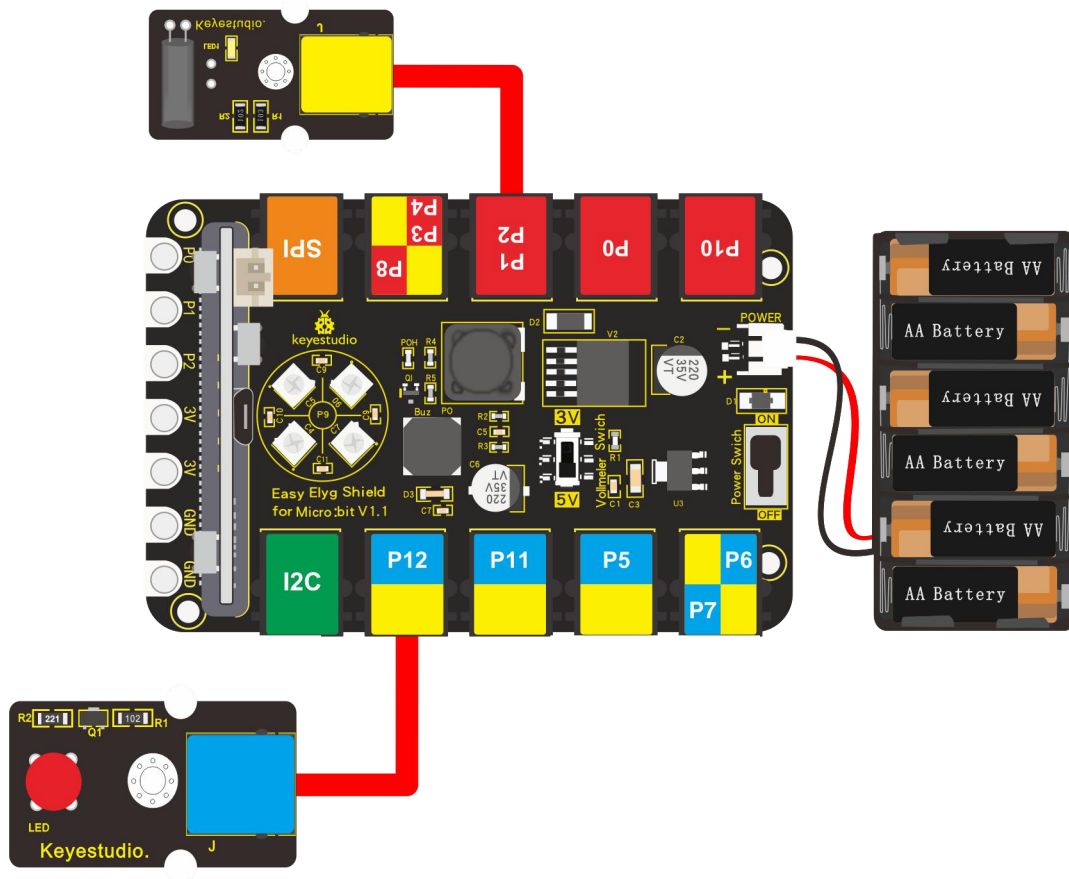
Sensor type: Digital

Dimensions: 39mm*20mm*18mm

Weight: 4.8g

4.Wiring Up:

Insert the micro:bit onto EASY Plug shield, connect a digital tilt sensor and an LED module to P1 and P12 port of shield. And plug in power.

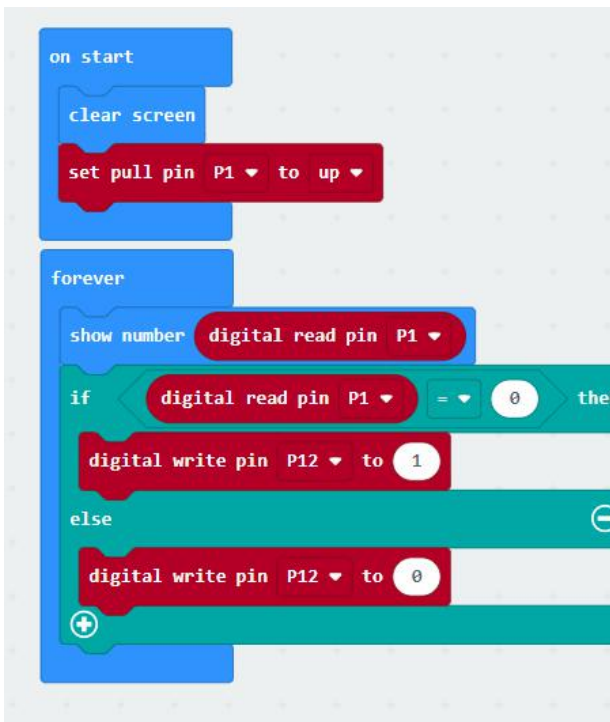


Note: Dial Voltmeter_Switch to 3V end

5.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.



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

Turn off LED dot matrix

Pull up the voltage of P1

The program under the block “forever” runs cyclically.

Micro:bit shows the digital signal read by tilt sensor

If the digital signals read by P1 is 0, execute the program under then block

Set P12 to high level(1), LED turns on

If the digital signals read by P1 is 1, execute the program under else block

Set P12 to low level(0), LED turns off

6.Test Result:

Wire up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

When tilt sensor is inclined to the right, micro:bit shows 0, and LED is on; when inclined to the left, micro:bit displays 1 and LED is off.

Project 20: Relay Module

1. Description:

Generally, we drive electronic devices with 220V alternating current and connect a switch in 220V circuit.

We design the Easy Plug relay module with NO and NC end to constrain from the risk of electricity leakage.

In this experiment, we will show you how to control a relay module.

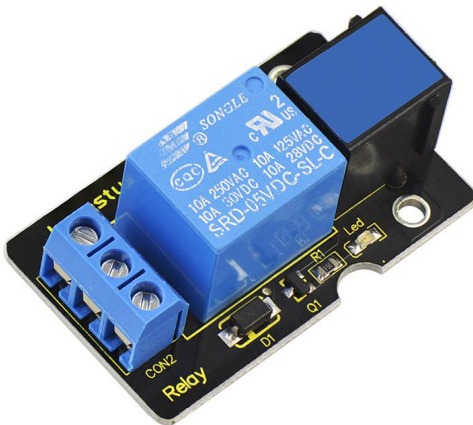
This module integrates a high-quality relay, basically an electrically controlled mechanical switch.

It can be controlled through the digital IO port, such as lamps, motors and other high current or high voltage devices.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug Relay Module*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Relay Module:



It is one of the most important controlled elements, which is widely used to control the lighting, communications, remote sensing, electrical and other equipment.

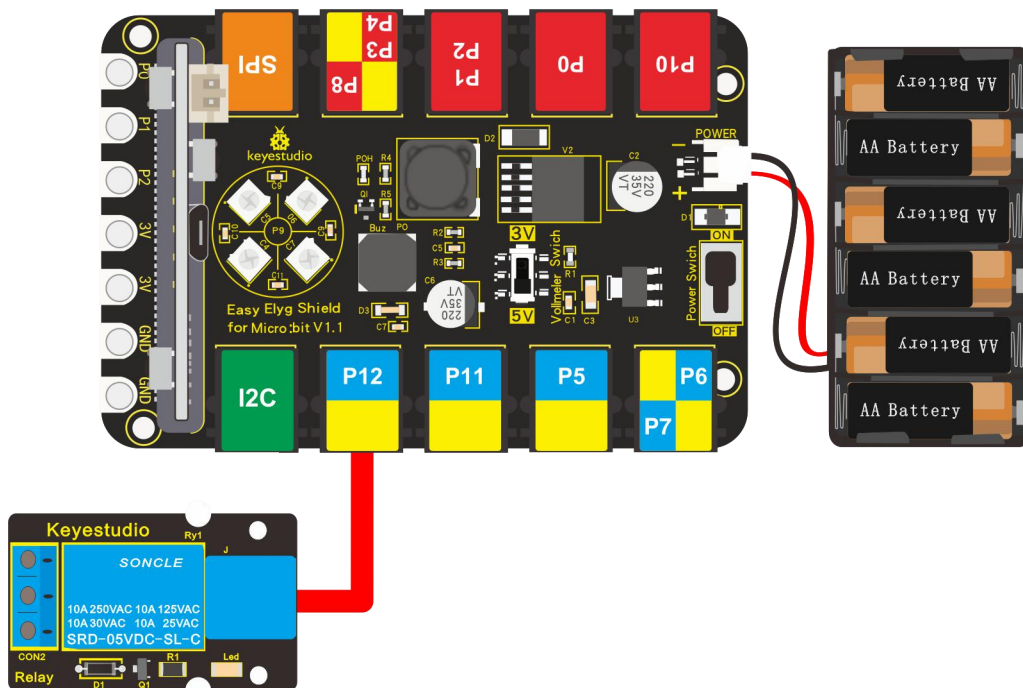
When S end is set high level, relay is driven, that is NO is connected, NC off; when S end is low level, relay is closed, that is NO is disconnected and NC is connected.

5. Specification:

- Type: Digital
- Rated current: 10A (NO) 5A (NC)
- Maximum switching voltage: 150VAC 24VDC
- Interface: Digital
- Control signal: TTL level
- Rated load: 8A 150VAC (NO), 10A 24VDC (NO), 5A 250VAC (NO/NC), 5A 24VDC (NO/NC)
- Maximum switching power: AC1200VA DC240W (NO), AC625VA DC120W (NC)
- Contact action time: 10ms
 - Size: 40*28mm
 - Weight: 15g

6. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect relay module to P12 of shield with a JR11 cable, plug in power.

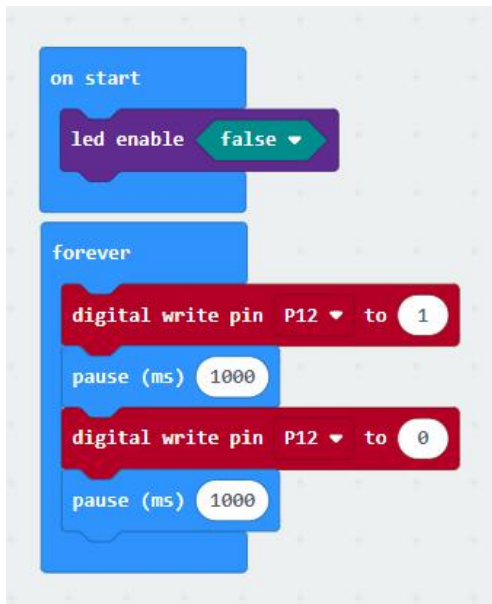


Note: Dial Voltmeter_Switch to 5V end

5.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.



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

Turn off dot matrix on micro:bit

The program under the block "forever" runs cyclically.

Set P12 to high level(1), relay module is connected

Delay in 1000ms

Set P12 to low level(0), relay module is disconnected

Delay in 1000ms

6.Test Result:

Wiring up, dial Voltmeter_Switch to 5V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

The relay module is connected and disconnected ceaselessly, with interval of 1s.

Project 21: Crash Sensor

1. Description:

We detect collision with a crash sensor. When the metal switch is touched, it will output low level signals; when the metal switch is not touched, high level will be remained.

We will control an LED with a collision sensor

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Crash Sensor*1
- EASY Plug White LED Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6



EASY Plug Crash Sensor:

Crash sensor, an electronic switch, is a digital switch input module.

1. When collision happens upfront of

where crash module is installed, module outputs low level signal; no collision, outputs high level signal.

2. With a mounting hole, convenient for fixation on any devices.

3. PCB size: 3.1cm * 2.1cm

4. With switch indicator light, if there is collision, LED on; if no collision, LED off.

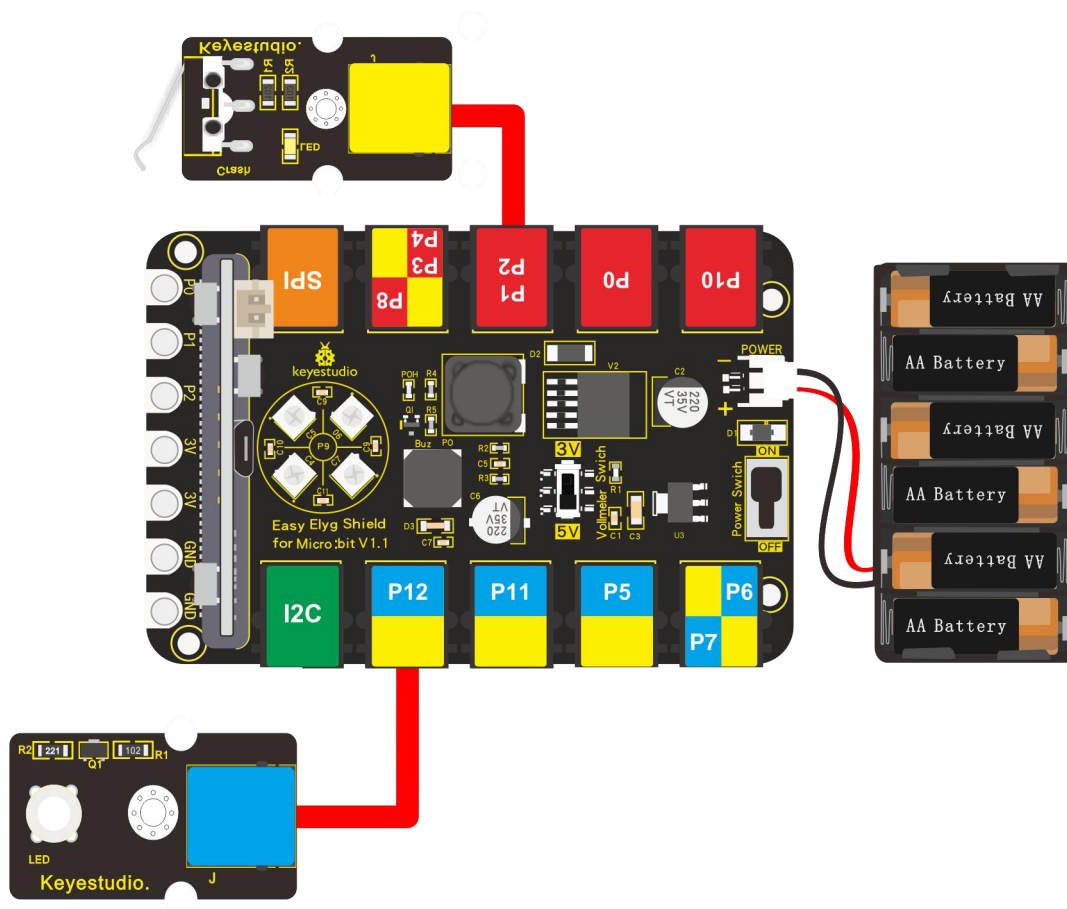
5.Connector: Easy plug

6.On-board status indicator LED

7.M3 mounting hole, convenient for fixation on other devices.

3. Wiring Up:

Insert micro:bit onto EASY Plug shield. Connect crash sensor and LED module to P1 and P12 port of shield. Plug in external power.



Note: Dial Voltmeter_Switch to 3V end

4. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program. The following test code is as for your reference.



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

Clear screen

The program under the block "forever" runs cyclically.

Micro:bit shows the digital signal (1/0) read by crash sensor

Micro:bit will show the digital signal read by

If the digital signal read by P1 is 0 (crash), execute the program under then block

Set P12 to high level (1), LED turns on

When the digital signal read by P1 is 1, no crash, execute the program under else block

Set P12 to low level (0), LED turns off

5. Test Result:

Wire up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

When the metal switch of crash sensor is pressed, the value low level(0) will be displayed and the LED will be on; on the contrary, micro:bit will show high level(1), LED will be off.

Project 22: Follow Black Line

1.Description:

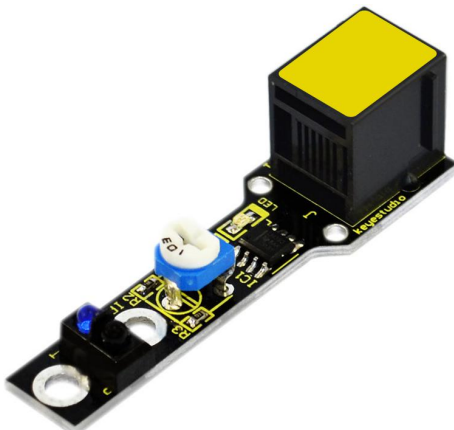
We will make smart robot follow the black lines to drive. Does it sound unbelievable?

Combine line tracking sensor with the micro:bit, then we could achieve what we want.

2.What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Line Tracking Sensor*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Line Tracking Sensor:



As an IR sensor, the line tracking sensor can detect black and white lines.

It has a TCRT5000 photoelectric sensor. Infrared reflectivity of color is different ,

which is applied to convert strong and weak echoed signal into current signal. The signal end will output high level when no object or black line is detected; otherwise, the low level will be output. As a result, we could determine color by high or low level from signal end.

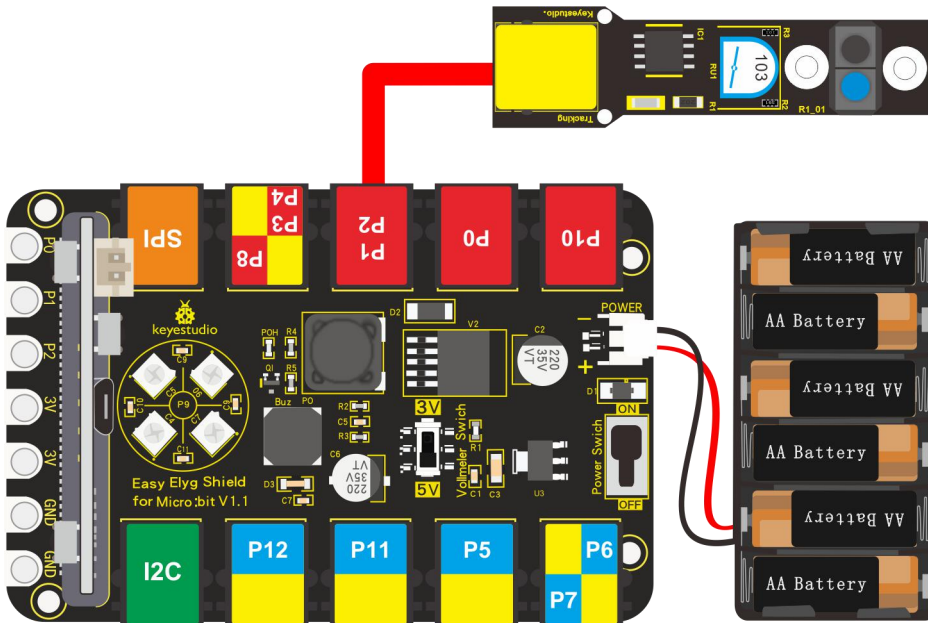
Note: rotate the potentiometer to keep LED in on-and-off state that stands for high sensitivity.

3. Specification:

- Power supply: +5V
- Operating current: <10mA
- Operating temperature range: 0°C ~ + 50°C
- Output interface: Easy plug
- Output Level: TTL (Black for HIGH output, White for LOW output)
- Detection Height: 0-3 cm

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect line tracking sensor to P1 port of shield, and plug in power.



Note: Dial Voltmeter_Switch to 3V end

5.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.

	<p>“on start” : command block only runs once to start program.</p> <p>Turn off LED dot matrix</p> <p>The program under the block “forever” runs cyclically.</p> <p>Micro: bit shows the digital signal read by line tracking sensor(1/0)</p> <p>Delay in 500ms</p>
--	--

6.Test Result:

Wire up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

The signal end will output high level when no object or black line is detected; otherwise, the low level will be output. As a result, we could determine color by high or low level from signal end.

Project 23: Magnetic Detection

1.Description:

Hall magnetic sensor has the characteristic of high sensitivity, quick-response, high reliability and high performance.

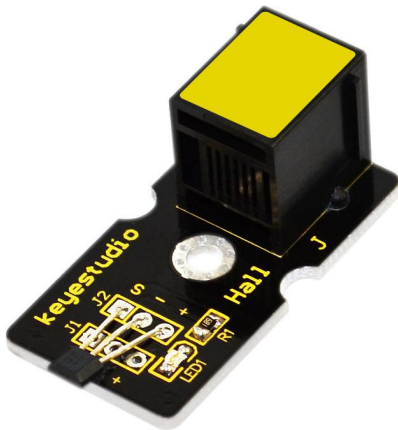
We will teach you how to control the on-and-off state of LED and detect if there is magnetic field with hall magnetic sensor.

2. What You Need:

- Micro:bit Board*1EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Hall Magnetic Sensor*1
- EASY Plug Blue LED Module*1

- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Hall Magnetic Sensor:



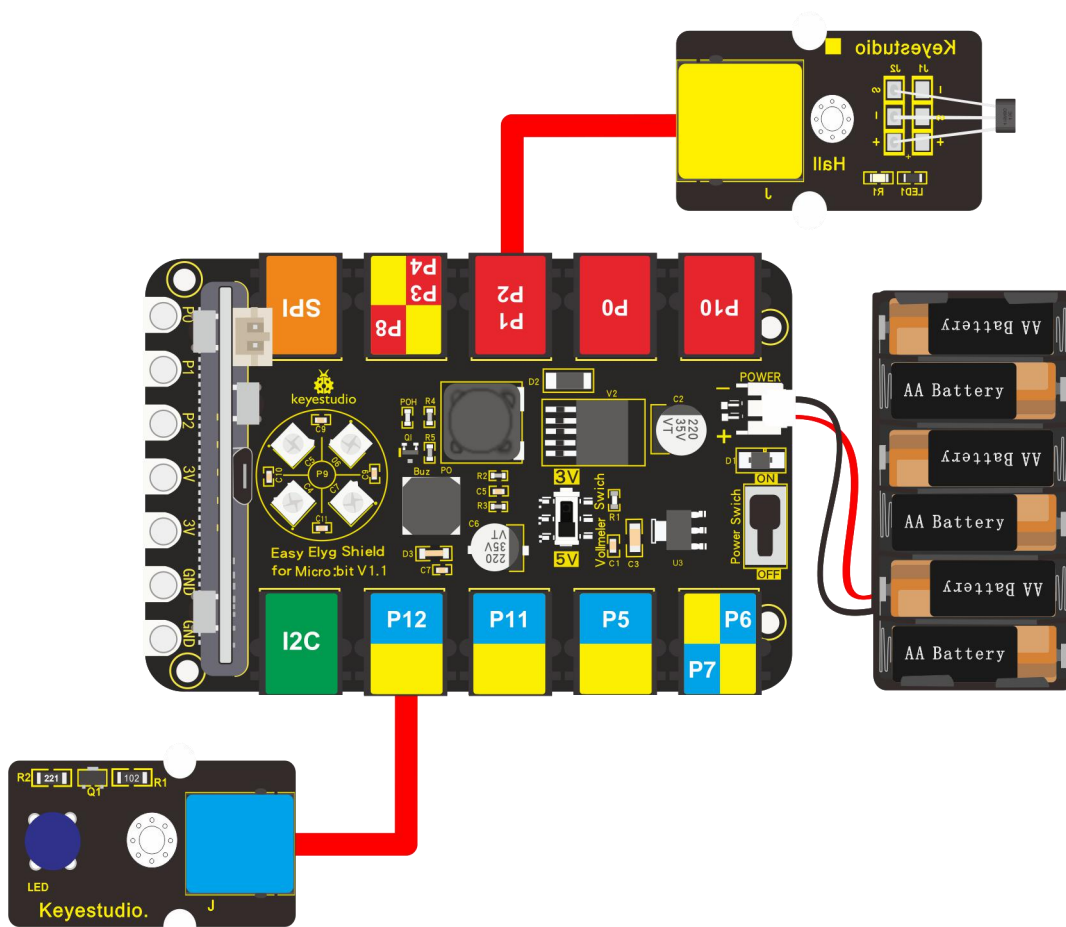
Applied to detect magnetic field and output digital signals, this Hall magnetic sensor adopts a A3144E component. It can detect if there exists magnetic field rather than how strong it is.

3. Specification:

- Power supply: +5V
- Sensing magnetic materials
- Detection range: up to 75px
- Output: Digital High/Low
- Detection range and magnetic field strength are proportional

4. Wiring Up:

Insert the micro:bit onto EASY Plug shield, connect the hall magnetic sensor and a blue LED module to P1 and P12 port of shield with 2 RJ11 cables. And plug in power.



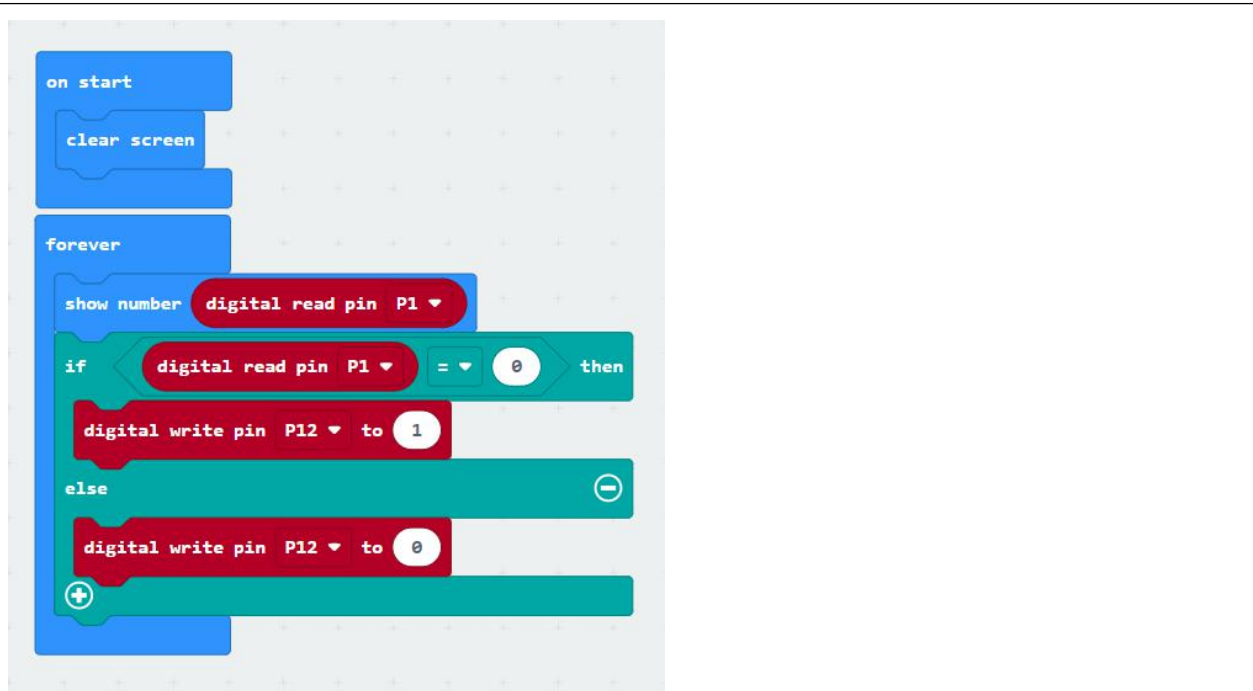
Note: Dial Voltmeter_Switch to 3V end

5. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.



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

Turn off LED dot matrix

The program under the block “forever” runs cyclically.

Micro:bit shows the digital signals read by hall magnetic sensor.

If the digital signal is 0, there exists magnetic field, execute the program under then block.

Set P12 to high level(1), LED turns on

If the digital signal is 1, there is no magnetic field, execute the program under else block.

Set P12 to low level(0), LED turns off

6. Test Result:

Wire up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Place a magnetic bead nearby the hall magnetic sensor, micro:bit will display 0(low level) and LED will be on if sensor detects the magnetic field; conversely, 1 will appear on micro:bit and LED will be off.

Project 24: 4-digit LED Display

1. Description:

In this lesson, we will teach you how to display numbers on EASY Plug 4-digit LED module.

2. What You Need:

- Micro:bit Board*1EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug 4-digit 8-segment Display Module*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY plug 4-digit LED display



This is a 4-digit 0.36'' common anode LED display module, a 12-pin display module with score points.

The driver chip used in the matrices is TM1637, using only two signal cables to make the microcontroller control the 4-digit LED display.

The four pins of LED display are GND、VCC、DIO、CLK. (GND is ground, VCC is for power supply, DIO is data IO pin, CLK is clock signal pin.)

The module pins are extended into Registered jack, so you can easily connect it to EASY Plug control board using a RJ11 cable.

This module should be used together with EASY plug control board.

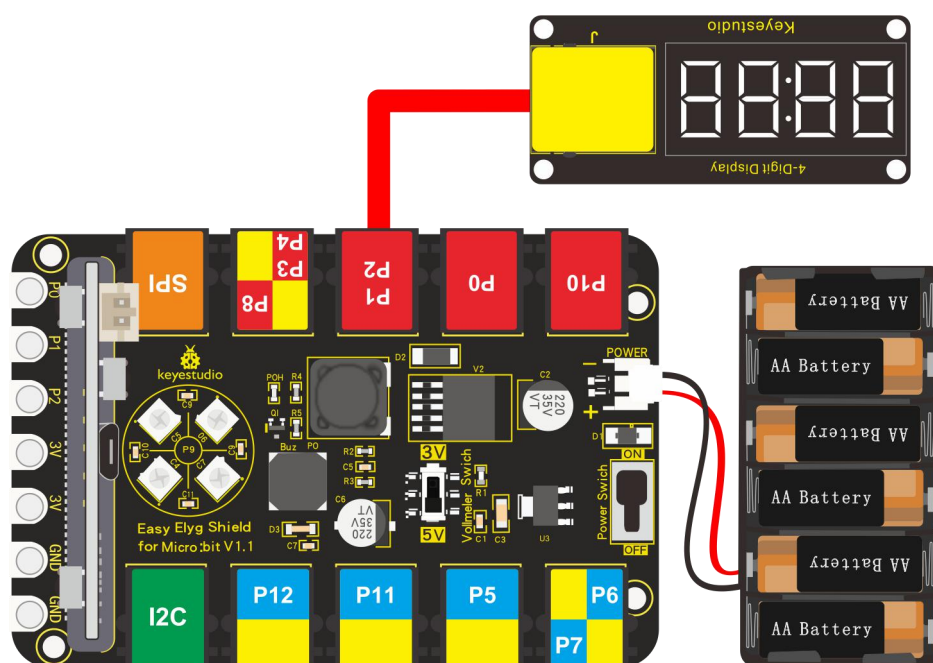
3. Specification:

- Specification:
- Working voltage: DC 5V
- Operating temperature

- range: -40 ~ +85°C
- Size: 49.6*23 MM
- Environmental protection attributes: ROHS

4. Wiring Up:

Insert the micro:bit onto the EASY Plug shield, and connect the 4-digit tube module to P1-P2 port of shield.



Note: Dial Voltmeter_Switch to 3V end

5. Test Code:

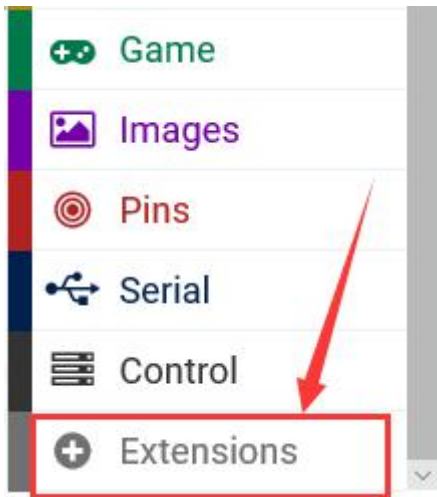
You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

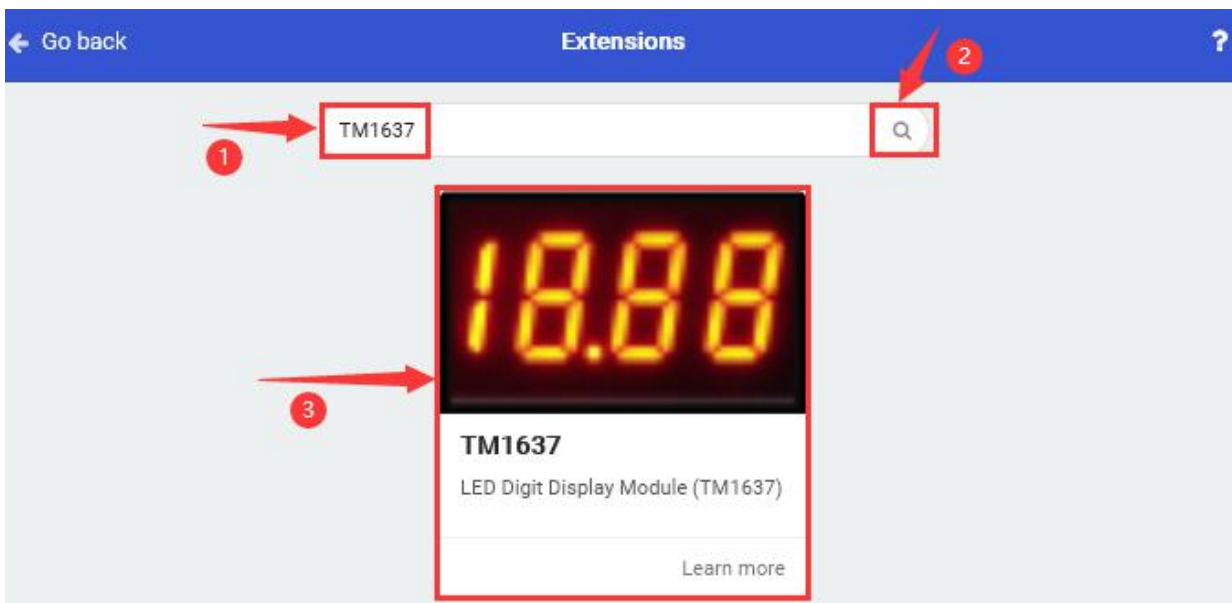
The following test code is as for your reference.

Add the library of 4-digit tube module, as shown below:

Click "Extensions"




Input "TM1637" and search, as shown below, click to download



After installing the library of 4-digit tube display module, you could view it in the editing blocks.

Search... Q

- Basic
- Input
- Music
- Led
- TM1637**
- Radio
- Loops
- Logic
- Variables
- Math
- Advanced
- Functions
- Arrays
- Text
- Game
- Images



TM1637

CLK P1

DIO P2

intensity 7

LED count 4

tm show number 0

tm show hex number 0

tm show digit 5 at 0

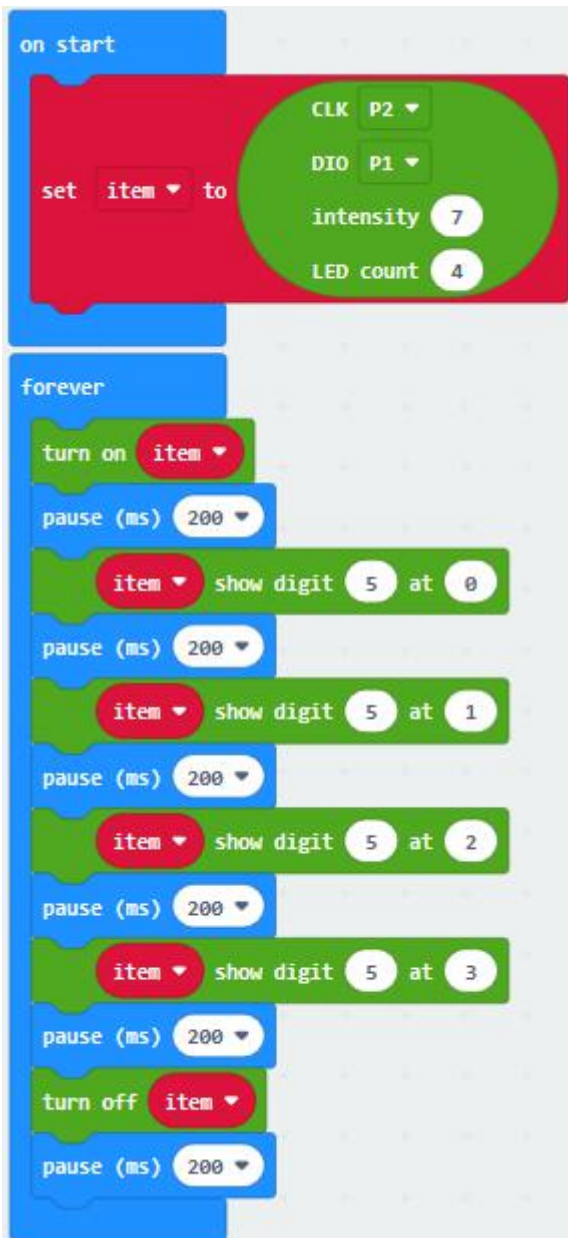
turn on tm

turn off tm

clear tm

tm DotPoint at 1 show true

tm set intensity 7



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

Set item to CLK P2 DIO P1 intensity 7 LED count 4

The program under the block "forever" runs cyclically.

Open 4-digit tube

Delay in 200ms

Show 5 at the zero bit on 4-digit tube module.

Delay in 200ms

Show 5 at the first bit on 4-digit tube module.

Delay in 200ms

Display 5 at the second bit on 4-digit tube module.

Delay in 200ms

Display 5 at the third bit on 4-digit tube module.

Delay in 200ms

6. Test Result:

Wiring up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

The 4-digit tube module shows "5" from 0 bit to the third bit(From left to right are 0, 1, 2, and 3 bits). then the number "5555" flashes.

Project 25: Light Interrupter

1. Description:

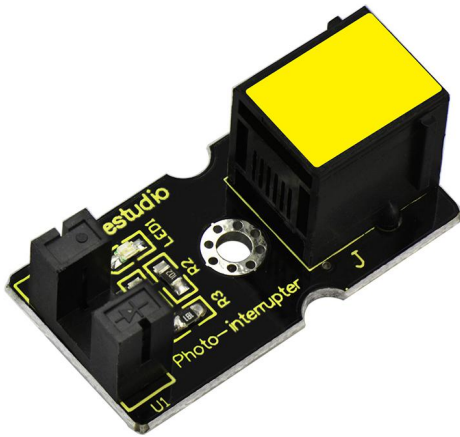
In our daily life, we often need to count and take measurement.

We could achieve goal by the combination of photo interrupter and microcontroller. We connect the photo interrupter to shield so as to control LED.

2. What You Need:

- Micro:bit Board*1 EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Light Interrupter*1
- EASY Plug Red LED Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Light Interrupter:



This is a high performance EASY Plug photo interrupter module.

The upright part on the module combines an infrared light emitting diode and shielded infrared detector.

By emitting a beam of infrared light from one end to another one, the sensor can detect an object when it passes through the beam.

Useful for many applications such as optical limit switches, pellet dispensing, general object detection, etc.

3. Specification:

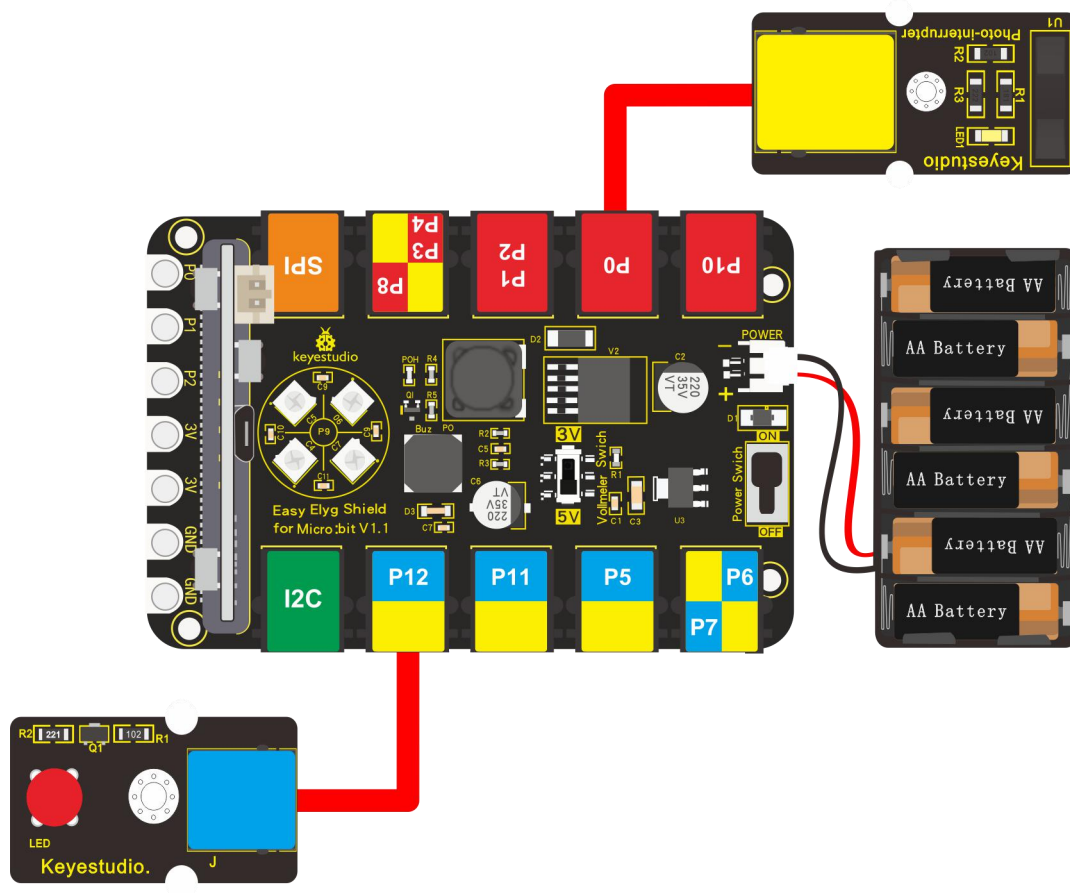
Support quick response; highly sensitive

Interface: Easy plug

Supply Voltage: 3.3V to 5V

4. Wiring Up:

Insert the micro:bit onto the EASY Plug shield, connect light interrupter module and LED module to P0 and P12 port of shield.



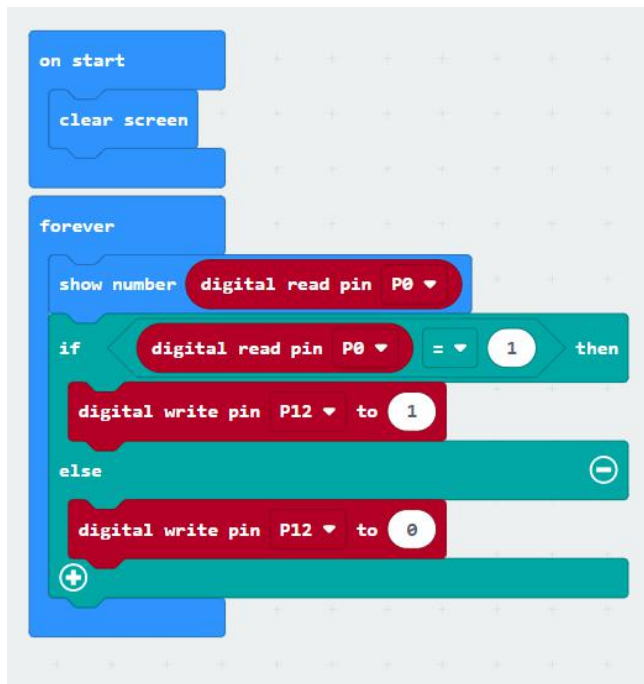
Note: Dial Voltmeter_Switch to 3V end

5. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.



6.Test Results:

Wire up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Micro:bit will display high level(1) and LED will be on if there is object goes through U slot of light interrupter; if not, LED will be off.

Project 26: EASY Plug Reed Switch Module

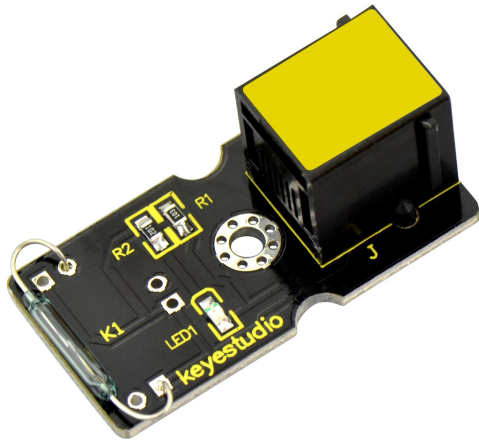
1.Description:

In this project, we will detect magnetic field with reed switch, shield and micro:bit. We've taken advantage of hall magnetic sensor to detect magnetic field in project 20. What's the difference between hall magnetic sensor and reed switch module? Let's get started.

2.What You Need:

- Micro:bit Board*1 EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug Magnetic Switch *1
- EASY plug Red LED Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Reed Switch Module:



This is a small device called a reed

switch on the module.

When the device is exposed to a magnetic field, the two ferrous materials inside the switch pull together and the switch closes.

When the magnetic field is removed, the reeds separate and the switch opens. This makes for a great non-contact switch.

You can mount reed switch on the door for alarming purpose or as switches.

This sensor needs to be used together with EASY plug control board.

The reed switch is applied widely in home appliance, automobile, communication, industrial, health care and security, as well as other electronic devices like door magnet, reed relay and level gauge.

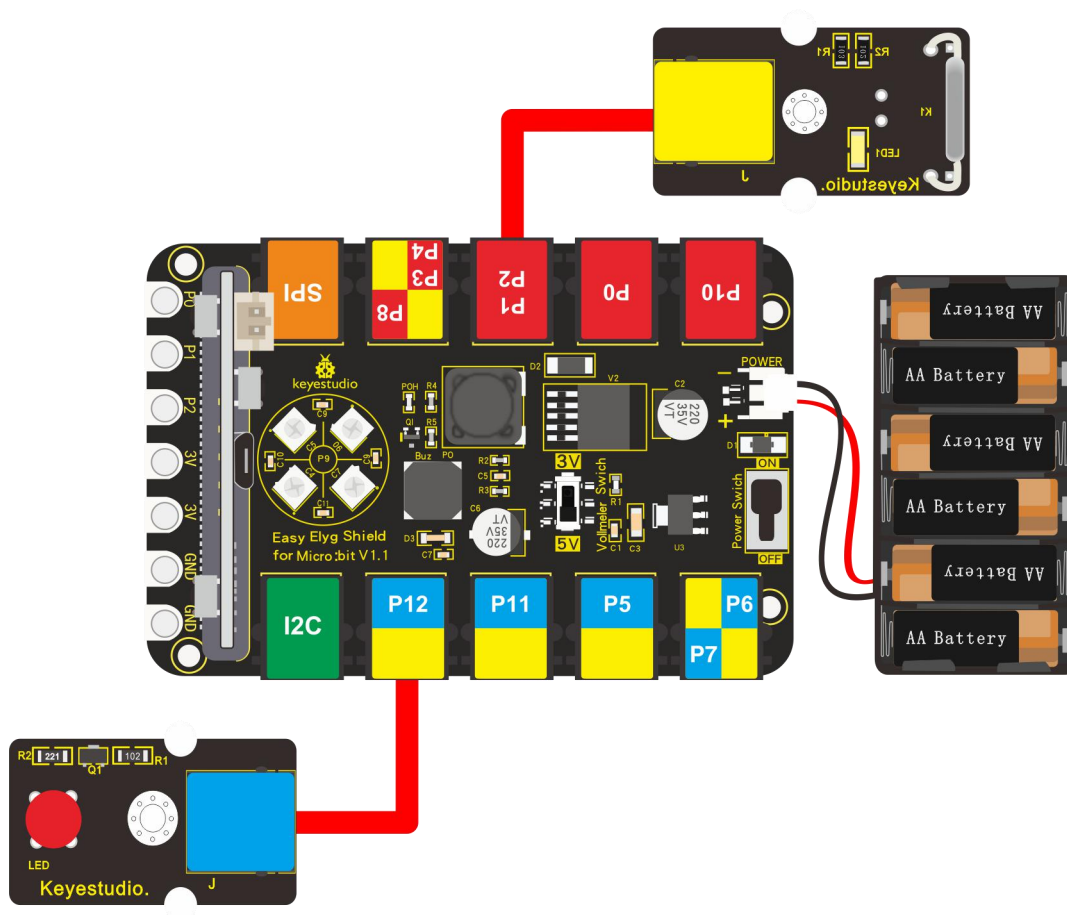
3. Specification:

- Interface: Easy plug
- Working voltage: DC 3.3V-5V
- Working current: $\geq 20\text{mA}$
- Working temperature: -10°C to $+50^{\circ}\text{C}$

- Detection distance: $\leq 10\text{mm}$

4. Wiring Up:

Insert the micro:bit onto the EASY Plug shield, connect reed switch and LED module to P1 and P12 port of shield.



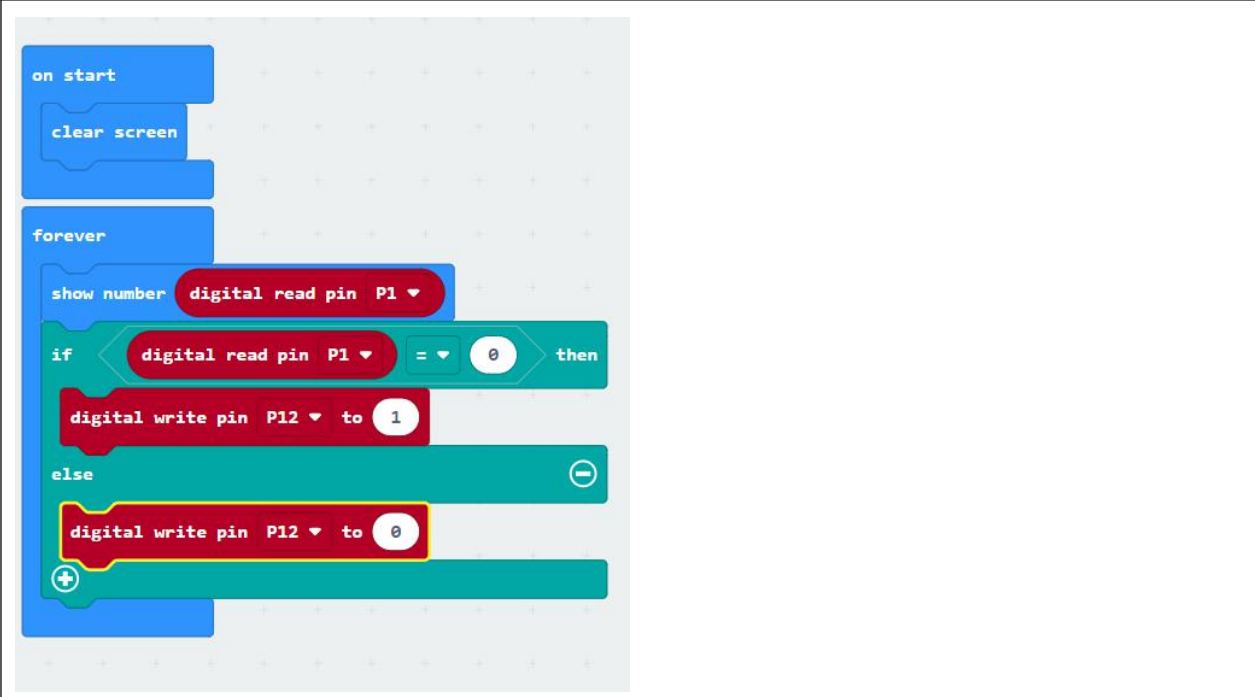
Note: Dial Voltmeter_Switch to 3V end

5. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.



```
on start
  clear screen

forever
  show number digital read pin P1
  if digital read pin P1 = 0 then
    digital write pin P12 to 1
  else
    digital write pin P12 to 0
```

The image shows a Scratch-style code editor with a light gray grid background. The code is written in blue, red, and teal blocks. It starts with an 'on start' block containing a 'clear screen' block. Below that is a 'forever' loop block. Inside the loop, there is a 'show number' block with a dropdown menu set to 'P1'. This is followed by an 'if' block with a dropdown menu set to 'P1', an equals sign, and a dropdown menu set to '0'. The 'then' branch contains a 'digital write pin' block with a dropdown menu set to 'P12' and a value of '1'. The 'else' branch contains a 'digital write pin' block with a dropdown menu set to 'P12' and a value of '0'. The code is enclosed in a rectangular frame.

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

Turn off LED dot matrix

The program under the block “forever” runs cyclically.

Micro:bit shows the digital signal(1/0)

If digital signal read by P1=0, there is magnetic field, execute the program under then block

Set P12 to high level(1), turn on LED

If digital signal read by P1=1, no magnetic field, execute the program under else block

Set P12 to low level(0), turn off LED

6. Test Result:

Wire up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end, download code to micro:bit. When the reed switch module detects magnetic field, micro:bit will show low level(0) and LED will be on; on the contrary, high level(1) will be displayed and LED will be off.

Project 27: Hear Footstep

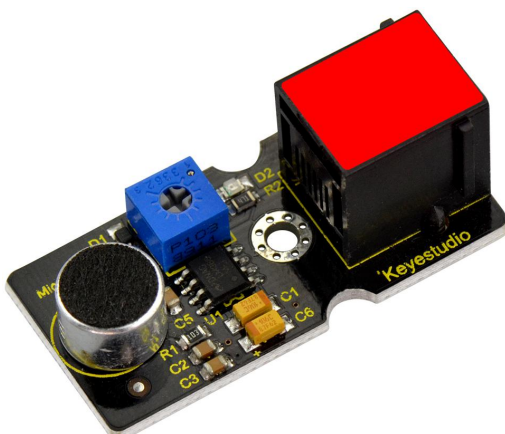
1. Description:

In this project, we will connect sound sensor to shield and read analog value by detecting sound with micro:bit. **The louder the sound is, the larger the analog value is.**

2. What You Need:

- Micro:bit main board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug analog sound sensor*1
- EASY plug White LED Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY plug Analog Sound Sensor:



The sound sensor mainly adopts a high-sensitivity microphone element and LM386 chip. High-sensitivity microphone components are used to detect external sounds. The LM386

chip can amplify the sound detected by the high-sensitivity microphone, and the maximum multiple is 200 times.

When in use, we can adjust the multiple of the sound by rotating the potentiometer on the sensor. Rotating potentiometer clockwise, the sound will be up to the maximum. This benefits us to make sound-activated robot, switch, alarm and so on.

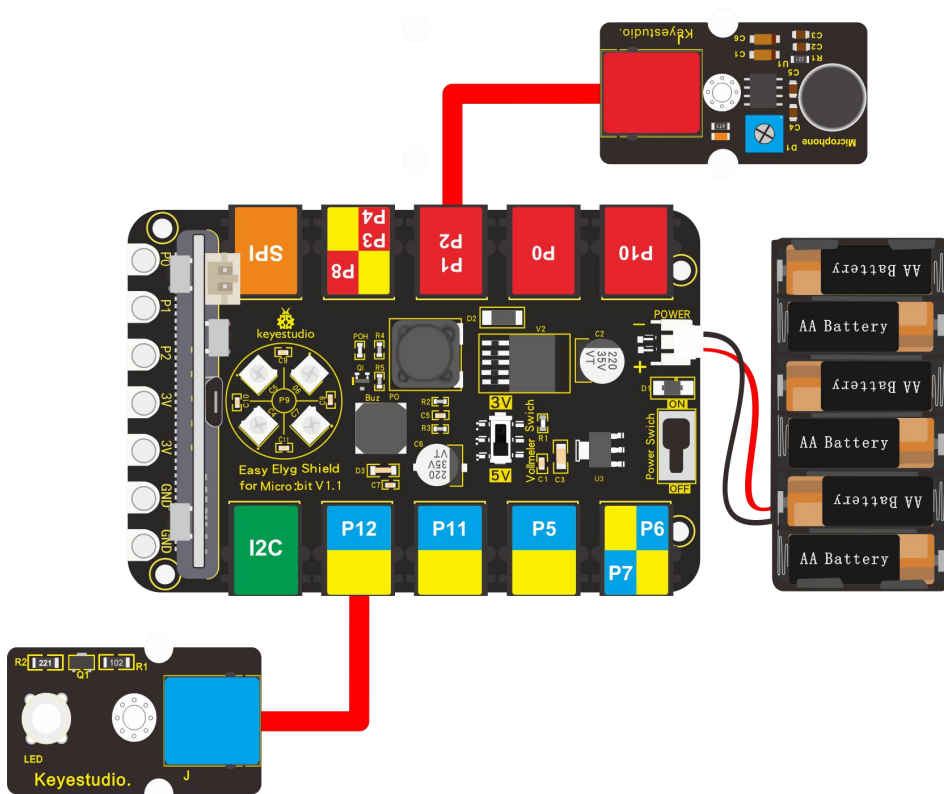
3. Specification:

Supply Voltage: 3.3V to 5V

Interface: Easy plug

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, respectively connect sound sensor and white LED to P1 and P12 of shield, and plug in power.



Note: Dial Voltmeter_Switch to 5V end.

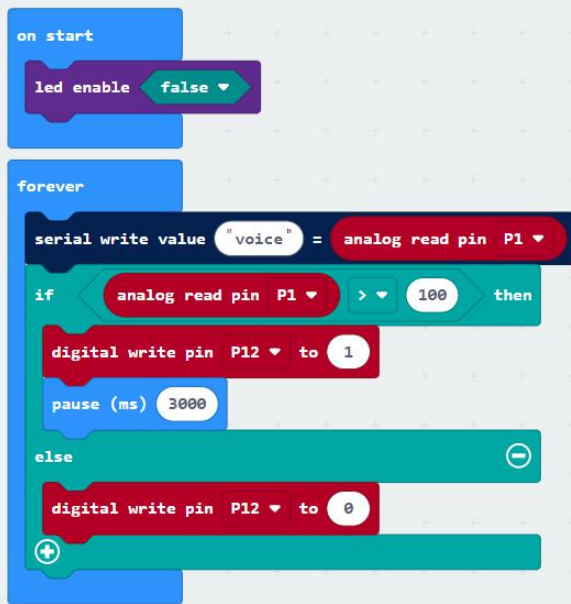
5. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.

(Note: sound analog value could be adjusted)



“on start” : command block only runs once to start program.
 Turn off LED dot matrix
 The program under the block “forever” runs cyclically.
 Serial writes voice=analog sound signals read by sound sensor
 If analog signals read by P1 >100, execute the program under then block
 Set P5 to high level (1) to turn on LED
 Delay in 3000ms
 If analog signals read by P1 <100, execute the program under else block
 Set P5 to low level (0) to turn off LED

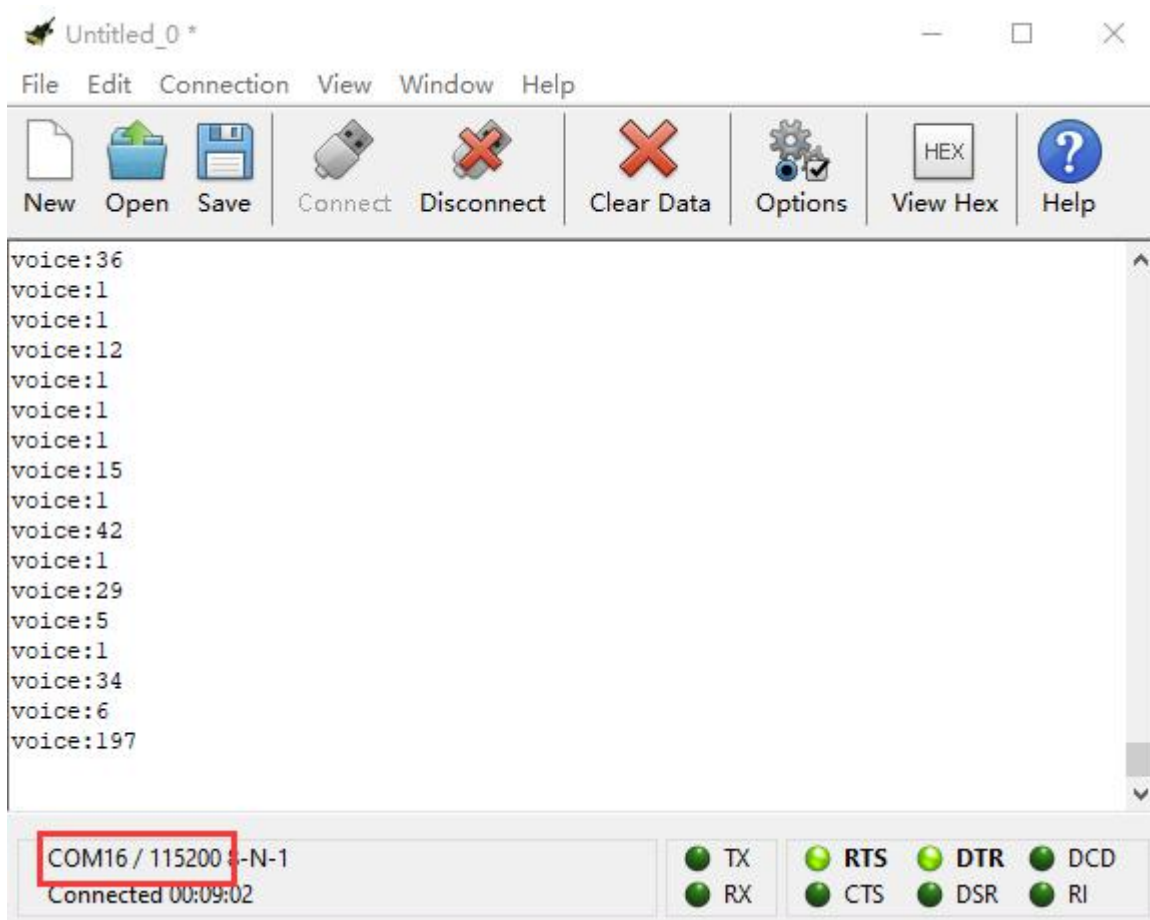
6. Test Result:

Wiring up, dial Voltmeter_Switch to 5V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Open CoolTerm, click Options and select SerialPort, set COM port and baud rate, set baud rate to 115200. Tap OK and Connect.

CoolTerm monitor will display the result, as shown below:

When the sound analog value is greater than 100, LED will be on, otherwise, LED will be off.



Project 28: Rotary Potentiometer

1. Description:

When doing experiments, we often use a 10K adjustable potentiometer. Rotating it can change analog value and you could check value on CoolTerm monitor. At same time, the brightness of LED connected to P10 gradually alters.

2. What You Need:

- Micro:bit main board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug Rotary Potentiometer*1
- EASY plug Red LED Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug Rotary Potentiometer:



This EASY Plug rotary potentiometer is counted as a changeable

resistor. In fact, it will change the resistance of changeable resistor when rotating potentiometer. We set the circuit, convert the change of resistance into the change of voltage.

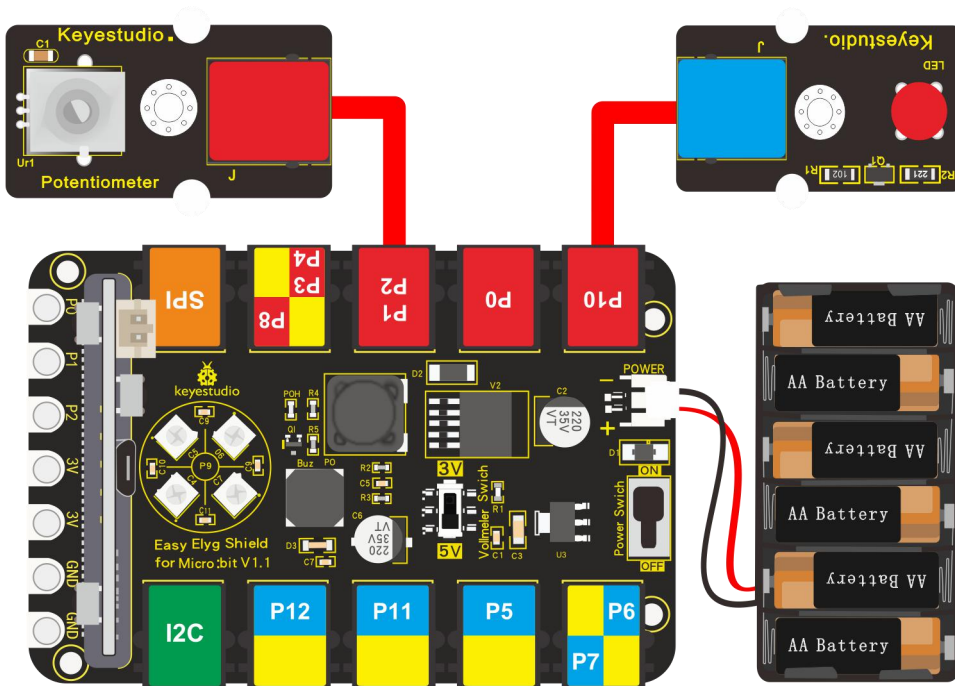
4. Specification:

Supply Voltage: 3.3V to 5V

Interface: EASY Plug

5. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect analog potentiometer and red LED module to P1 and P10 port of shield. And plug in power.



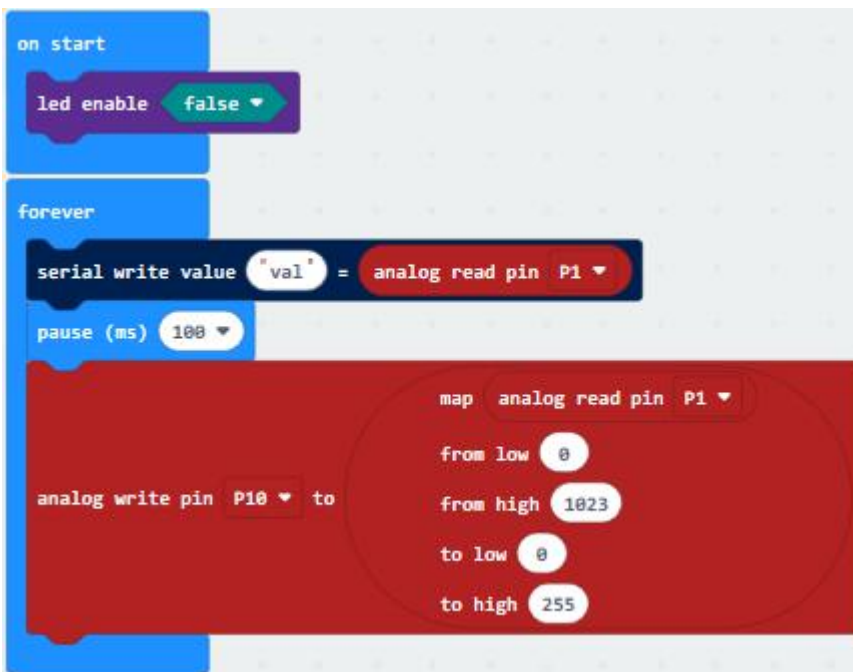
Note: Dial Voltmeter_Switch to 3V end.

6. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.



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

Turn off micro:bit

The program under the block "forever" runs cyclically.

Serial writes the analog signal of adjustable potentiometer

Delay in 100ms

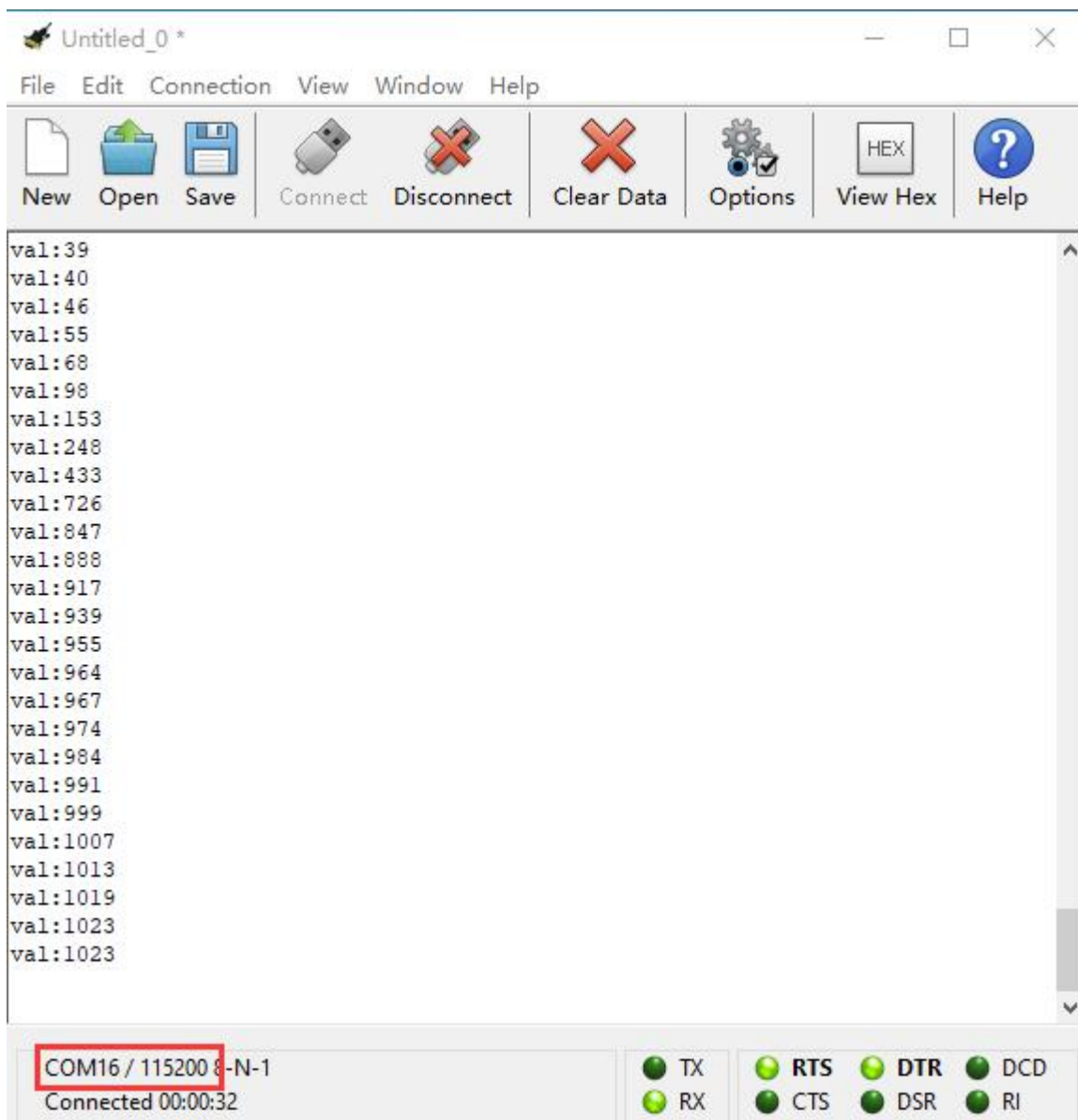
Map the analog signal (0-1023) of potentiometer to analog value (0-255) of LED connected P10

7. Test Result:

Wiring up, dial Voltmeter_Switch to 5V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Open CoolTerm, click Options and select SerialPort, set COM port and baud rate, set baud rate to 115200. Tap OK and Connect.

CoolTerm monitor shows the detected value, rotate the potentiometer to adjust analog value. As the analog value rises, LED gradually gets bright; when the value reduces, LED gets dimmer.



Project 29: Alcohol Content in the Air

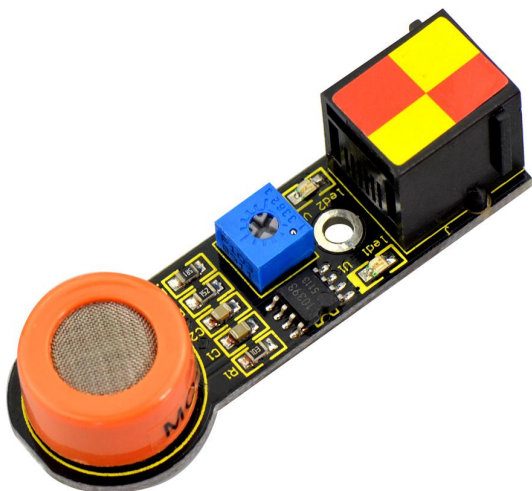
1. Description:

In this program, we will conduct you how to detect alcohol content.

2. What You Need:

- Micro:bit Board*1 EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug Analog Alcohol Sensor*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Analog Alcohol Sensor:



This analog gas sensor - MQ3 is suitable for detecting alcohol. It can be used in a Breath analyzer. Also it has high sensitivity to alcohol and low sensitivity to gas. You could

adjust the sensitivity by rotating the potentiometer of sensor.

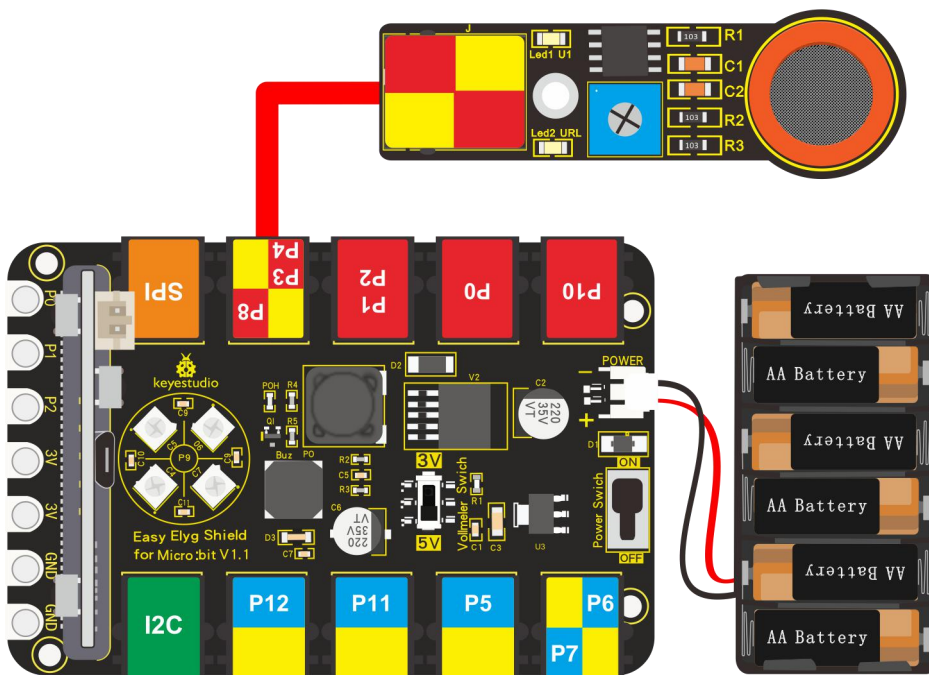
The higher the alcohol content is, the larger the analog value at A0.

3. Specification:

- Power supply: 5V
- Interface type: EASY Plug
- Simple drive circuit
- Stable and long service life
- Quick response and High sensitivity

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect alcohol sensor to P4 of shield and plug in external power.



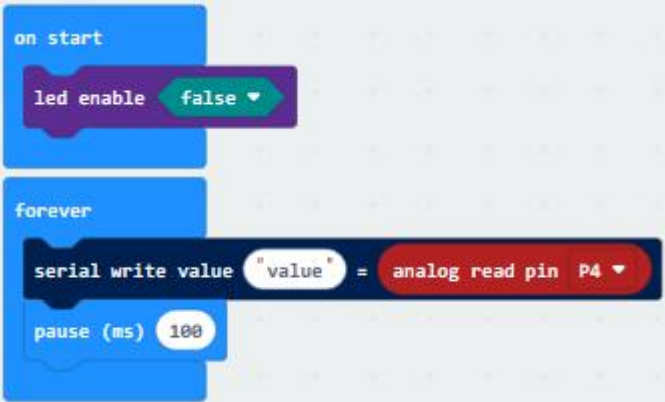
Dial Voltmeter_Switch to 5V end.

5. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.



The image shows a Scratch-style code editor with the following blocks:

- on start** block containing a **led enable** block set to **false**.
- forever** loop block containing:
 - serial write value** block with **"value"** and **analog read pin P4**.
 - pause (ms)** block set to **100**.

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

Turn off dot matrix on micro:bit

The program under the block "forever" runs cyclically.

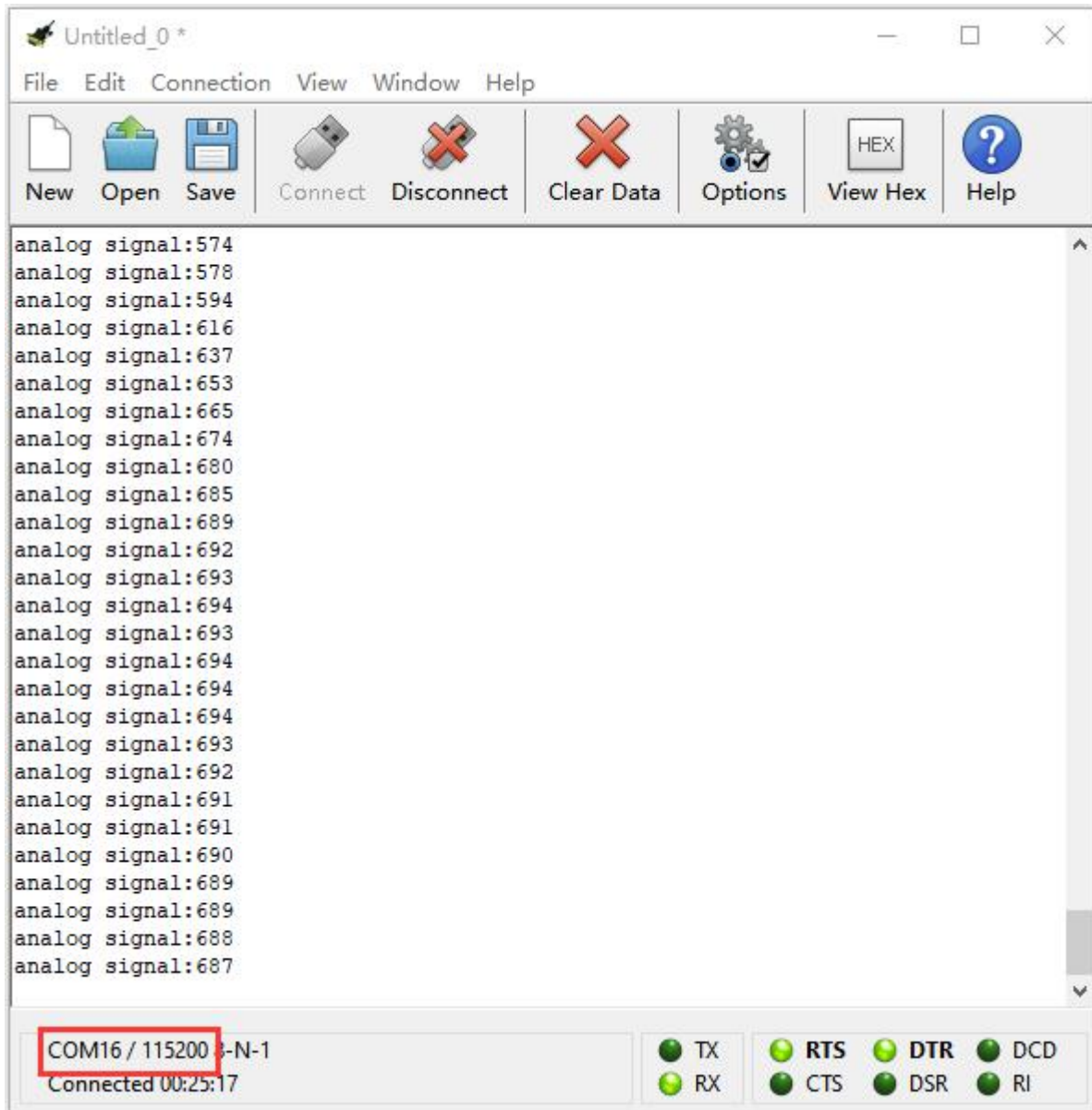
Serial writes analog signals read by alcohol sensor

Delay in 100ms

6. Test Result:

Wiring up, dial Voltmeter_Switch to 5V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.(LED2 of sensor shows green color, and you could adjust potentiometer to keep LED on module in off-and-on state(the sensitivity is highest) Open CoolTerm, click Options and select SerialPort, set COM port and baud rate, set baud rate to 115200. Tap OK and Connect.

Make alcohol gas close to alcohol sensor, CoolTerm serial monitor indicates that the analog value gets larger and larger and LED1 is on; on the contrary, the analog value gets smaller, LED1 will be off.



Project 30: Ambient Temperature Detection

1. Description:

We will detect the current temperature with EASY plug LM35 linear temperature sensor and display the results on CoolTerm monitor and dot matrix.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug LM35 Temperature Sensor*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug LM35 Linear Temperature Sensor:



Based on semiconductor LM35 temperature sensor, LM35 linear temperature Sensor can be used to detect ambient temperature.

It can detect temperature between 0°~100°. Sensitivity is 10mV per degree Celsius. The output voltage is proportional to the temperature.

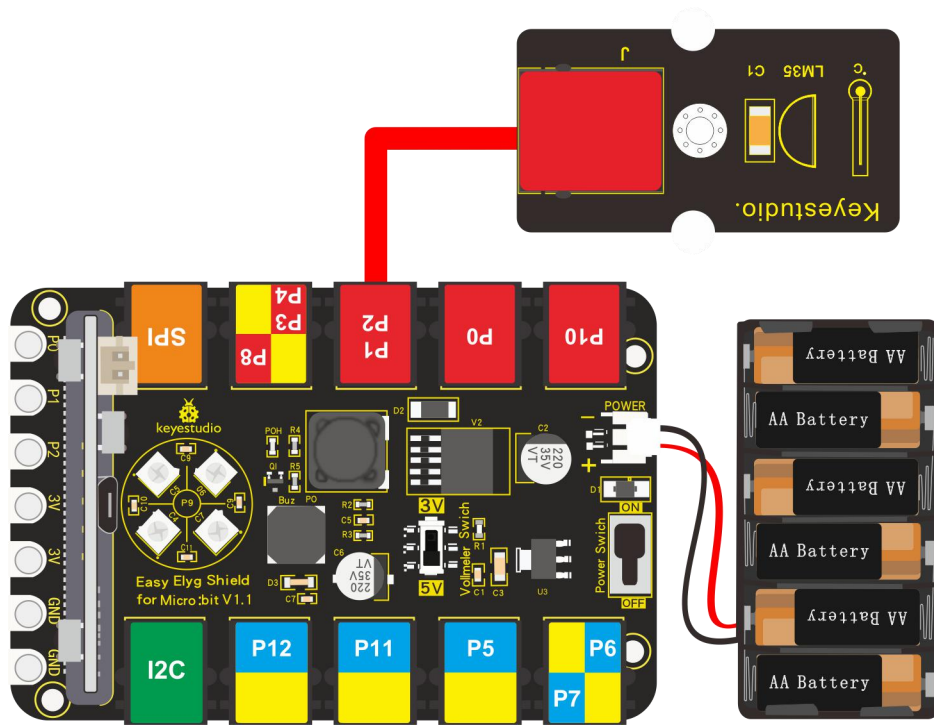
3. Specification:

Sensitivity: 10mV per degree Celsius

Functional Range: 0°C to 100°C

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect LM35 temperature sensor to P1 of EASY Plug shield with a RJ11 cable, and plug in external power.



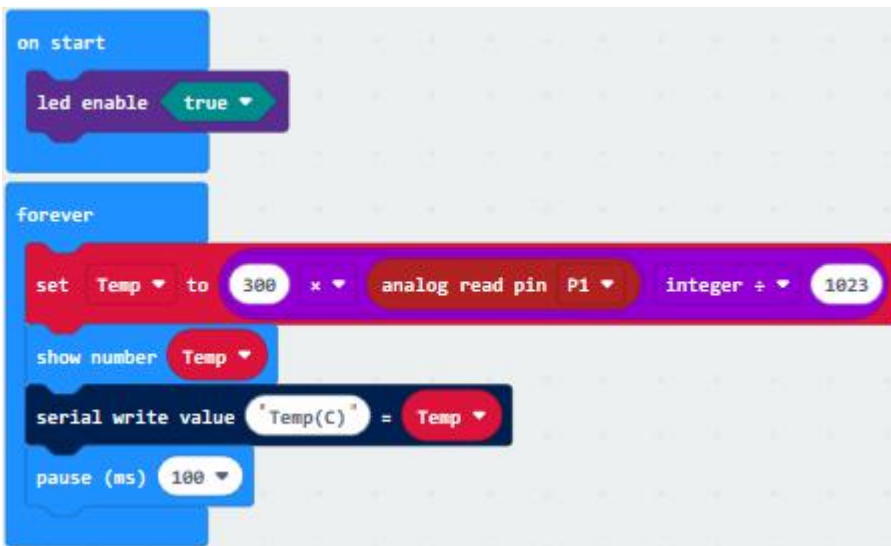
Note: Dial Voltmeter_Switch to 3V end.

5.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.



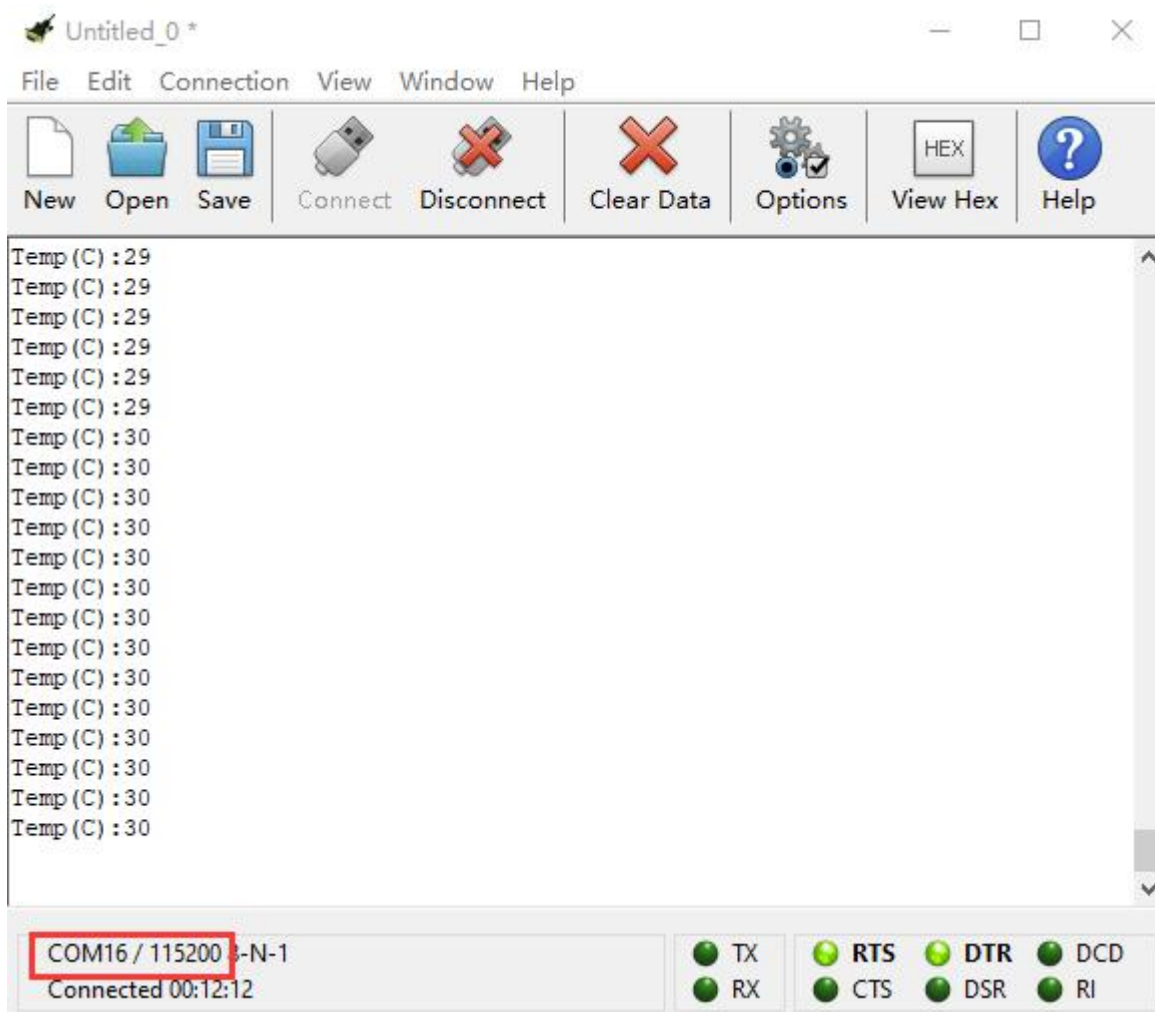
"on start" : command block only runs once to start program.
 open LED dot matrix on micro:bit
 The program under the block "forever" runs cyclically.
 Set temp to $300 \times \text{analog read pin P1} \div 1023$
 Display temperature value on micro:bit
 Serial writes temperature value
 Delay in 100ms

6.Test Result:

Wiring up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Open CoolTerm, click Options and select SerialPort, set COM port and baud rate, set baud rate to 115200. Tap OK and Connect.

Micro:bit and CoolTerm monitor will display the current temperature, as shown below:



Project 31: Water Level Alarm

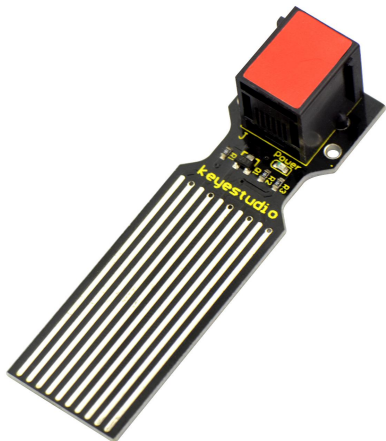
1.Description:

In this lesson, we will do an experiment with a water level sensor and a passive buzzer. Measure the water level in the cup, if water level rises up, LED will flash and passive buzzer will emit sound.

2. What You Need:

- Micro:bit Board*1 EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug Water Level Sensor*1
- EASY plug Red LED Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Water Level Sensor:



The water level sensor is easy- to-use, portable and cost-effective, designed to identify and detect water level and water drop.

This sensor measures the volume of water drop and water quantity through an array of traces of exposed parallel wires.

Characteristic:

Conversion of water volume and simulated water volume;

Strong flexibility, can output analog value;

Low power consumption, high sensitivity;

Can be directly connected to microprocessors or other logic circuits, suitable for Arduino controllers, STC microcontrollers, AVR

microcontrollers and other development boards and controllers;

Production process: FR4 double-sided tin plating;

Shape design: non-slip half-moon groove.

3. Specification:

Working voltage: DC 5V;

Working current: 20mA;

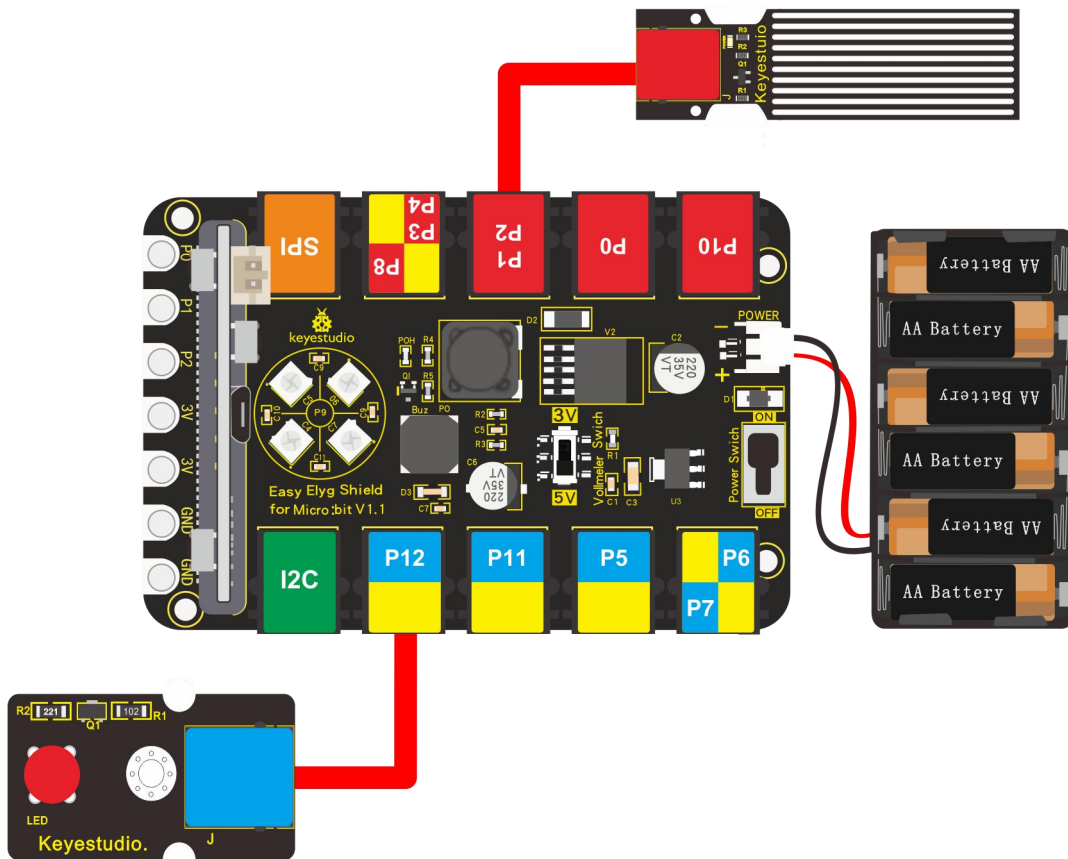
Sensor type: analog signal;

Working temperature: 10°C -30°C;

Working humidity: 10%-90% non-condensing

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect water level sensor and LED module to P1 and P12 port of shield with two RJ11 cables.



Note: Dial Voltmeter_Switch to 3V end.


5.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.

(Note: the analog value can be adjusted)



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

Turn off dot matrix on micro:bit

The program under the block “forever” runs cyclically.

Set the analog signals read by water level sensor to variable item

Serial writes value=analog signals read by water level sensor

If the analog signals read by P1>400, execute the program under then block

Play tone high C for 1 beat to make passive buzzer emit sound

Set P12 to high level (1) to make LED light on

Delay in 200ms

Set P12 to low level(0). LED turns off

Delay in 200ms

When the analog signals read by P1≤400, execute the program under else program

Set P12 to low level(0), turn off LED

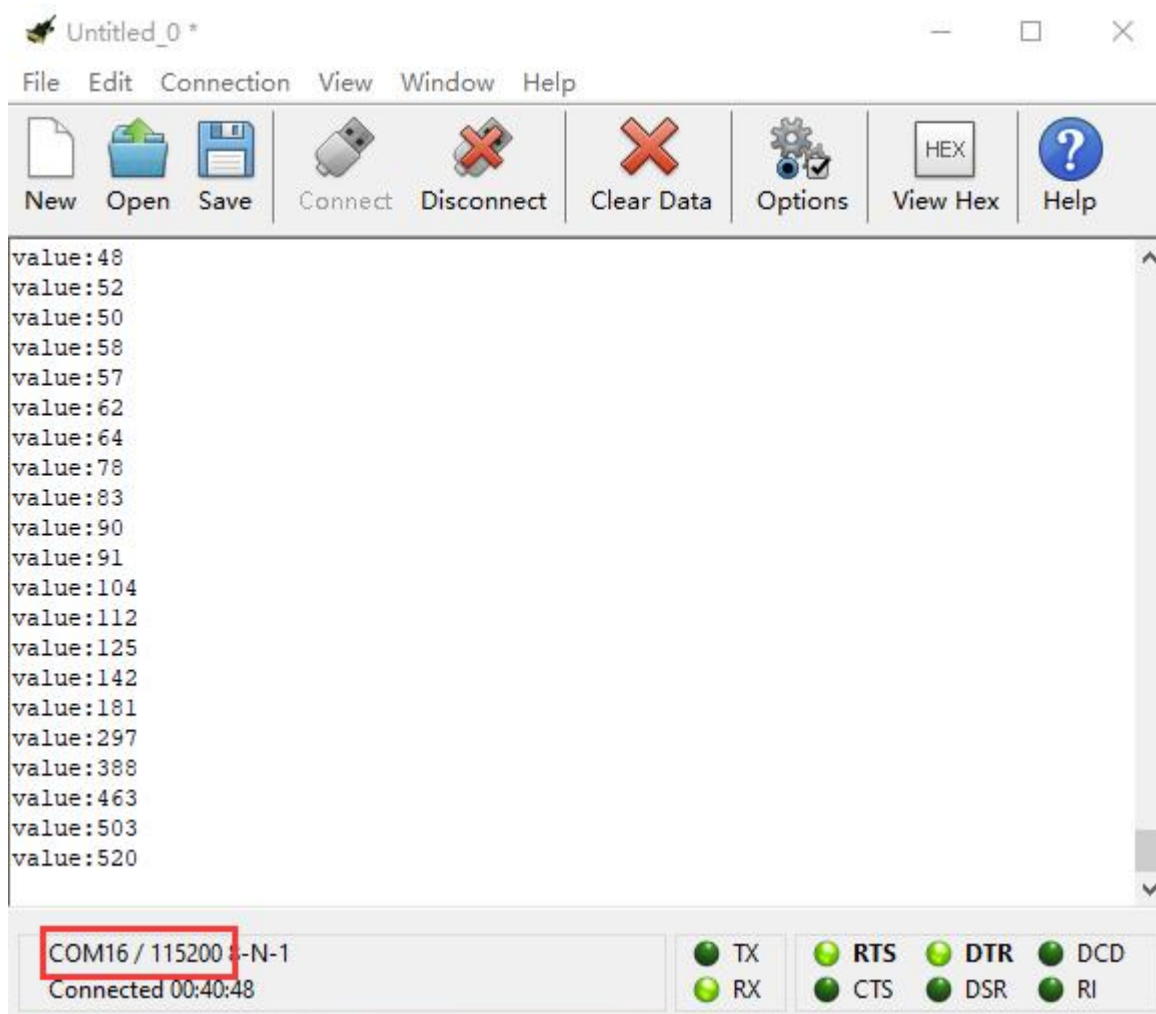
Set P12 to low level (0), turn off buzzer

6. Test Result:

Wiring up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Open CoolTerm, click Options and select SerialPort, set COM port and baud rate, set baud rate to 115200. Tap OK and Connect.

The deeper the water level sensor is immersed, the analog value gets larger and larger; on the contrary, the analog value will plummet. When the analog value is greater than 400, passive buzzer will sound and LED will flash; if not, LED will be off and buzzer won' t emit sound.



Project 32: 1602 LCD Display

1. Description:

1602 I2C can be used as display, in this project, we will connect it to shield and teach you how to make it display “keyestudio” and numbers.

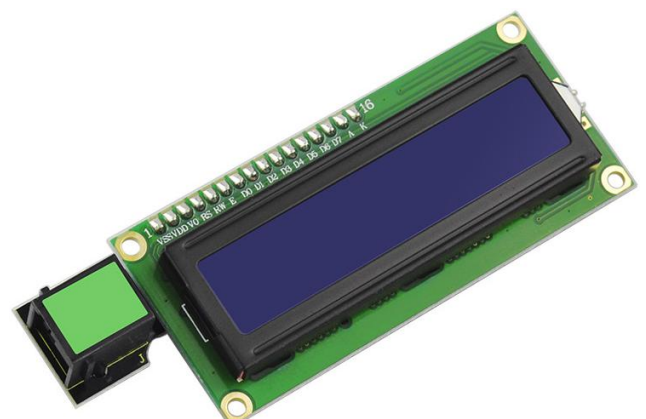
2. What You Need:

- Micro:bit*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug LCD 1602 I2C Module*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug LCD 1602 I2C Module:

This module is a LCD 16x2 display, useful for creating standalone projects.

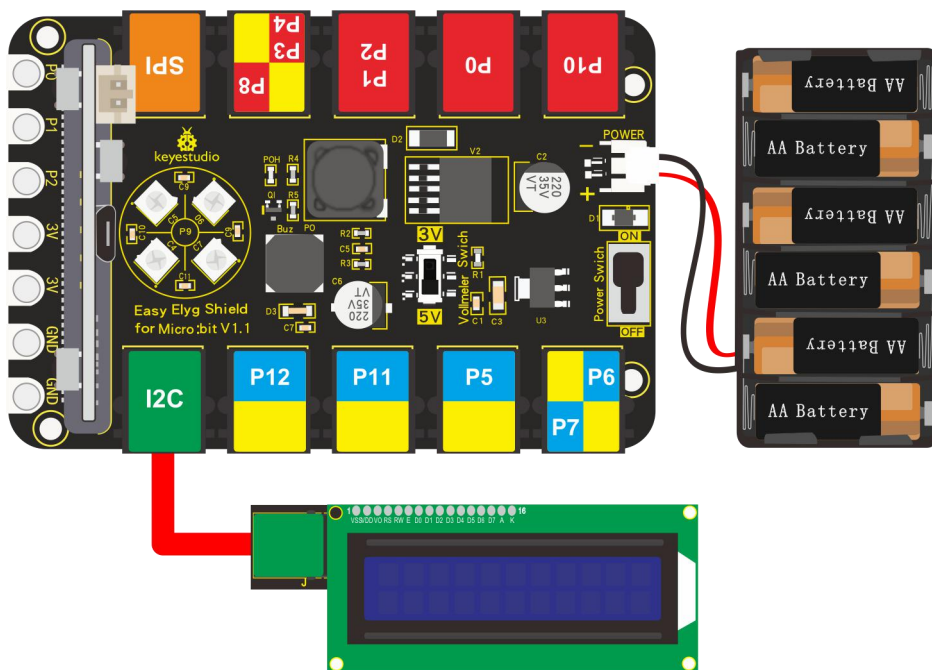
- 16 characters wide, 2 rows;
- White text on blue background;
- Chip Operating Voltage: 4.5-5.5V
- Working Current: 2.0mA (5.0V)
- Optimum working voltage of the module is 5.0V



- Single LED backlight included can be dimmed easily with a resistor.
- Built in character set supports English text
- Comes with necessary contrast potentiometer

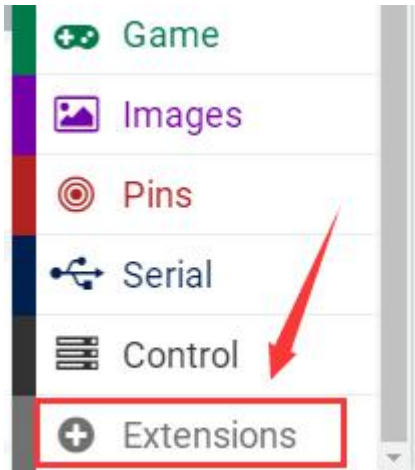
4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect LCD 1602 I2C to I2C port of shield with a RJ11 cable, and plug in power.



5. Test Code:

Enter link: <https://makecode.micro:bit.org/> to edit program, and set test code with library, as shown below:



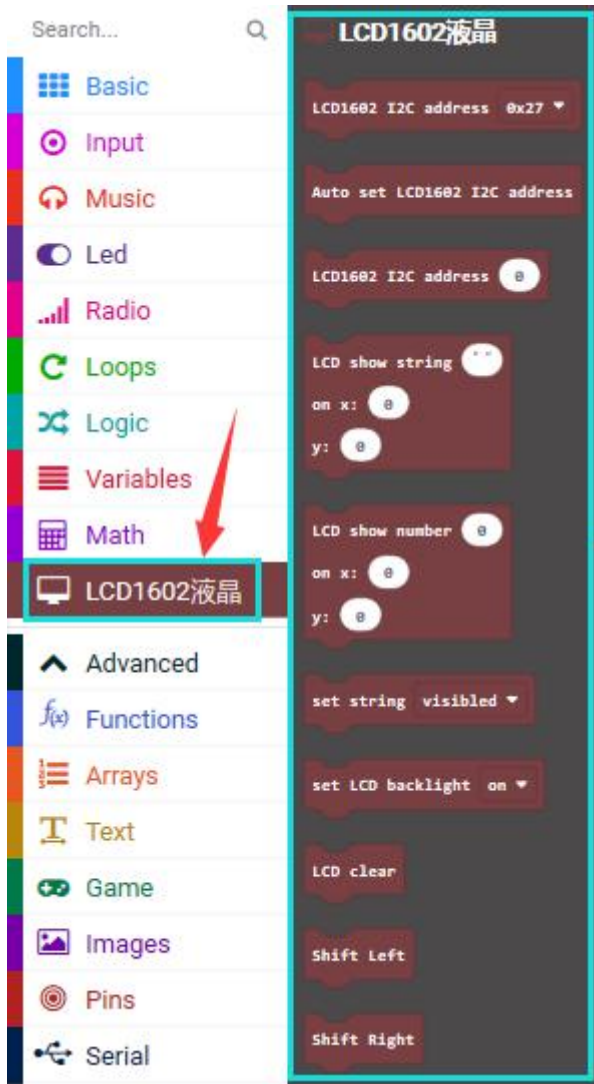
Download library:

https://github.com/xuefengedu/pxt-lcd1602_CN

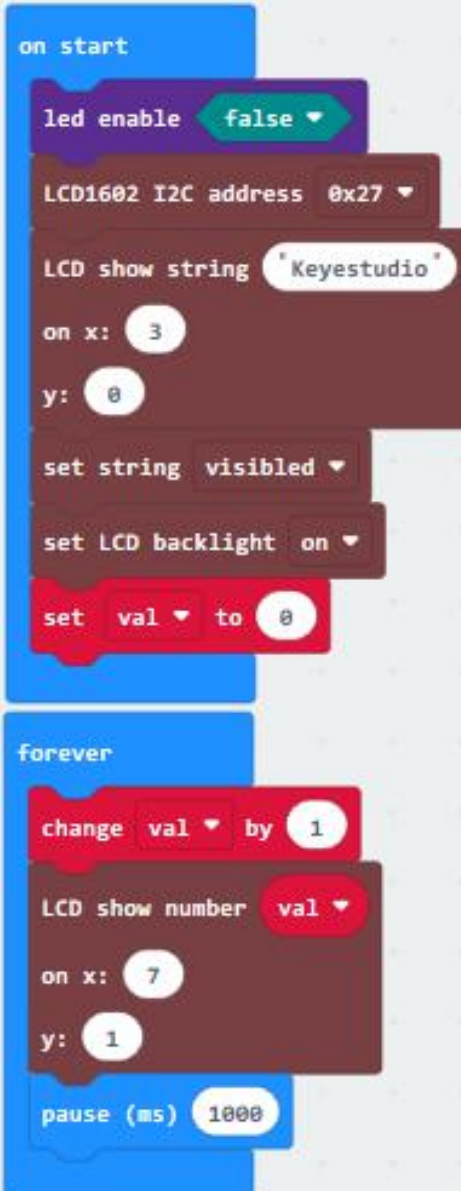
Copy this link in the search box to search:



Tap “**lcd1602**” to download, then LCD 1602 module will be added in the editing blocks, as shown below:



Test Code:



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

Turn off LED dot matrix

Set I2C address of LCD1602 to 0x27

Display Keyestudio at the first row and the fourth column on LCD module

Set backlight of LCD on

Set val to 0

The program under the block “forever” runs cyclically.

Change val by 1

Start displaying val at x:7 y:1

Delay in 1000ms

6.Test Result:

Wiring up, dial Voltmeter_Switch to 5V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

You will see “Keyestudio” at first row and numbers at the second row.

(Note: Press the reset button if there is random code on 1602 LCD module)

Project 33: Vapor in the Air

1.Description:

We could use vapor sensor to detect the vapor content in the air. The analog value will be displayed on 1602 LCD and CoolTerm serial monitor.

2.What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Vapor Sensor*1
- EASY Plug LCD 1602 I2C Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Vapor Sensor:



Vapor sensor is an analog sensor and can be made as a simple rainwater detector and liquid level switch. When humidity on the

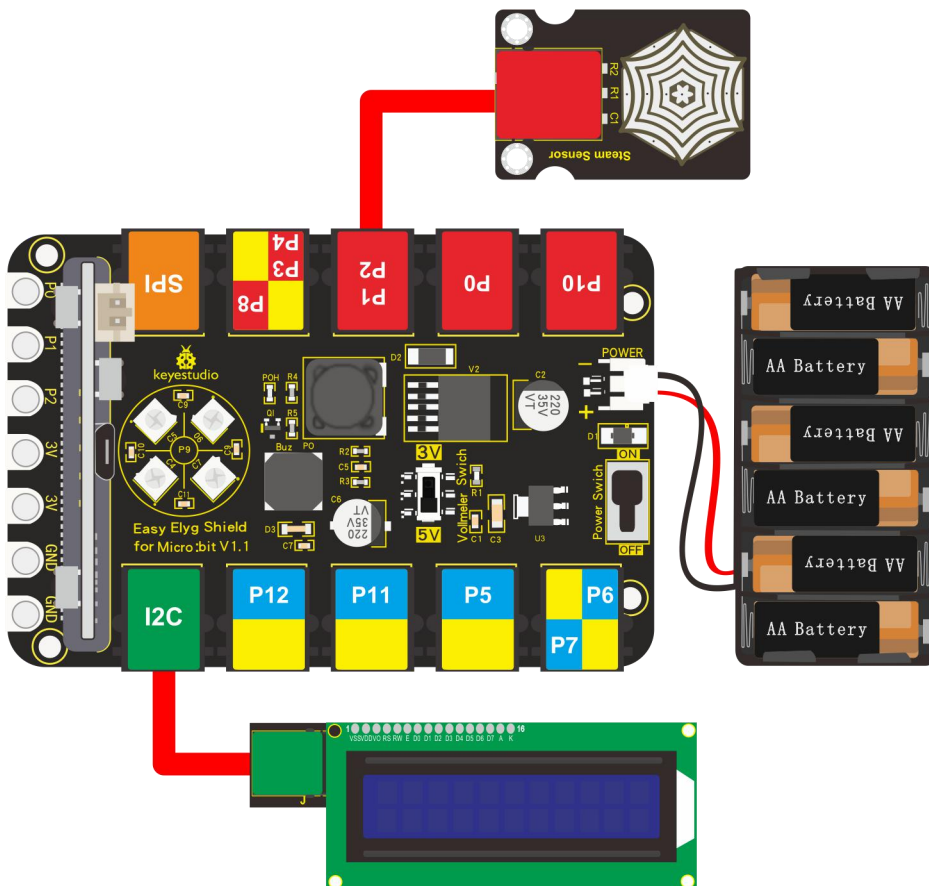
face of this sensor rises, the output voltage will increase.

3. Specification:

- Working Voltage: 3.3V-5V
- Working Current: <20mA
- Working Temperature: - 10°C ~ + 70°C
- Interface Type: EASY plug

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect vapor sensor and 1602LCD to P1 and I2C port of shield.



Note: Dial Voltmeter_Switch to 5V end.

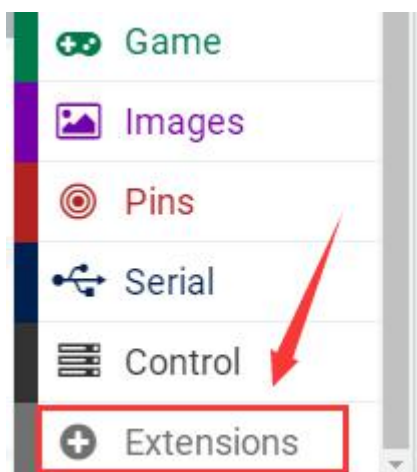
5.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.

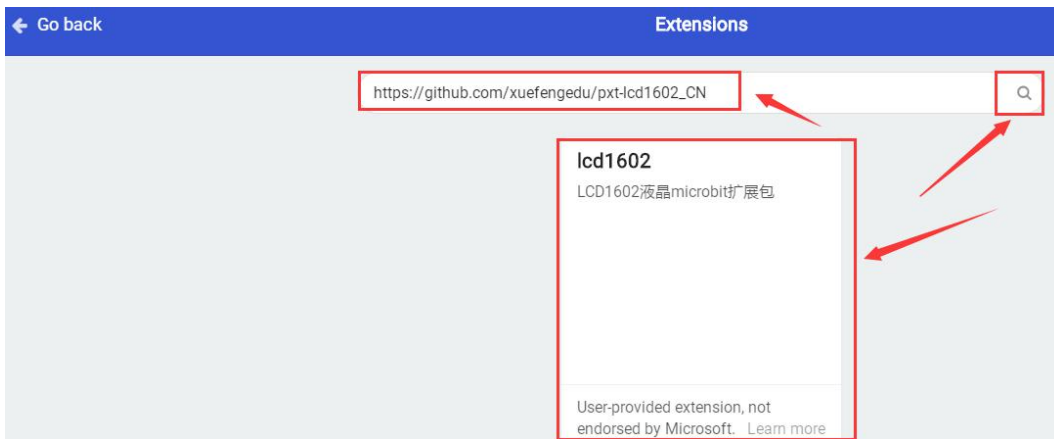
The below example code is as for your reference, you need to add the library of LCD 1602 module



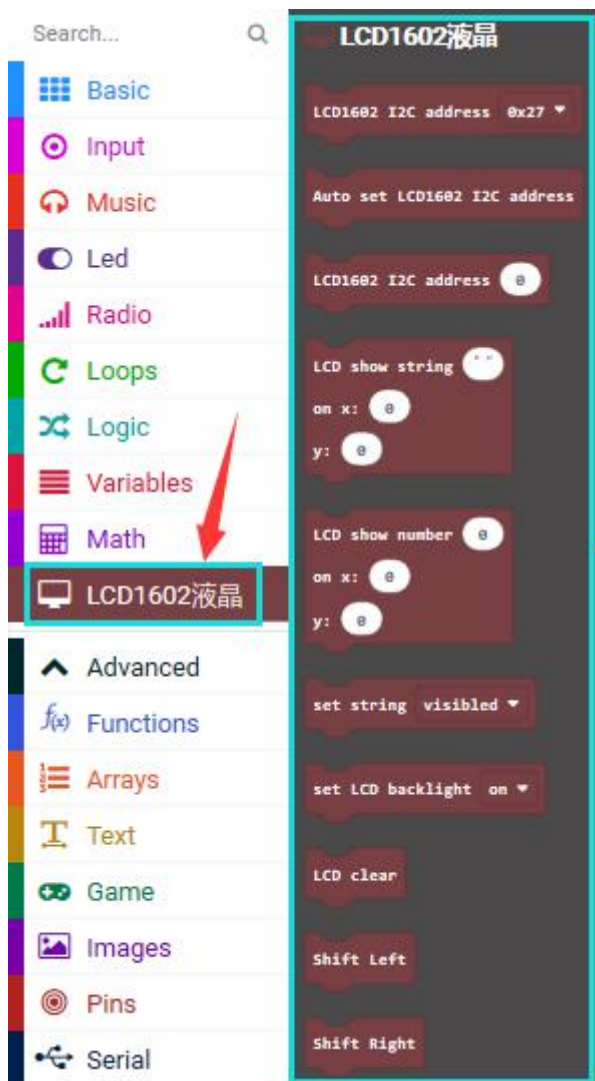
Download library:

https://github.com/xuefengedu/pxt-lcd1602_CN

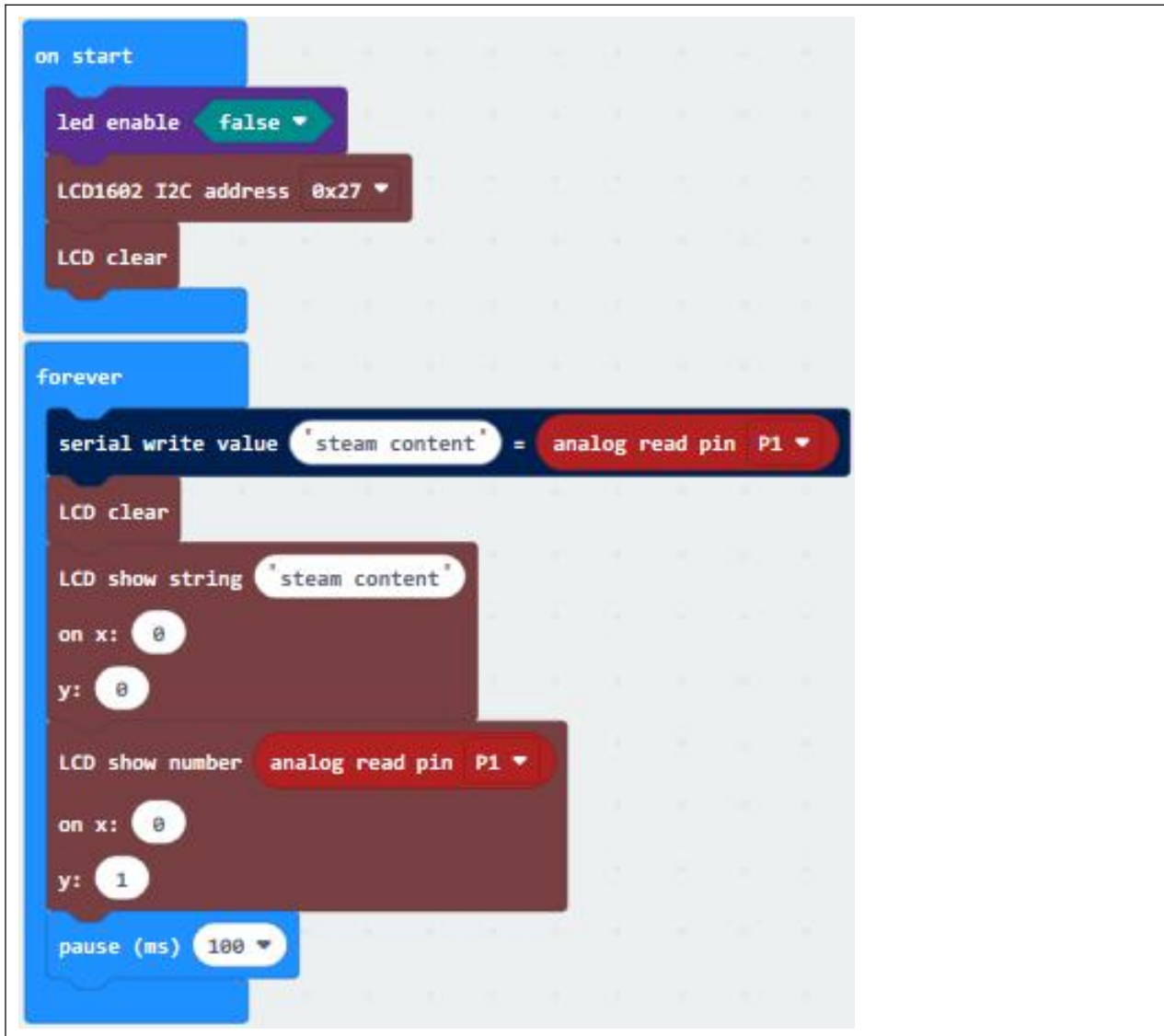
Copy this link in the search box to search.



Click “**lcd1602**” to download, then LCD 1602 module will be added in editing blocks



6. Test Code:



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

Turn off LED dot matrix

Set I2C address of LCD1602 to 0x27

Clear LCD screen

The program under the block “forever” runs cyclically.

Serial writes value=analog signals read by steam sensor

Clear LCD screen

Show the character string at the first row and the first column on LCE module

Steam content

Show the analog signals read by P1 at the second row and the first column on LCE module

Delay in 100ms

6.Test Result:

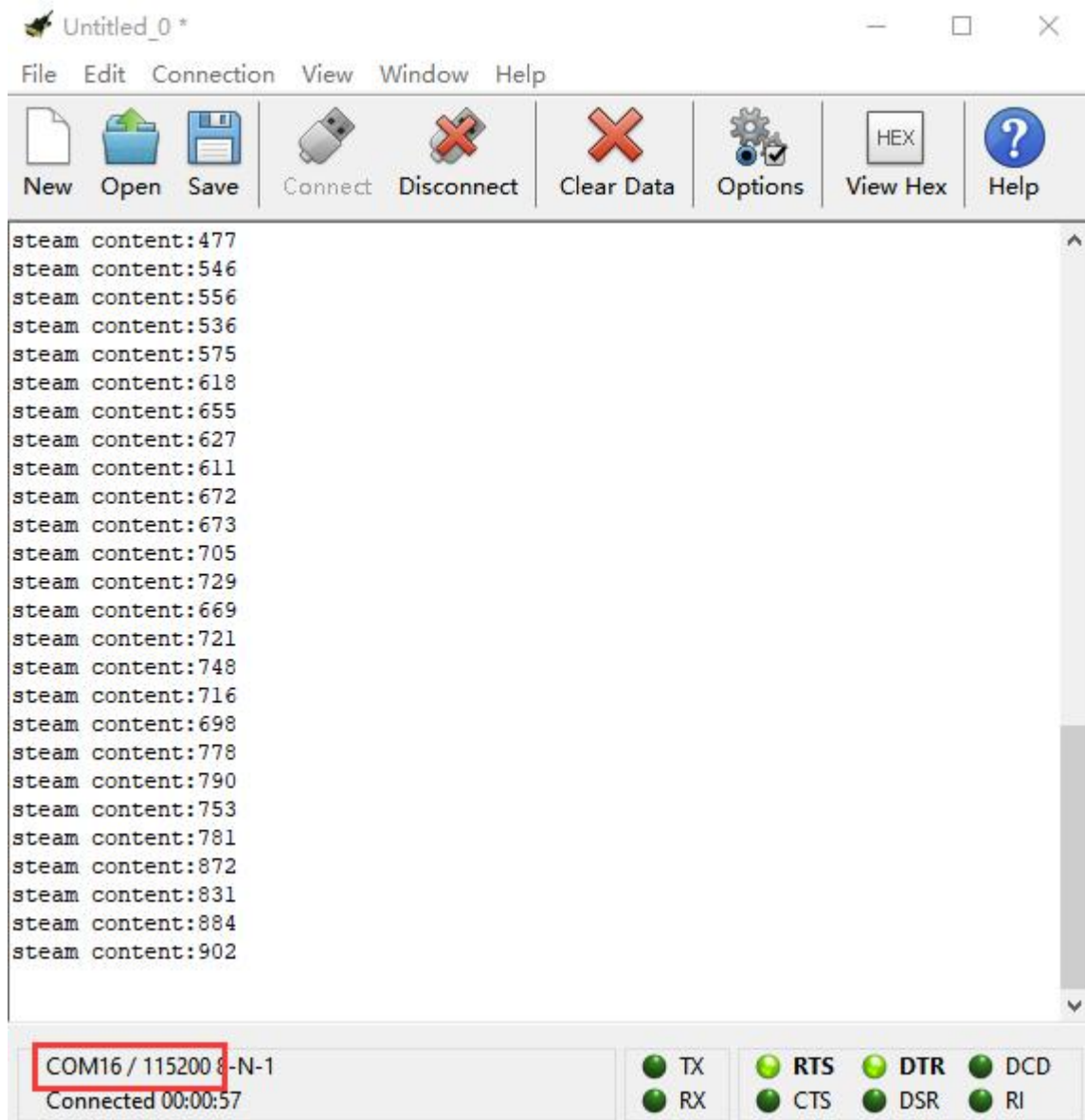
Wiring up, dial Voltmeter_Switch to 5V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Open CoolTerm, click Options and select SerialPort, set COM port and baud rate, set baud rate to 115200. Tap OK and Connect.

CoolTerm monitor and 1602 LCD module display the analog signals read by steam sensor, the higher the vapor content is, the larger the analog value, as shown below:

(Note: Press the reset button if there is random code on 1602 LCD module)

CoolTerm monitor shows the analog signals read by vapor sensor, the higher the vapor content, the larger the analog value is, as shown below:



Project 34: Pressure Detection

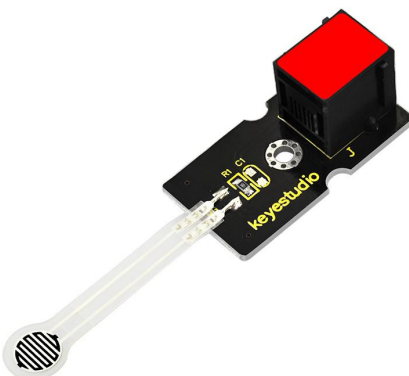
1. Description:

We've learned different sensors with specific characteristics, like sound sensor, gas sensor and so on. In this experiment, we will measure the pressure with thin-film pressure sensor and micro:bit.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug thin-film pressure sensor*1
- RJ11 Cable*1
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug Thin-film Pressure Sensor



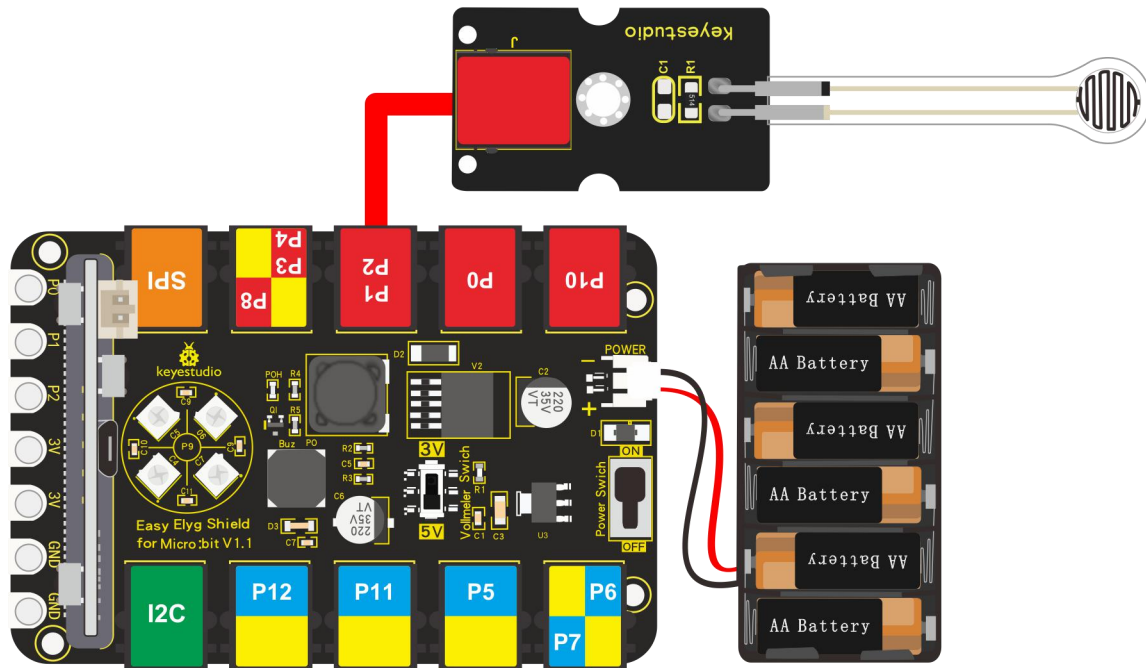
This EASY plug pressure sensor adopts the flexible Nano pressure-sensitive material with an ultra-thin film pad. It has the functions of water-proof and pressure detection. The force sensors are ultra-thin and flexible printed circuits, which can be easily integrated into force measurement applications.

3. Specification:

- Range: 0-10KG
- Working Voltage: DC 3.3V—5V
- Thickness: < 0.25mm
- Response Point: < 20g
- Repeatability: < $\pm 5.8\%$ (50% load)
- Accuracy: $\pm 2.5\%$ (85% range interval)
- Durability: > 100 thousand times
- Initial Resistance: > 100M Ω (no load)
- Response Time: < 1ms
- Recovery Time: < 15ms
- Working Temperature: - 20°C—60°C

4.Wiring Up:

Insert micro:bit onto EASY Plug shield, connect thin-film pressure sensor to P1 port of shield.



Note: Dial Voltmeter_Switch to 3 V end.

5.Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.



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

Turn off LED dot matrix

The program under the block "forever" runs cyclically.

Micro:bit shows the analog signals read by thin-film pressure sensor

Delay in 100ms

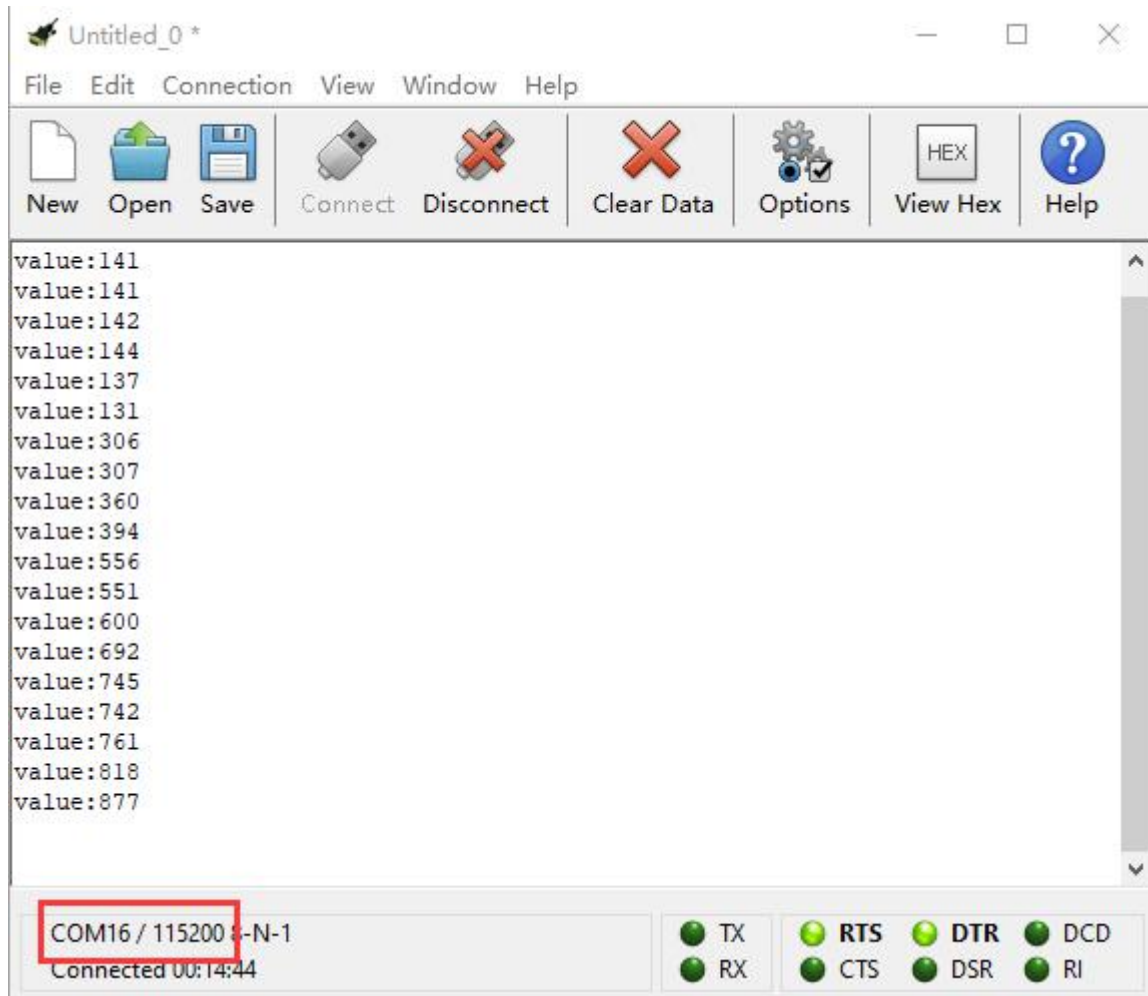
Serial writes value=analog signals read by thin-film pressure sensor

6. Test Result:

Wiring up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Open CoolTerm, click Options and select SerialPort, set COM port and baud rate, set baud rate to 115200. Tap OK and Connect.

Read the analog value of P1 end when the thin film is not pressed; when pressed, the bigger the pressure on thin film is, the larger the analog value is, as shown below:



Project 35: Make A Thermo-hygrometer

1. Description:

This DHT11 Temperature and Humidity Sensor is a composite sensor which contains a calibrated digital signal output of the temperature and humidity.

Its technology ensures high reliability and excellent long-term stability.

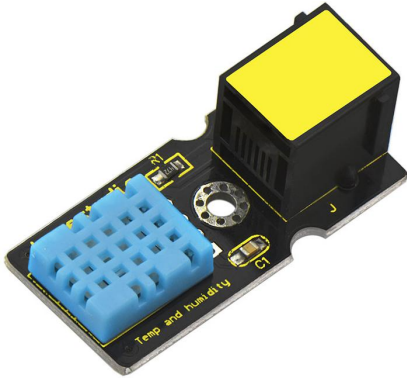
In this experiment, we connect DHT11 temperature and humidity sensor to P1 of micro:bit. We could calculate the current temperature and humidity value with specific formula to read the data.

And CoolTerm and 1602 LCD module will display temperature and humidity value too.

2. What You Need:

- Micro:bit Board*1 EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY plug DHT11 Temperature and Humidity Sensor*1
- EASY plug 1602 LCD Module*1
- RJ11 Cable*2
- 6 AA Battery Holder*1
- 1.5V AA Battery*6

EASY Plug DHT11 Temperature and Humidity Sensor:



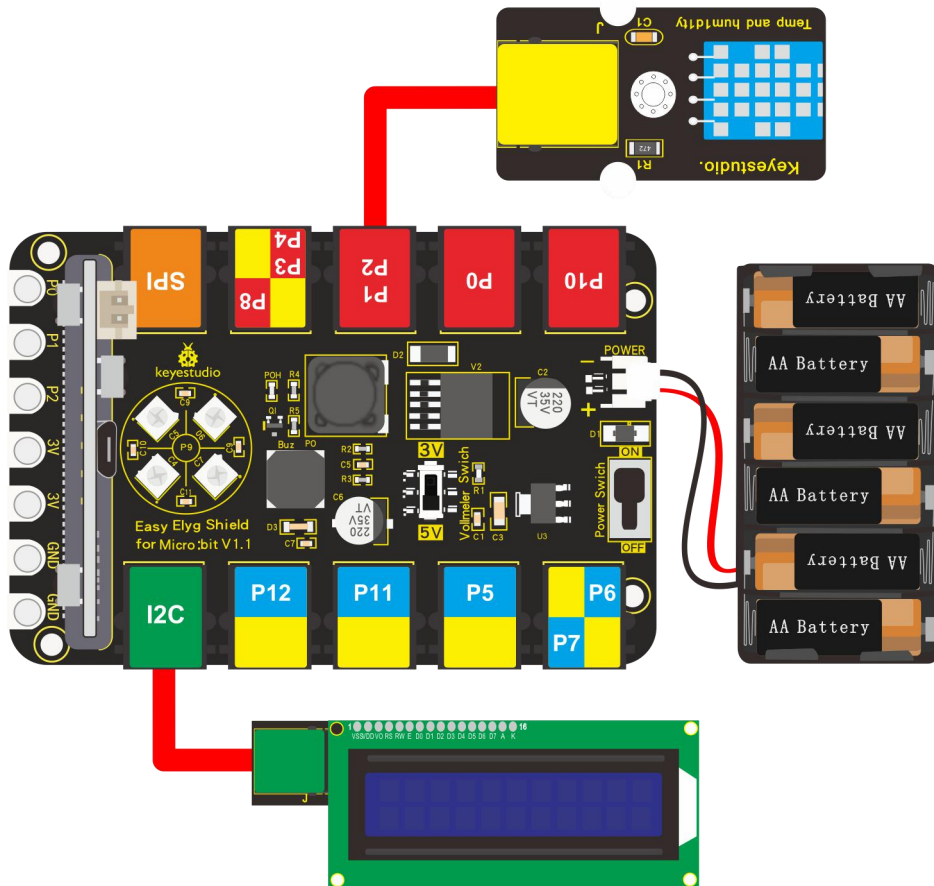
DHT11 temperature and humidity sensor, super low cost, adopts a capacitive humidity sensor and a thermistor to measure data and output a pre-calibrated digital signal. It has characteristics with high response, anti-interference ability, reliability and long-term stability.

3. Specification:

- Power supply voltage: DC5V;
- Relative humidity and temperature measurement;
- Suitable for humidity reading 20%-90%, accuracy: 5%;
- Suitable for temperature readings of 0-50°C, accuracy: $\pm 2^{\circ}\text{C}$
- Interface: EASY plug
- Low cost

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect DHT11 temperature and humidity sensor and 1602 I2C to P1 and I2C port of shield.



Dial Voltmeter_Switch to 5V end.

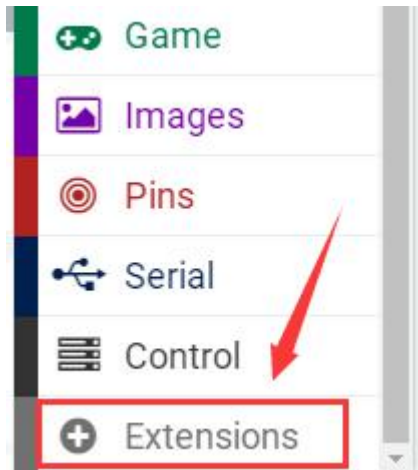
5. Test Code:

You could navigate <https://makecode.micro:bit.org/reference> to have access to more details.

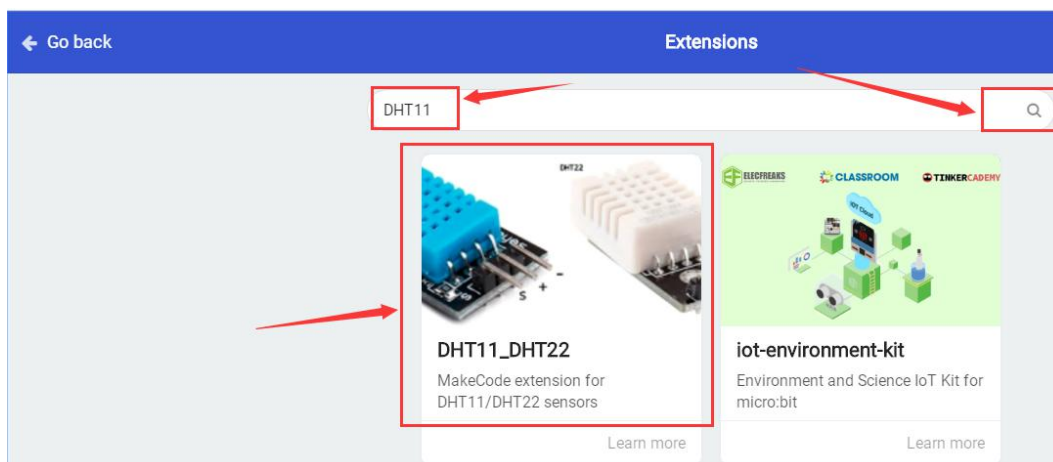
Browse link <https://makecode.micro:bit.org/> to edit your program.

The following test code is as for your reference.

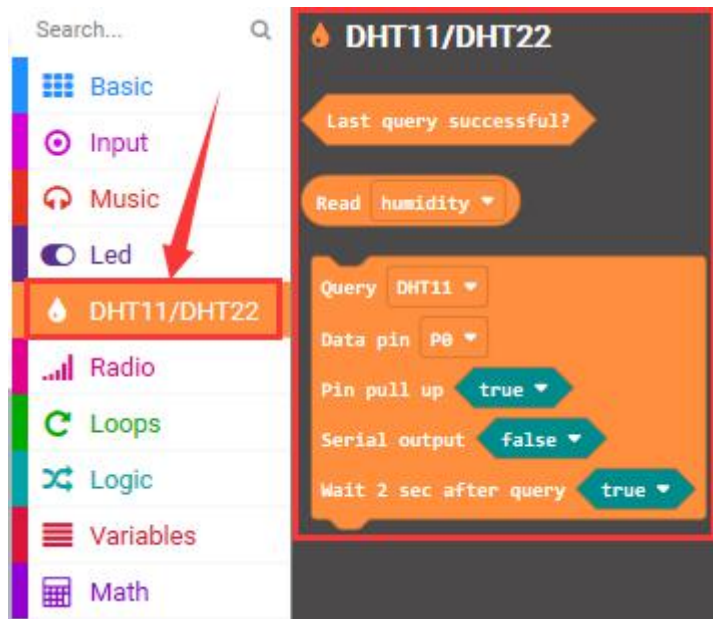
We need to add the library of DHT11 temperature and humidity sensor.



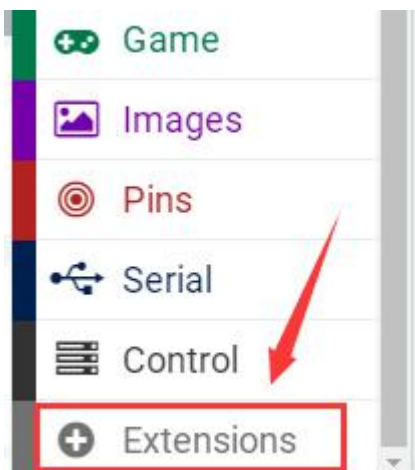
Search "DHT11" , as shown below, select and click "DHT11_DHT22" to install library



After the successful installation, the DHT11 module will be added into the editing module, as shown below:



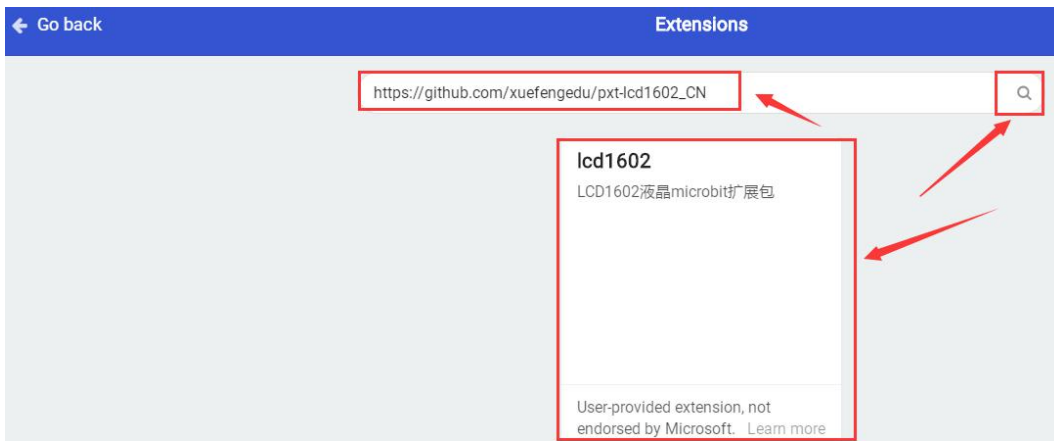
Then we add the library of 1602 I2C LCD module in same way.



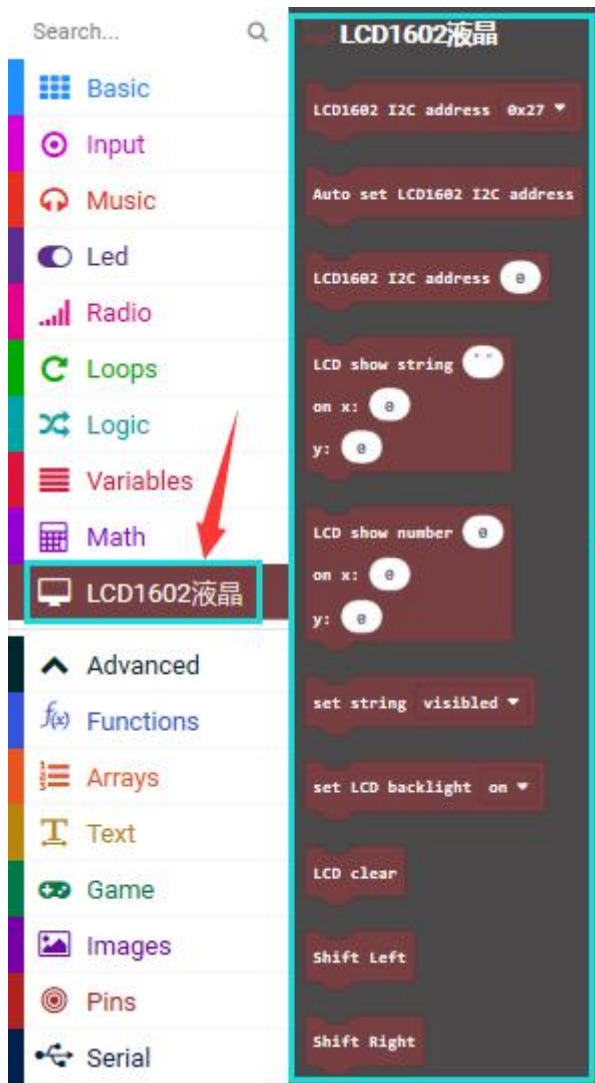
Download library:

https://github.com/xuefengedu/pxt-lcd1602_CN

Copy this link in the search box to search, as shown below:



Click “**lcd1602**” to download, LCD 1602 module will be added.



6. Test Code:

```

on start
  led enable false
  LCD1602 I2C address 0x27
  LCD clear

forever
  Query DHT11
  Data pin P1
  Pin pull up true
  Serial output false
  Wait 2 sec after query true

  serial write value "humid:" = Read humidity
  serial write value "temper:" = Read temperature
  serial write line ""

  LCD clear
  LCD show string "humid:"
  on x: 0
  y: 0
  LCD show number Read humidity
  on x: 7
  y: 0
  LCD show string "temper:"
  on x: 0
  y: 1
  LCD show number Read temperature
  on x: 8
  y: 1

```

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

turn off LED dot matrix

Set I2C address of LCD1602 to 0x27

Clear LCD screen

The program under the block "forever" runs cyclically.

Search data from P1 of DHT11 sensor

Serial writes " humid : " =the humidity read by P0

Serial writes "temperature: " =the temperature read by P0

Clear LCD screen

Show character string at the first row and column

Display the humidity read by P1 at the first row and the eighth column

Display temper at the second row and the first column

Show the temperature read by P1 at the second row and the ninth column

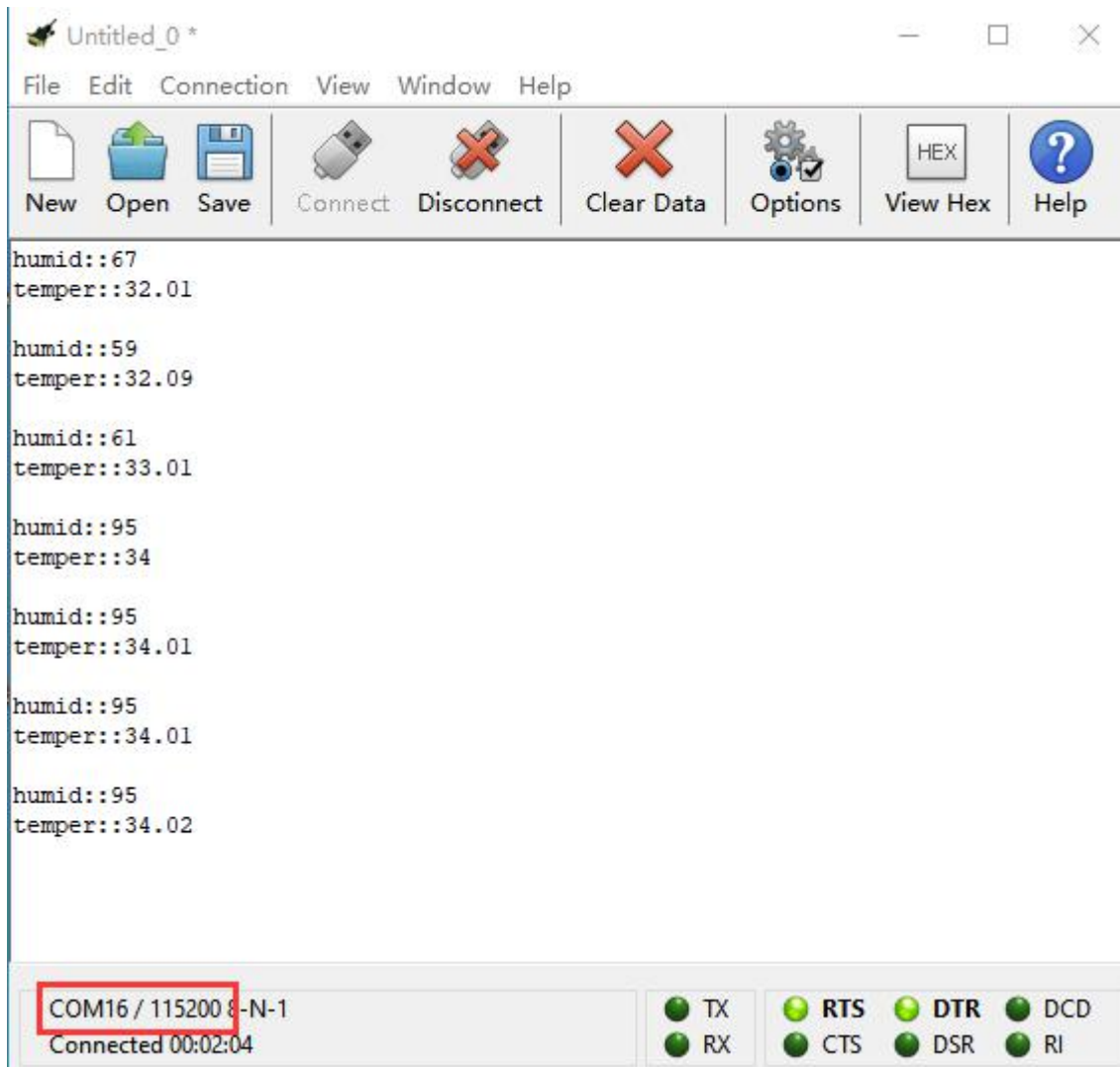
7. Test Result:

Wiring up, dial Voltmeter_Switch to 5V end, plug in external power and dial Power_Switch to ON end and upload code to micro:bit.

Open CoolTerm, click Options and select SerialPort, set COM port and baud rate, set baud rate to 115200. Tap OK and Connect.

CoolTerm serial monitor and 1602 LCD module show the detected temperature and humidity, as shown below:

(Note: You could press reset button if there is random code on 1602 LCD module)



7.Resources

<https://fs.keyestudio.com/KS4020-4021>

