Micro:bit Starter Kit

keyestudio

Nov 30, 2023

KS4022 KS4023 EASY PLUG ULTIMATE STARTER KIT FOR BBC MICRO:BIT STEM EDU

CHAPTER ONE

RESOURCE

Download code ,CoolTerm software ,Driver and more details, please refer to the following link: https://fs.keyestudio.com/KS4022-4023

CHAPTER

TWO

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.

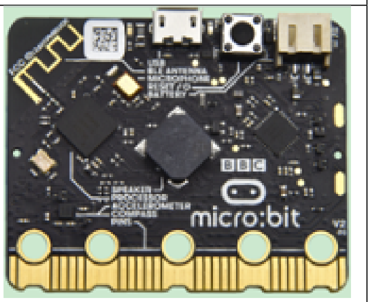
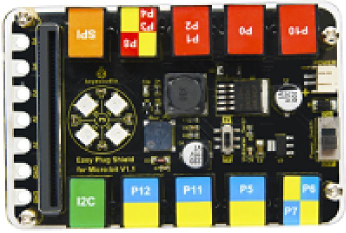
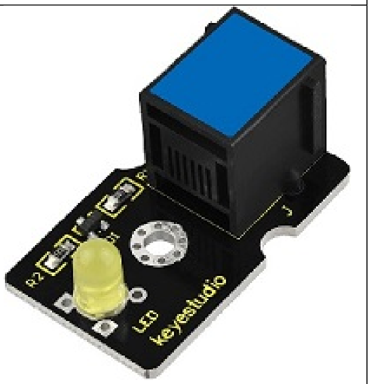
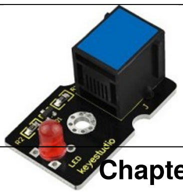
To make you deeply know the micro:bit, we also provide test code and projects.

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

CHAPTER

THREE

2. KIT LIST

#	Component	QTY	Picture
0	Micro:bit main board **is not included** in KS4022 KitMicro:bit main board is **included** in KS4023 Kit	1	
1	EASY Plug Shield for micro bit V1.1	1	
2	EASY Plug Green LED Module	1	
3	EASY Plug Yellow LED Module	1	
4	EASY Plug Red LED Module	1	
5	EASY Plug Photoresistor	1	

CHAPTER

FOUR

3. INTRODUCTION

(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.

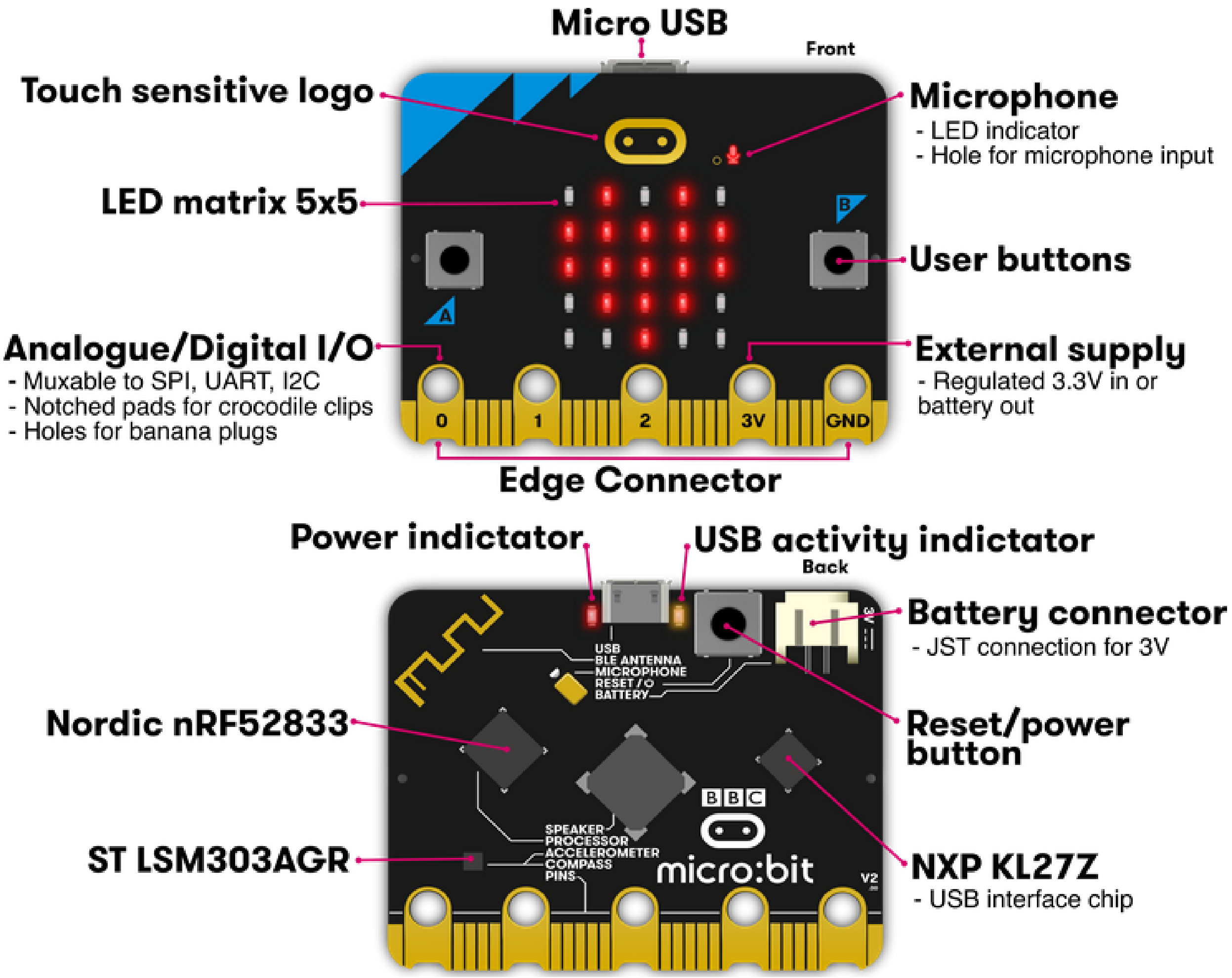
It is equipped with loads of components,including a 5*5 LED dot matrix, 2 programmable buttons, a compass, a Micro USB interface and a Bluetooth module and others. Though it is just the size of a credit card, it boasts multiple functions. To name just a few, it can be applied in programming video games,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

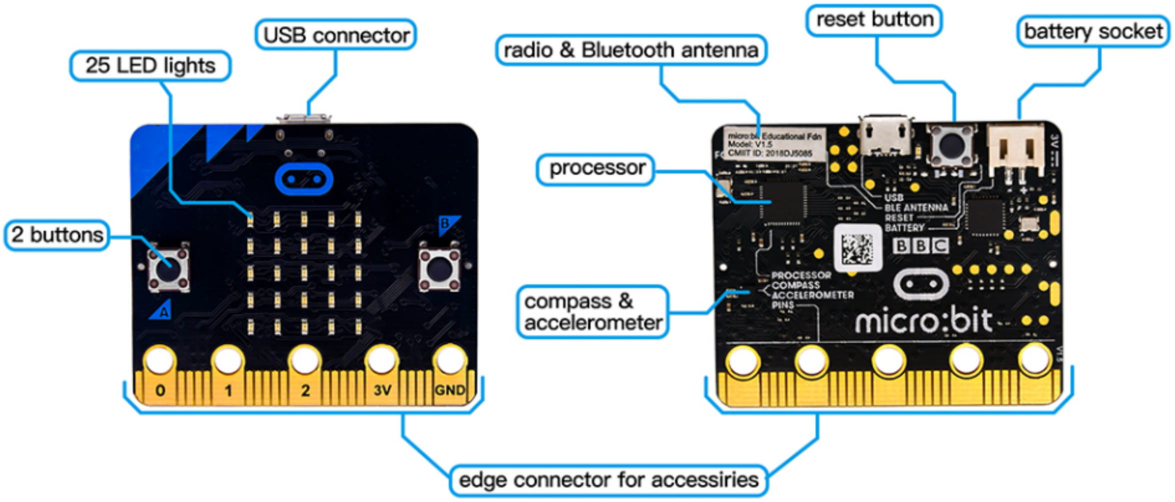
In final analysis, the Micro:bit main board V2 can allow customers to explore more functions so as to make more innovative products.

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

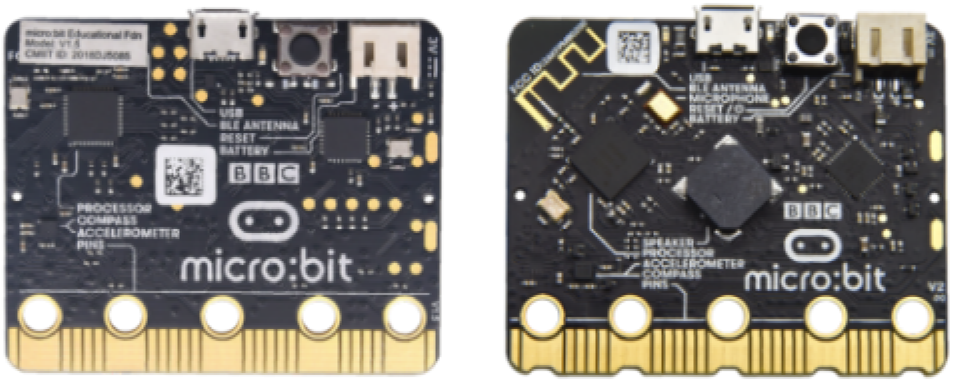
Micro:bit main Board V2.0

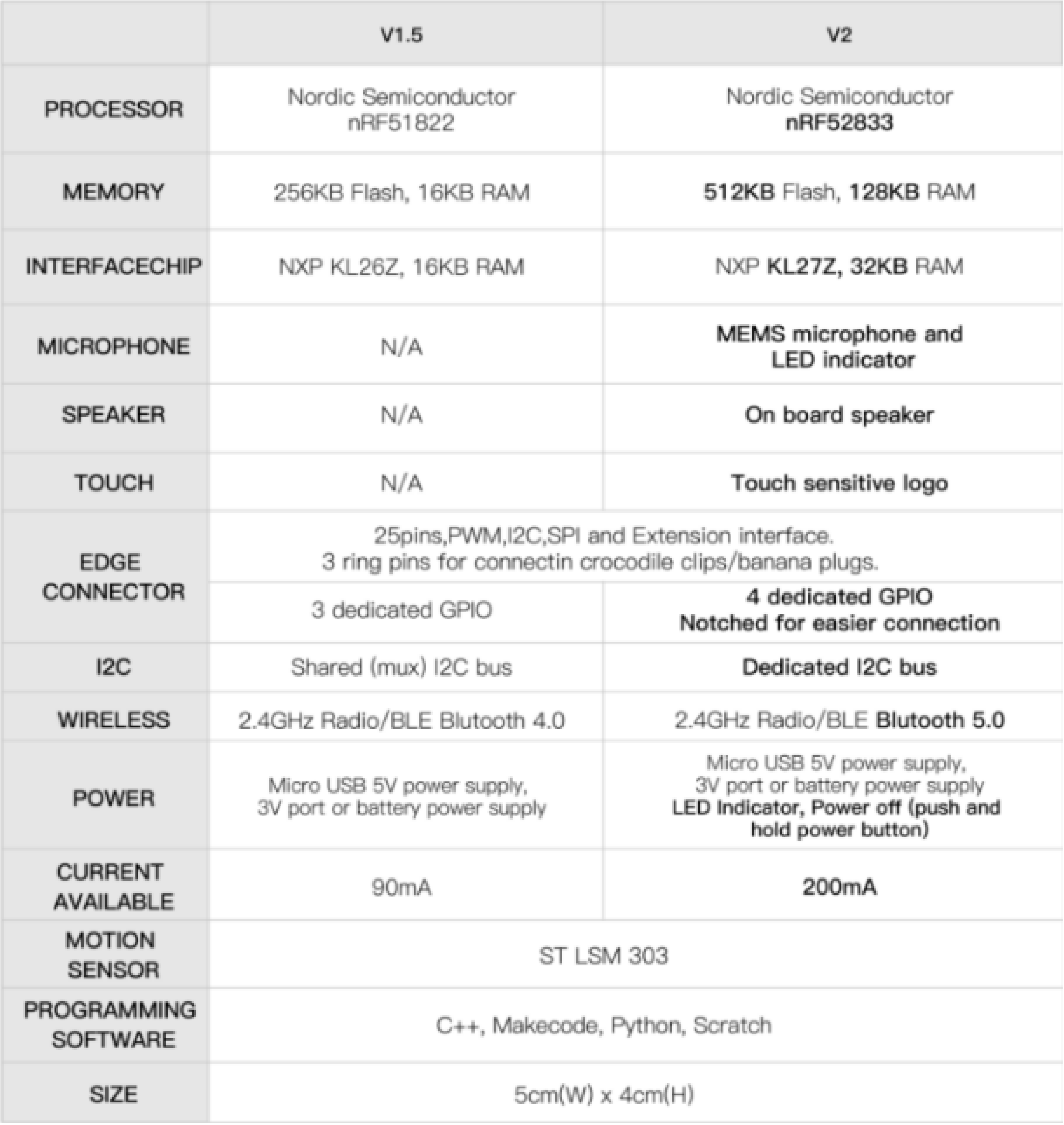


Micro:bit main Board V1.5



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 Blutooth 4.0	2.4GHz Radio/BLE **Blutooth 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)	

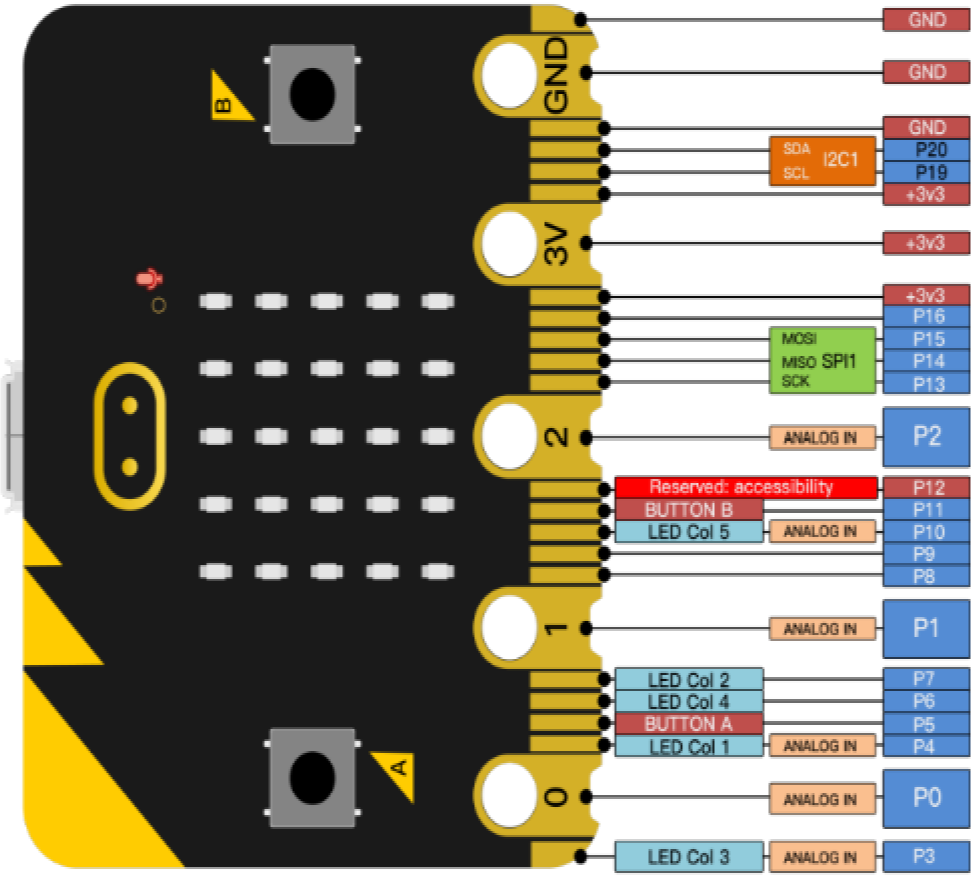
For the Micro: Bit main board V2, pressing the Reset & Power button , it will reset the Micro: Bit and rerun the program. If you hold it tight, the red LED will slowly get darker. When the power indicator becomes darker, releasing the button and your Micro: Bit board will enter sleep mode for power saving .This will make your battery more durable. And you could press this button again to 'wake up' your Micro:bit.

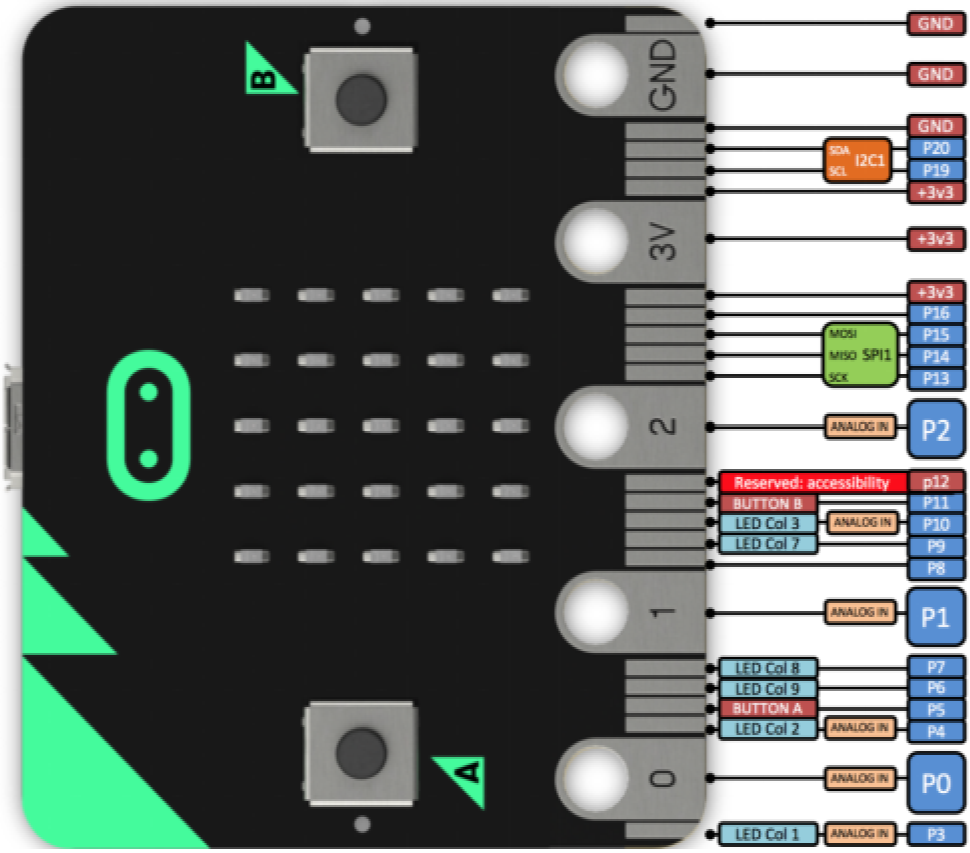
For more information,please resort to following links

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.

CHAPTER

FIVE

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/KS4022-4023 to download the driver file of micro:bit, .

CHAPTER

SIX

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.

6.1 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:

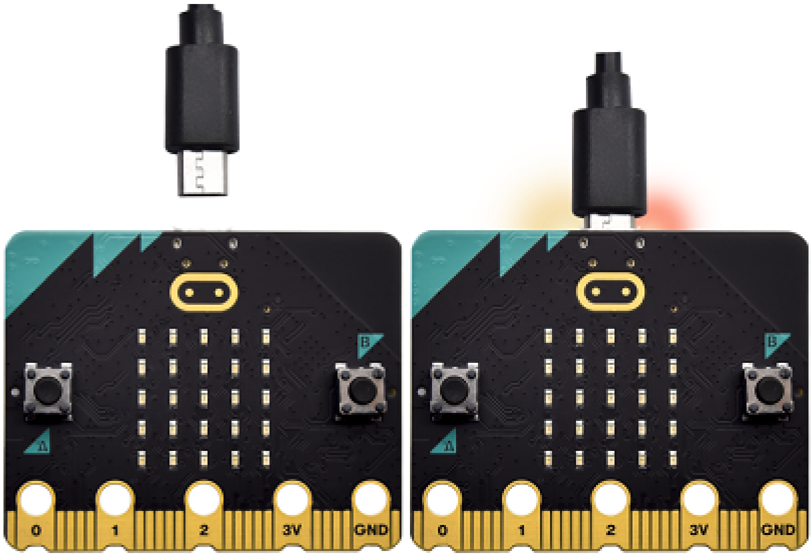
https://microbit.org/guide/quick/

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

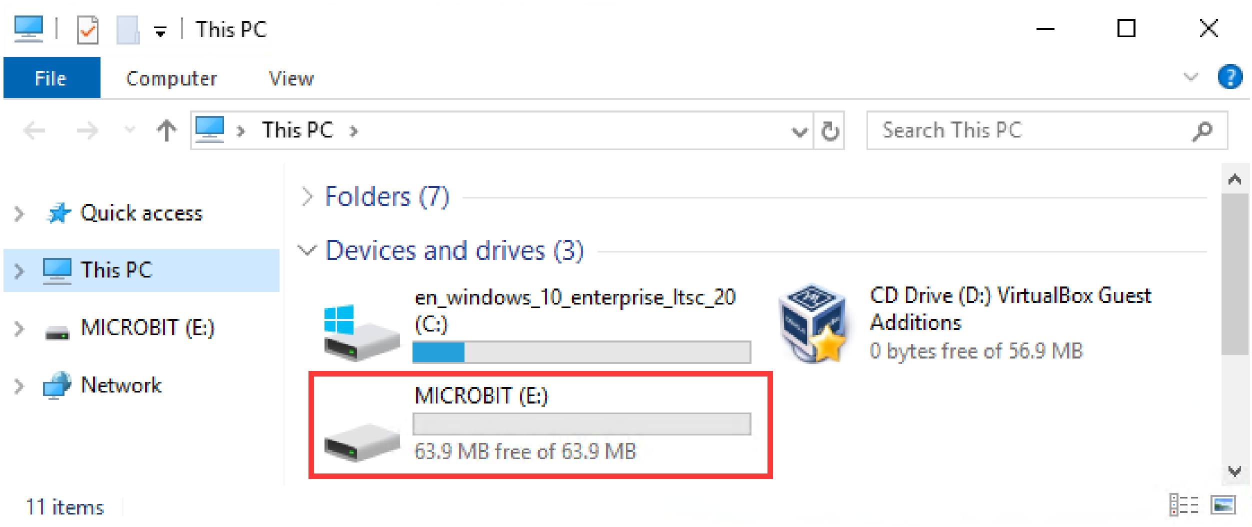
Firstly, link the Micro: Bit main board V2 with your computer via the USB cable. Macs, PCs, Chromebooks and Linux including Raspberry Pisystems are all compatible with the Micro: Bit main board V2.

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.

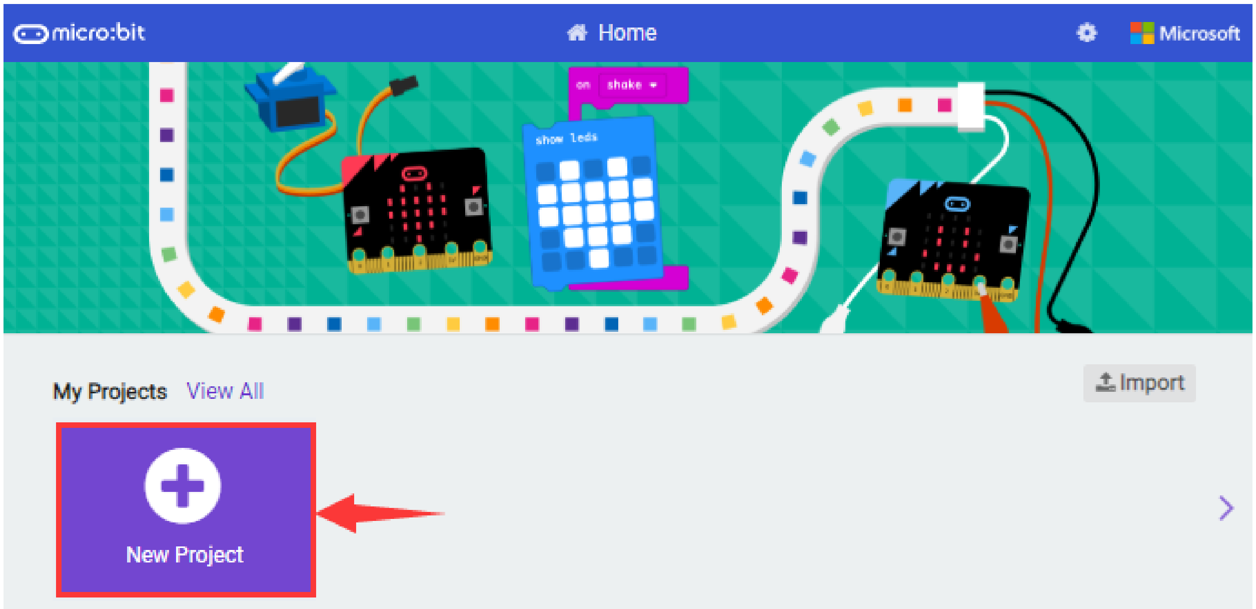


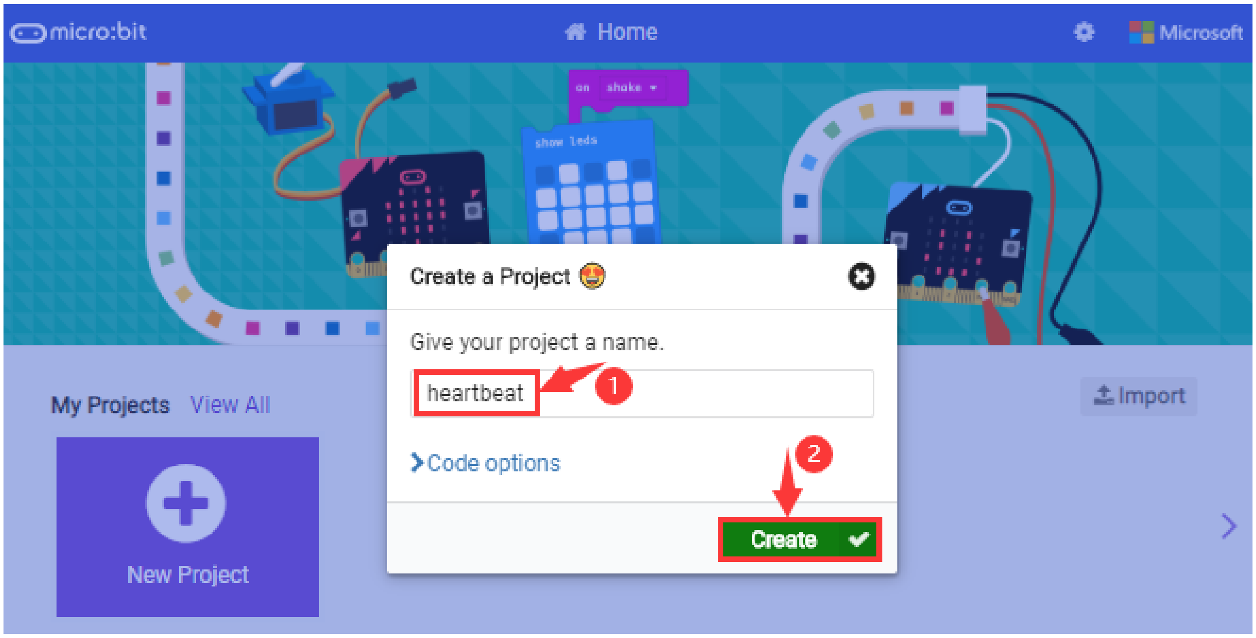
Step 2: writing programs

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)





Create a Project

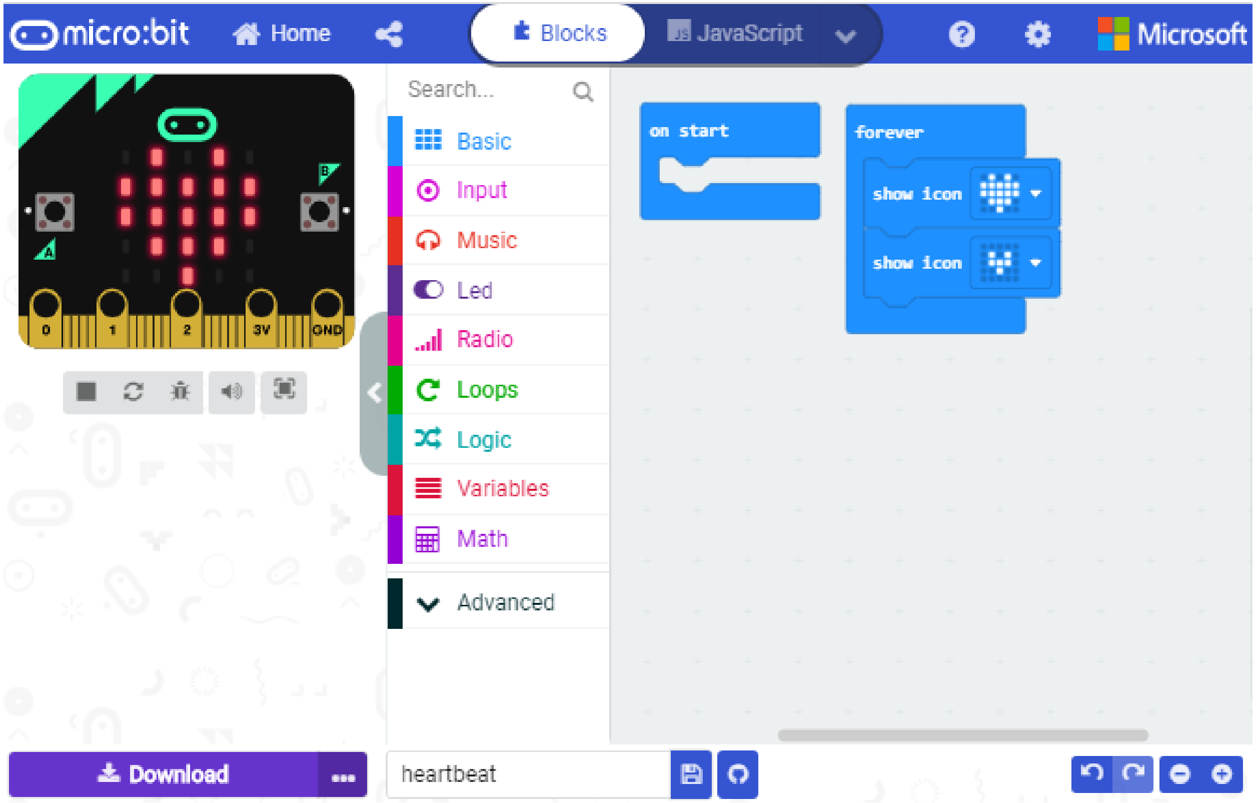
Give your project a name.

heartbeat

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.



```
on start

forever
    show icon
    show icon
```

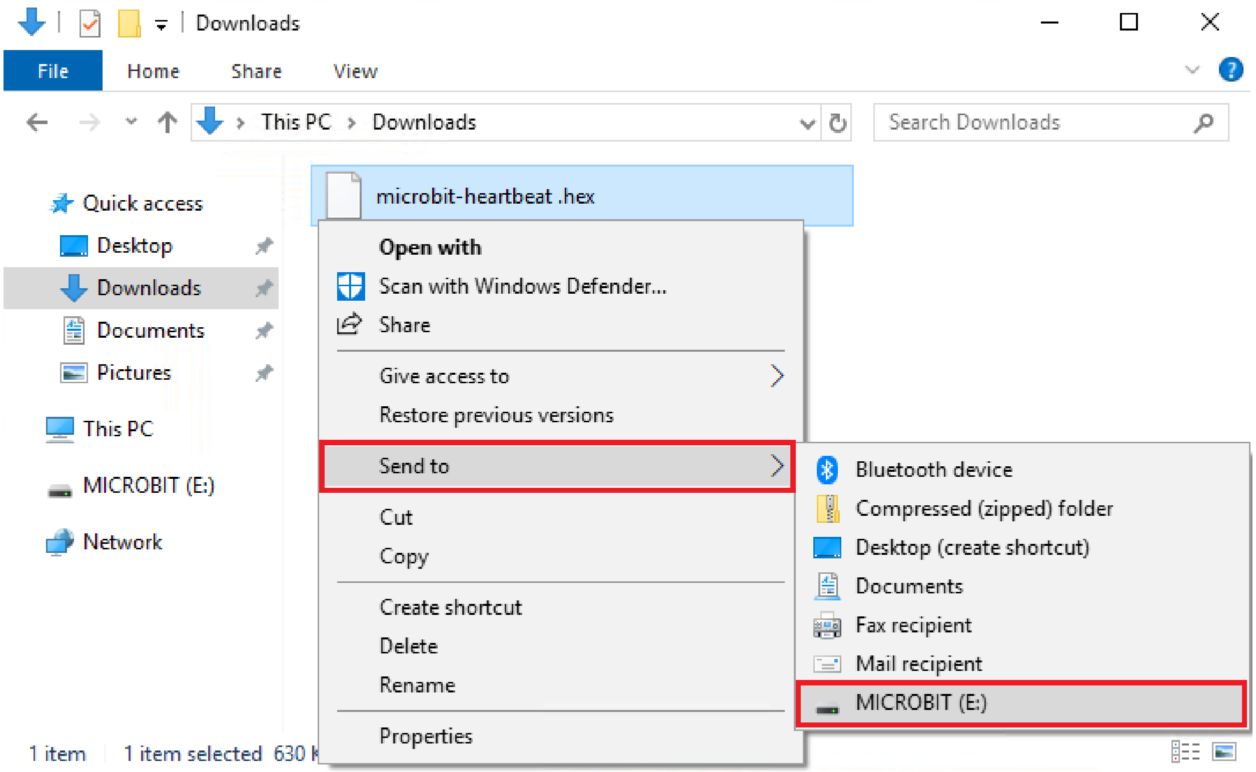
heartbeat

Step 3: download test code

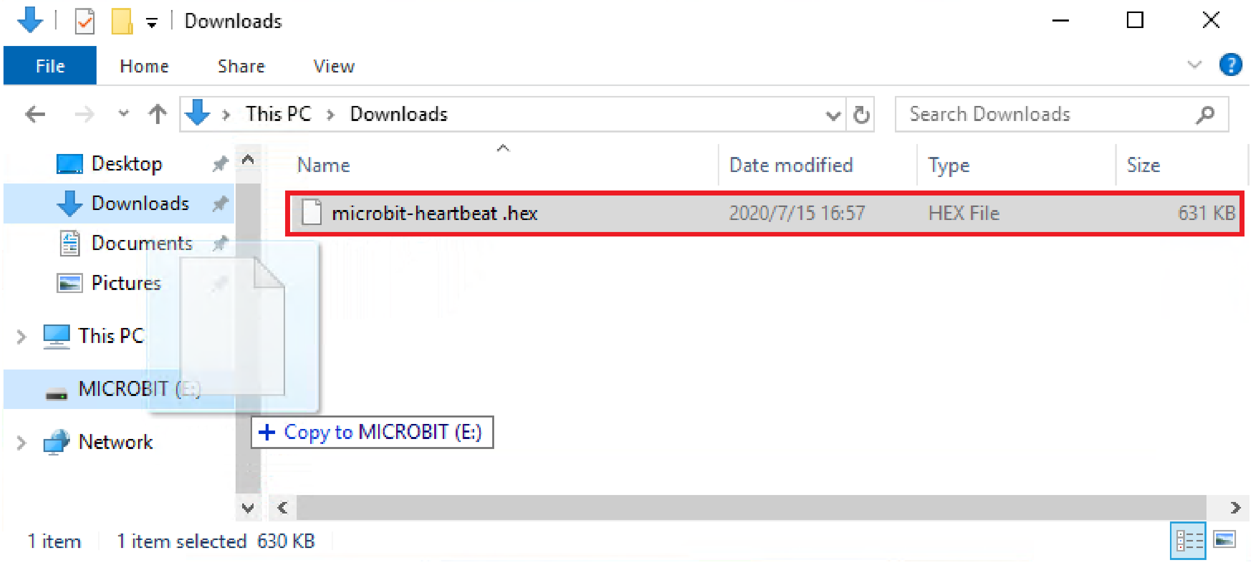
If your computer is Windows 10 and you have downloaded the APP MakeCode for micro:bit to write program, what you will have to do to download the program to your Micro: Bit main board V2 is merely clicking the ‘Download’ button, then all is done.

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

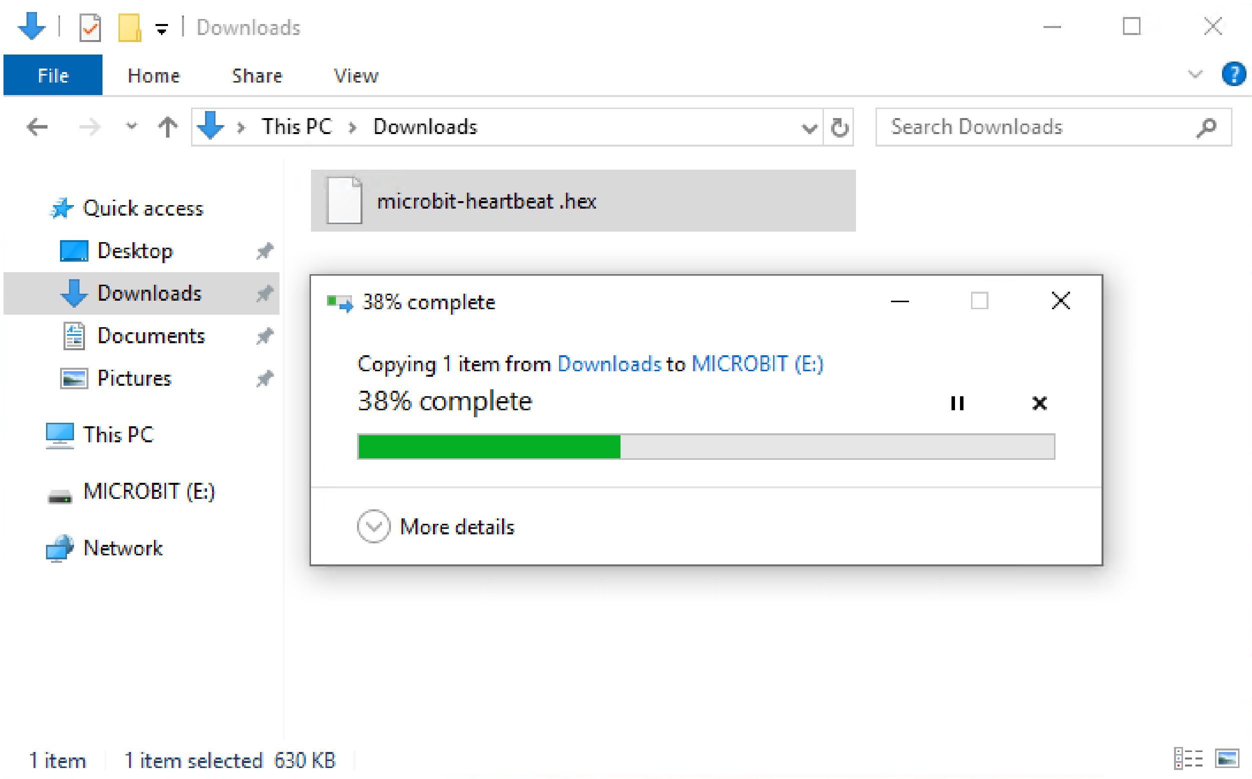
Click the ‘Download’ in the editor to download a “hex” file, which is a compact program format that the Micro: Bit main board can read. Once the hexadecimal file is downloaded, copy it to your board V2 just like the process that you copy the file to the USB driver. If you are running Windows system, 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.



Name	Date modified	Type	Size
microbit-heartbeat .hex	2020/7/15 16:57	HEX File	631 KB



38% complete

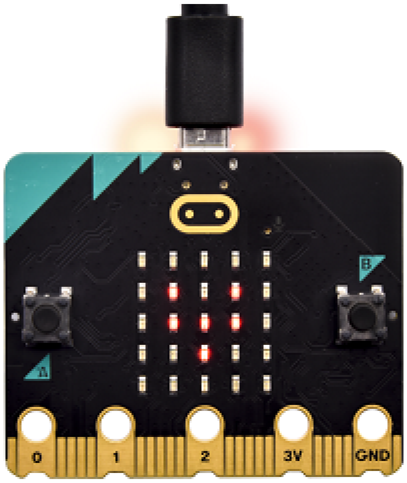
Copying 1 item from Downloads to MICROBIT (E:)

38% complete

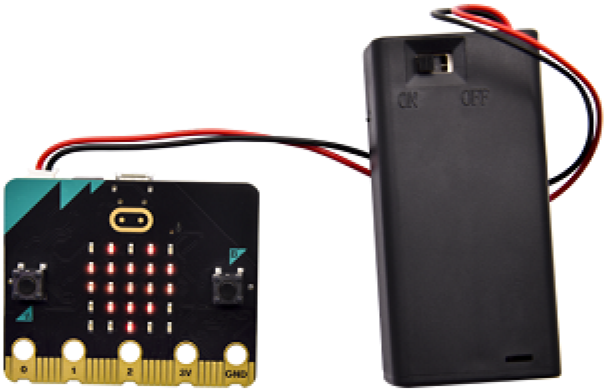
During the process of copying the downloaded hex file to the Micro: Bit main board V2, the yellow signal light on the back side of the board flashes. When the copy is completed, the yellow signal light will stop flashing and remain on.

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 power3V

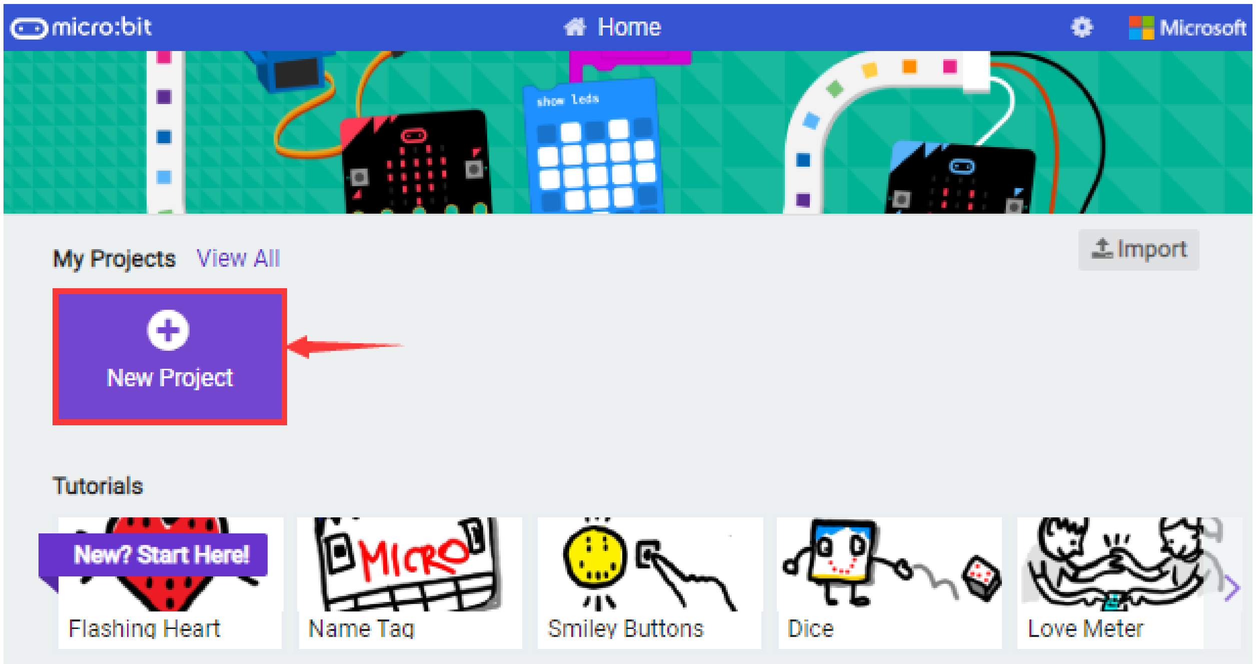
Step 5: other programming languages

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

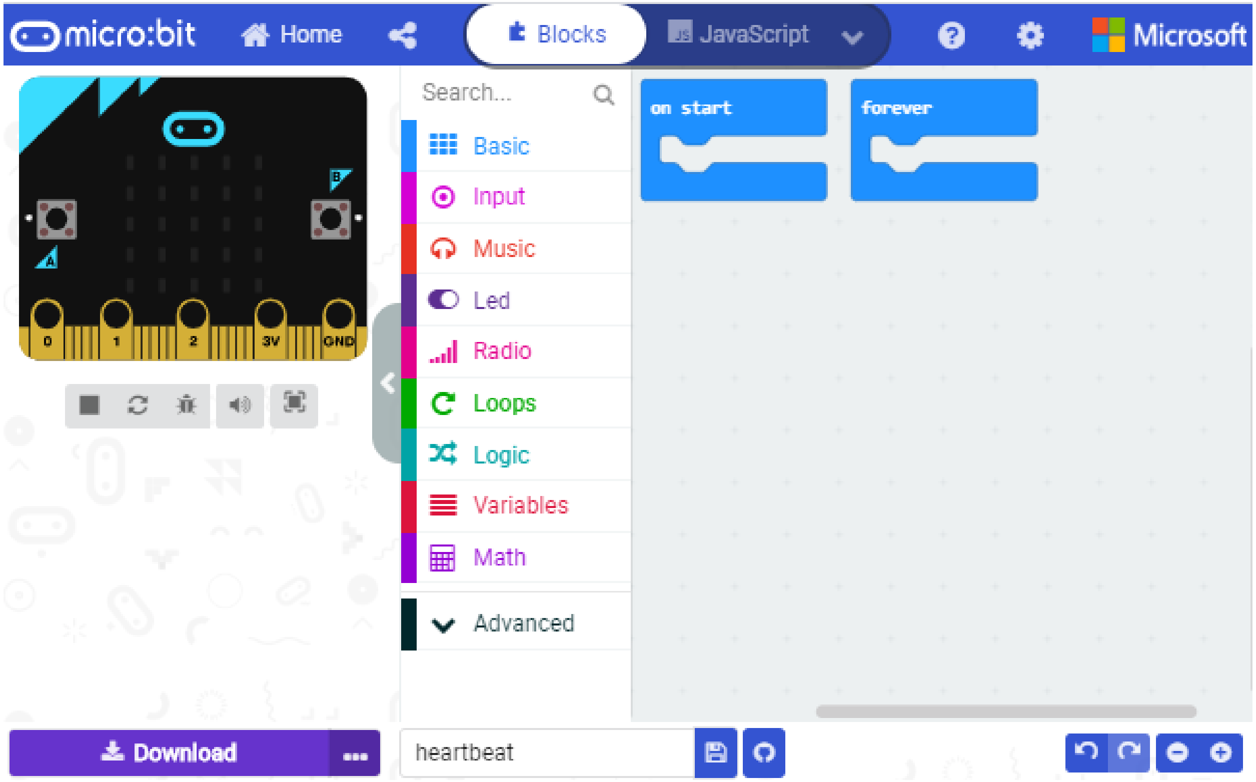
But except for the Makecode graphical programming introduced you can also write Micro: Bit programs in other languages. Go to the link: https://microbit.org/code/ to know about other programming languages, or view the link: https://microbit.org/projects/, to find something you want to have a go.

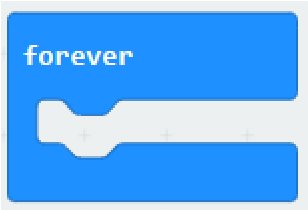
6.2 5.2 Makecode

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



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



There are block and  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.

6.3 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.

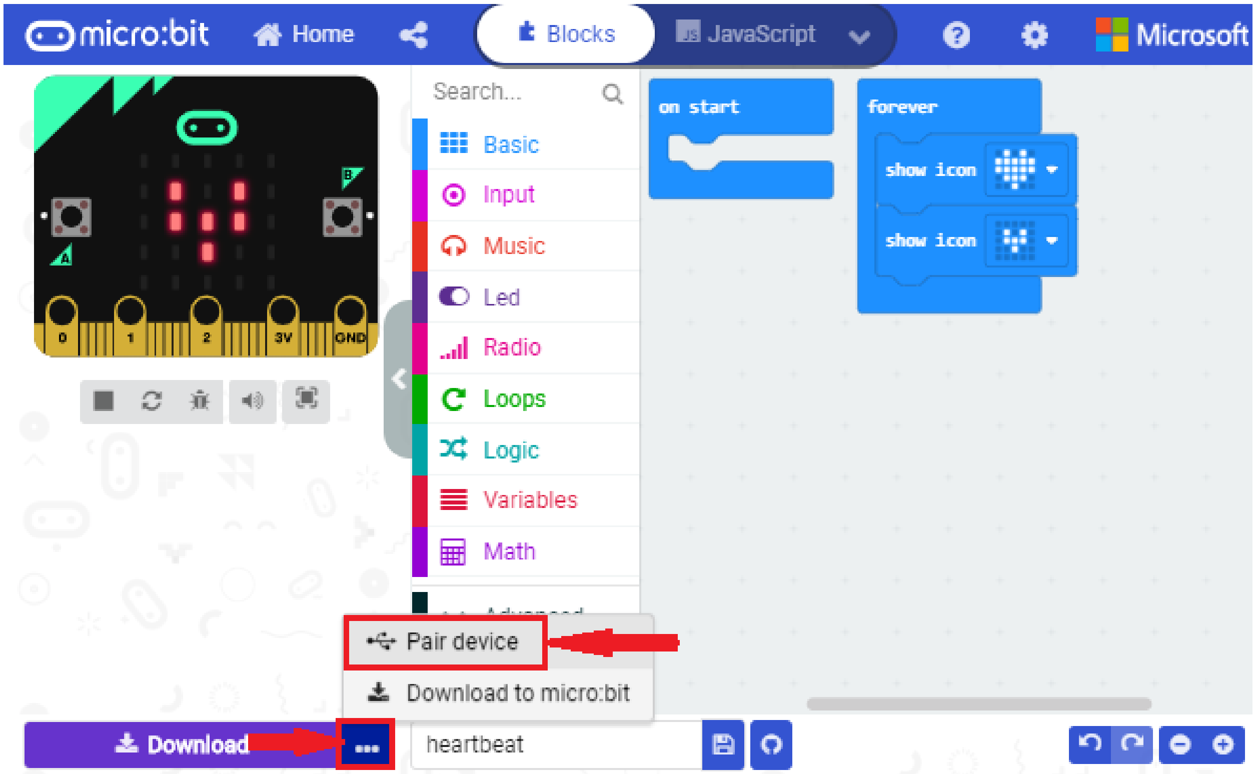
While it is a little more trickier if you are using a browser to enter makecode. However, if you use Google Chrome, suitable for Linux, macOS and Windows 10, the process can be quicker too.

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

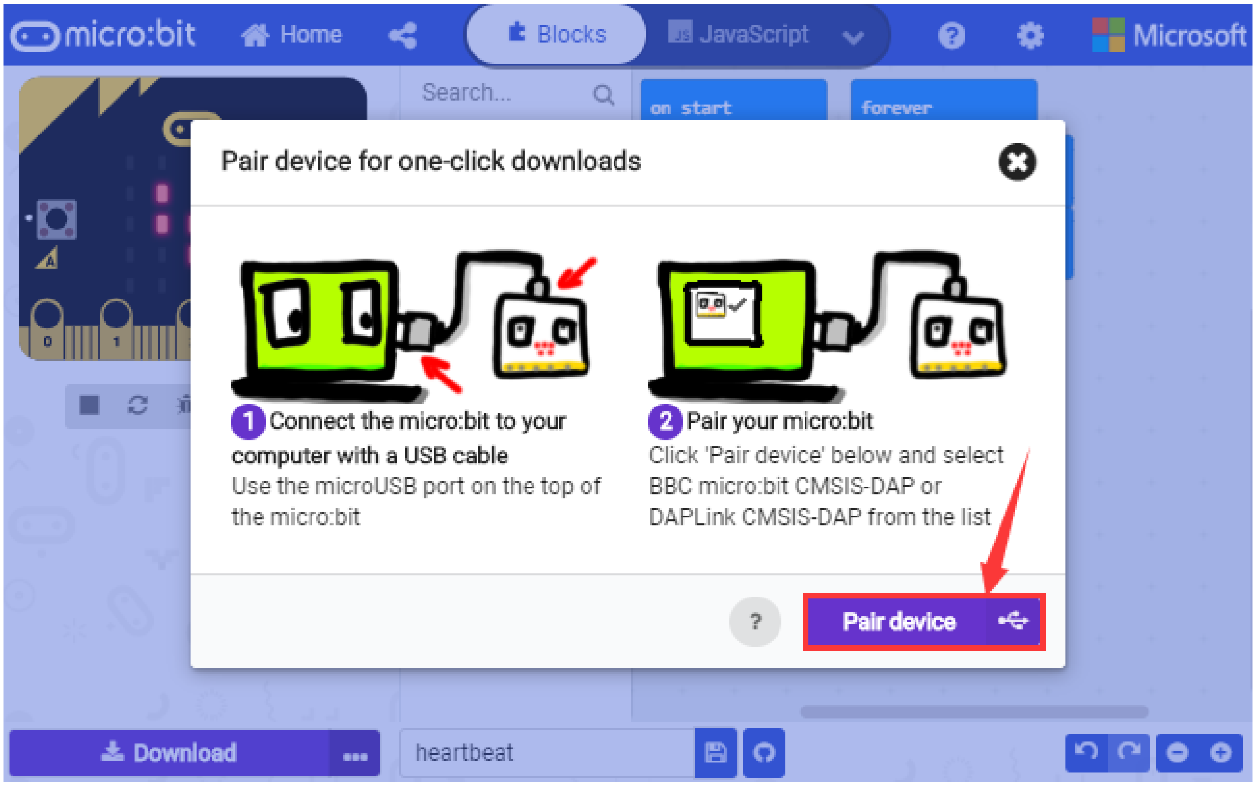
You could refer to the following steps to connect and pair devices.

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.

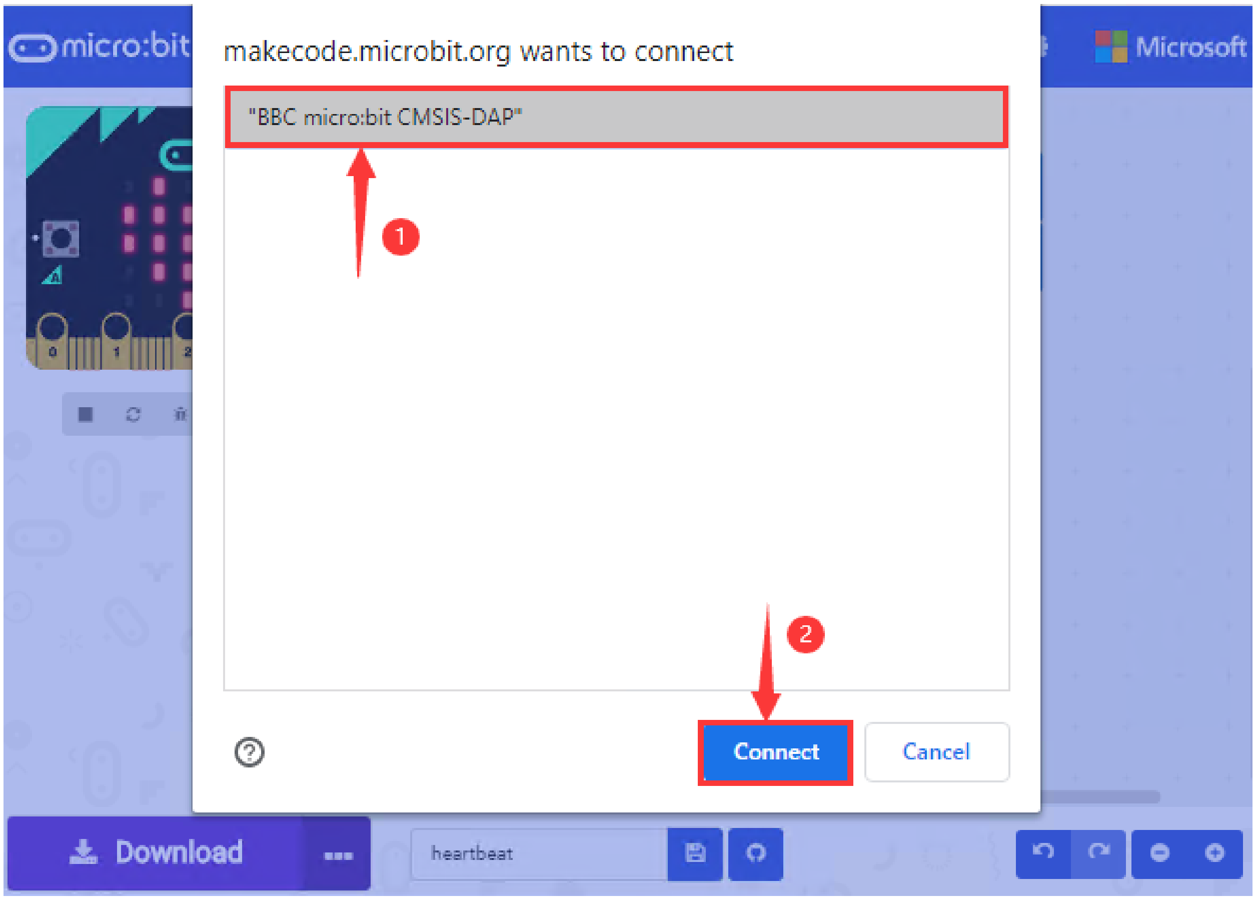


Pair device for one-click downloads

1 Connect the micro:bit to your computer with a USB cable
Use the microUSB port on the top of the micro:bit

2 Pair your micro:bit
Click 'Pair device' below and select BBC micro:bit CMSIS-DAP or DAPLink CMSIS-DAP from the list

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



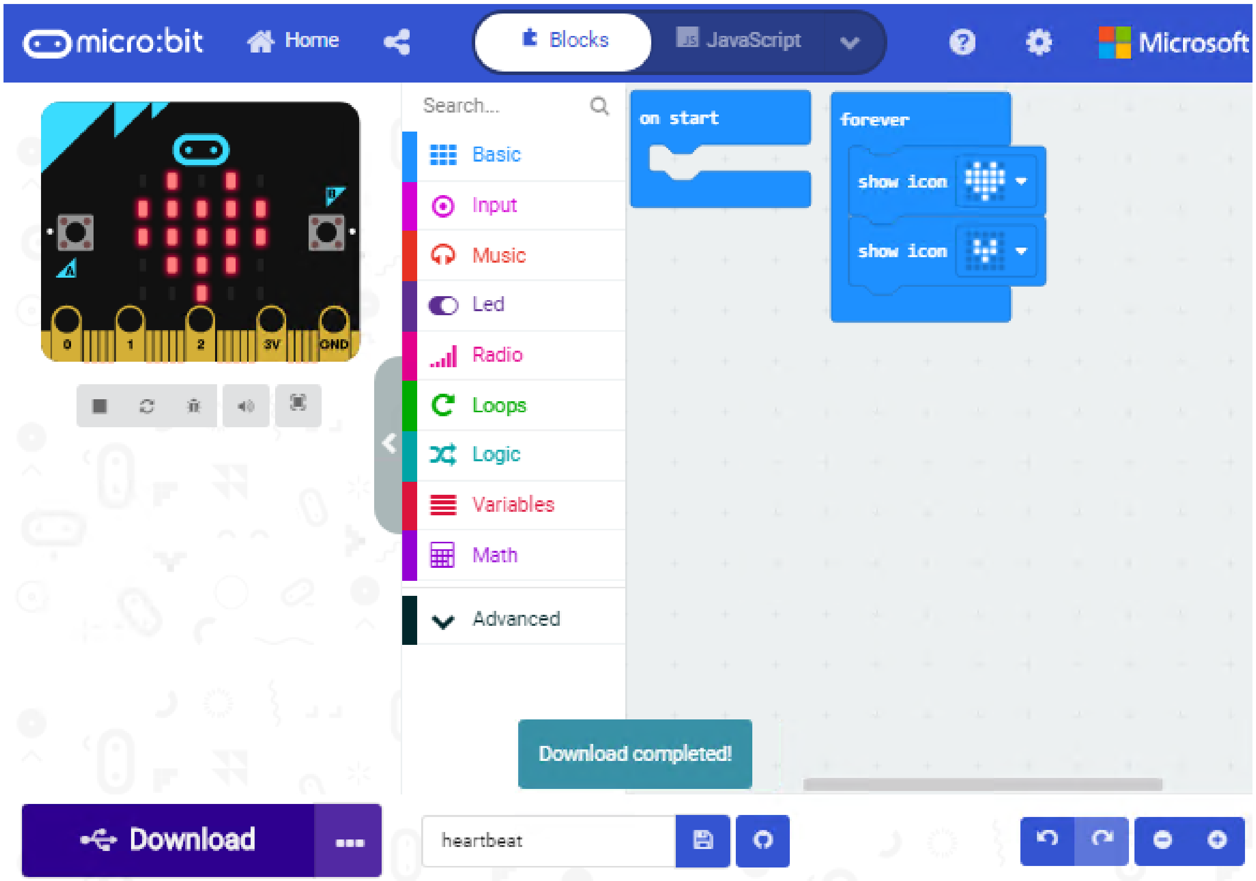
makecode.microbit.org wants to connect

"BBC micro:bit CMSIS-DAP"

We also provide ☑ 6. Troubleshooting-WebUSB in the resource link.

What's more, if you don't know how to update the firmware of micro:bit, refer to the link: https://microbit.org/guide/firmware/ or browse folder ☑ 4.Upgrad the Firmware we provide.

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



```
on start

forever
    show icon
    show icon
```

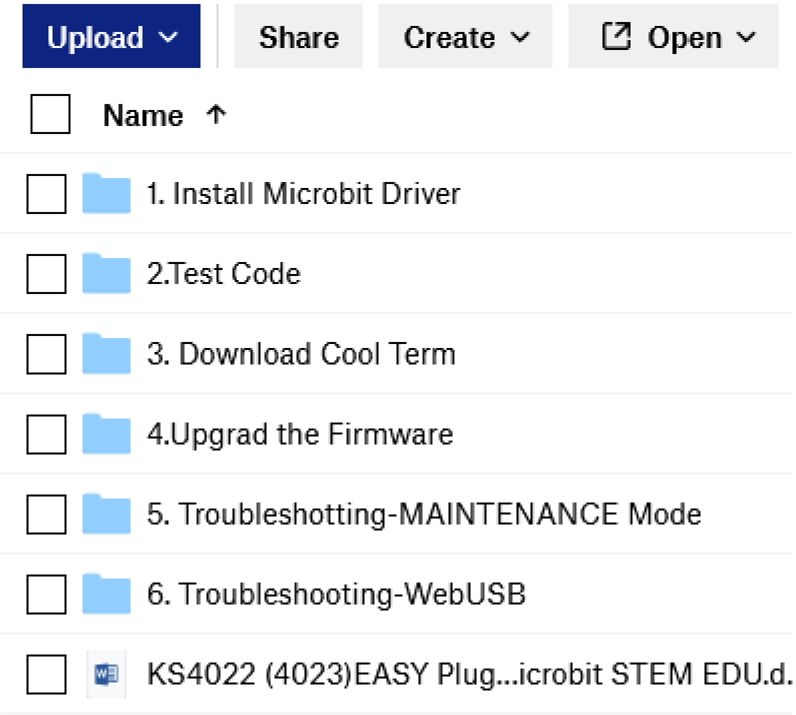
Download completed!

heartbeat

6.4 5.4 Resources and test code

Tools ,test code and other resources can be downloaded via the link https://fs.keyestudio.com/KS4022-4023

Download and unzip the file, you will see a file clip named KS4022 (4023)EASY Plug Ultimate Starter Kit for BBC Micro:bit STEM EDU, and it contains following files:



□ 1. Install Microbit Driver

□ 2.Test Code

□ 3. Download Cool Term

□ 4.Upgrad the Firmware

□ 5. Troubleshotting-MAINTENANCE Mode

□ 6. Troubleshooting-WebUSB

□ KS4022 (4023)EASY Plug...icrobit STEM EDU.d.

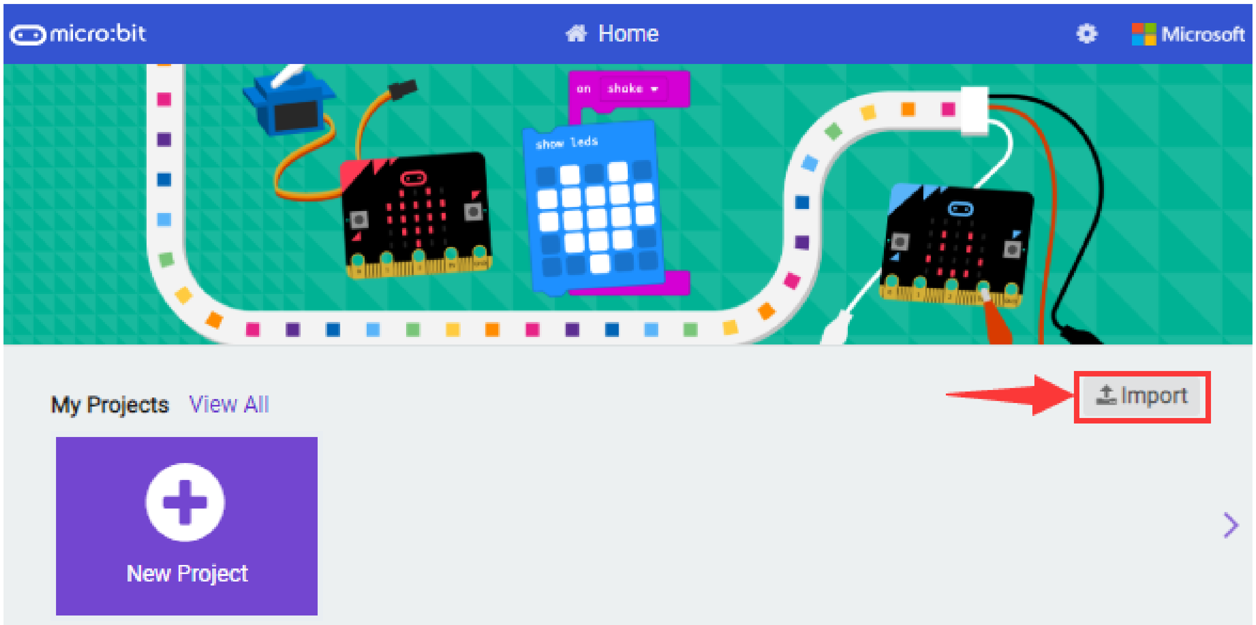
6.5 5.5 Import test code

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.

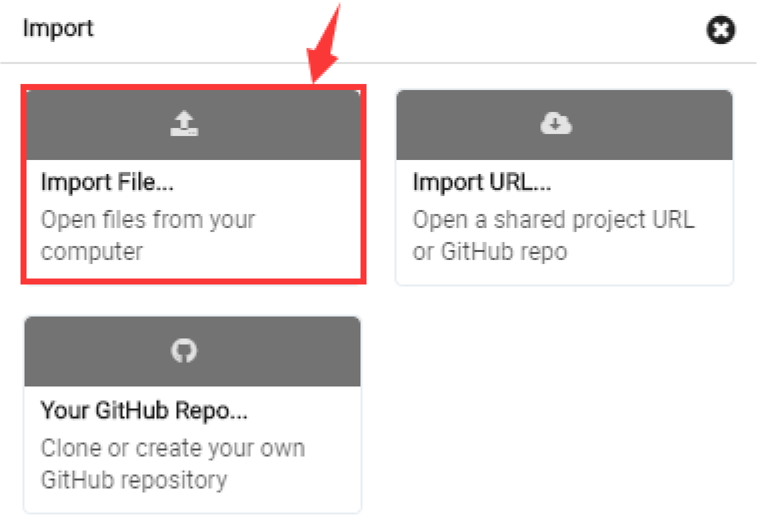
For simple projects, dragging a block of code to complete the program is recommended. For complex projects, it is recommended to conduct the program by importing the hexadecimal code file we provide.

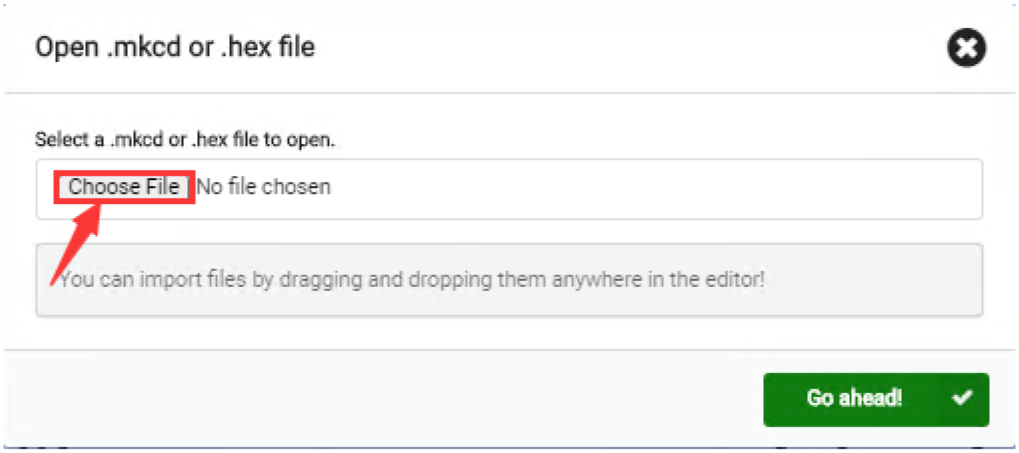
Let's take the "Heatbeat" project as an example to show how to load the code.

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



Click “Import File”;


Import
Import File...
Open files from your computer
Import URL...
Open a shared project URL or GitHub repo
Your GitHub Repo...
Clone or create your own GitHub repository

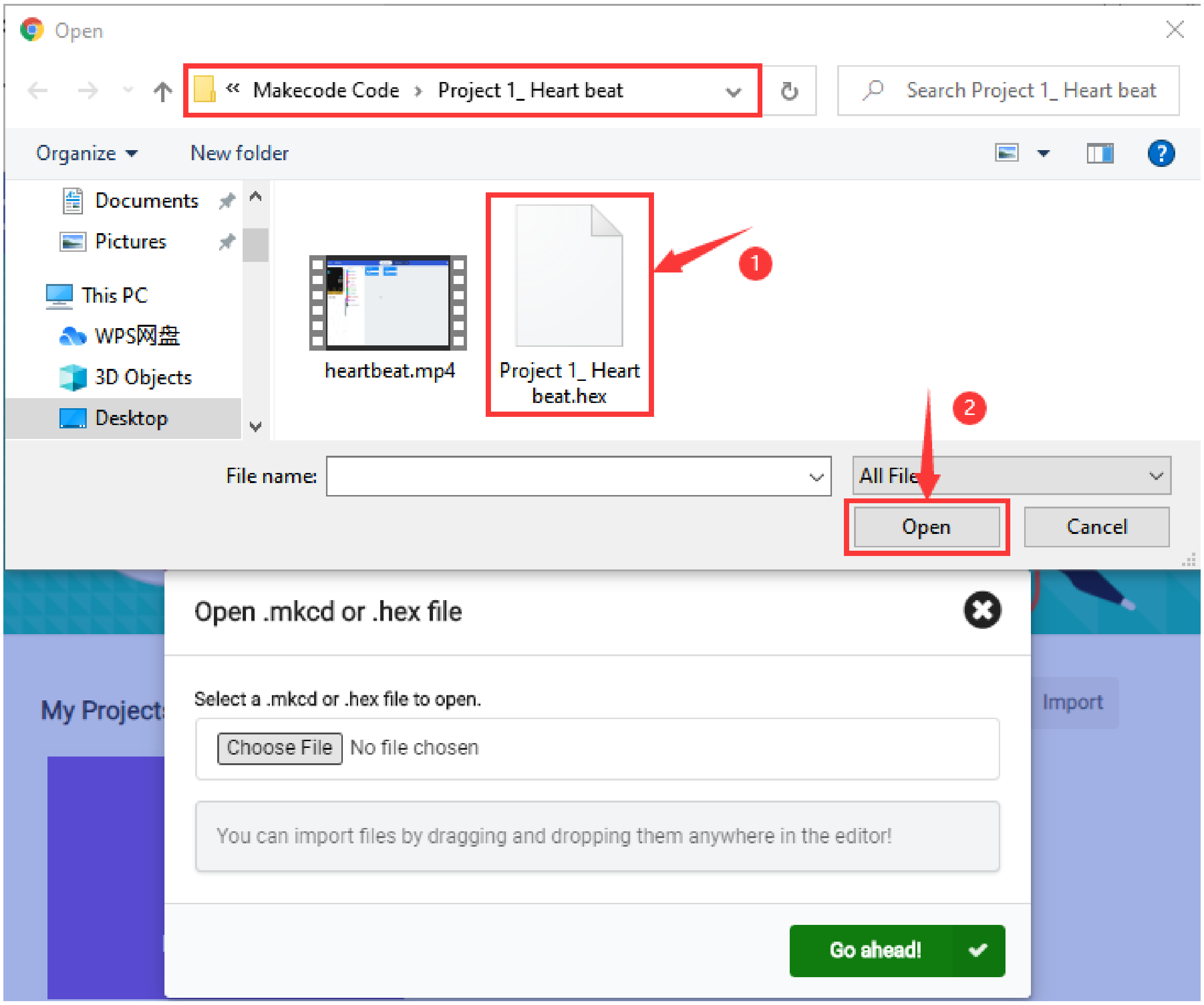

Open .mkcd or .hex file

Select a .mkcd or .hex file to open.

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

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

Then click Go ahead! ✓.

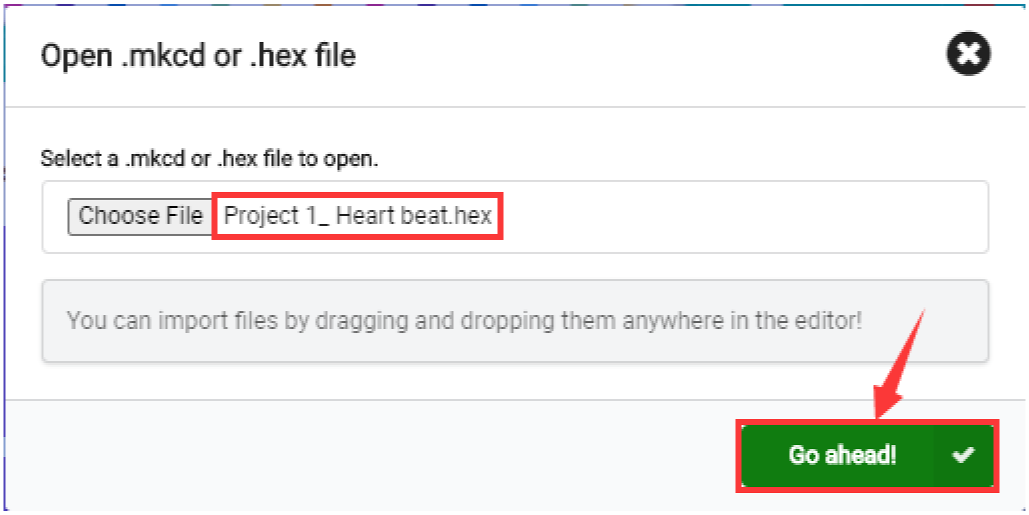


Open .mkcd or .hex file

Select a .mkcd or .hex file to open.

No file chosen

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



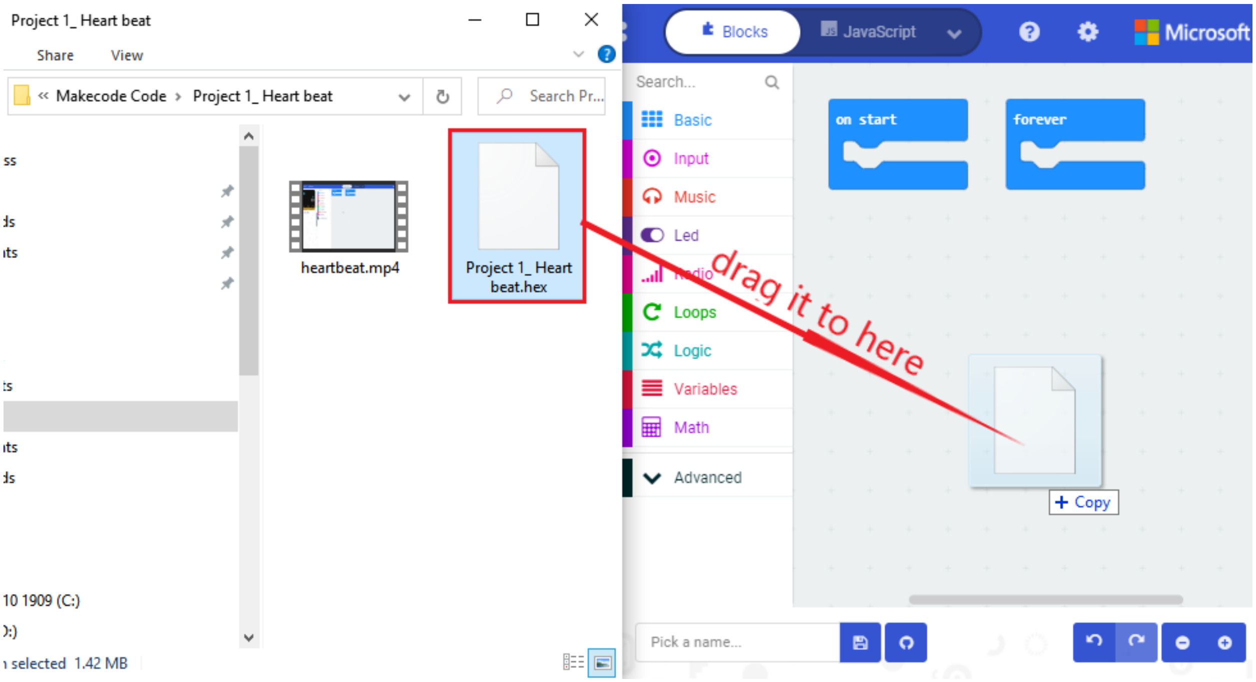
Open .mkcd or .hex file

Select a .mkcd or .hex file to open.

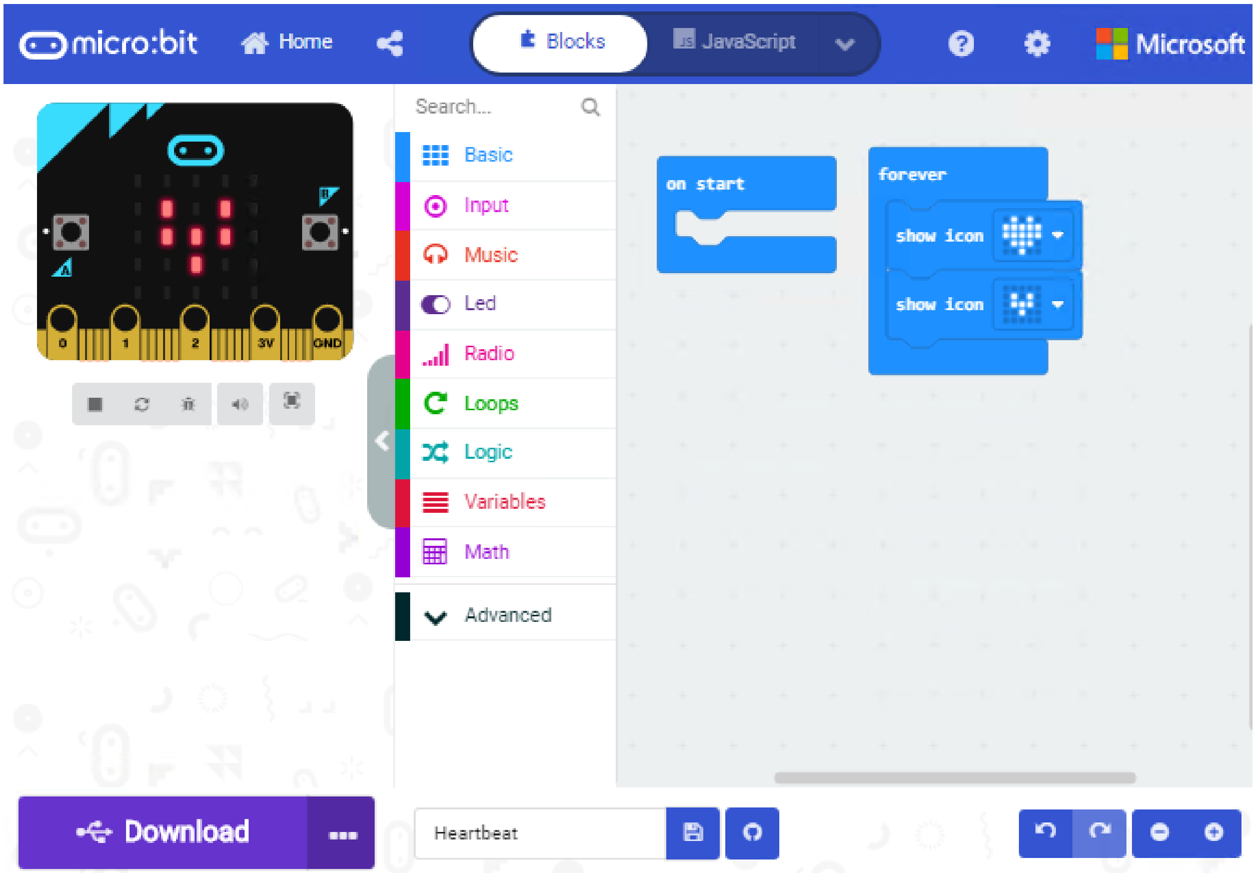
Project 1_ Heart beat.hex

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

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



After a few seconds, it is done.



Note: if your computer system is Windows 7 or 8 instead of Windows 10, the pairing cannot be done via Google Chrome. Therefore, digital signal or analog signal of sensors and modules cannot be shown on the serial port simulator. However, you need to read the corresponding digital signal or analog signal.

So what can we do? You can use the CoolTerm software to read the serial port data of the micro:bit. Next chapter is about how to install CoolTerm.

6.6 5.6 CoolTerm Installation

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

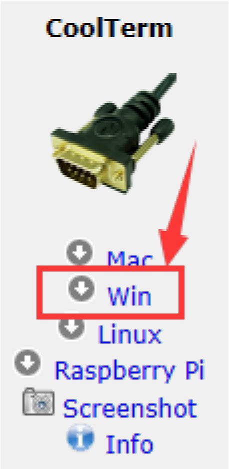
Download CoolTerm program:

Link of Download: https://freeware.the-meiers.org/

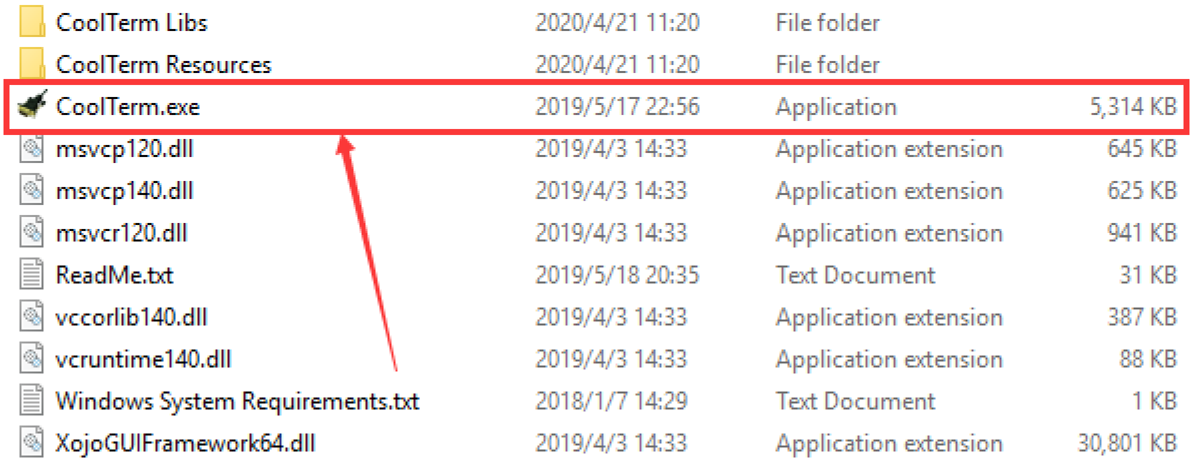
1.After the download, we need to install CoolTerm program file, below is Window system taken as an example.

2.Choose "win Win" to download the zip file of CoolTerm

3.Unzip file and open it. (also suitable for Mac and Linux system)

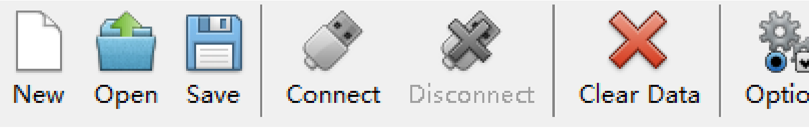


4.Double-click CoolTerm.exe .



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

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



ICON	Function
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
HEX View Hex	Displays the Terminal Data in Hexadecimal Format
? Help	Displays the Help Window

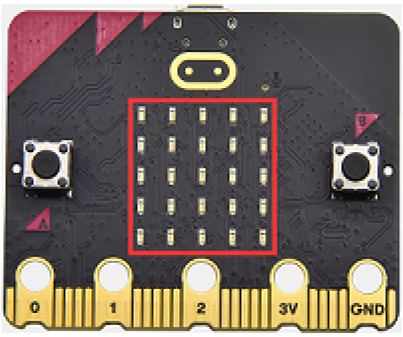
CHAPTER

SEVEN

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)

7.1 Project 1: Heartbeat



1. Project Description:

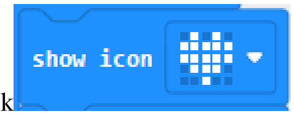
This project is easy to conduct with a micro:bit V2 main board, a Micro USB cable and a computer. The micro:bit LED dot matrix will display a relatively big heart-shaped pattern and then a smaller one. This alternative change of this pattern is like heart beating. This experiment serves as a starter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

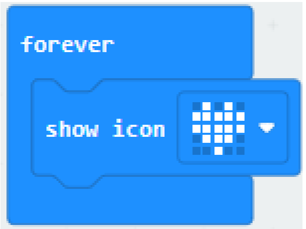
2. Components Needed:

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

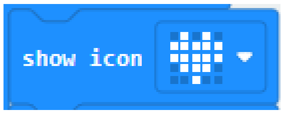
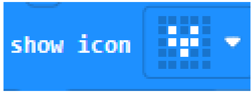
3. Test Code:

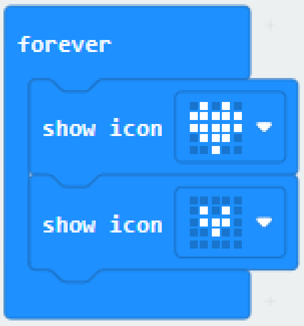
Attach the Micro:bit main board V2 to your computer via the Micro USB cable and begin editing.

Firstly, click ▦ Basic and find and drag the block  show icon to “forever”;

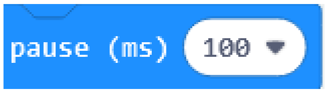


```
forever
  show icon
```

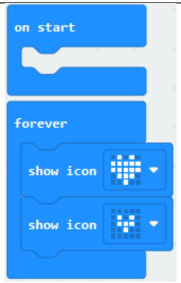
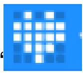
Secondly, click Basic again and find and drag the block  show icon to module “forever” and click the little triangle to select  show icon;



forever
show icon
show icon

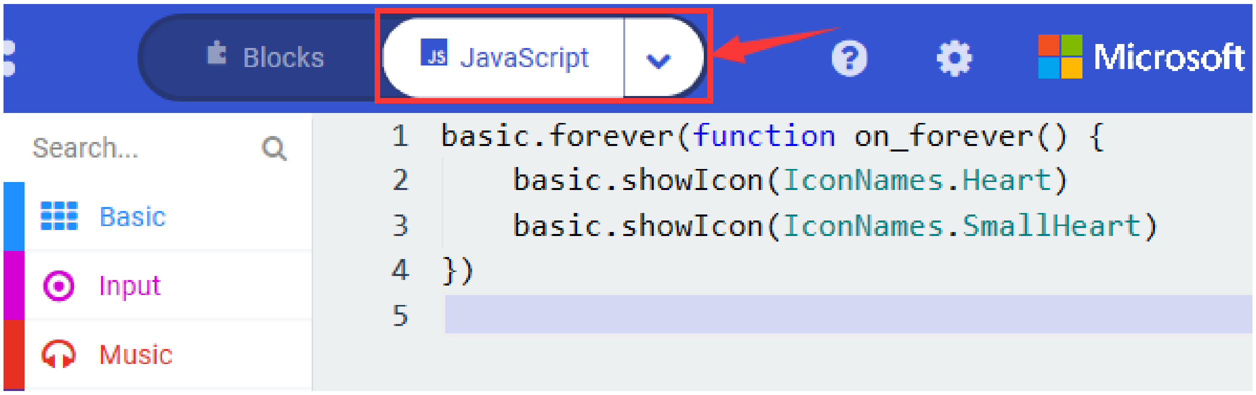
Thirdly, click Basic and find and drag the block  pause (ms) 100 to the code block and click the littler triangle to select 500;

Complete Program

 on start forever show icon show icon	In“on start”the program only runs once; In”forever”the program runs cyclically; The LED dot matrix displays pattern “”;The LED dot matrix displays pattern“ ”

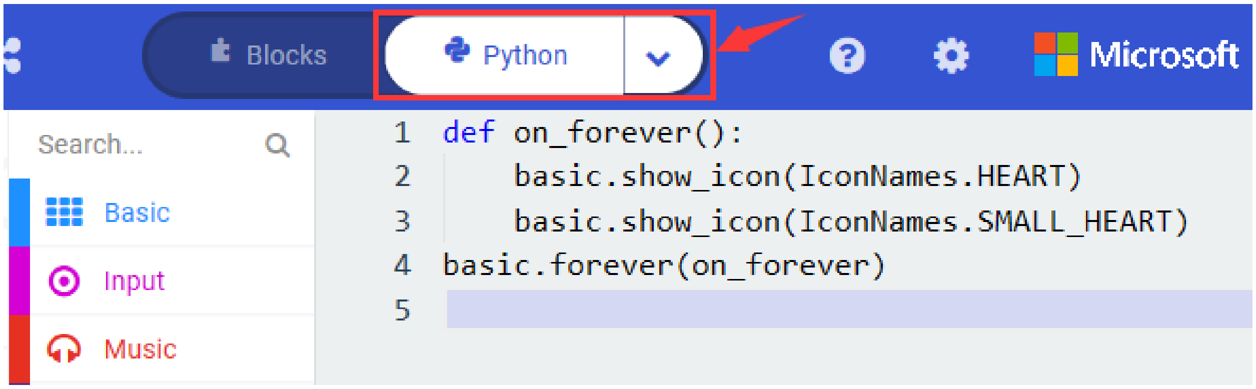
Note: the “on start” means that blocks in the code are only executed once, “forever” implies that code will run cyclically.

Click JavaScript, you will find the corresponding programming languages.



```
basic.forever(function on_forever() {
    basic.showIcon(IconNames.Heart)
    basic.showIcon(IconNames.SmallHeart)
})
```

Click ⌄ to choose Python, you will find the corresponding Python programming languages.



```
def on_forever():
    basic.show_icon(IconNames.HEART)
    basic.show_icon(IconNames.SMALL_HEART)
basic.forever(on_forever)
```

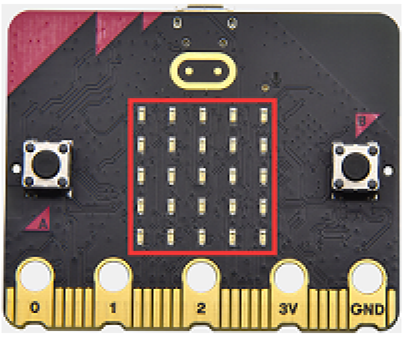
4. Test Results:

Uploading test code to micro:bit main board V2 and keeping the connection with the computer to power the main board, the LED dot matrix shows pattern “♥”and then “♥”alternatively.

(Please refer to chapter 5.3 to know how to download test code quickly.)

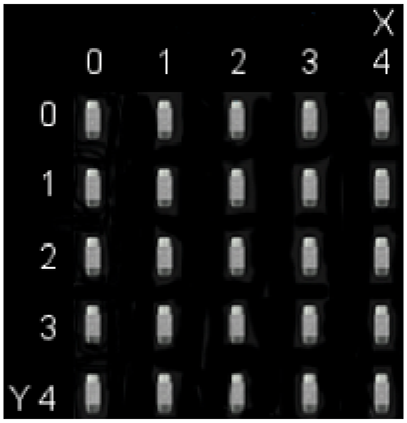
If the downloading is not smooth, please remove the micro USB from the main board and then reconnect them and reopen Makecode to try again.

7.2 Project 2: Light A Single LED



1. Project Description:

The LED dot matrix consists of 25 LEDs arranged in a 5 by 5 square. In order to locate these LEDs quickly, as the figure shown below, we can regarded this matrix as a coordinate system and create two aces 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 (1,0and the LED positioned in the fifth of the fourth column is (3,4and others likewise.

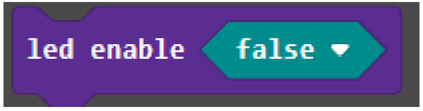
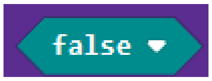


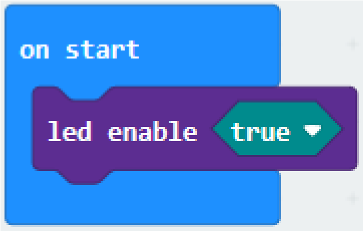
2. Components Needed:

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

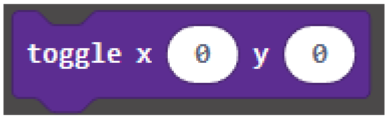
3. Test Code:

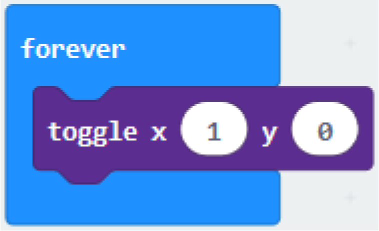
Attach the Micro:bit main board V2 to your computer via the Micro USB cable and begin editing.

Firstly, click Led and then the ••• more to find and drag the block  led enable false to block “on start”; and click  false to select true;

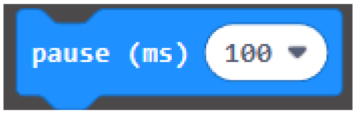


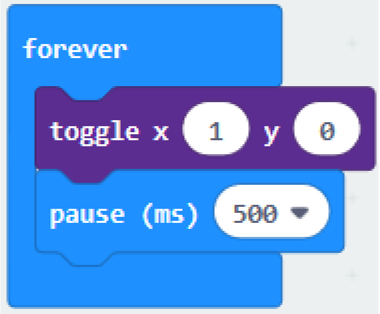
```
on start
    led enable true
```

Secondly, click Led and to find and drag the block  toggle x 0 y 0 to block “forever” and alter “x0” to”x1”;

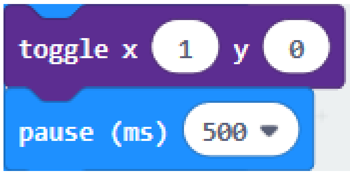


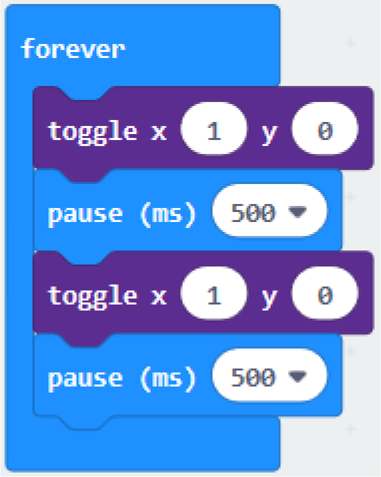
```
forever
    toggle x 1 y 0
```

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

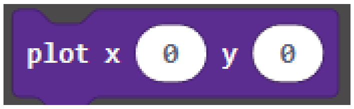


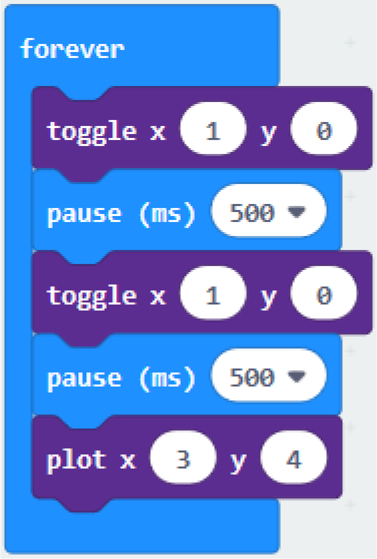
```
forever
  toggle x 1 y 0
  pause (ms) 500
```

Fourthly, copy the block  toggle x 1 y 0 pause (ms) 500 and place it into “forever” block;



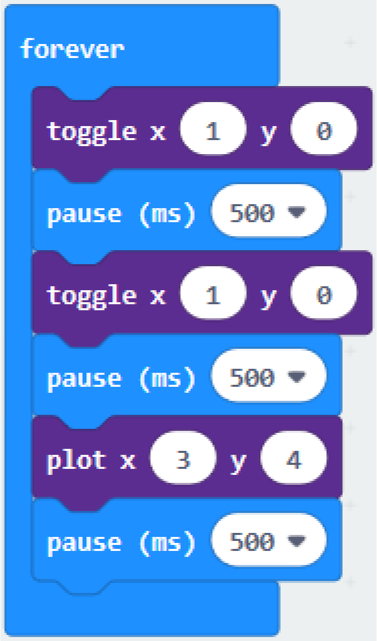
```
forever
  toggle x 1 y 0
  pause (ms) 500
  toggle x 1 y 0
  pause (ms) 500
```

Fifthly, click Led 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”;

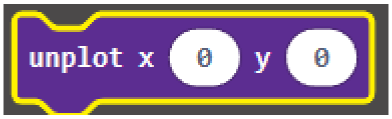


```
forever
    toggle x 1 y 0
    pause (ms) 500
    toggle x 1 y 0
    pause (ms) 500
    plot x 3 y 4
```

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

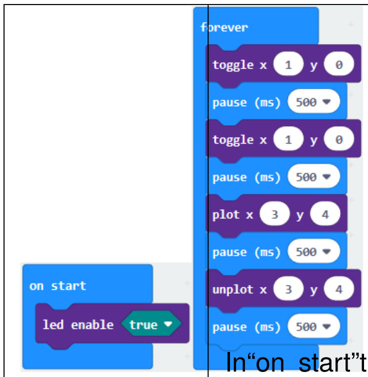


```
forever
    toggle x 1 y 0
    pause (ms) 500
    toggle x 1 y 0
    pause (ms) 500
    plot x 3 y 4
    pause (ms) 500
```



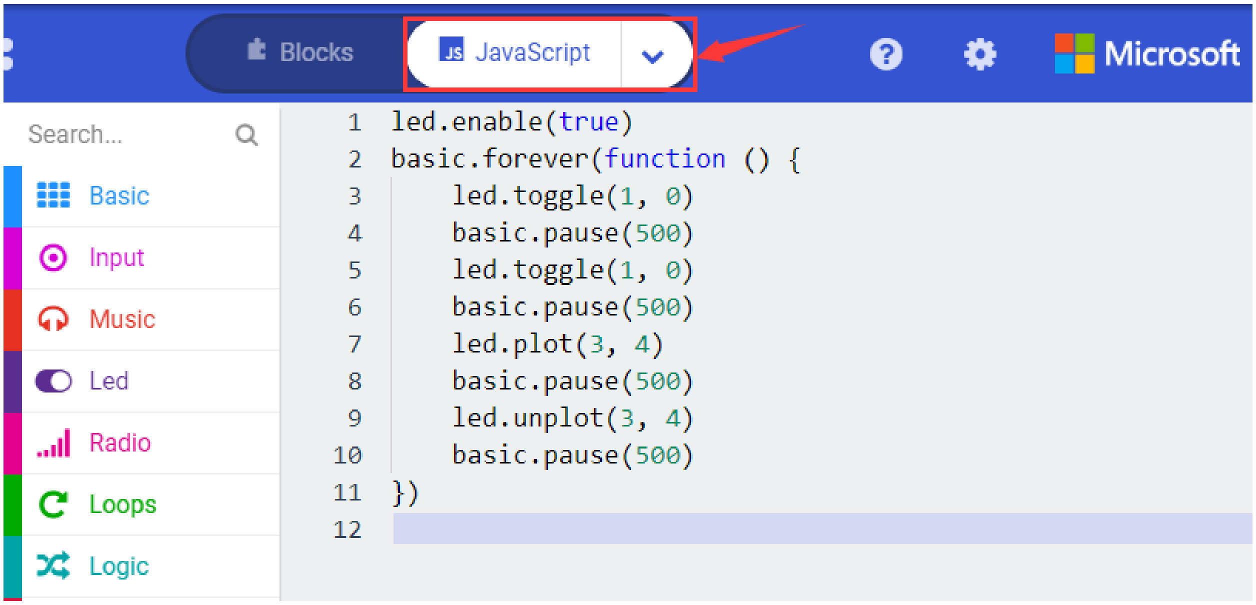
Lastly, click Led to find and drag the block unplot x 0 y 0 to "forever" and change "x 0 y 0" to "x 3 y 4"; and copy and place the block "pause(ms)500" to block "forever";

Complete Program

	In“on start”the program only runs once;Open the LED dot matrix on the micro:bit board;In“forever”the program runs cyclically;Change the brightness of the LED positioned in (1,0)Delay in 500ms;Change the brightness of the LED positioned in (1,0)Delay in 500ms;Change the brightness of the LED positioned in (3,4)Delay in 500ms;Turn off the LED in(3,4);Delay in 500ms;

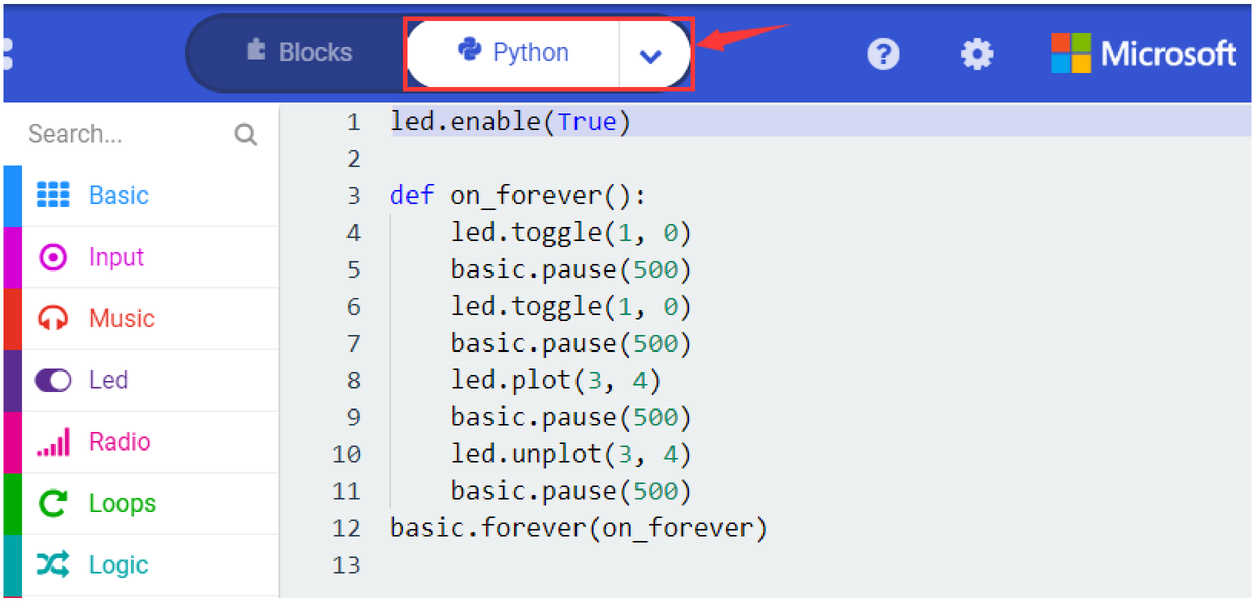
```
on start
    led enable true

forever
    toggle x 1 y 0
    pause (ms) 500
    toggle x 1 y 0
    pause (ms) 500
    plot x 3 y 4
    pause (ms) 500
    unplot x 3 y 4
    pause (ms) 500
```

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



```
led.enable(true)
basic.forever(function () {
    led.toggle(1, 0)
    basic.pause(500)
    led.toggle(1, 0)
    basic.pause(500)
    led.plot(3, 4)
    basic.pause(500)
    led.unplot(3, 4)
    basic.pause(500)
})
```

Click ⌄ to choose Python, you will find the corresponding Python programming languages.

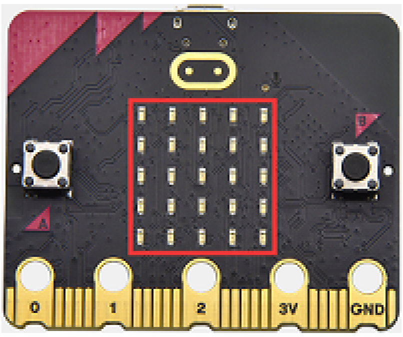


```
led.enable(True)

def on_forever():
    led.toggle(1, 0)
    basic.pause(500)
    led.toggle(1, 0)
    basic.pause(500)
    led.plot(3, 4)
    basic.pause(500)
    led.unplot(3, 4)
    basic.pause(500)
basic.forever(on_forever)

```

4. Test Results

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

7.3 Project 3: LED Dot Matrix



1. Project Description:

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

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

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

2. Components Needed:

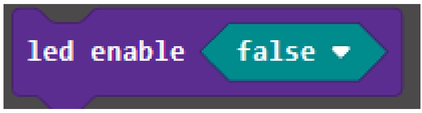
- Micro:bit main board V2 *1
- Micro USB cable*1

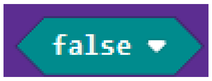
3. Test Code 1:

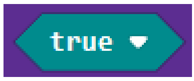
Link computer with micro:bit board by micro USB cable, and program in MakeCode editor.

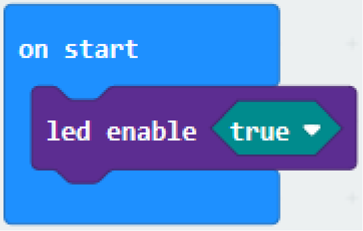
Enter
Led

→
••• more

→ 
```
led enable false ▾
```

; Click 
```
false ▾
```

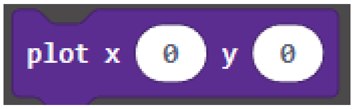
to select 
```
true ▾
```

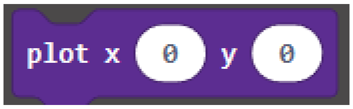
: 
```
on start
    led enable true ▾
```

Combine it with "on start" block

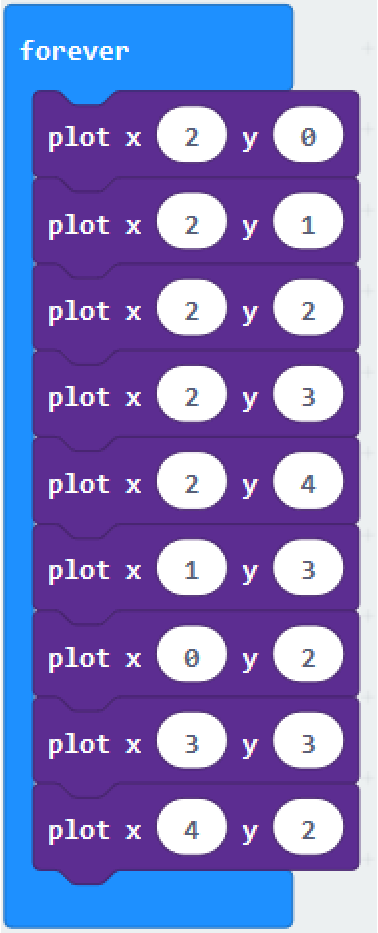
****

Click
Led

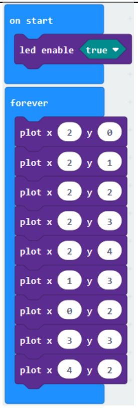
to move 
```
plot x 0 y 0
```

into"forever", then replicate 
```
plot x 0 y 0
```

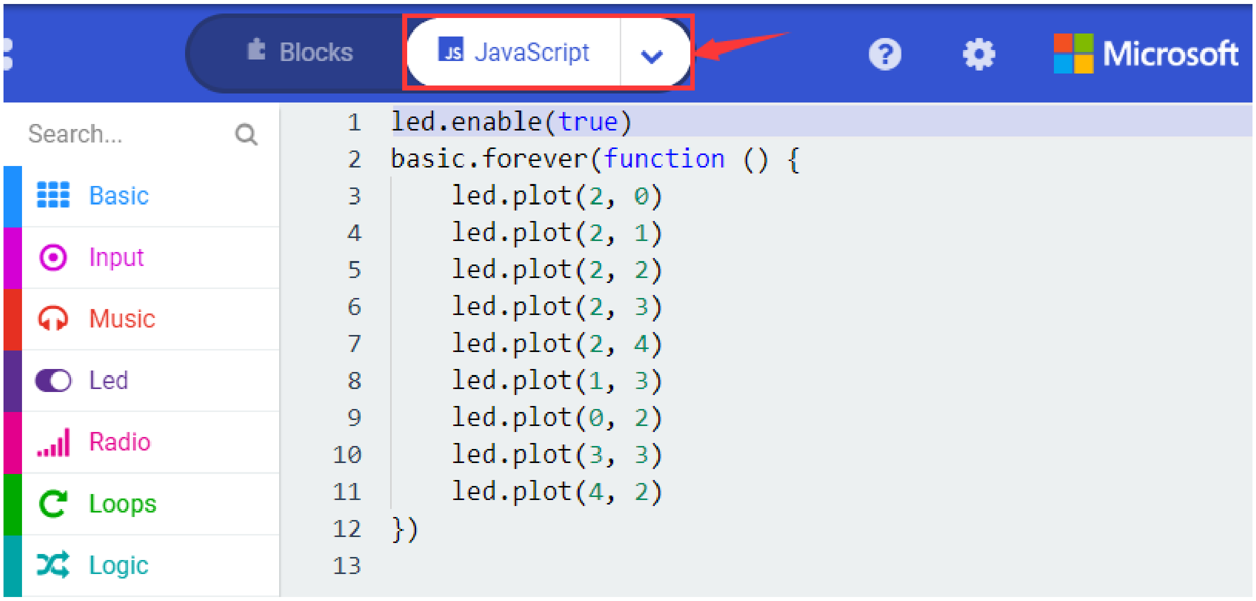
for 8 times, respectively set to"x 2"y 0", "x 2"y 1", "x 2"y 2", "x 2"y 3", "x2"y 4", "x 1"y 3", "x 0"y 2", "x 3"y 3", "x 4"y 2".


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

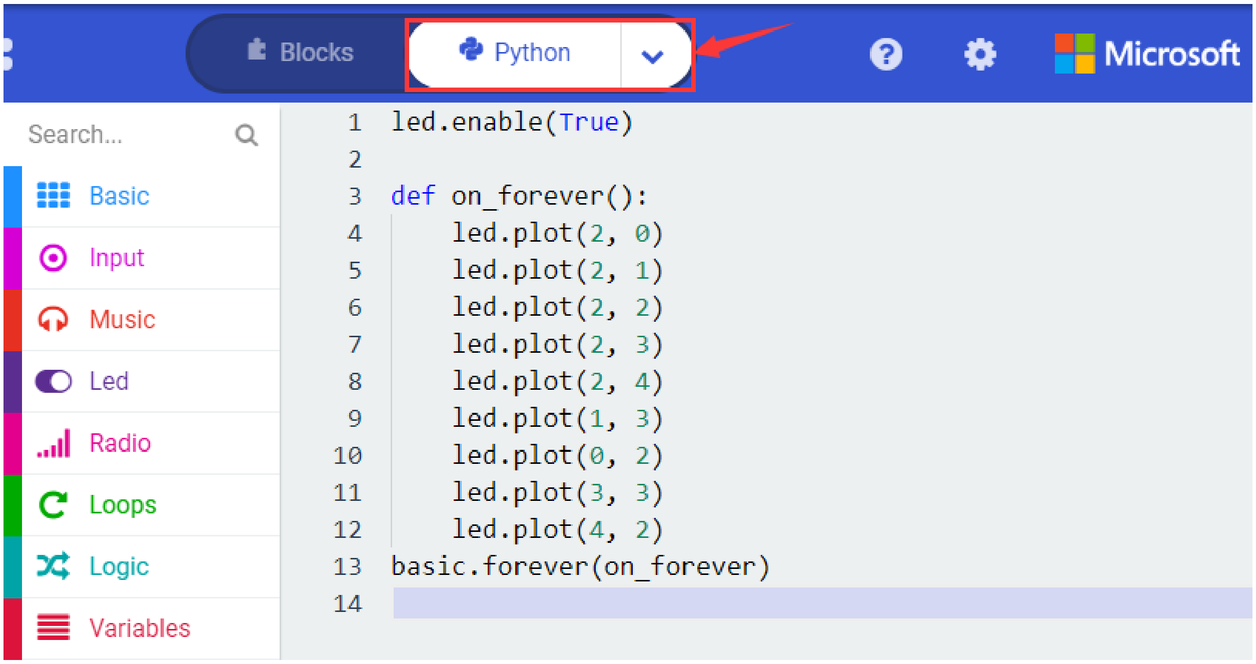
Complete Program

 on start led enable true forever plot x 2 y 0 plot x 2 y 1 plot x 2 y 2 plot x 2 y 3 plot x 2 y 4 plot x 1 y 3 plot x 0 y 2 plot x 3 y 3 plot x 4 y 2	"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 2y 0", "x 2y 1", "x 2y 2", "x 2y 3", "x 2y 4", "x 1y 3", "x 0y 2", "x 3y 3", "x 4y 2".

Select and to switch into JavaScript and Python code:

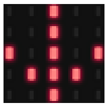


```
led.enable(true)
basic.forever(function () {
    led.plot(2, 0)
    led.plot(2, 1)
    led.plot(2, 2)
    led.plot(2, 3)
    led.plot(2, 4)
    led.plot(1, 3)
    led.plot(0, 2)
    led.plot(3, 3)
    led.plot(4, 2)
})

```



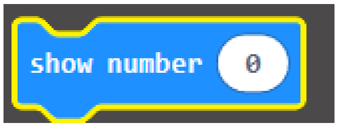
```
led.enable(True)

def on_forever():
    led.plot(2, 0)
    led.plot(2, 1)
    led.plot(2, 2)
    led.plot(2, 3)
    led.plot(2, 4)
    led.plot(1, 3)
    led.plot(0, 2)
    led.plot(3, 3)
    led.plot(4, 2)
basic.forever(on_forever)
```

4. Test Results 1:

Upload code 1 and power on , we will see the  icon.

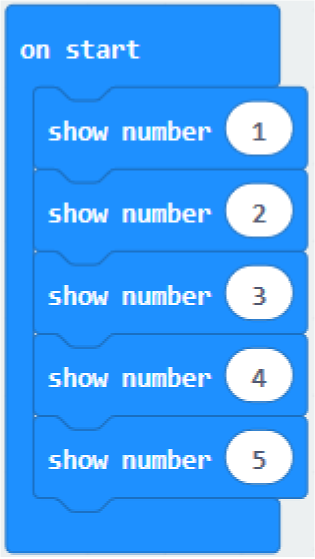
5. Test Code 2:

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

Enter Basic → 

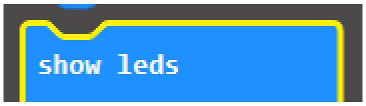
```
show number 0
```

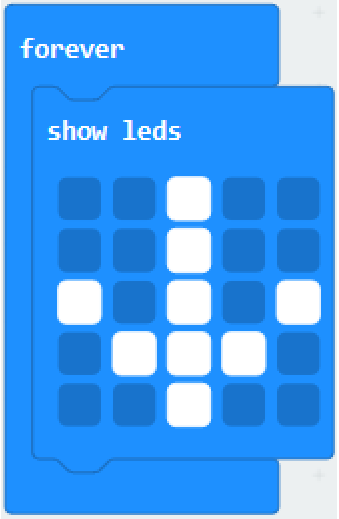
→ Duplicate it for 4 times, separately set number to 1, 2, 3, 4, 5.



```
on start
    show number 1
    show number 2
    show number 3
    show number 4
    show number 5
```

****

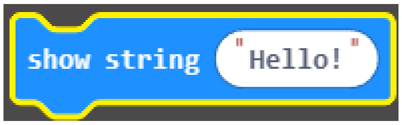
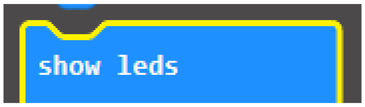
Click Basic →  show leds → put it into "forever" block, tick blue boxes to light LED and generate "↓" pattern.

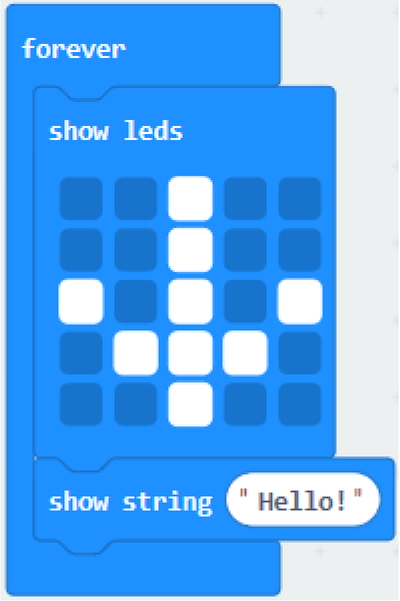


forever

show leds

****

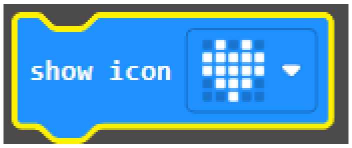
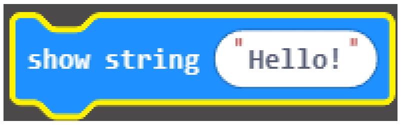
Move out the block  show string "Hello!" from Basic block, and leave it beneath the  show leds block.

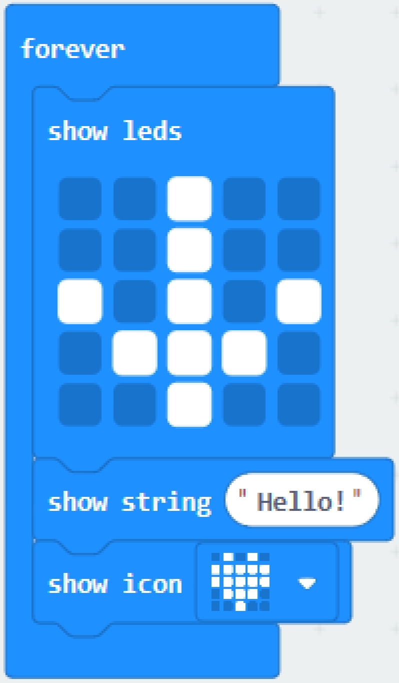


forever

show leds

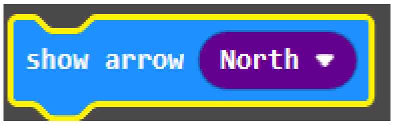
show string "Hello!"

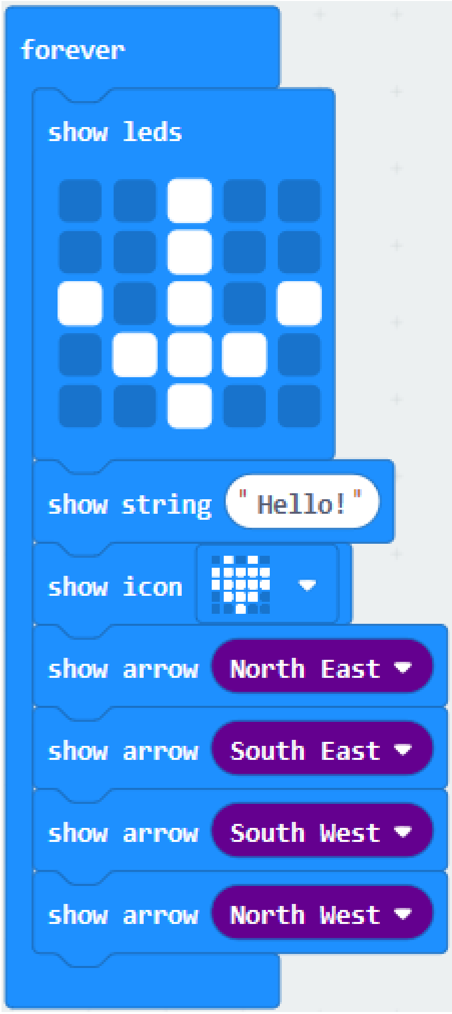
Choose  show icon from ▦ Basic, and leave it beneath  show string "Hello!"



```
forever
  show leds
    . . # . .
    . . # . .
    # . # . #
    . # # # .
    . . # . .
  show string "Hello!"
  show icon
```

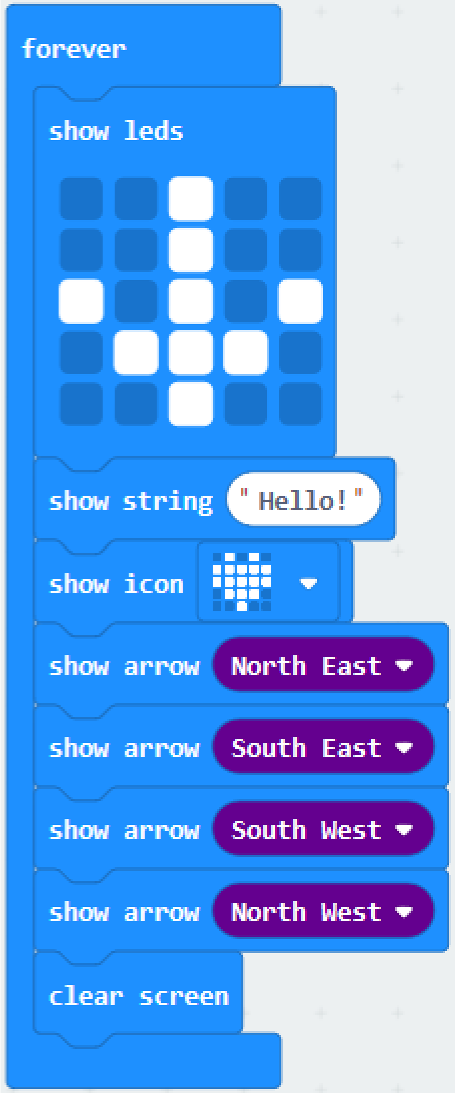
****

Enter ▦ Basic →  show arrow North → replicate for 3 times, respectively set to "North East", "South East", "South West" "North West".



```
forever
  show leds
    . . # . .
    . . # . .
    # . # . #
    . # # # .
    . . # . .
  show string "Hello!"
  show icon ▾
  show arrow North East ▾
  show arrow South East ▾
  show arrow South West ▾
  show arrow North West ▾
```

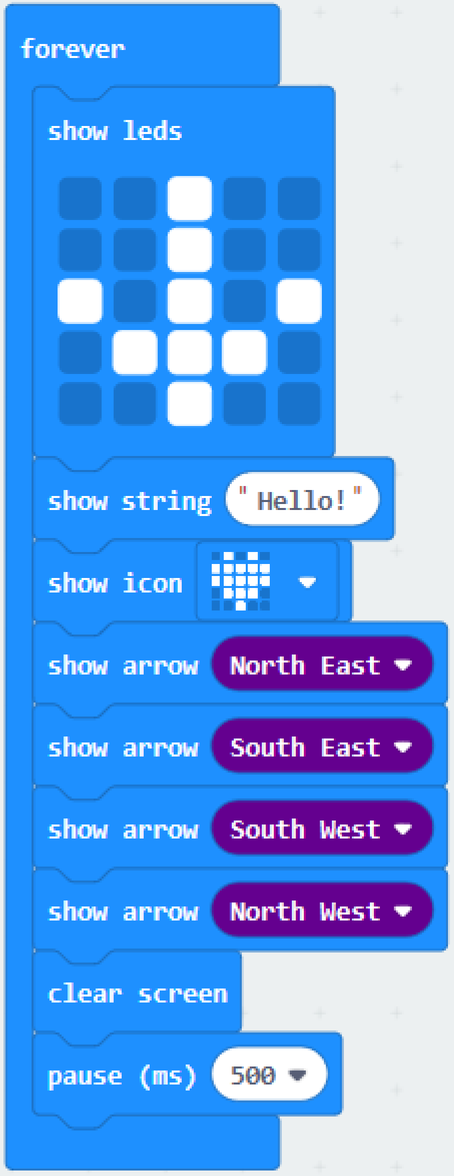
Click Basic to get `clear screen` then remain it below `show arrow North West ▾`



```
forever
    show leds
    show string "Hello!"
    show icon
    show arrow North East
    show arrow South East
    show arrow South West
    show arrow North West
    clear screen
```

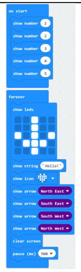
****

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

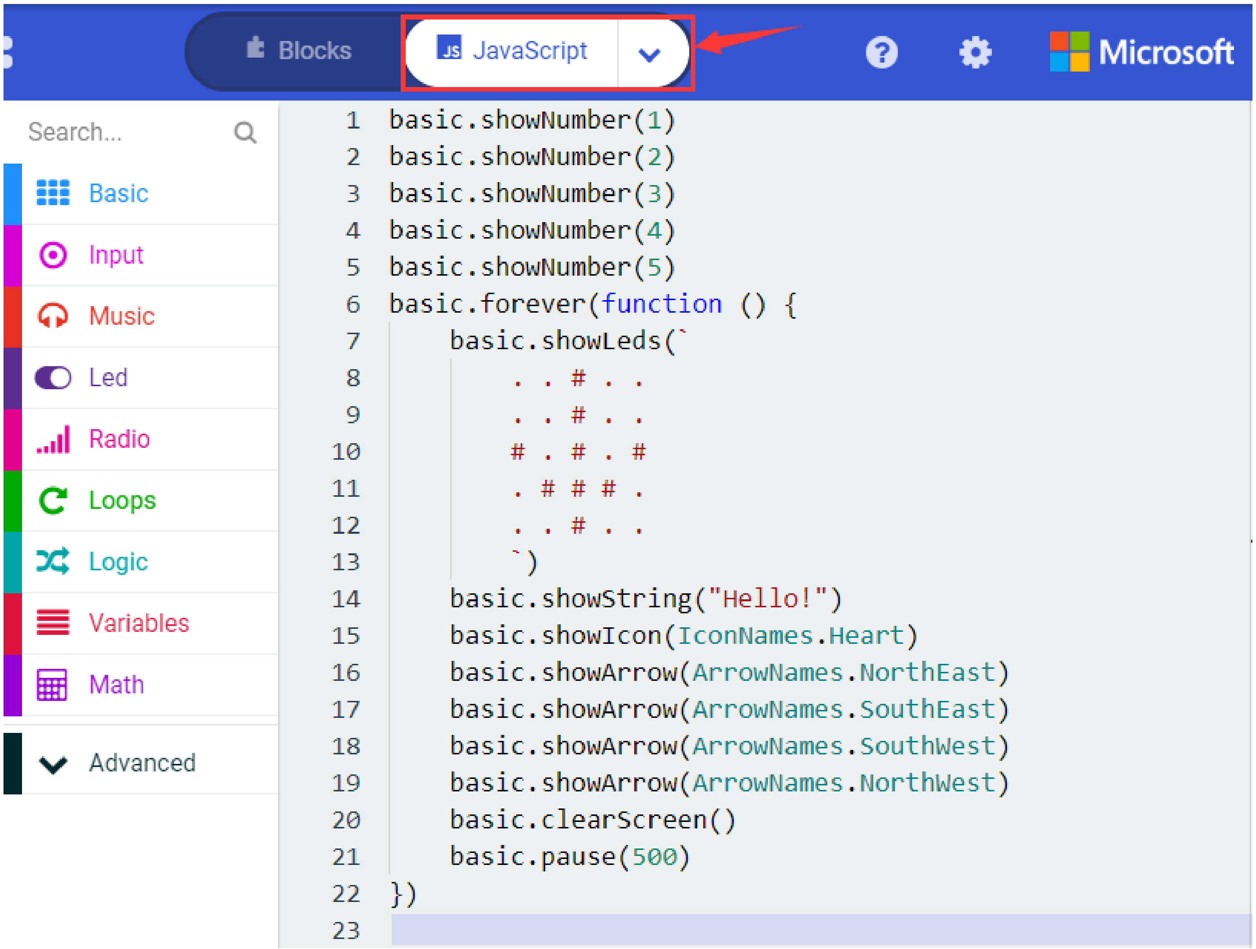


```
forever
  show leds
  show string "Hello!"
  show icon
  show arrow North East
  show arrow South East
  show arrow South West
  show arrow North West
  clear screen
  pause (ms) 500
```

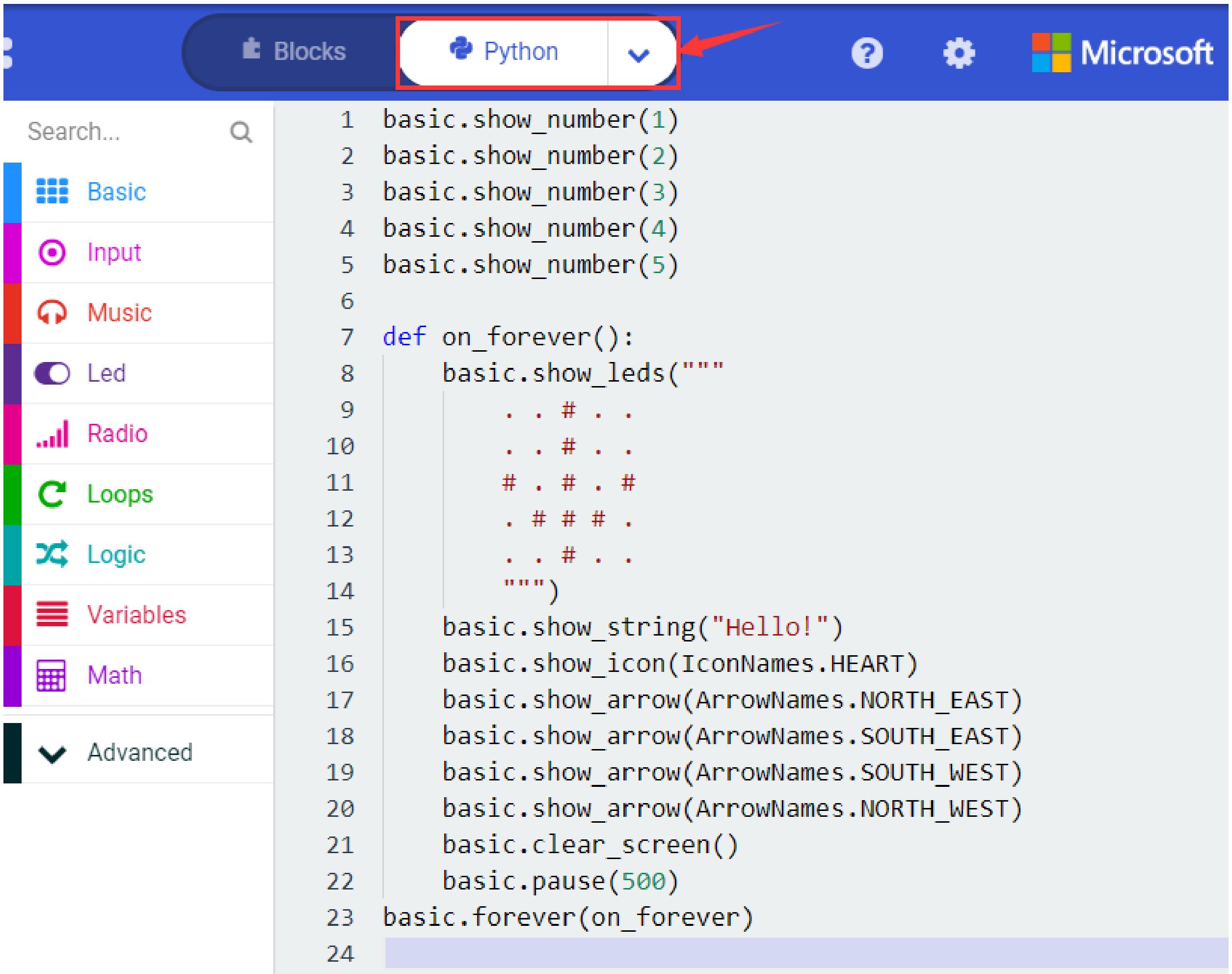
Complete Program:

 on start show number 1 show number 2 show number 3 show number 4 show number 5 forever show leds show string "Hello!" show icon show arrow North East show arrow South East show arrow South West show arrow North West clear screen pause (ms) 500	“on start”: command block only runs once to start program.LED dot matrix displays 1,2,3,4,5Under the block “forever”program runs cyclically.Dot matrix shows the“↓” patternDot matrix scrolls to show “Hello!”“”is shown on dot matrixLED dot matrix displays “North East” arrow.The “South East” arrow shows up on LED dot matrixThe “South West” arrow appears up on LED dot matrix-The“North West”arrow is displayed on LED dot matrixClear the screenDelay in 500ms

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

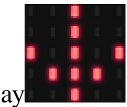
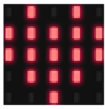
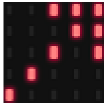
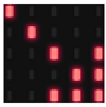
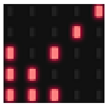
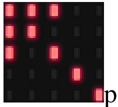


```
basic.showNumber(1)
basic.showNumber(2)
basic.showNumber(3)
basic.showNumber(4)
basic.showNumber(5)
basic.forever(function () {
    basic.showLeds(`
        . . # . .
        . . # . .
        # . # . #
        . # # # .
        . . # . .
        `)
    basic.showString("Hello!")
    basic.showIcon(IconNames.Heart)
    basic.showArrow(ArrowNames.NorthEast)
    basic.showArrow(ArrowNames.SouthEast)
    basic.showArrow(ArrowNames.SouthWest)
    basic.showArrow(ArrowNames.NorthWest)
    basic.clearScreen()
    basic.pause(500)
})
```

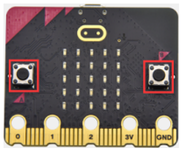


```
basic.show_number(1)
basic.show_number(2)
basic.show_number(3)
basic.show_number(4)
basic.show_number(5)

def on_forever():
    basic.show_leds("""
        . . # . .
        . . # . .
        # . # . #
        . # # # .
        . . # . .
        """)
    basic.show_string("Hello!")
    basic.show_icon(IconNames.HEART)
    basic.show_arrow(ArrowNames.NORTH_EAST)
    basic.show_arrow(ArrowNames.SOUTH_EAST)
    basic.show_arrow(ArrowNames.SOUTH_WEST)
    basic.show_arrow(ArrowNames.NORTH_WEST)
    basic.clear_screen()
    basic.pause(500)
basic.forever(on_forever)

```

6. Test Results 2:

Upload code 2 and plug micro:bit to power. Micro: bit starts showing number 1, 2, 3, 4, and 5, then cyclically display,"Hello!", , , ,  and patterns.

7.4 Project 4: Programmable Buttons



1. Project Description:

Buttons can be used to control circuits. In an integrated circuit with a button, the circuit is connected when pressing the button and it is open the other way around.

Micro: Bit main board V2 boasts three buttons, two are programmable buttons(marked with A and B), and the one on the other side is a reset button. By pressing the two programmable buttons can input three different signals. We can press button A or B alone or press them together and the LED dot matrix shows A,B and AB respectively. Let's get started.

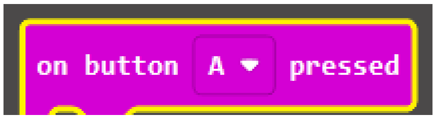
2. Components Needed:

- 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,

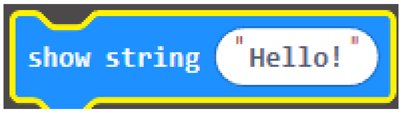
Delete "on start" and "forever" firstlythen click ● Input →



on button A ▾ pressed

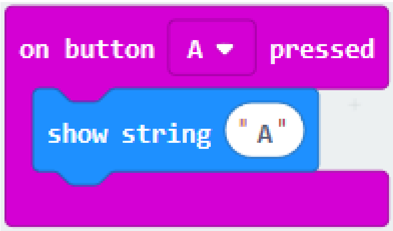
****

A. Click Basic →



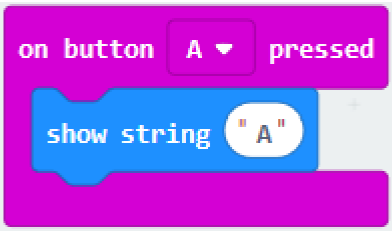
show string "Hello!"

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

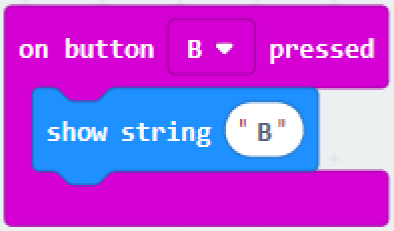


on button A ▾ pressed
show string "A"

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



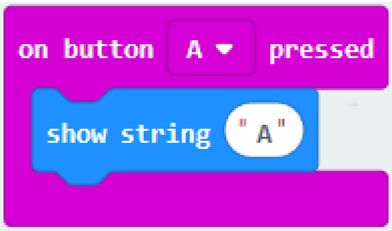
on button A ▾ pressed
show string "A"



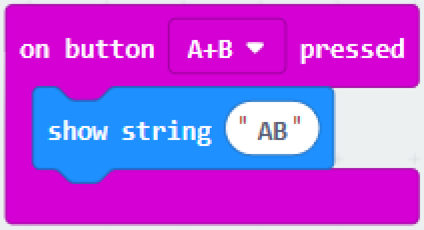
on button B ▾ pressed
show string "B"

****

1. Copy onceand set to"on button A+B pressed"and"show string "AB"



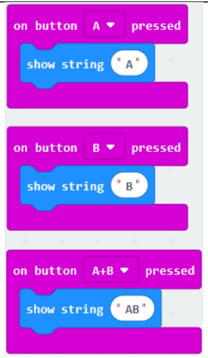
on button A ▾ pressed
show string "A"



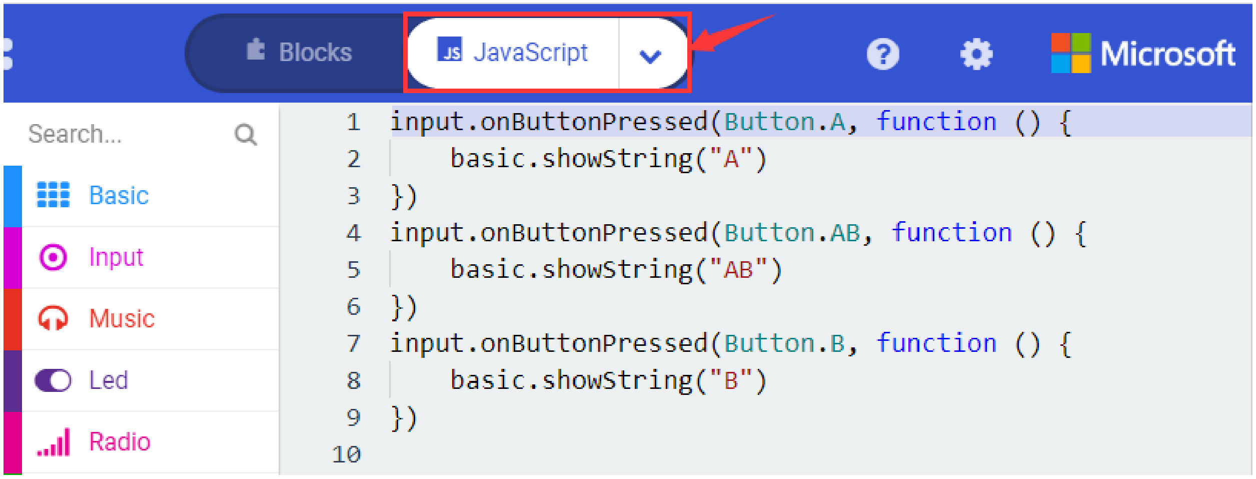
```
on button A+B ▾ pressed
    show string "AB"
```

****

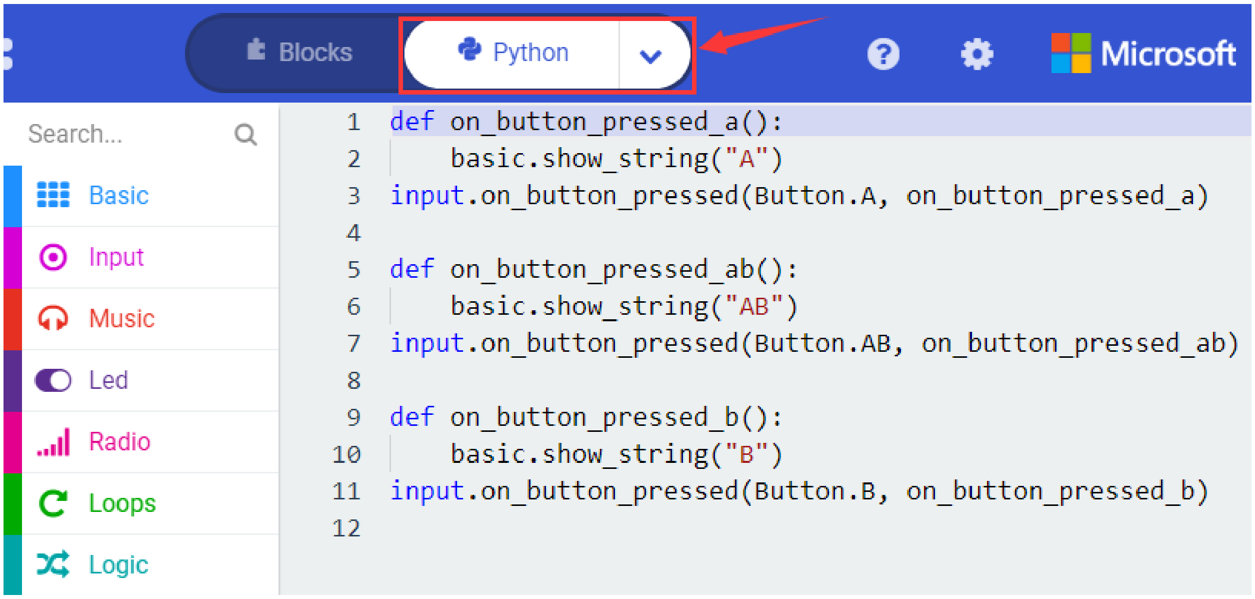
Complete Code:

 on button A ▾ pressed show string "A" on button B ▾ pressed show string "B" on button A+B ▾ pressed show string "AB"	Press button A on Micro: bit main boardShow the character “A”Press button B on Micro: bit main boardShow the character “B”Press button A and B at same time Display the character “AB”

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



```
input.onButtonPressed(Button.A, function () {
    basic.showString("A")
})
input.onButtonPressed(Button.AB, function () {
    basic.showString("AB")
})
input.onButtonPressed(Button.B, function () {
    basic.showString("B")
})

```



```
def on_button_pressed_a():
    basic.show_string("A")
input.on_button_pressed(Button.A, on_button_pressed_a)

def on_button_pressed_ab():
    basic.show_string("AB")
input.on_button_pressed(Button.AB, on_button_pressed_ab)

def on_button_pressed_b():
    basic.show_string("B")
input.on_button_pressed(Button.B, on_button_pressed_b)
```

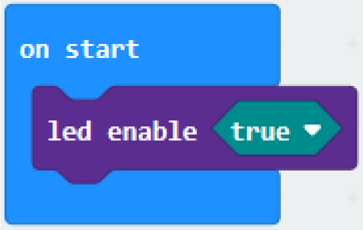
4. Test Results 1:

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

5. Test Code 2:

A. Click “Led” → “more” → “led enable false”,

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



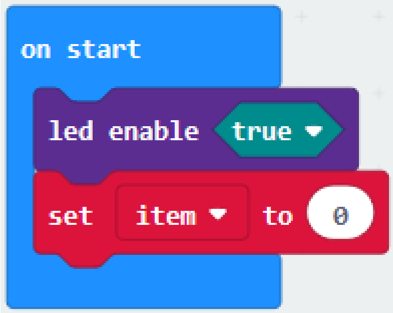
```
on start
    led enable true
```

.

****

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



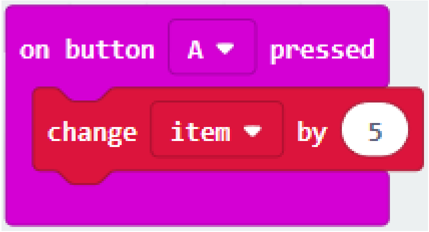
```
on start
    led enable true
    set item to 0
```

****

A. Click “Input” → “on button A pressed”.

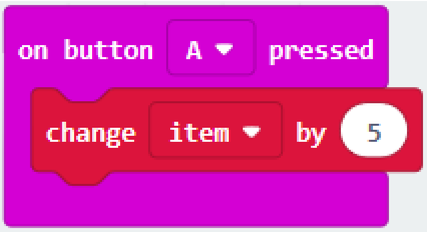
B. Go to “Variables”→“ change item by 1 ”

C. Place it into“on button A pressed”and 1 is modified into 5.



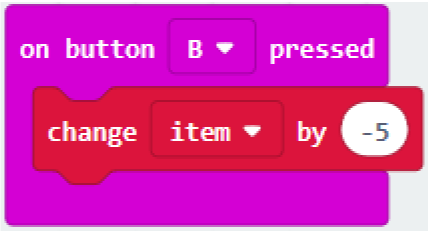
```
on button A ▾ pressed
    change item ▾ by 5
```

****



```
on button A ▾ pressed
    change item ▾ by 5
```

Duplicate code string onceclick the drop-down button to select“B”then set“change item by -5”.



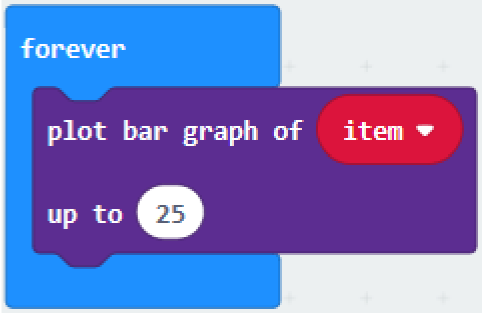
```
on button B ▾ pressed
    change item ▾ by -5
```

****

A. Enter“Led”→“plot bar graph of 0 up to 0”

B. Keep it into“forever”block

C. Go to“Variables”to move“item”into 0 boxchange 0 into 25.



```
forever
    plot bar graph of item ▾
    up to 25
```

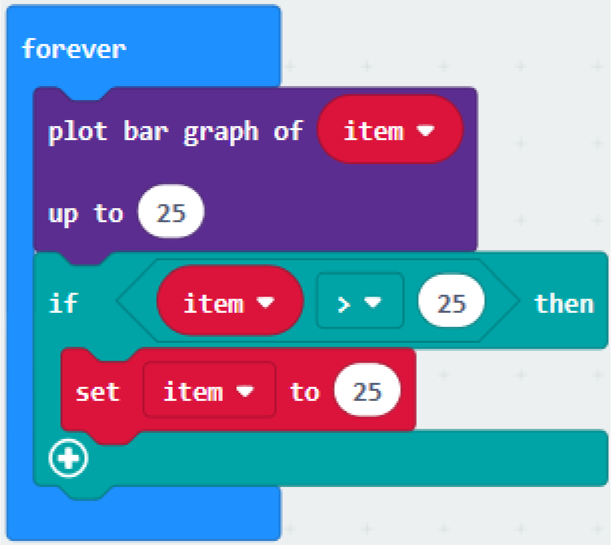
****

(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

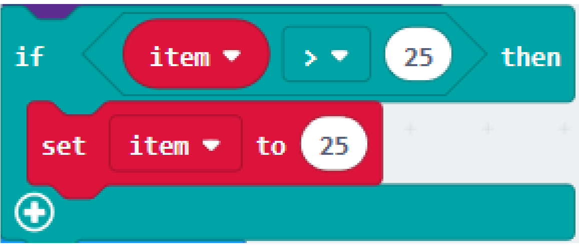
D. Enter“Variables”to drag“set item to 0”block into“if…true..then…”, alter 0 into 25.



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

****

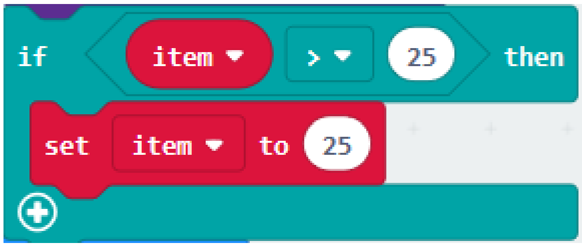
(7) A. Replicate code string once



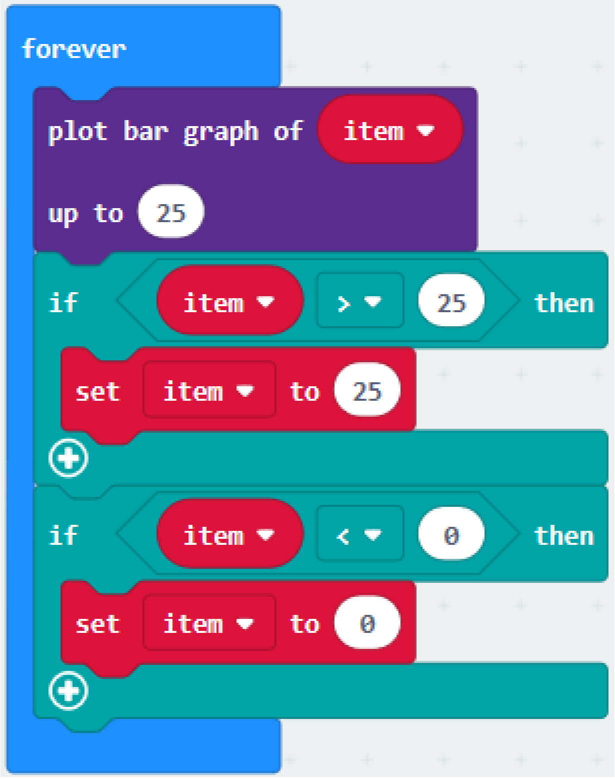
```
if item > 25 then
    set item to 25
```

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

C. Leave it beneath code string.

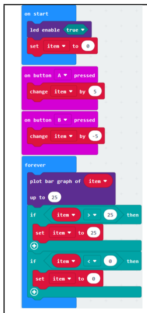


```
if item > 25 then
    set item to 25
```

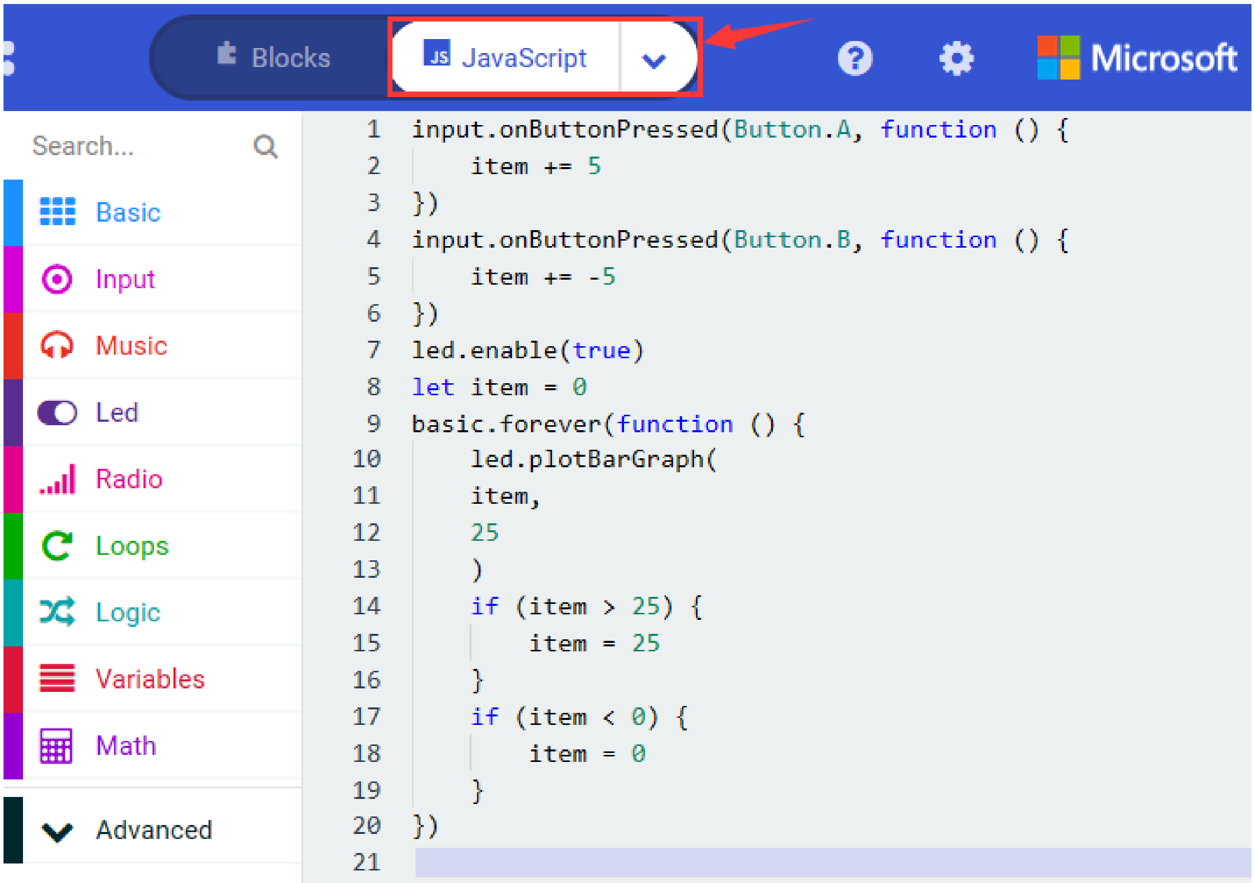


```
forever
    plot bar graph of item ▾
    up to 25
    if item ▾ > ▾ 25 then
        set item ▾ to 25
    ⊕
    if item ▾ < ▾ 0 then
        set item ▾ to 0
    ⊕
```

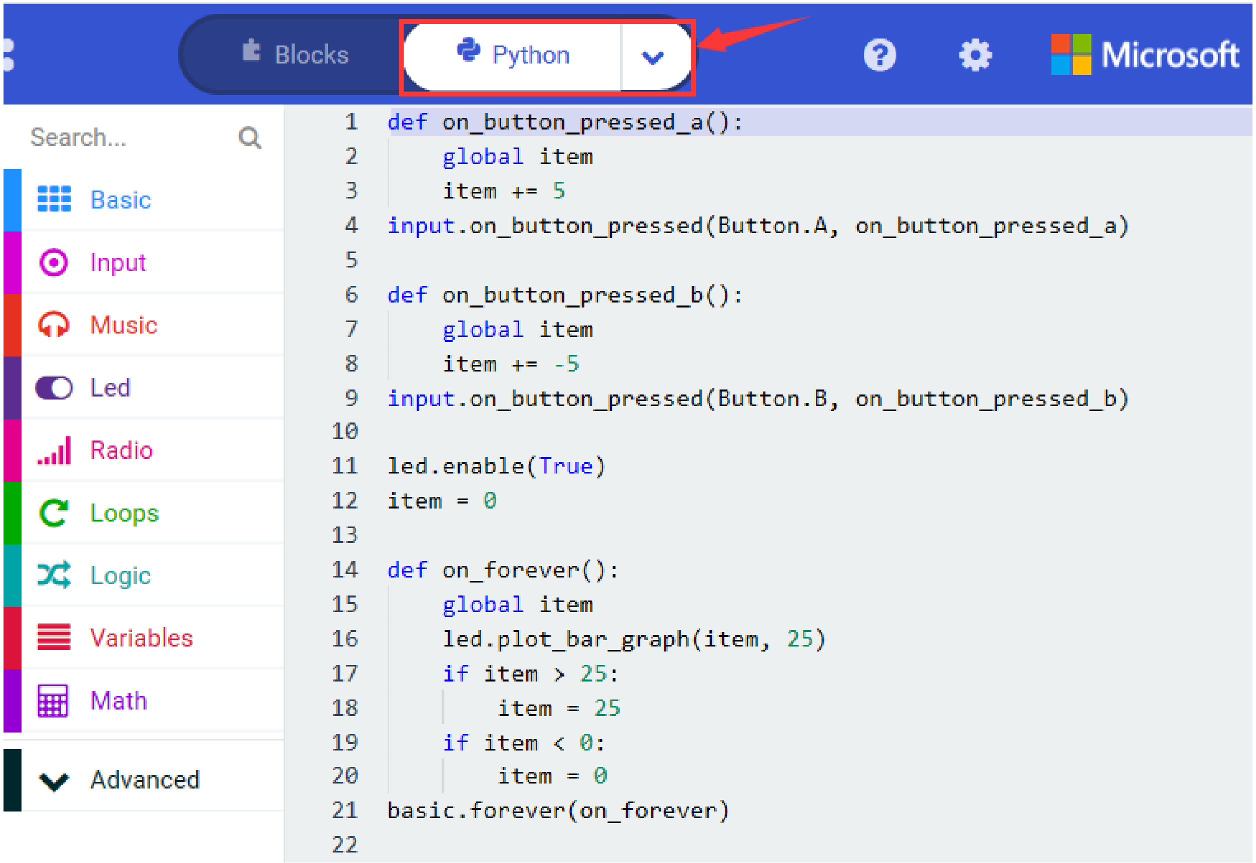
Complete Program

 on start led enable true ▾ set item ▾ to 0 on button A ▾ pressed change item ▾ by 5 on button B ▾ pressed change item ▾ by -5 forever plot bar graph of item ▾ up to 25 if item ▾ > ▾ 25 then set item ▾ to 25 ⊕ if item ▾ < ▾ 0 then set item ▾ to 0 ⊕	"on start": command block runs once to start program.Turn on LED dot matrixSet the initial value of item to 0Press button A on Micro:bit boardChange item by 5Press button B on Micro:bit boardChange item by -5The program under the block "forever" runs cyclically.Light on LED in dot matrix to draw bar graph, light up up to 25 LEDsIf item is greater than 25Then set item to 25If item is less than 0Then Set item to 0

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



```
input.onButtonPressed(Button.A, function () {
    item += 5
})
input.onButtonPressed(Button.B, function () {
    item += -5
})
led.enable(true)
let item = 0
basic.forever(function () {
    led.plotBarGraph(
    item,
    25
    )
    if (item > 25) {
        item = 25
    }
    if (item < 0) {
        item = 0
    }
})
```

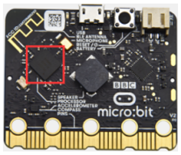


```
def on_button_pressed_a():
    global item
    item += 5
input.on_button_pressed(Button.A, on_button_pressed_a)

def on_button_pressed_b():
    global item
    item += -5
input.on_button_pressed(Button.B, on_button_pressed_b)

led.enable(True)
item = 0

def on_forever():
    global item
    led.plot_bar_graph(item, 25)
    if item > 25:
        item = 25
    if item < 0:
        item = 0
basic.forever(on_forever)

```

6. Test Results 2:

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

7.5 Project 5: Temperature Detection



1. Project Description:

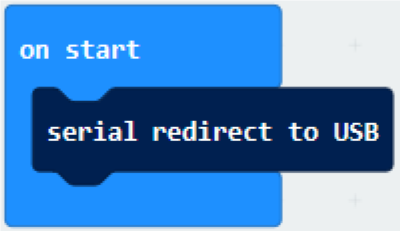
Actually ,the Micro:bit main board V2 is not equipped with a temperature sensor, but uses the temperature sensor built into NFR52833 chip for temperature detection. Therefore, the detected temperature is more closer to the temperature of the chip, and there maybe deviation from the ambient temperature.

2. Components Needed:

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

3. Test Code 1:

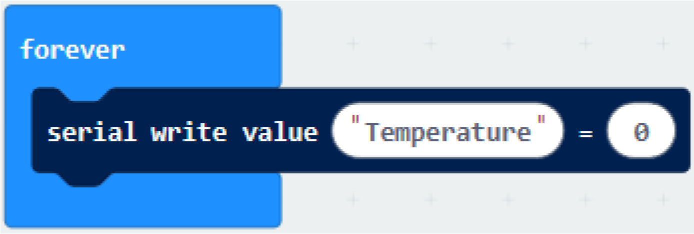
Click “Advanced” → ”Serial” → “serial redirect to USB” into “on start”.



```
on start
  serial redirect to USB
```

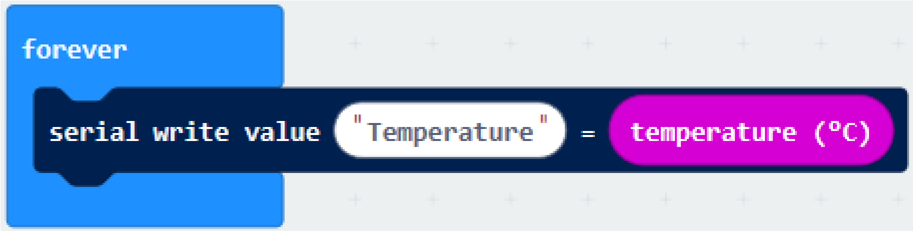
****

Go to “Serial” → “serial write value “x”=0” into “forever”



```
forever
  serial write value "Temperature" = 0
```

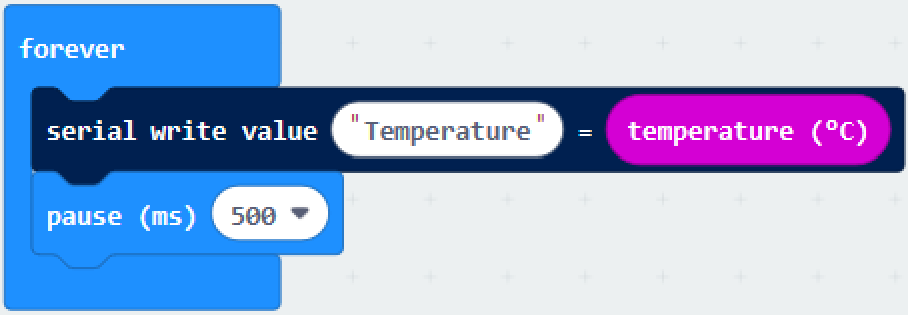
Click “Input” → “temperature(°C)” into “serial write value“x”=0 and change”0”into “temperature”



```
forever
  serial write value "Temperature" = temperature (°C)
```

****

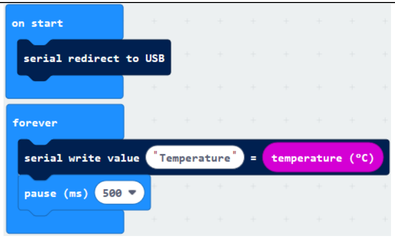
Go to“Basic”→“pause (ms) 100”into “forever”and set pause to 500



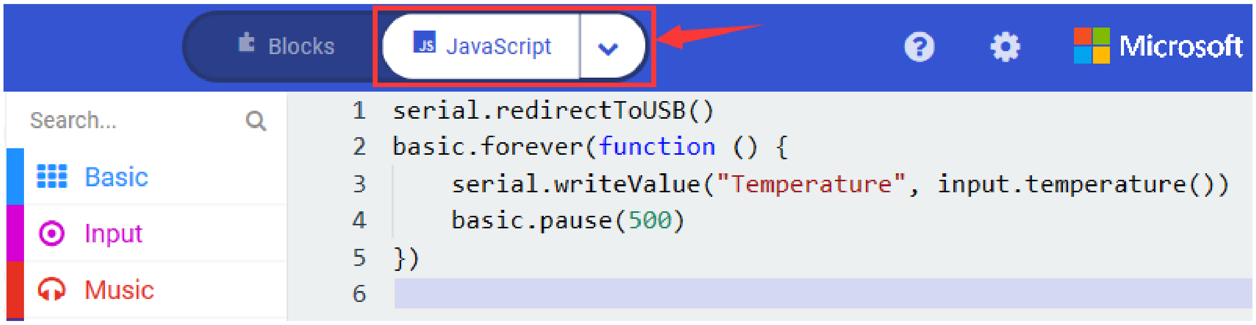
```
forever
  serial write value "Temperature" = temperature (°C)
  pause (ms) 500
```

****

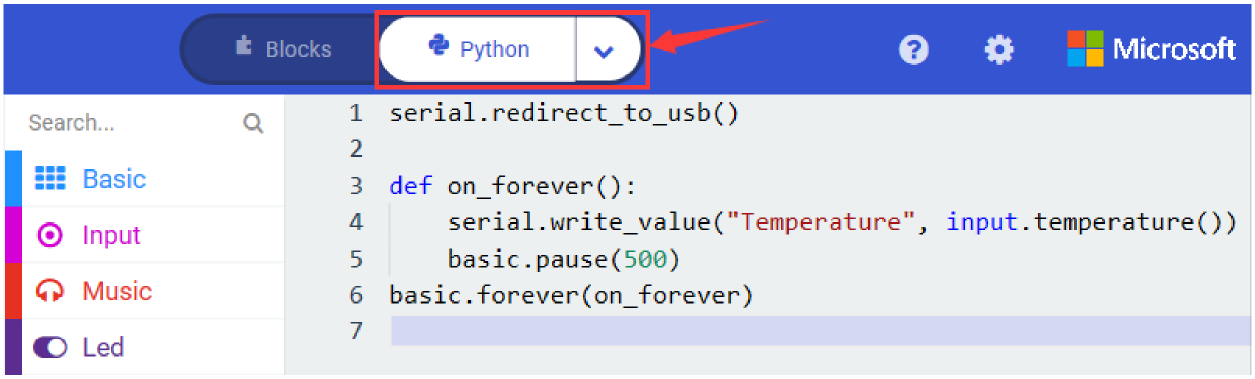
Complete Program

 on start serial redirect to USB forever serial write value "Temperature" = temperature (°C) pause (ms) 500	In“on start”the program only runs once;Redirect serial to USB;In“forever”the program runs cyclically;The serial writes the temperature value detected by the sensor;Delay in 50ms.

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



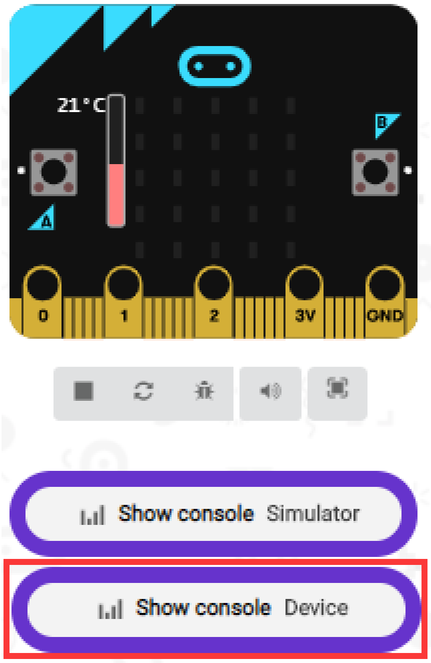
```
serial.redirectToUSB()
basic.forever(function () {
    serial.writeValue("Temperature", input.temperature())
    basic.pause(500)
})
```

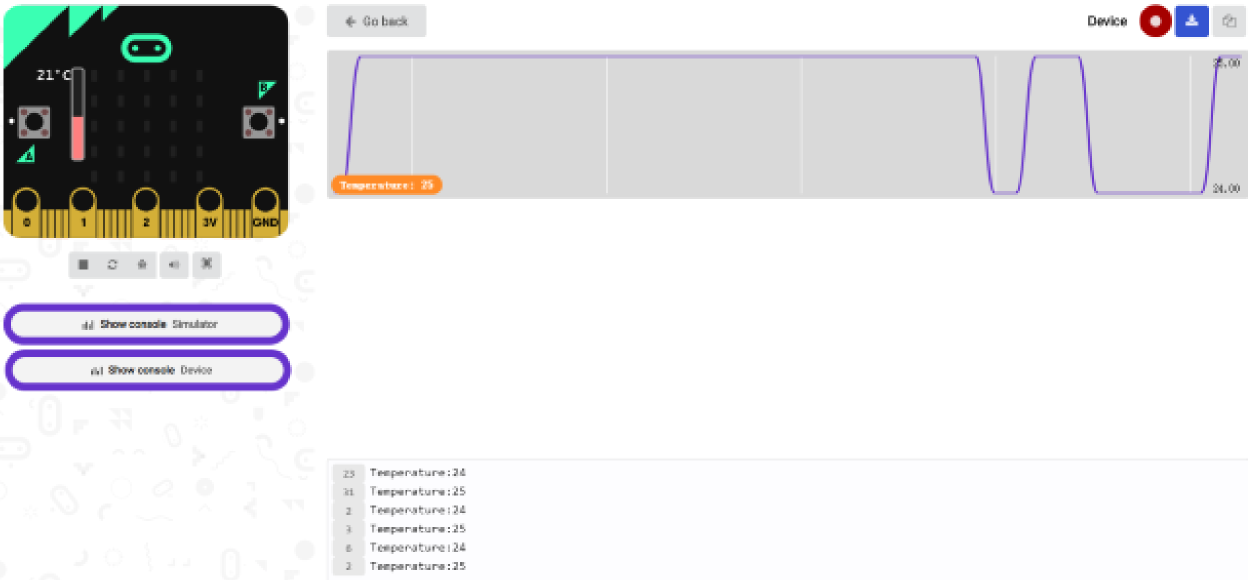


```
serial.redirect_to_usb()

def on_forever():
    serial.write_value("Temperature", input.temperature())
    basic.pause(500)
basic.forever(on_forever)
```

4. Test Results 1:

Uploading test code 1 to micro:bit main board V2, powering the main board via the USB cable, and clicking “Show console Device”, the data of temperature shows in the serial monitor page as shown below.

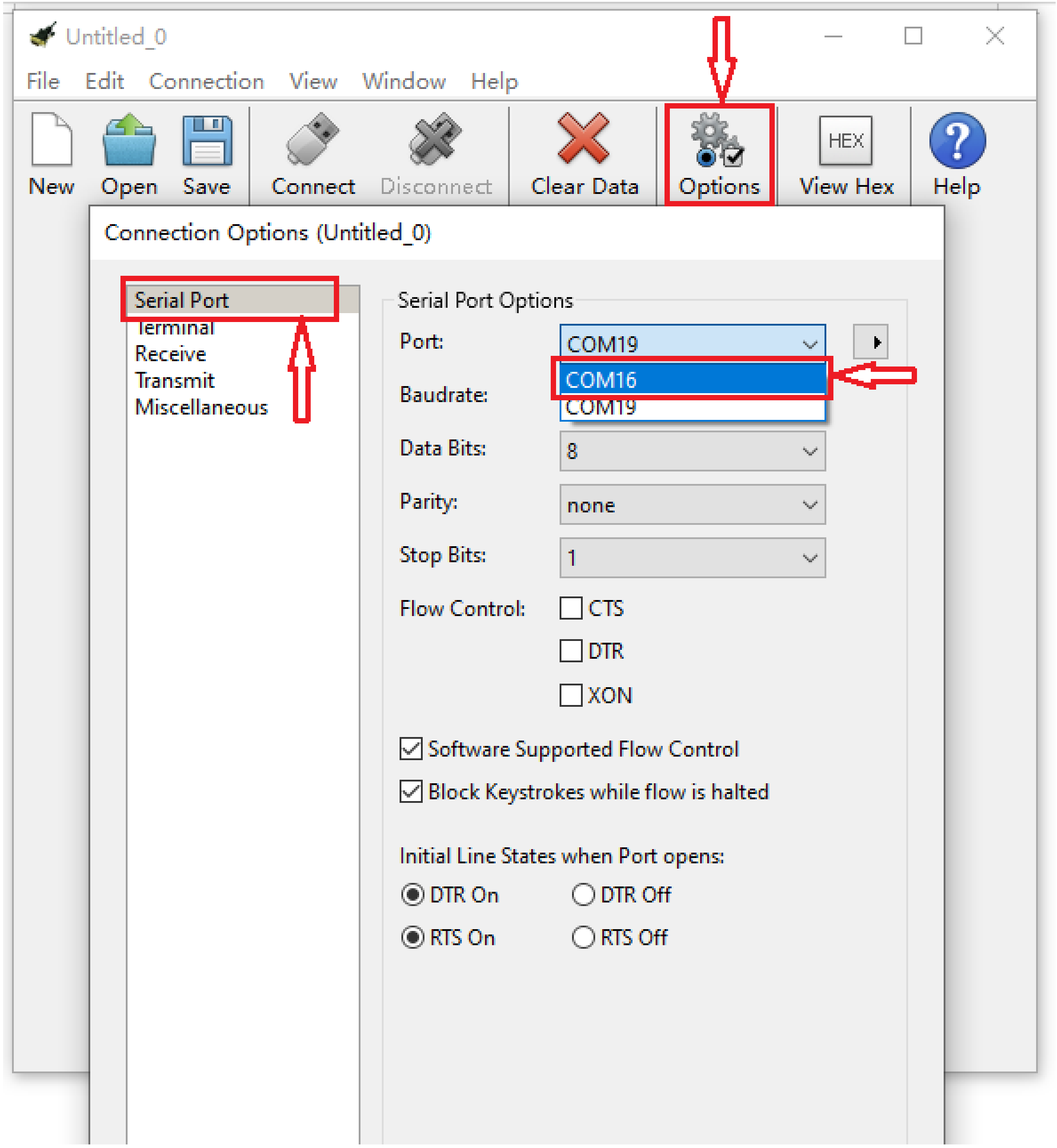




```
23  Temperature:24
31  Temperature:25
2   Temperature:24
3   Temperature:25
6   Temperature:24
2   Temperature:25
```

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 change of temperature in the current environment, as shown in the figures below :



Connection Options (Untitled_0)

Serial Port
Terminal
Receive
Transmit
Miscellaneous

Serial Port Options

Port: COM19
COM16
COM19

Baudrate:

Data Bits: 8

Parity: none

Stop Bits: 1

Flow Control: □ CTS
□ DTR
□ XON

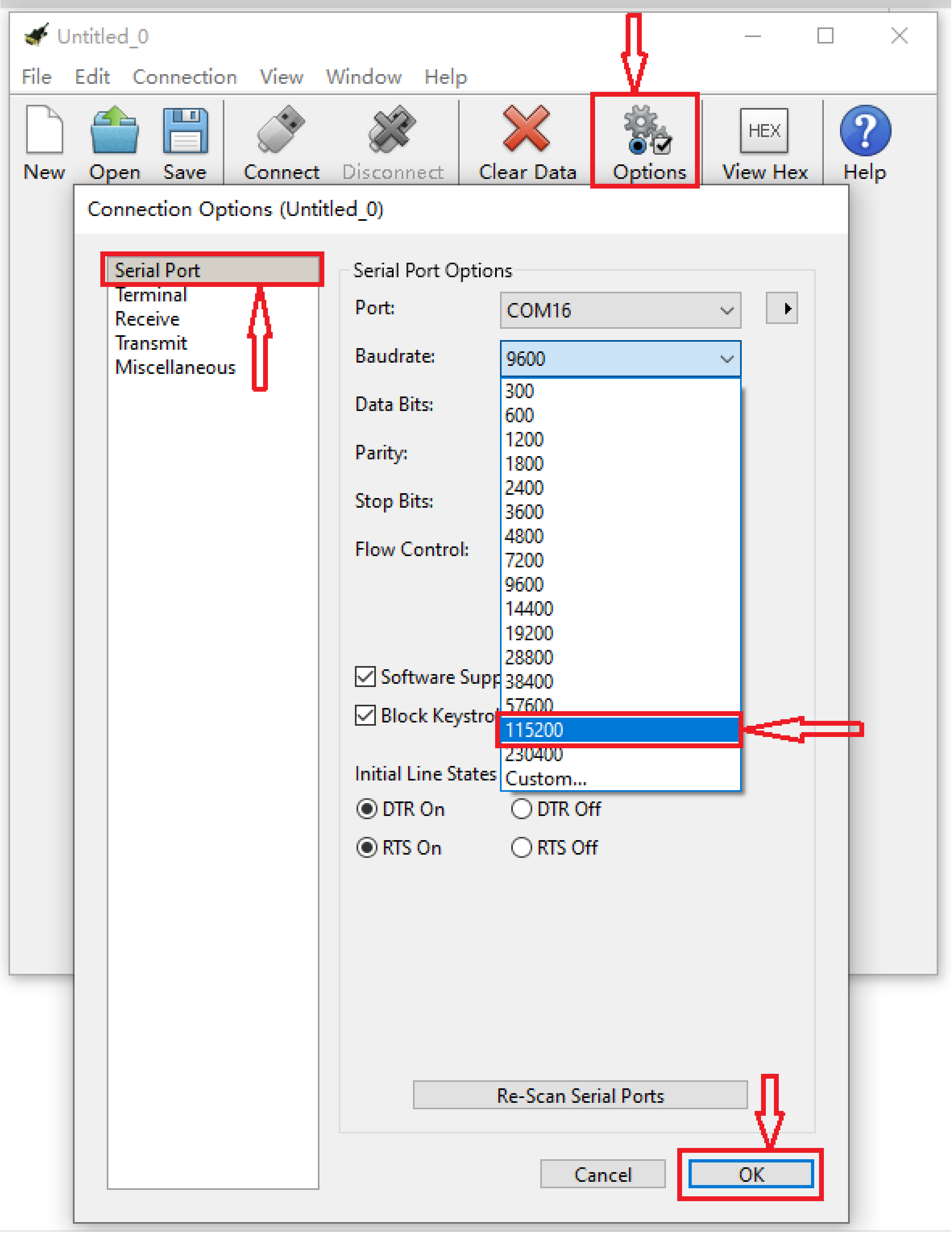
☑ Software Supported Flow Control

☑ Block Keystrokes while flow is halted

Initial Line States when Port opens:

● DTR On ○ DTR Off

● RTS On ○ RTS Off



Connection Options (Untitled_0)

Serial Port Options

Port: COM16

Baudrate: 9600

Data Bits:

Parity:

Stop Bits:

Flow Control:

300
600
1200
1800
2400
3600
4800
7200
9600
14400
19200
28800
38400
57600
115200
230400
Custom...

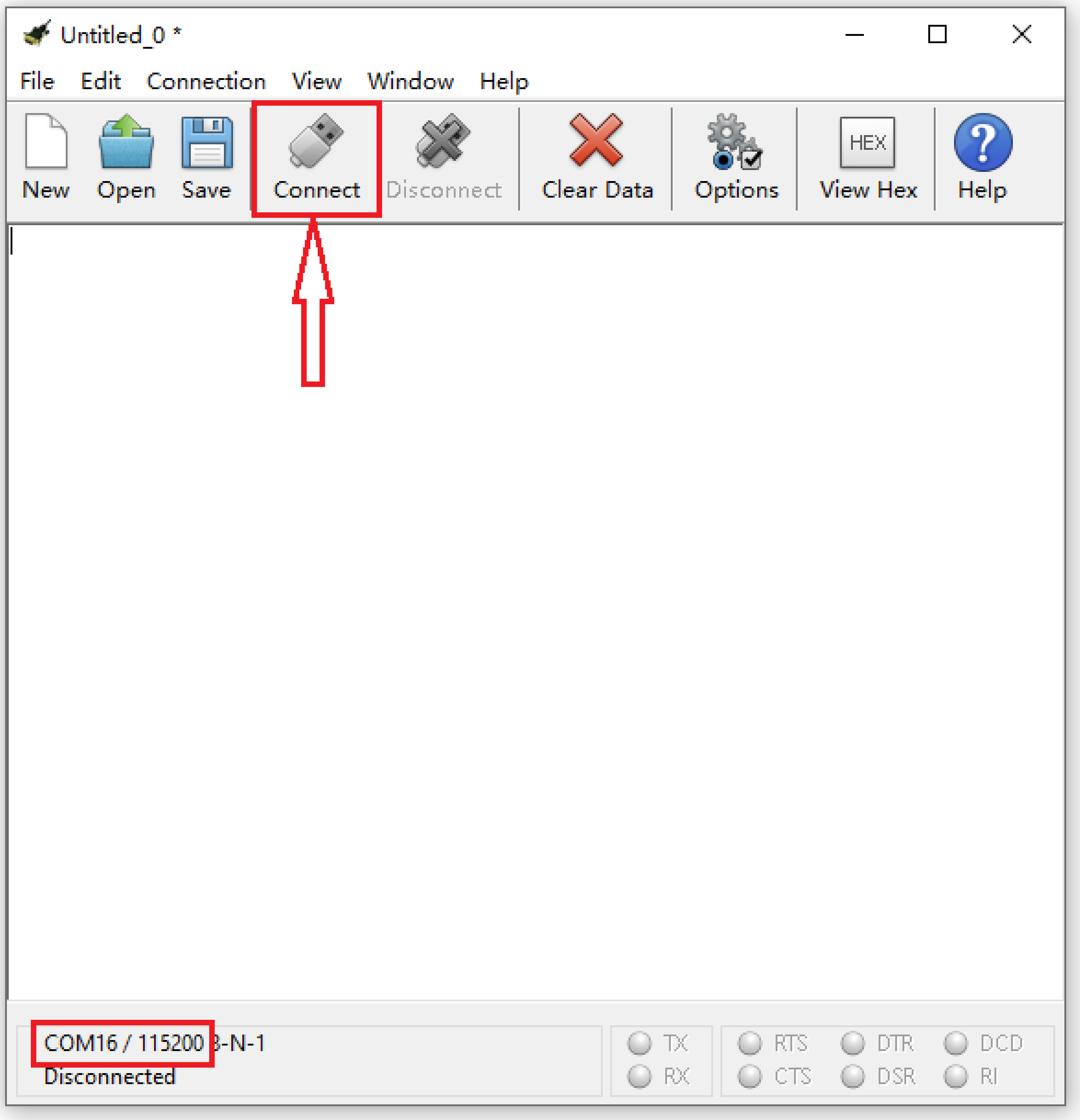
☑ Software Supp

☑ Block Keystro

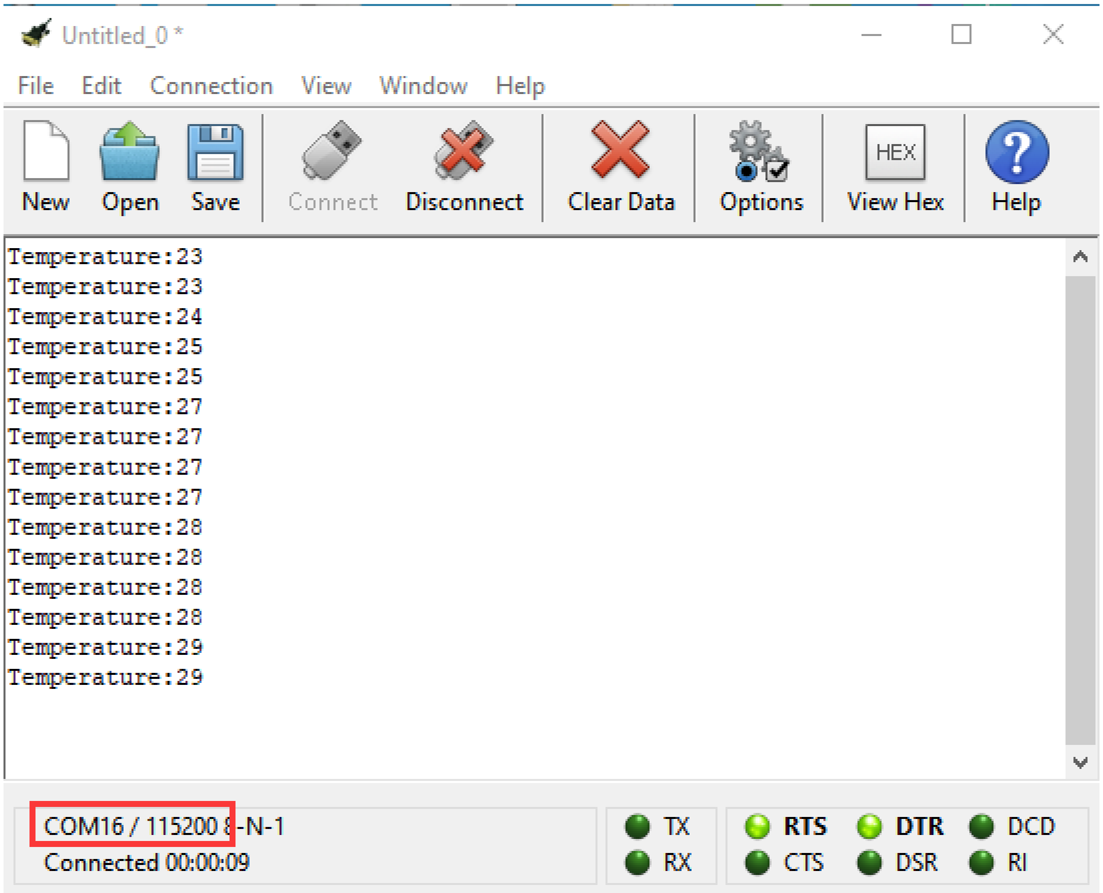
Initial Line States

● DTR On ○ DTR Off

● RTS On ○ RTS Off



COM16 / 115200 8-N-1
Disconnected


Untitled_0 *
File Edit Connection View Window Help
New Open Save Connect Disconnect Clear Data Options View Hex Help

```
Temperature:23
Temperature:23
Temperature:24
Temperature:25
Temperature:25
Temperature:27
Temperature:27
Temperature:27
Temperature:27
Temperature:28
Temperature:28
Temperature:28
Temperature:28
Temperature:29
Temperature:29
```

COM16 / 115200 8-N-1
Connected 00:00:09

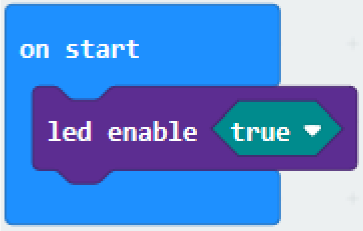
TX
RX
RTS
CTS
DTR
DSR
DCD
RI

5. Test Code 2:

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

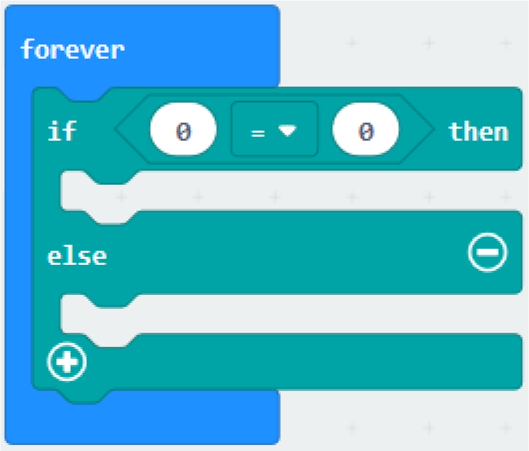
A. Go to“Led”→“more”→“led enable false”block,

B. Keep it into the“on start”blocktap the triangle button to select“true”.



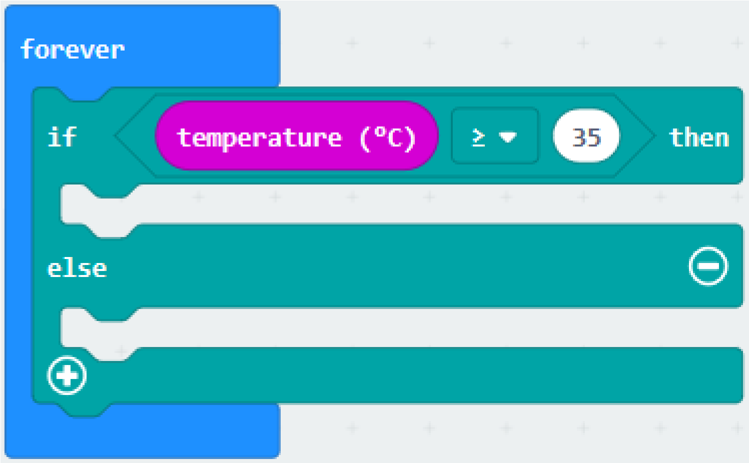
****

Tap “Logic” and drag “if…then…else” into “forever” block; and then drag “=” into “true”



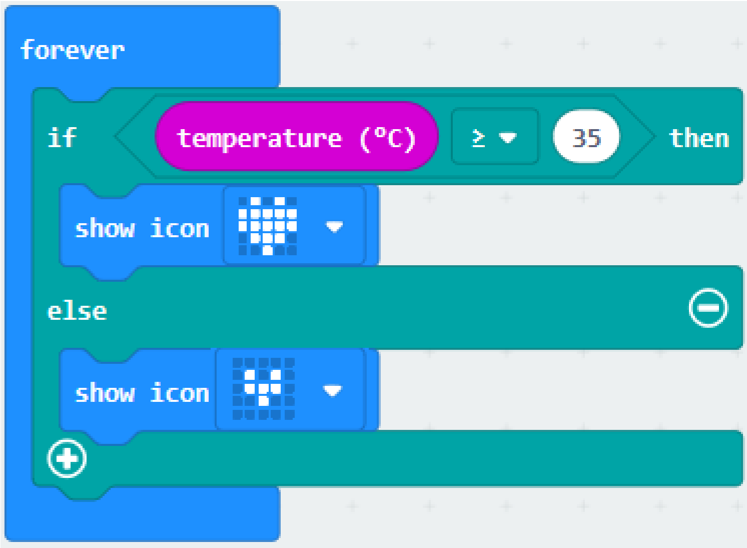
```
forever
  if 0 = 0 then

  else

```

Enter “Input” to move “temperature(℃)” into the left side of “=”; click the little triangle of “=” to choose “”, and change the “0” to “35”



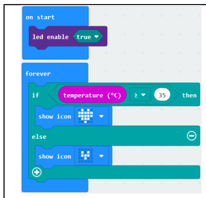
```
forever
  if temperature (°C) ≥ 35 then

  else

```

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 “♥”

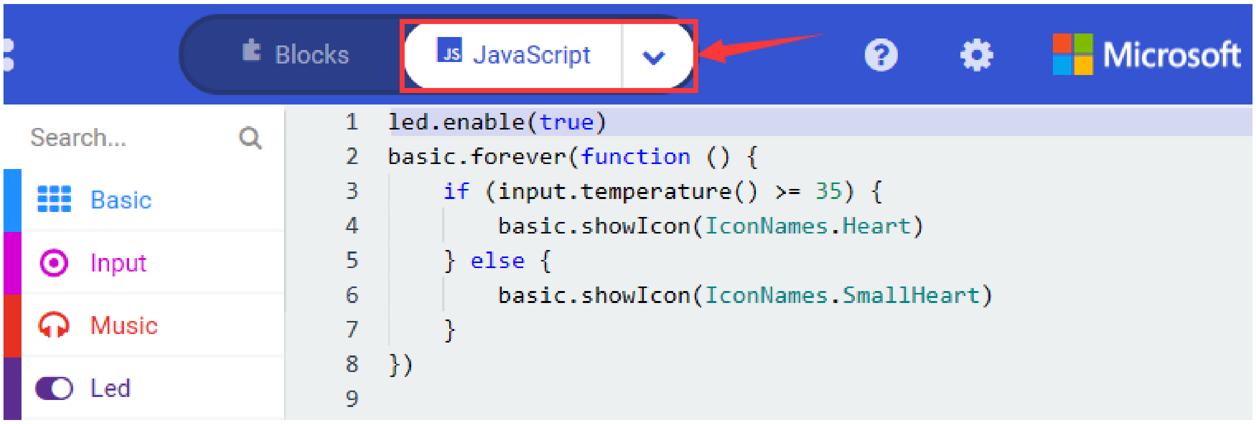


```
forever
  if  temperature (°C)  ≥ ▾  35  then
    show icon ▾
  else
    show icon ▾
```

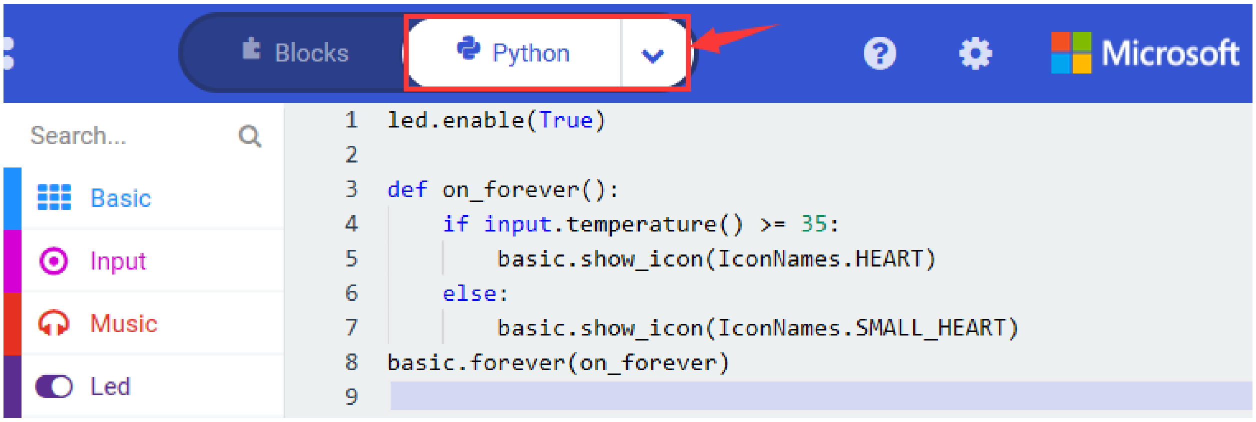
Complete Program

 on start led enable true ▾ forever if temperature (°C) ≥ ▾ 35 then show icon ▾ else show icon ▾	In“on start”the program only runs once;Open the LED dot matrix on the micro:bit board;In“forever”the program runs cyclically;When the condition that the detected temperature is 35℃, the program in then runs;LED dot matrix displays pattern“♥”;When the condition that the detected temperature is not 35℃, the program in else runs;LED dot matrix displays pattern“♥”.

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

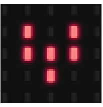
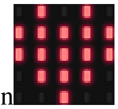


```
led.enable(true)
basic.forever(function () {
    if (input.temperature() >= 35) {
        basic.showIcon(IconNames.Heart)
    } else {
        basic.showIcon(IconNames.SmallHeart)
    }
})
```



```
led.enable(True)

def on_forever():
    if input.temperature() >= 35:
        basic.show_icon(IconNames.HEART)
    else:
        basic.show_icon(IconNames.SMALL_HEART)
basic.forever(on_forever)
```

6. Test Results 2:

Uploading the code 2, when the ambient temperature is less than 35℃, 5*5LED will show . When the temperature is equivalent to or greater than 35℃, the pattern  will appear.

7.6 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:

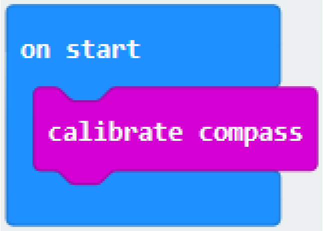
- Micro:bit main board V2 *1
- Micro USB cable*1

3. Test Code 1:

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

A. Click“Input”→“more”→“calibrate compass”

B. Lay down it into block“on start”.

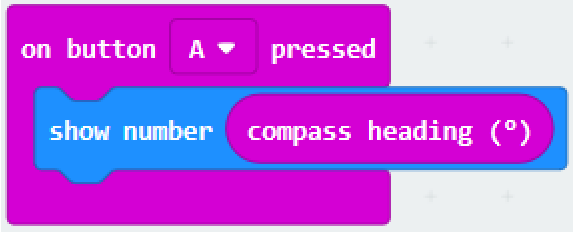


```
on start
  calibrate compass
```

A. Go to“Input”→“on button A pressed”.

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

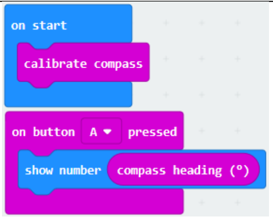
C. Tap“Input”→“compass heading(℃)” and place it into“show number”



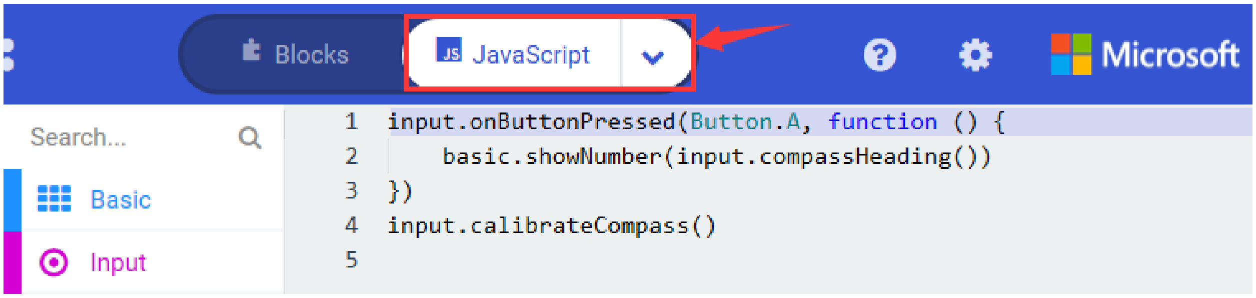
```
on button A ▾ pressed
  show number compass heading (°)
```

****

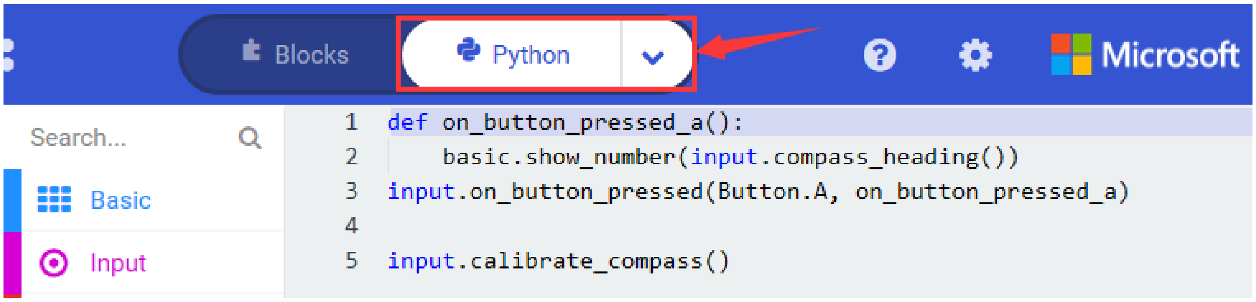
Complete Program

 on start calibrate compass on button A ▾ pressed show number compass heading (°)	“on start”: command block only runs once to start program.Calibrate compass-Press button A on Micro:bit main boardDot matrix shows the direction of compass heading

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



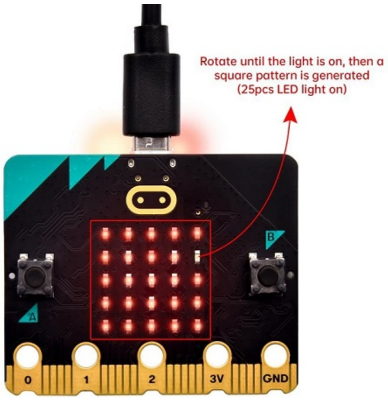
```
input.onButtonPressed(Button.A, function () {
    basic.showNumber(input.compassHeading())
})
input.calibrateCompass()
```



```
def on_button_pressed_a():
    basic.show_number(input.compass_heading())
input.on_button_pressed(Button.A, on_button_pressed_a)

input.calibrate_compass()
```

4. Test Results 1:

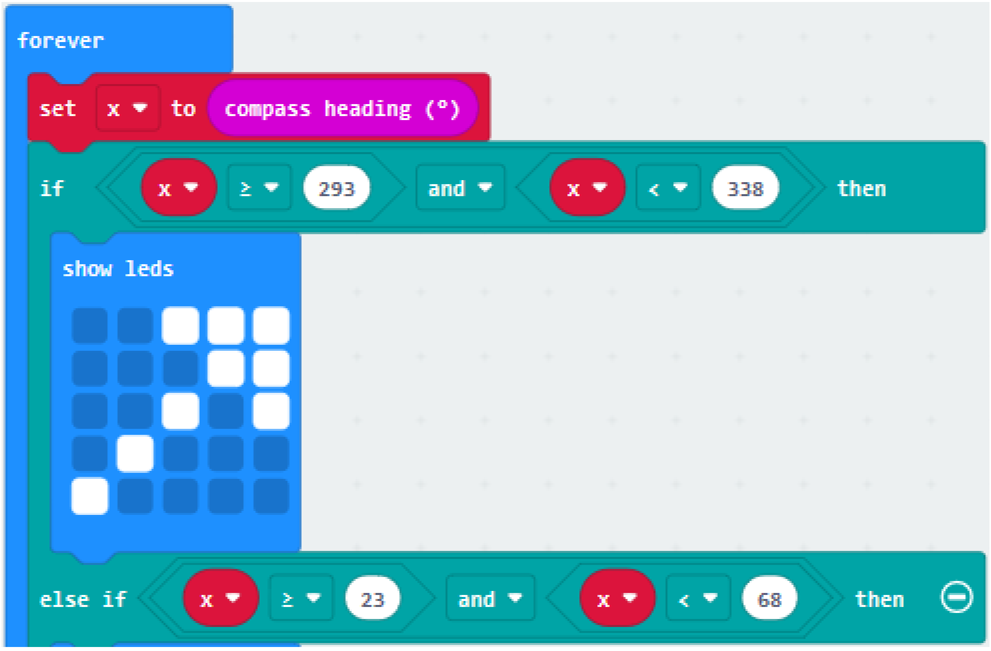
Uploading test code to micro:bit main board V2 and powering the board via the USB cable, and pressing the button A, the board asks us to calibrate compass and the LED dot matrix shows "TILT TO FILL SCREEN". Then enter the calibration page. Rotate the board until all 25 LEDs are on red as shown below.



Rotate until the light is on, then a square pattern is generated (25pcs LED light on)

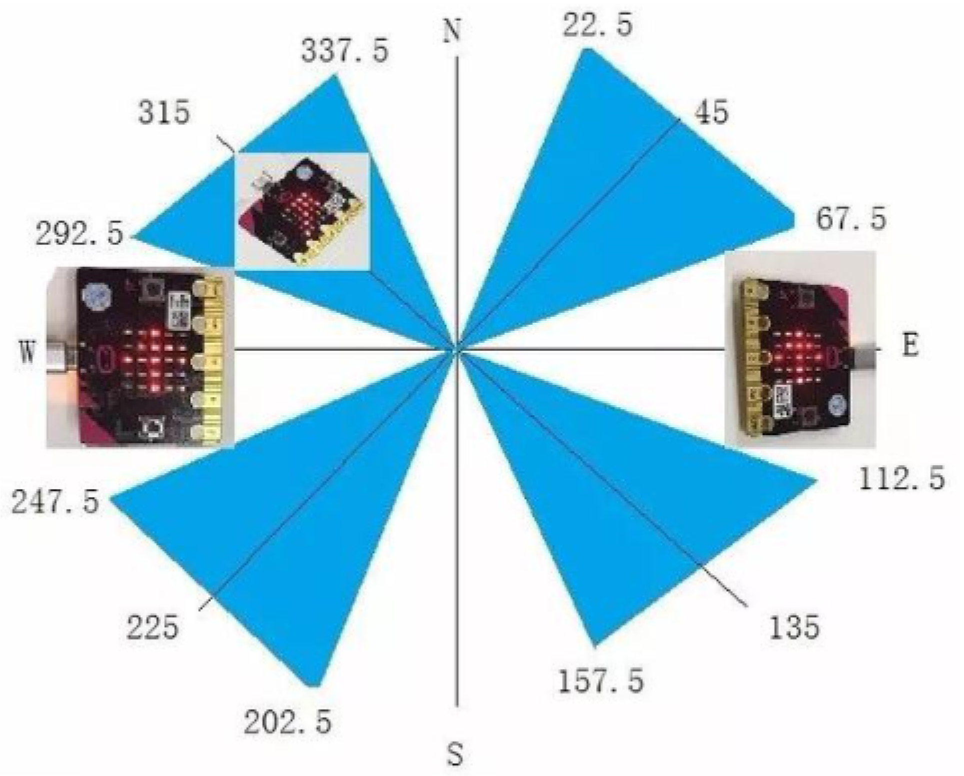
After that, a smile pattern appears, which implies the calibration is done. When the calibration process is completed, pressing the button A will make the magnetometer reading display directly on the screen. And the direction north, east, south and west correspond to 0°, 90°, 180° and 270°.

5. Test Code 2:



```
forever
    set x ▾ to compass heading (°)
    if x ▾ ≥ ▾ 293 and ▾ x ▾ < ▾ 338 then
        show leds
            . . # # #
            . . . # #
            . . # . #
            . # . . .
            # . . . .
    else if x ▾ ≥ ▾ 23 and ▾ x ▾ < ▾ 68 then ⊖
```

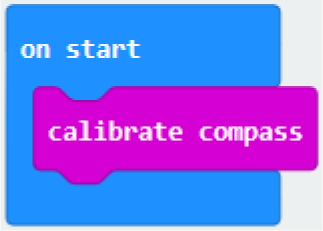
This module can keep readings to determine direction, so does point to the current magnetic North Pole by arrow.



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

Enter"Input"→ "more"→"calibrate compass". Move"calibrate compass"into"on start"



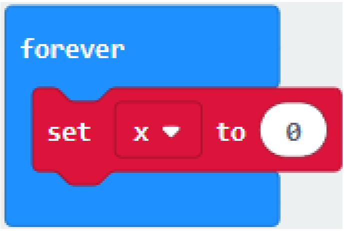
```
on start
    calibrate compass
```

****

A. Click"Variables"→"Make a Variable…"→"New variable name"

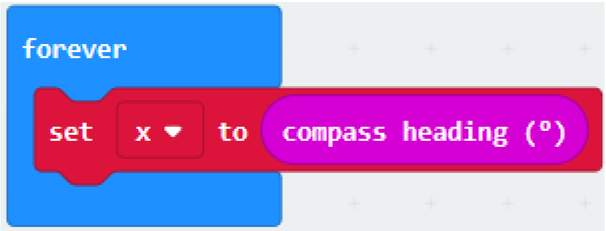
B. Input"x"in the blank box and click"OK", and the variable "x" is generated.

C. Drag out"set x to"into"forever"block



```
forever
    set x ▾ to 0
```

Go to“Input”→“compass heading(℃)”, and keep it into“0”box



```
forever
    set x ▾ to compass heading (°)
```

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

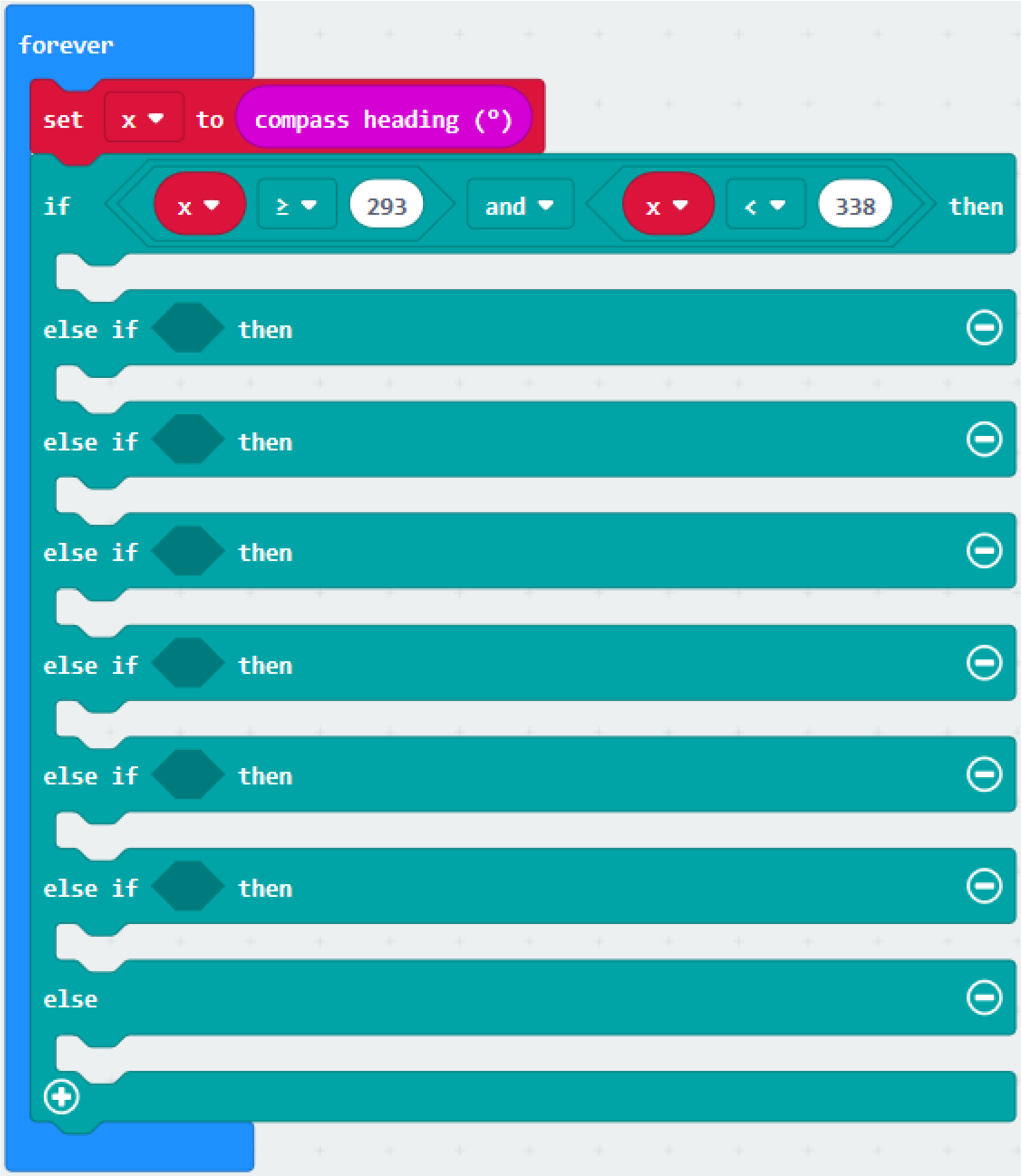
****

A. Place“and”into“true”block

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

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

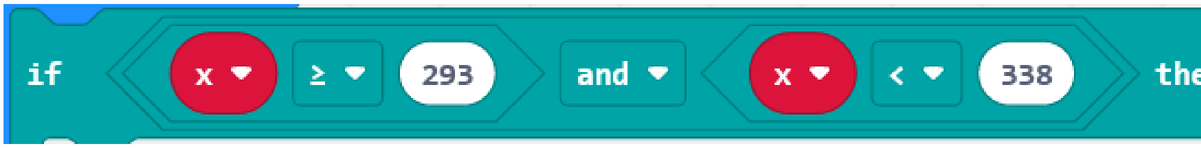
D. Then copy“x293”once and leave it to the right “0”box and set to“x<338”



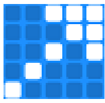
```
forever
  set x ▾ to compass heading (°)
  if x ▾ ≥ ▾ 293 and ▾ x ▾ < ▾ 338 then

  else if then ⊖

  else if then ⊖

  else if then ⊖

  else if then ⊖

  else if then ⊖

  else if then ⊖

  else ⊖

  ⊕
```

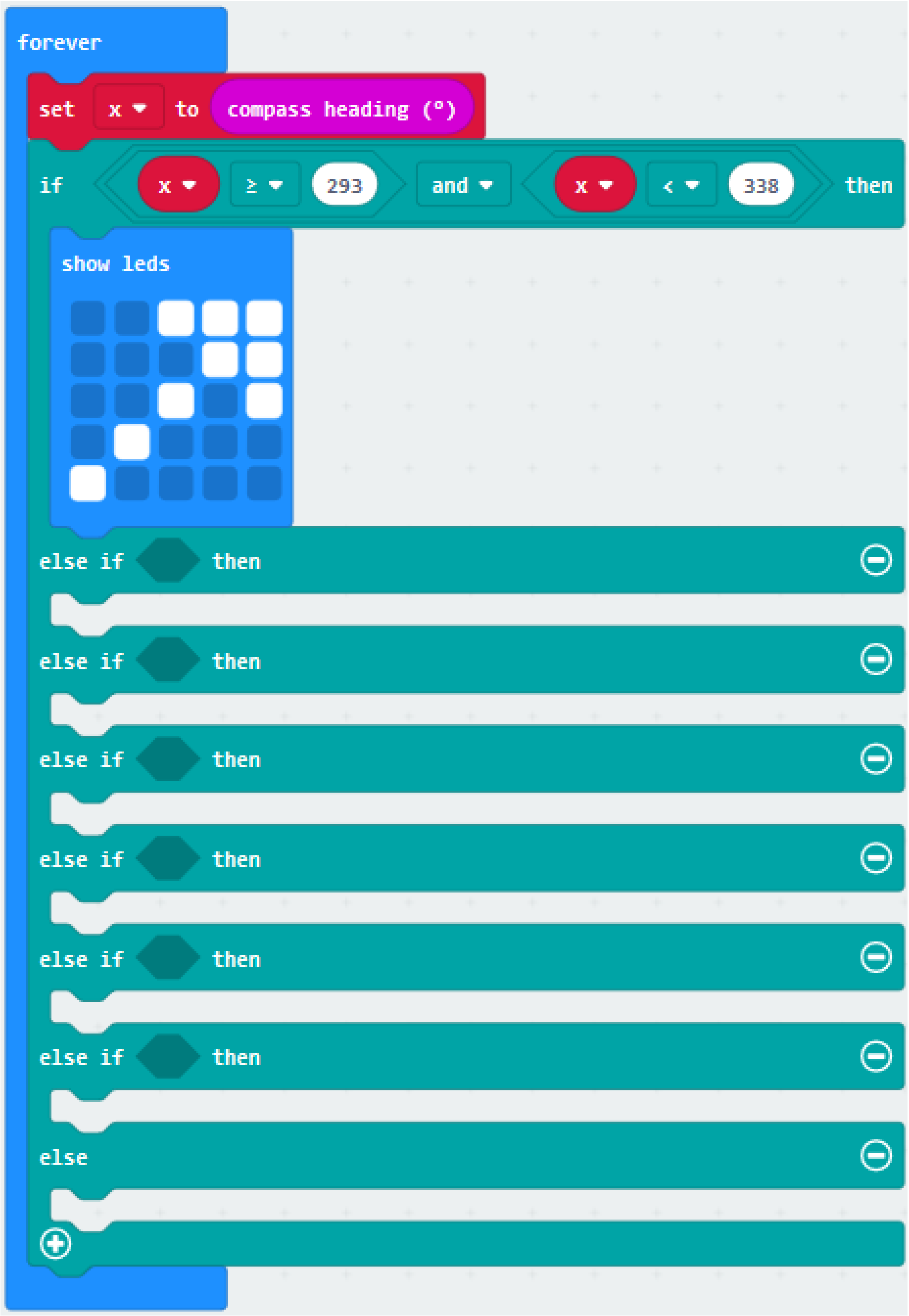
****

A. Go to“Basic”→“show leds”

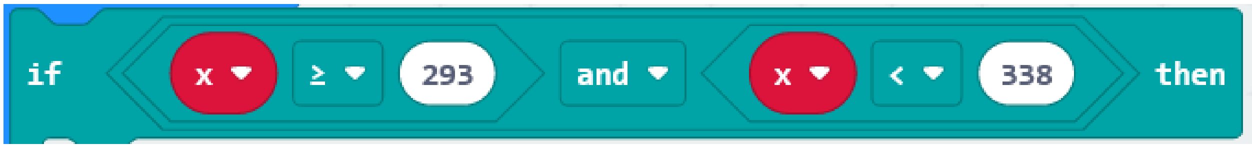
B. Lay it down beneath 

```
if x ▾ ≥ ▾ 293 and ▾ x ▾ < ▾ 338 the
```

then click“show leds”and the pattern  appears.



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

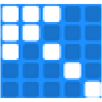
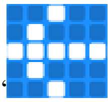
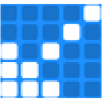
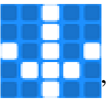
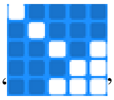
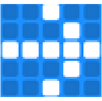
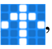
A. Duplicate  for 6 times.

```
if x ≥ 293 and x < 338 then
```

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

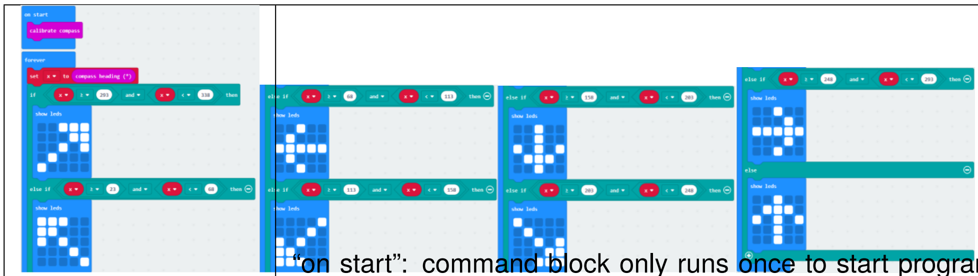
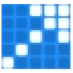
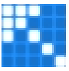
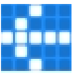
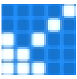
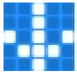
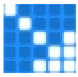
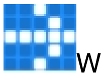
C. Set to“x23 and x<68”,“x68 and x<113 ”,“x113 and x<158 ”,“x158 and x<203 ”,“x203 and x<248 ”,“x248 and x<293 ”respectively.

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

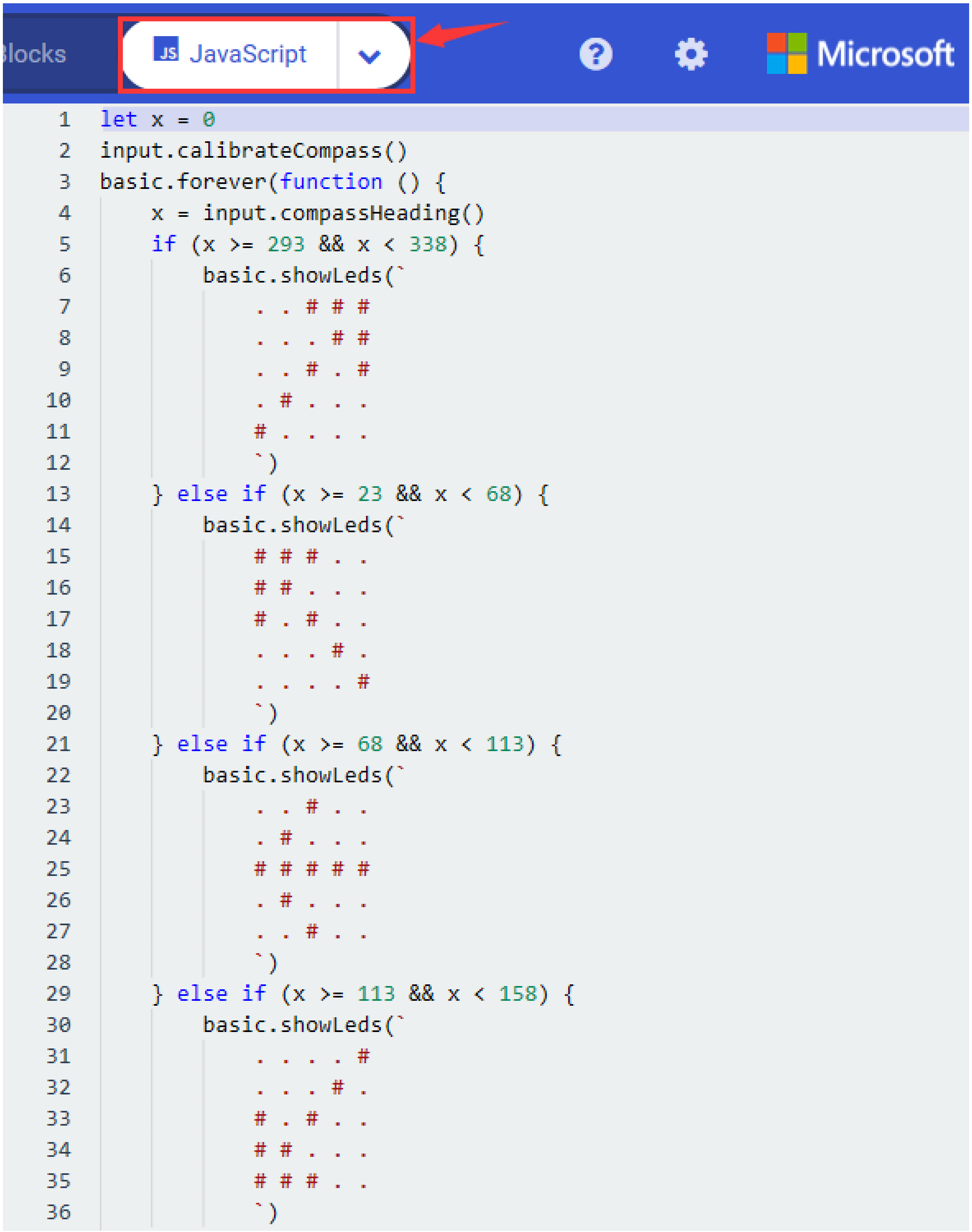
E. Click the blue boxes to form the pattern“”, “”, “”, “”, “”, “”and “”.

****

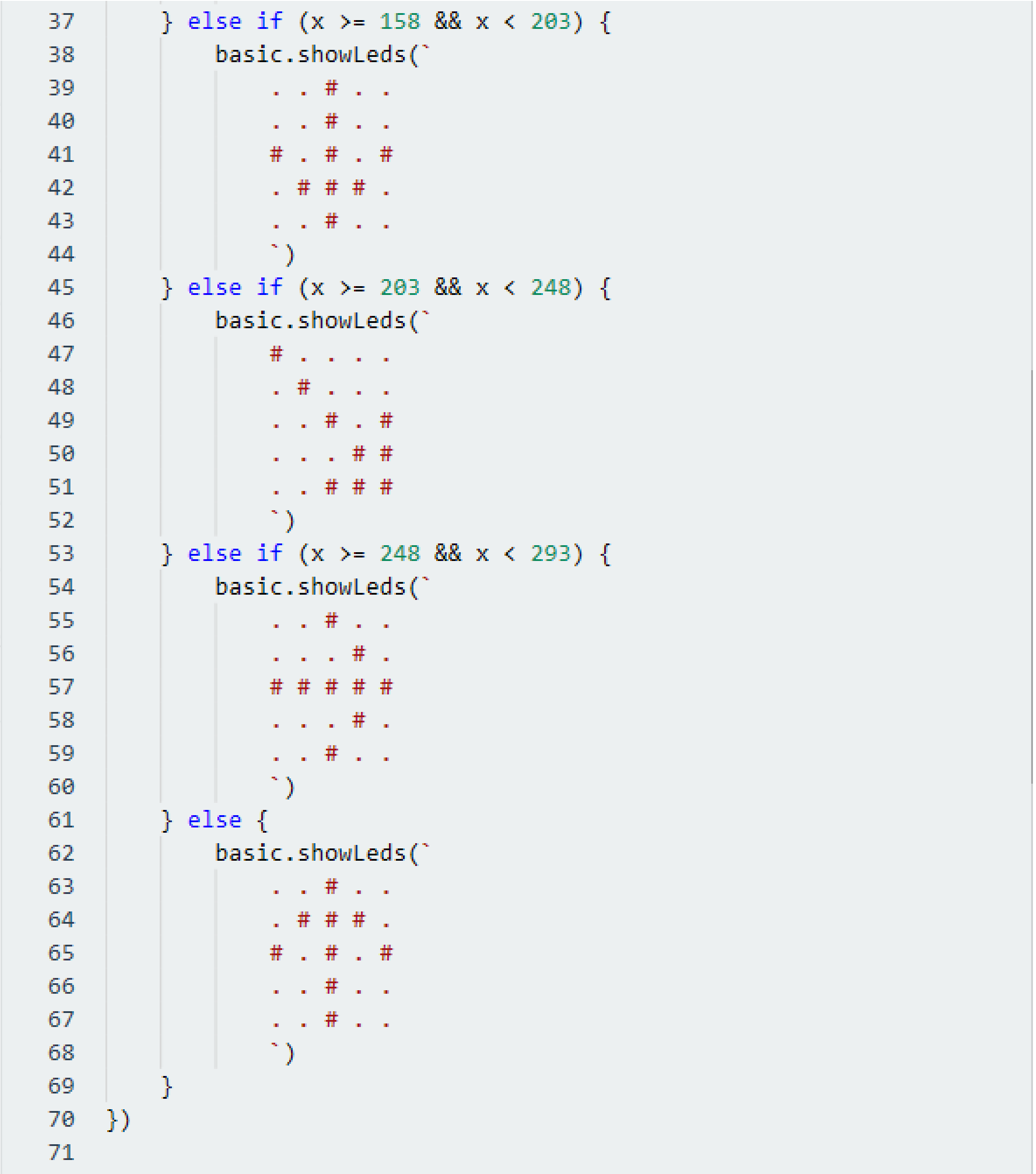
Complete Program

<table>
<tr>
<td>
<pre>
on start
  calibrate compass
forever
  set x to compass heading (°)
  if x ≥ 293 and x &lt; 338 then
    show leds
  else if x ≥ 23 and x &lt; 68 then
    show leds
  else if x ≥ 68 and x &lt; 113 then
    show leds
  else if x ≥ 113 and x &lt; 158 then
    show leds
  else if x ≥ 158 and x &lt; 203 then
    show leds
  else if x ≥ 203 and x &lt; 248 then
    show leds
  else if x ≥ 248 and x &lt; 293 then
    show leds
  else
    show leds
</pre></td>
<td>“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 xWhen 293x<338, the next program will be executedappears on the dot matrixWhen 23x<68, the next program will be executedis displayed on dot matrixWhen 68x<113, the next program will be executed is shown on dot matrixWhen 113x<158, the next program will be executedpattern appearsWhen 158x<203, the next program will be executed.Dot matrix shows When 203x<248, the next program will be executed.Dot matrix displays When 248x<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</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</table>

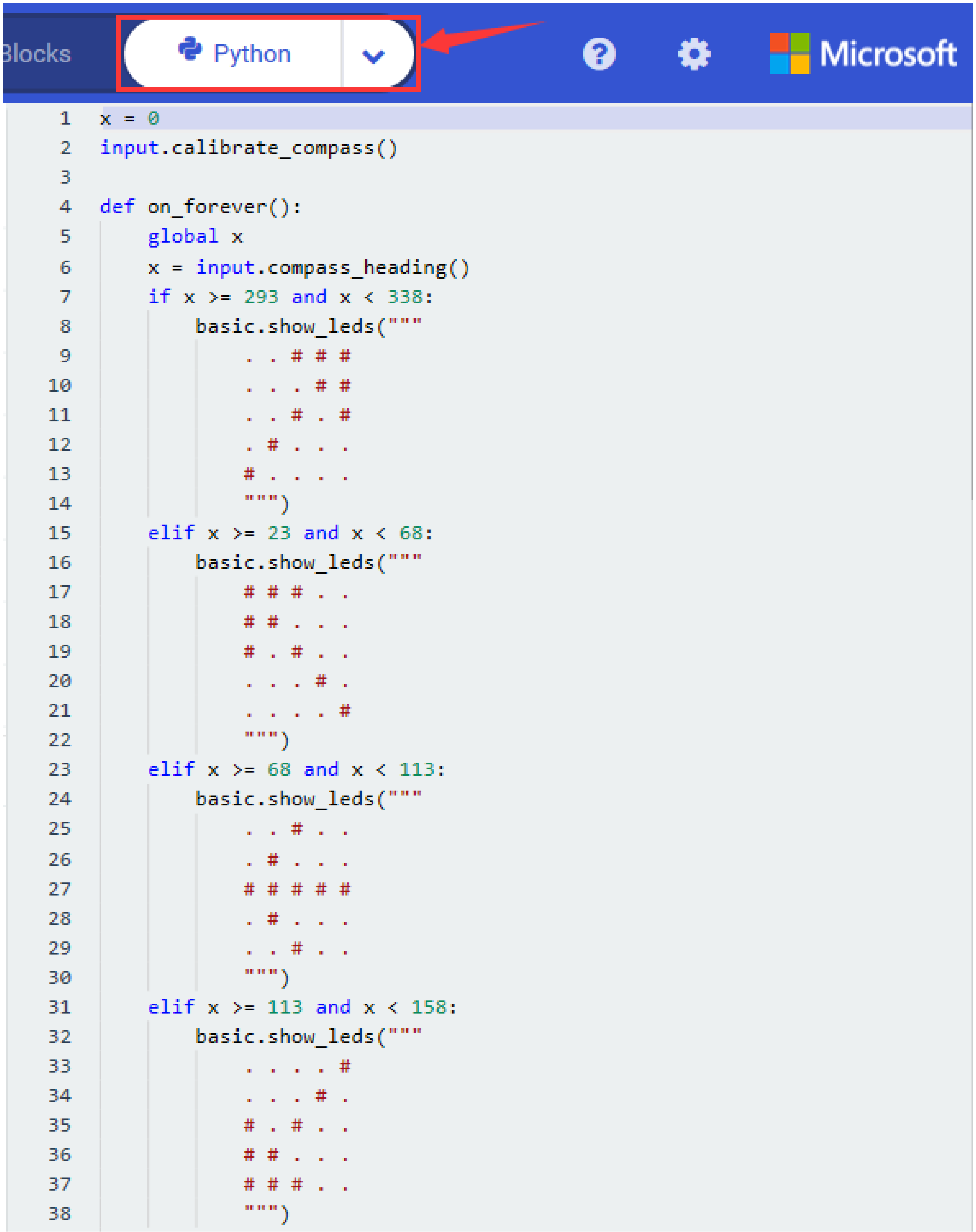
Select“JavaScript” and“Python”to switch into JavaScript and Python language code:



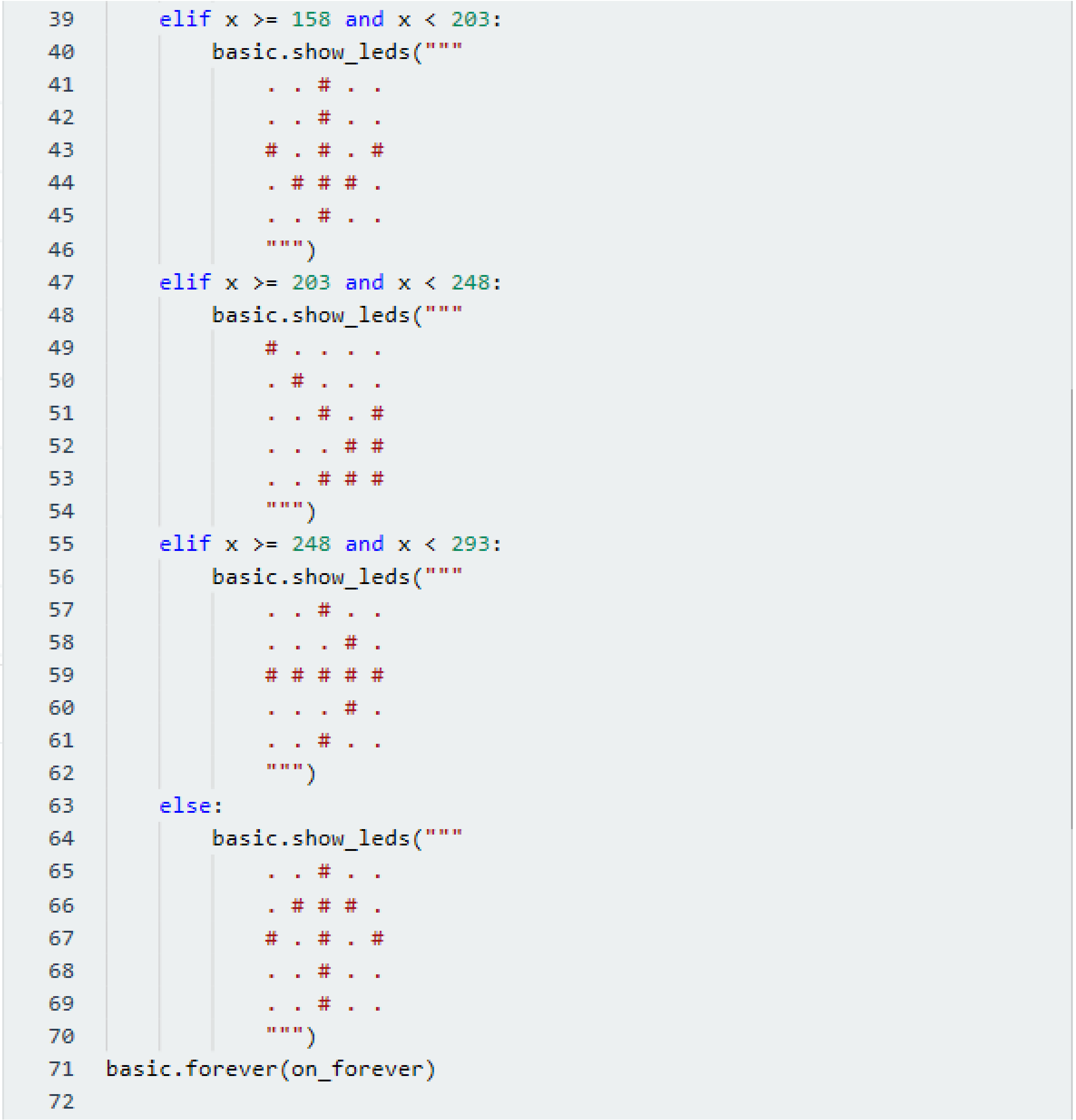
```
let x = 0
input.calibrateCompass()
basic.forever(function () {
    x = input.compassHeading()
    if (x >= 293 && x < 338) {
        basic.showLeds(`
            . . # # #
            . . . # #
            . . # . #
            . # . . .
            # . . . .
            `)
    } else if (x >= 23 && x < 68) {
        basic.showLeds(`
            # # # . .
            # # . . .
            # . # . .
            . . . # .
            . . . . #
            `)
    } else if (x >= 68 && x < 113) {
        basic.showLeds(`
            . . # . .
            . # . . .
            # # # # #
            . # . . .
            . . # . .
            `)
    } else if (x >= 113 && x < 158) {
        basic.showLeds(`
            . . . . #
            . . . # .
            # . # . .
            # # . . .
            # # # . .
            `)
```



```
    } else if (x >= 158 && x < 203) {
        basic.showLeds(`
            . . # . .
            . . # . .
            # . # . #
            . # # # .
            . . # . .
            `)
    } else if (x >= 203 && x < 248) {
        basic.showLeds(`
            # . . . .
            . # . . .
            . . # . #
            . . . # #
            . . # # #
            `)
    } else if (x >= 248 && x < 293) {
        basic.showLeds(`
            . . # . .
            . . . # .
            # # # # #
            . . . # .
            . . # . .
            `)
    } else {
        basic.showLeds(`
            . . # . .
            . # # # .
            # . # . #
            . . # . .
            . . # . .
            `)
    }
})

```



```
x = 0
input.calibrate_compass()

def on_forever():
    global x
    x = input.compass_heading()
    if x >= 293 and x < 338:
        basic.show_leds("""
            . . # # #
            . . . # #
            . . # . #
            . # . . .
            # . . . .
            """)
    elif x >= 23 and x < 68:
        basic.show_leds("""
            # # # . .
            # # . . .
            # . # . .
            . . . # .
            . . . . #
            """)
    elif x >= 68 and x < 113:
        basic.show_leds("""
            . . # . .
            . # . . .
            # # # # #
            . # . . .
            . . # . .
            """)
    elif x >= 113 and x < 158:
        basic.show_leds("""
            . . . . #
            . . . # .
            # . # . .
            # # . . .
            # # # . .
            """)
```



```
    elif x >= 158 and x < 203:
        basic.show_leds("""
            . . # . .
            . . # . .
            # . # . #
            . # # # .
            . . # . .
            """)
    elif x >= 203 and x < 248:
        basic.show_leds("""
            # . . . .
            . # . . .
            . . # . #
            . . . # #
            . . # # #
            """)
    elif x >= 248 and x < 293:
        basic.show_leds("""
            . . # . .
            . . . # .
            # # # # #
            . . . # .
            . . # . .
            """)
    else:
        basic.show_leds("""
            . . # . .
            . # # # .
            # . # . #
            . . # . .
            . . # . .
            """)
basic.forever(on_forever)
```

6. Test Results 2:

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

7.7 Project 7: Accelerometer

1. Project Description:

The Micro: Bit main board 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:

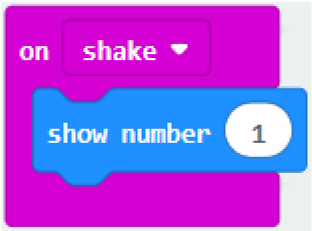
- Micro:bit main board V2 *1
- Micro USB cable*1

3. Test Code 1:

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

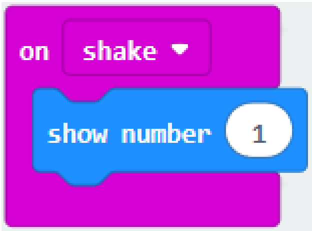
(1) A. Enter"Input"→"on shake"

B. Click"Basic"→"show number", place it into"on shake"block, then change 0 into 1.



```
on shake ▾
    show number 1
```

****

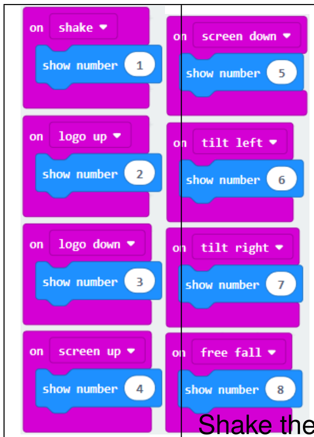


```
on shake ▾
    show number 1
```

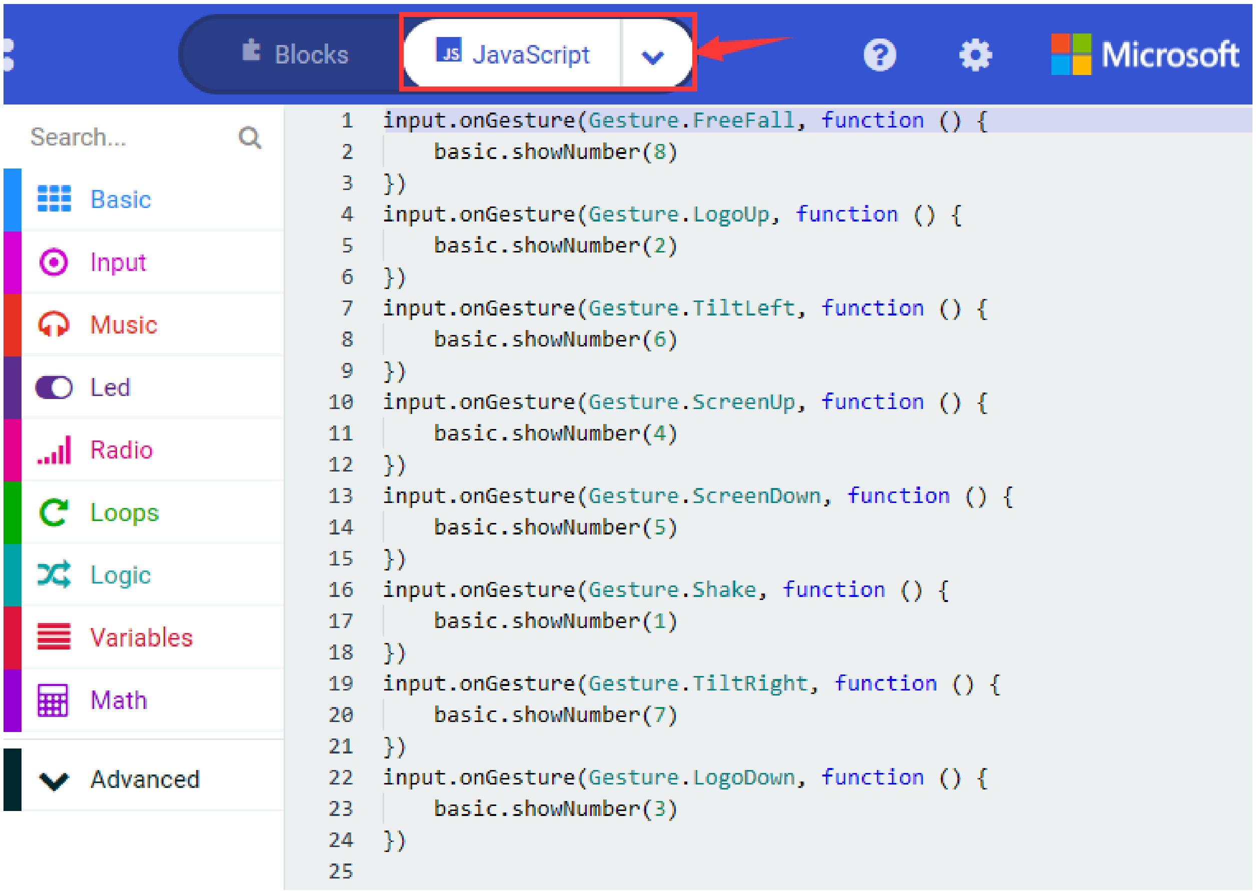
(2) Copy code string for 7 times; separately click the triangle button to select"logo up","logo down","screen up","screen down","tilt left","tilt right"and"free fall", then respectively change 1 into 2, 3, 4, 5, 6, 7, 8.

****

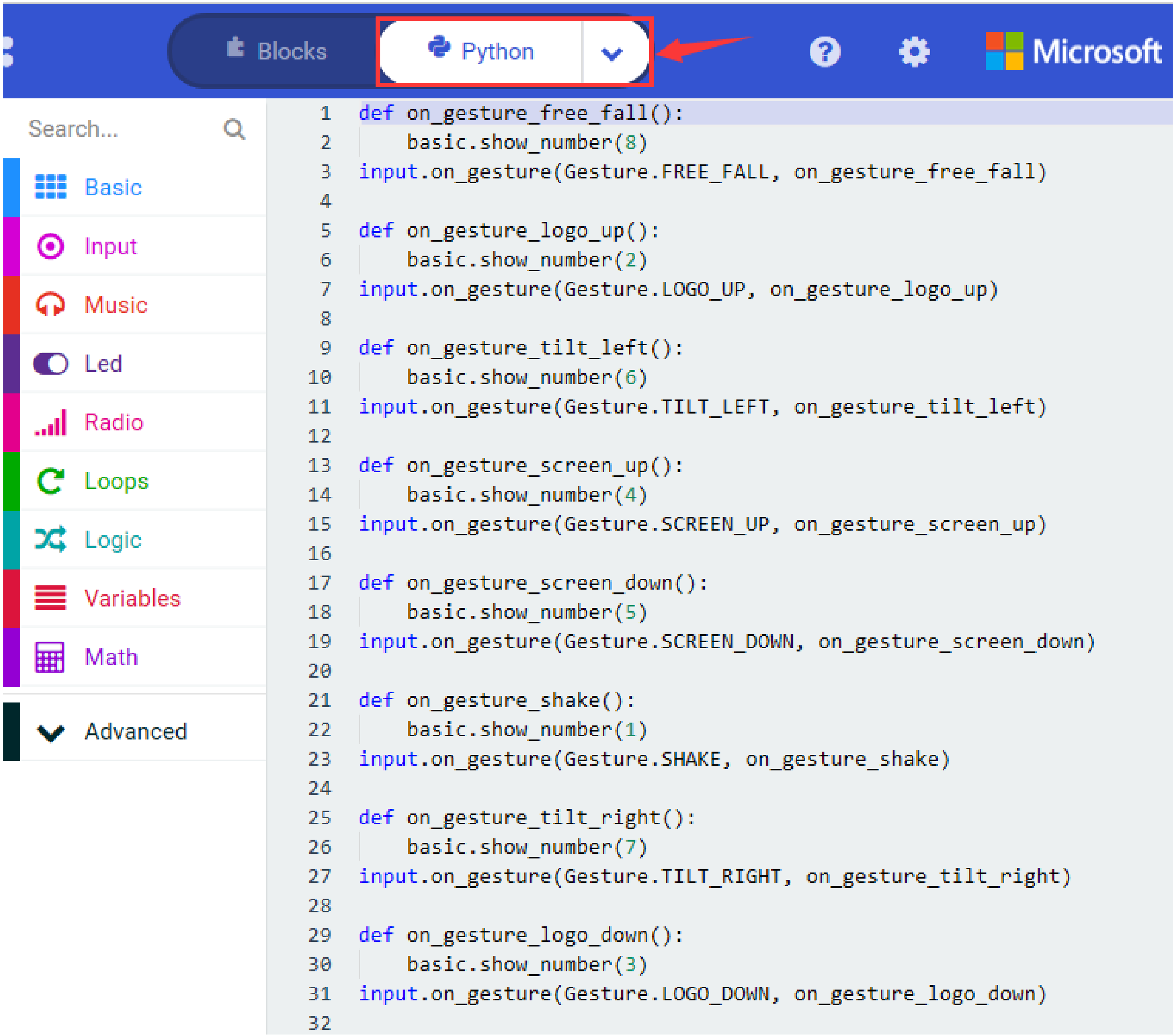
Complete Program

 on shake show number 1 on logo up show number 2 on logo down show number 3 on screen up show number 4 on screen down show number 5 on tilt left show number 6 on tilt right show number 7 on free fall show number 8	Shake the Micro:bit boardLED dot matrix displays 1The log is up LED dot matrix displays 2The logo is down LED dot matrix displays 3The screen is upLED dot matrix displays 4The screen is downNumber 5 is shownThe Micro:bit board is tilt to the leftNumber 6 is displayedThe Micro:bit board is tilt to the rightNumber7 is displayedWhen the Micro:bit board is free fallLED dot matrix shows 8

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



```
input.onGesture(Gesture.FreeFall, function () {
    basic.showNumber(8)
})
input.onGesture(Gesture.LogoUp, function () {
    basic.showNumber(2)
})
input.onGesture(Gesture.TiltLeft, function () {
    basic.showNumber(6)
})
input.onGesture(Gesture.ScreenUp, function () {
    basic.showNumber(4)
})
input.onGesture(Gesture.ScreenDown, function () {
    basic.showNumber(5)
})
input.onGesture(Gesture.Shake, function () {
    basic.showNumber(1)
})
input.onGesture(Gesture.TiltRight, function () {
    basic.showNumber(7)
})
input.onGesture(Gesture.LogoDown, function () {
    basic.showNumber(3)
})

```



```python
def on_gesture_free_fall():
    basic.show_number(8)
input.on_gesture(Gesture.FREE_FALL, on_gesture_free_fall)

def on_gesture_logo_up():
    basic.show_number(2)
input.on_gesture(Gesture.LOGO_UP, on_gesture_logo_up)

def on_gesture_tilt_left():
    basic.show_number(6)
input.on_gesture(Gesture.TILT_LEFT, on_gesture_tilt_left)

def on_gesture_screen_up():
    basic.show_number(4)
input.on_gesture(Gesture.SCREEN_UP, on_gesture_screen_up)

def on_gesture_screen_down():
    basic.show_number(5)
input.on_gesture(Gesture.SCREEN_DOWN, on_gesture_screen_down)

def on_gesture_shake():
    basic.show_number(1)
input.on_gesture(Gesture.SHAKE, on_gesture_shake)

def on_gesture_tilt_right():
    basic.show_number(7)
input.on_gesture(Gesture.TILT_RIGHT, on_gesture_tilt_right)

def on_gesture_logo_down():
    basic.show_number(3)
input.on_gesture(Gesture.LOGO_DOWN, on_gesture_logo_down)
```

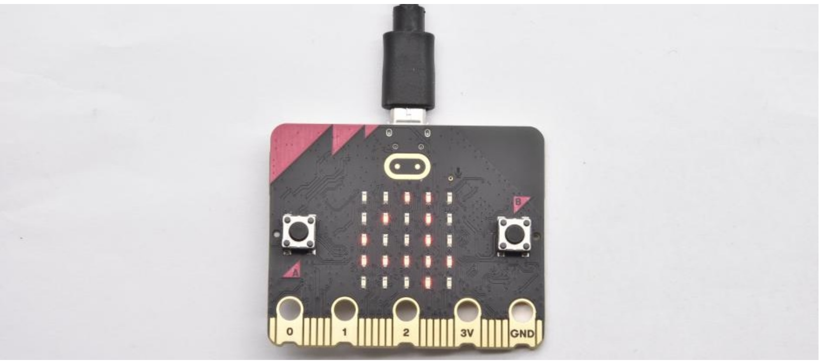
4. Test Results 1:

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

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

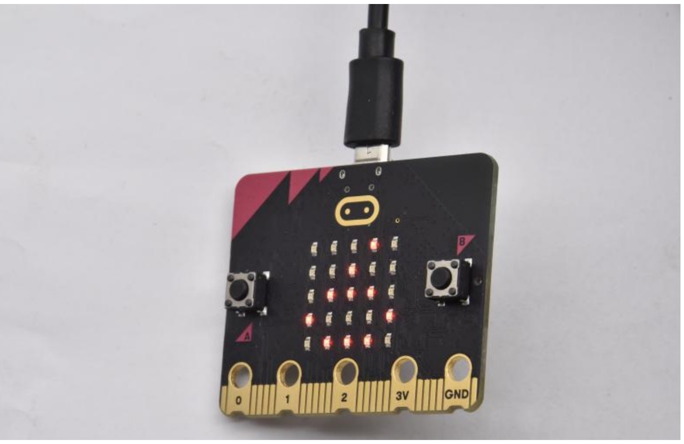
When it is kept upside down(its logo below the LED dot matrix) , it will show as below.

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

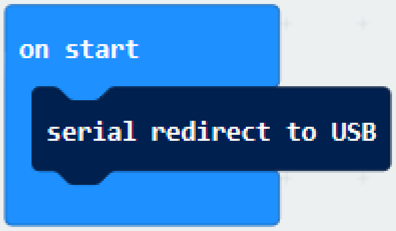
When the board is knocked to the floor, this process can be considered as a free fall and the LED dot matrix shows the number 8. (please note that this test is not recommended for it may damage the main board.)

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:

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

B. Drag it into"on start"

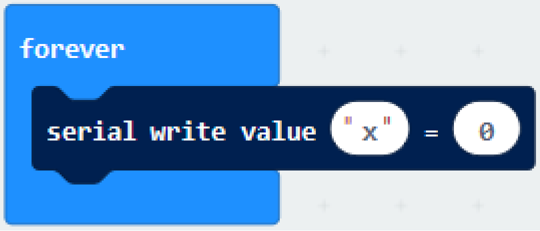


```
on start
  serial redirect to USB
```

****

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

B. Leave it into"forever"block

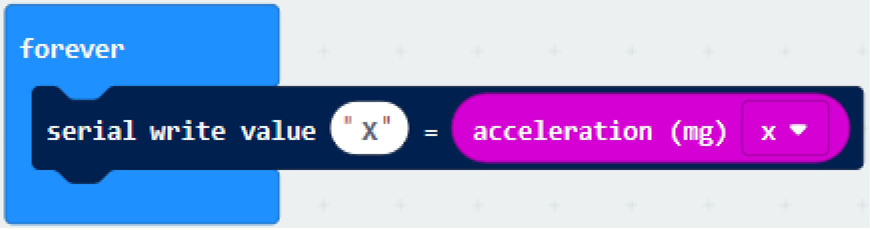


```
forever
  serial write value "x" = 0
```

****

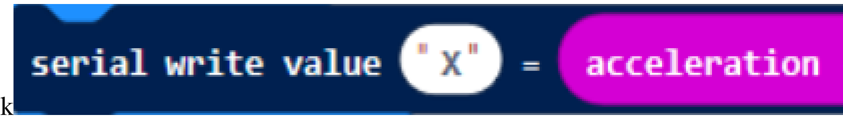
A. Click"Input"→"acceleration(mg) x"

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



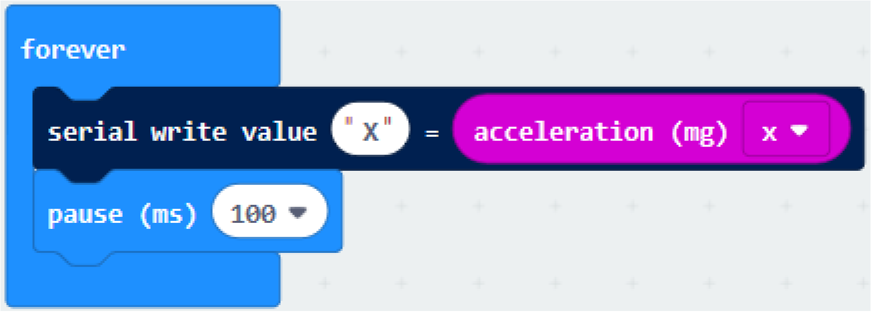
```
forever
  serial write value "X" = acceleration (mg) x ▾
```

****



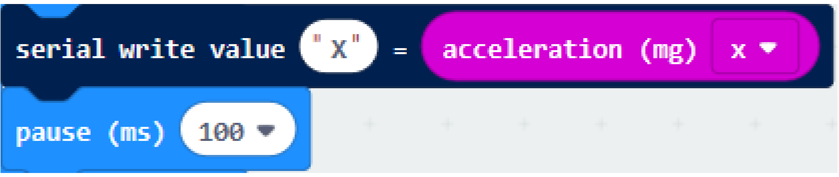
```
serial write value "X" = acceleration
```

Go t"Basic"and move out"pause (ms) 100"below the block then set to 100ms.



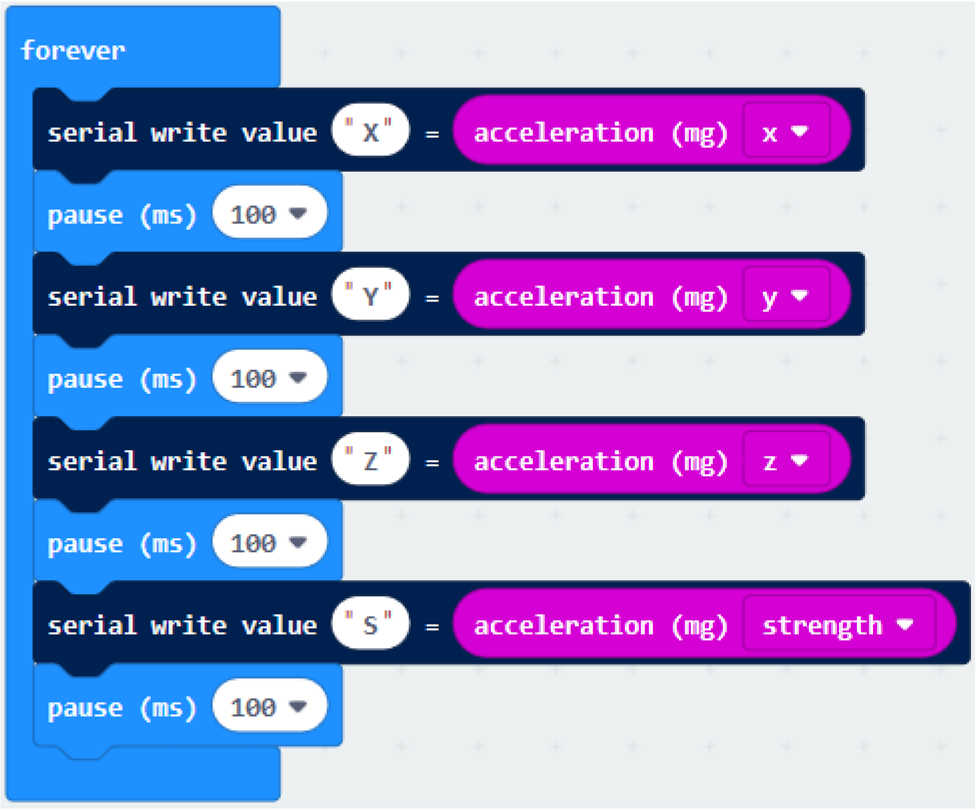
```
forever
    serial write value "X" = acceleration (mg) x
    pause (ms) 100
```

****



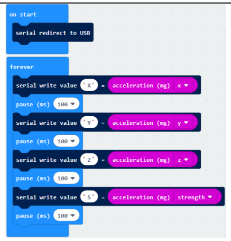
```
serial write value "X" = acceleration (mg) x
pause (ms) 100
```

Replicate code string for 3 times and keep them into“forever”blockseparately set the whole code string as follows:

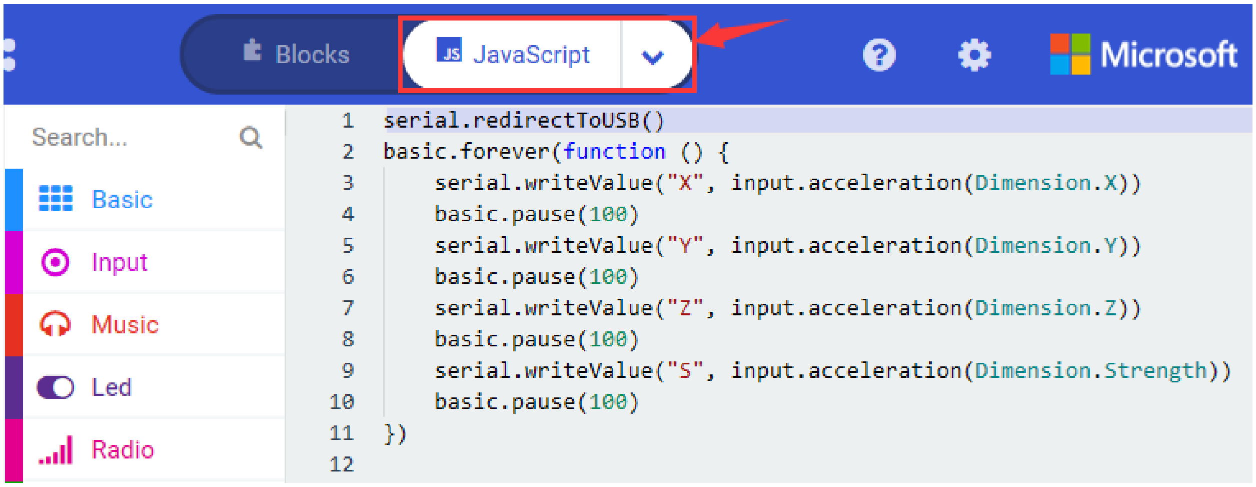


```
forever
    serial write value "X" = acceleration (mg) x
    pause (ms) 100
    serial write value "Y" = acceleration (mg) y
    pause (ms) 100
    serial write value "Z" = acceleration (mg) z
    pause (ms) 100
    serial write value "S" = acceleration (mg) strength
    pause (ms) 100
```

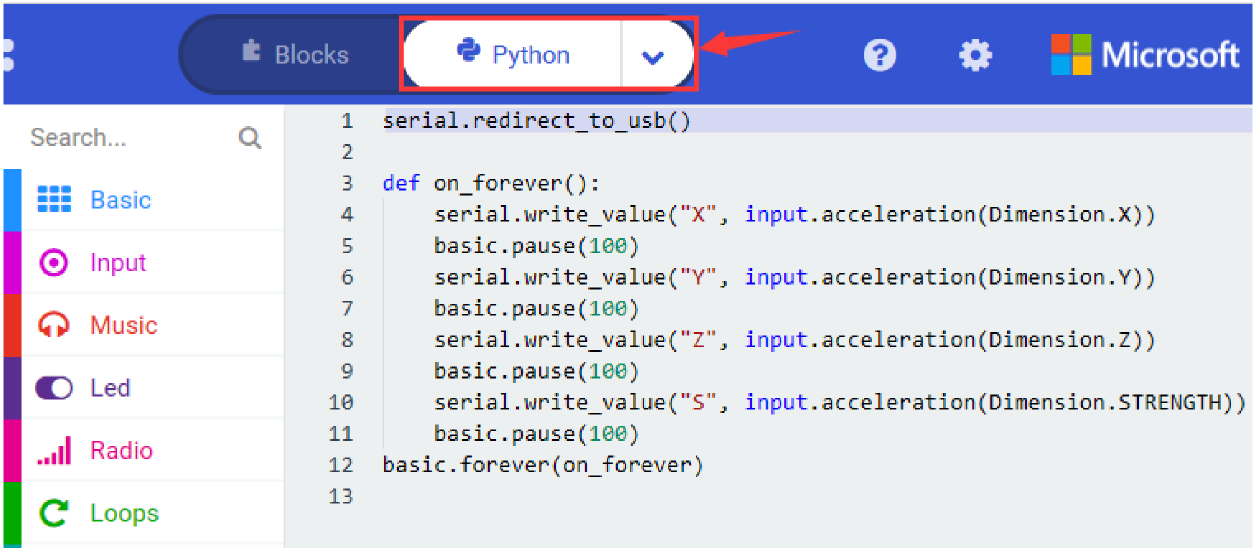
Complete Program

 on start serial redirect to USB forever serial write value "X" = acceleration (mg) x pause (ms) 100 serial write value "Y" = acceleration (mg) y pause (ms) 100 serial write value "Z" = acceleration (mg) z pause (ms) 100 serial write value "S" = acceleration (mg) strength pause (ms) 100	"on start": command block runs once to start program.Serial redirects to USBThe program under the block "forever" runs cyclically.Serial write value "X"=acceleration value on x axisDelay 100msSerial write value "Y"=acceleration value on y axisDelay 100msSerial write value "Z"=acceleration value on z axisDelay 100msSerial write value "S"=acceleration value on s axisDelay 100ms

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



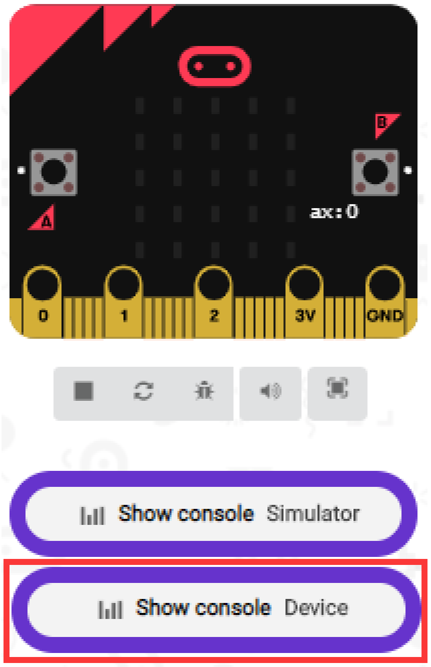
```
serial.redirectToUSB()
basic.forever(function () {
    serial.writeValue("X", input.acceleration(Dimension.X))
    basic.pause(100)
    serial.writeValue("Y", input.acceleration(Dimension.Y))
    basic.pause(100)
    serial.writeValue("Z", input.acceleration(Dimension.Z))
    basic.pause(100)
    serial.writeValue("S", input.acceleration(Dimension.Strength))
    basic.pause(100)
})
```



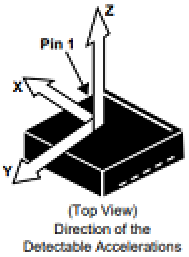
```
serial.redirect_to_usb()

def on_forever():
    serial.write_value("X", input.acceleration(Dimension.X))
    basic.pause(100)
    serial.write_value("Y", input.acceleration(Dimension.Y))
    basic.pause(100)
    serial.write_value("Z", input.acceleration(Dimension.Z))
    basic.pause(100)
    serial.write_value("S", input.acceleration(Dimension.STRENGTH))
    basic.pause(100)
basic.forever(on_forever)
```

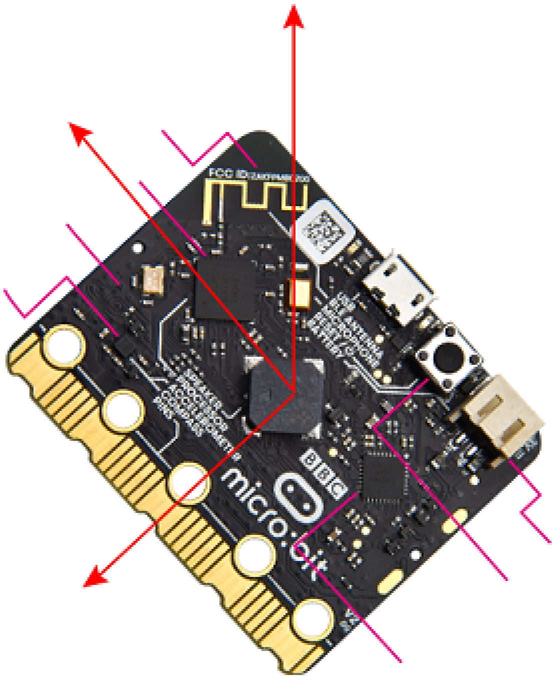
6. Test Results 2:

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

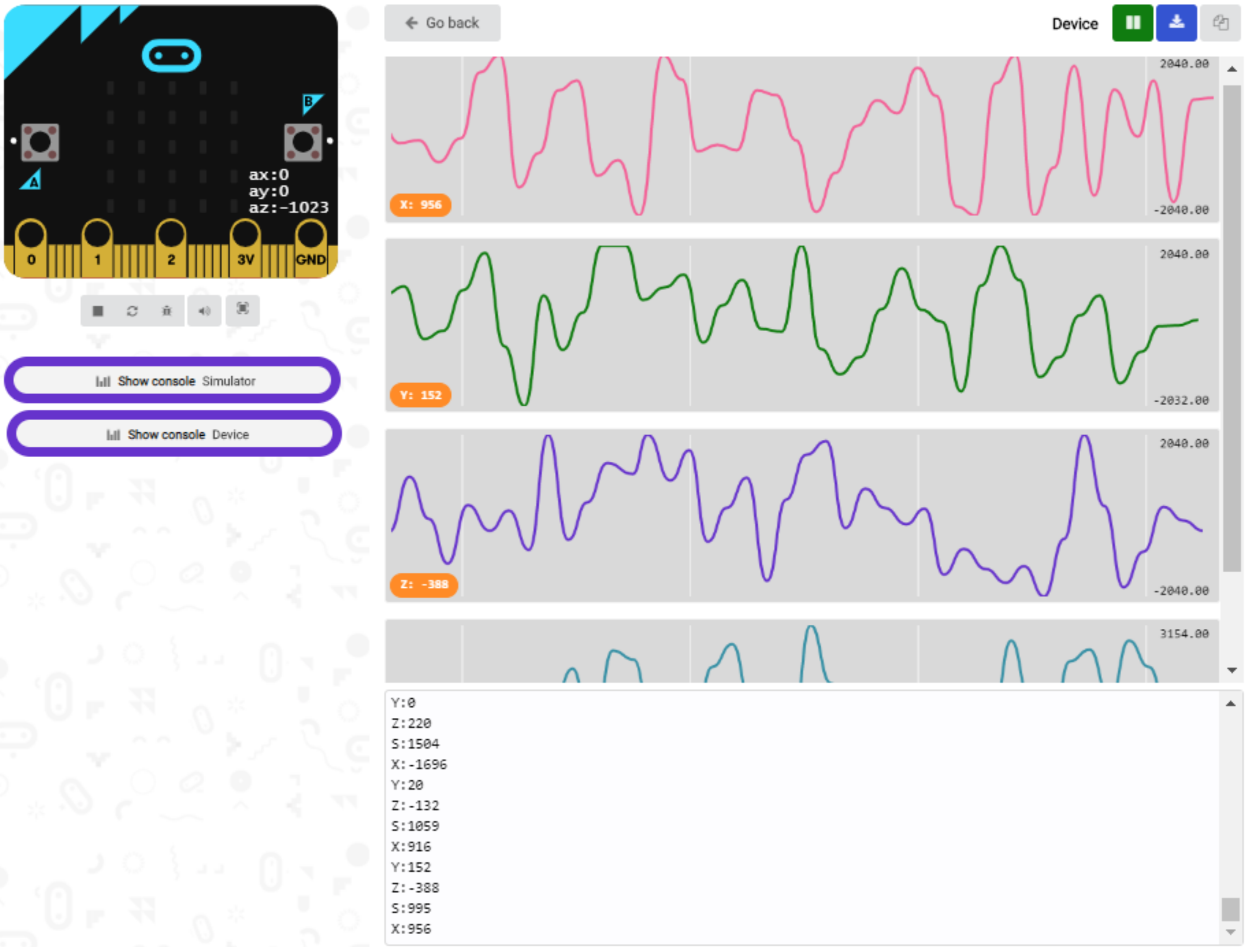


After referring to the MMA8653FC data manual and the hardware schematic diagram of the Micro: Bit main board V2, the accelerometer coordinate of the Micro: Bit V2 motherboard are shown in the figure below:





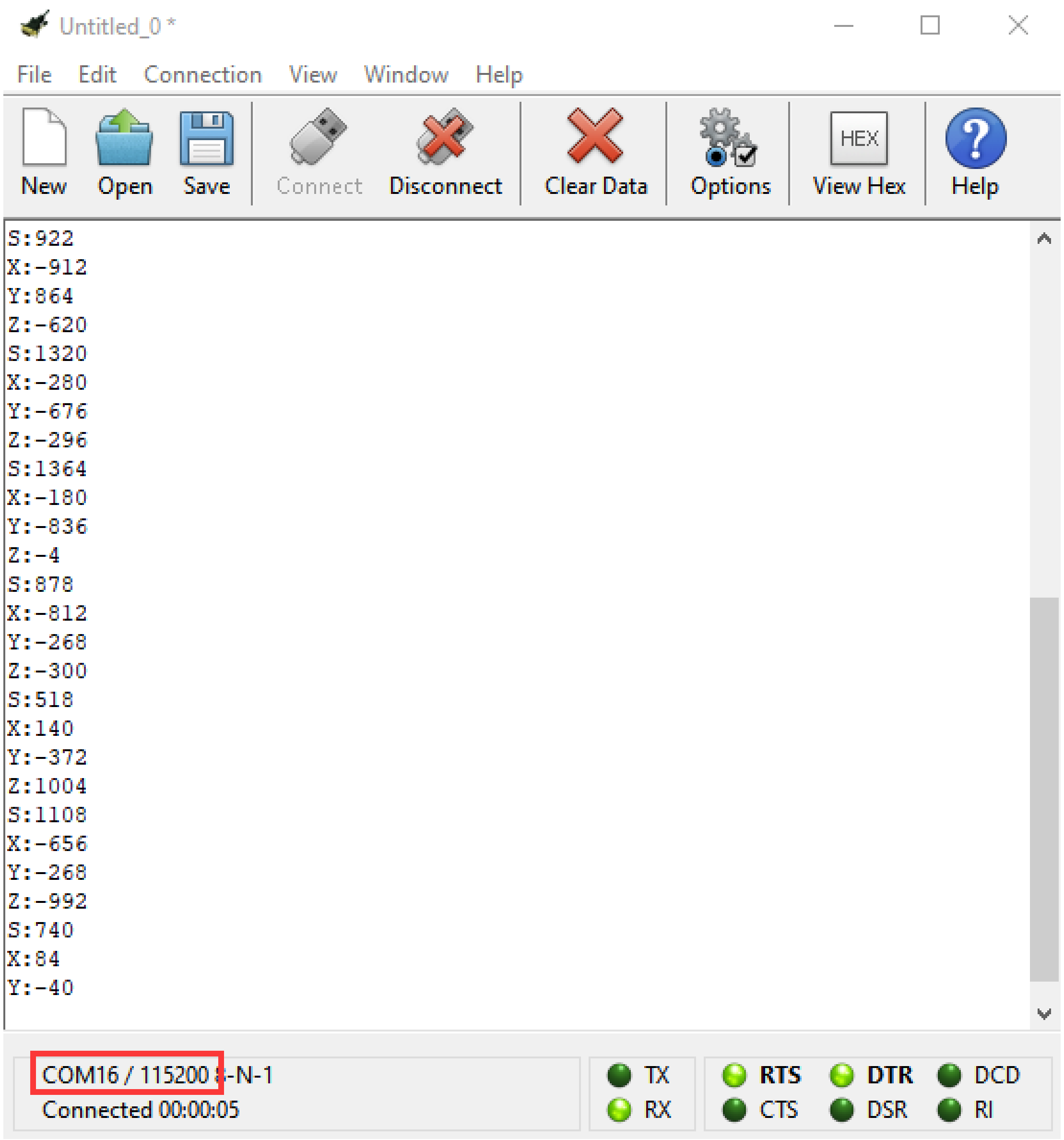
The following interface shows the decomposition value of acceleration in X axis, Y axis and Z axis respectively, as well as acceleration synthesis (acceleration synthesis of gravity and other external forces).



```
Y:0
Z:220
S:1504
X:-1696
Y:20
Z:-132
S:1059
X:916
Y:152
Z:-388
S:995
X:956
```

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.

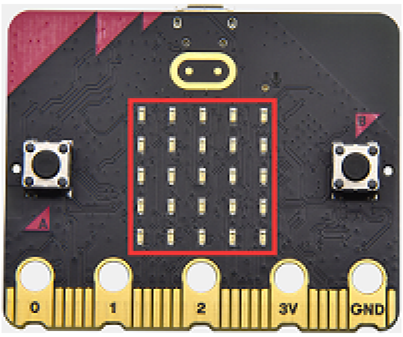
You could open CoolTerm software, click Options, select SerialPort, set COM port and baud rate to 115200 (after testing, the baud rate of USB SerialPort communication on Micro: Bit main board V2 is 115200), click OK, and Connect. The CoolTerm serial monitor shows the data of X axis, Y axis and Z axis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



```
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 8-N-1
Connected 00:00:05

7.8 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.

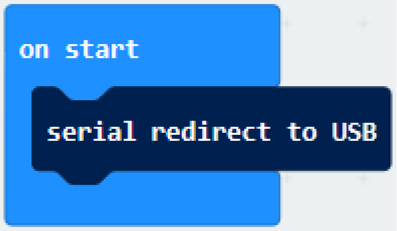
2. Components Needed:

- 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”;

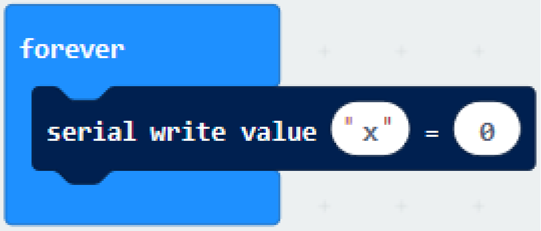


```
on start
  serial redirect to USB
```

B. Drag it into“on start”block.

****

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

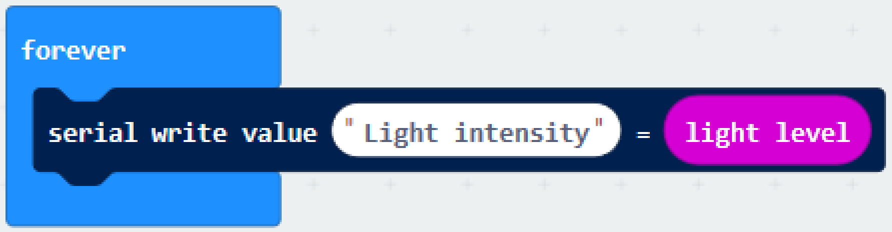


```
forever
  serial write value "x" = 0
```

B. Move it into“forever”

A. Click“Input”→“acceleration(mg) x”

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

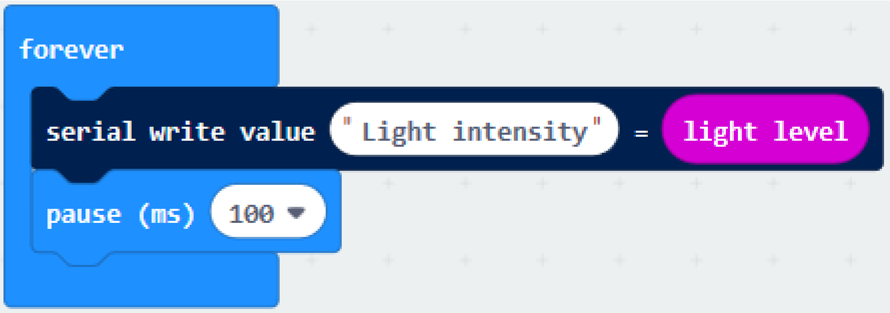


```
forever
  serial write value "Light intensity" = light level
```

****

A. Click“Basic”→“pause (ms) 100”;

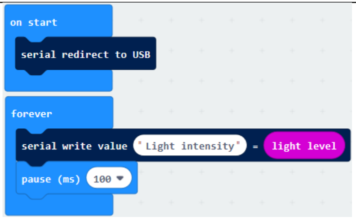
B. Lay it down into“forever”and set to 100ms.



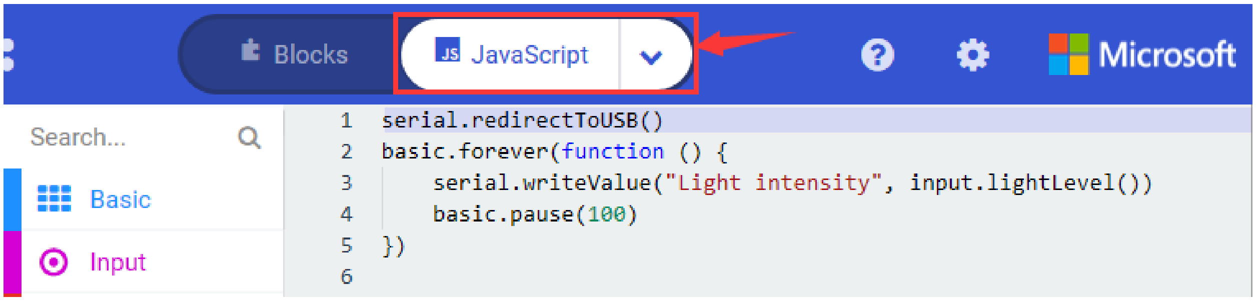
```
forever
    serial write value " Light intensity " = light level
    pause (ms) 100 ▾
```

****

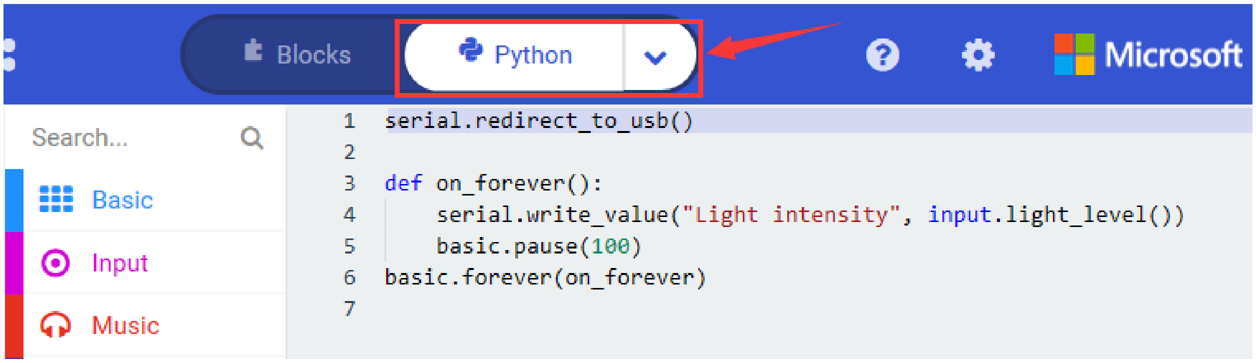
Complete Program

 on start serial redirect to USB forever serial write value " Light intensity " = light level pause (ms) 100 ▾	"on start": command block runs once to start program.Serial redirects to USBThe program under the block "forever" runs cyclically.Serial write value "Light intensity"= light levelDelay in 100ms

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



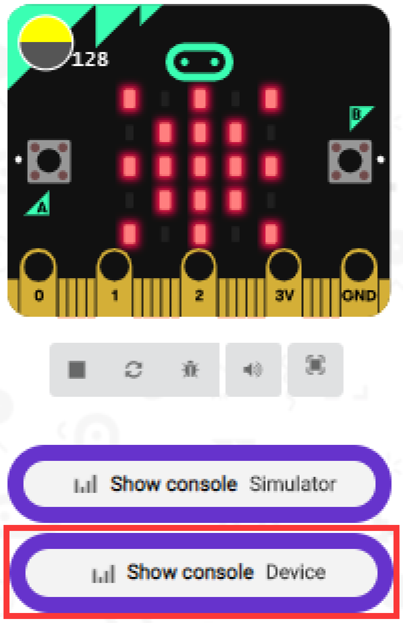
```
serial.redirectToUSB()
basic.forever(function () {
    serial.writeValue("Light intensity", input.lightLevel())
    basic.pause(100)
})
```



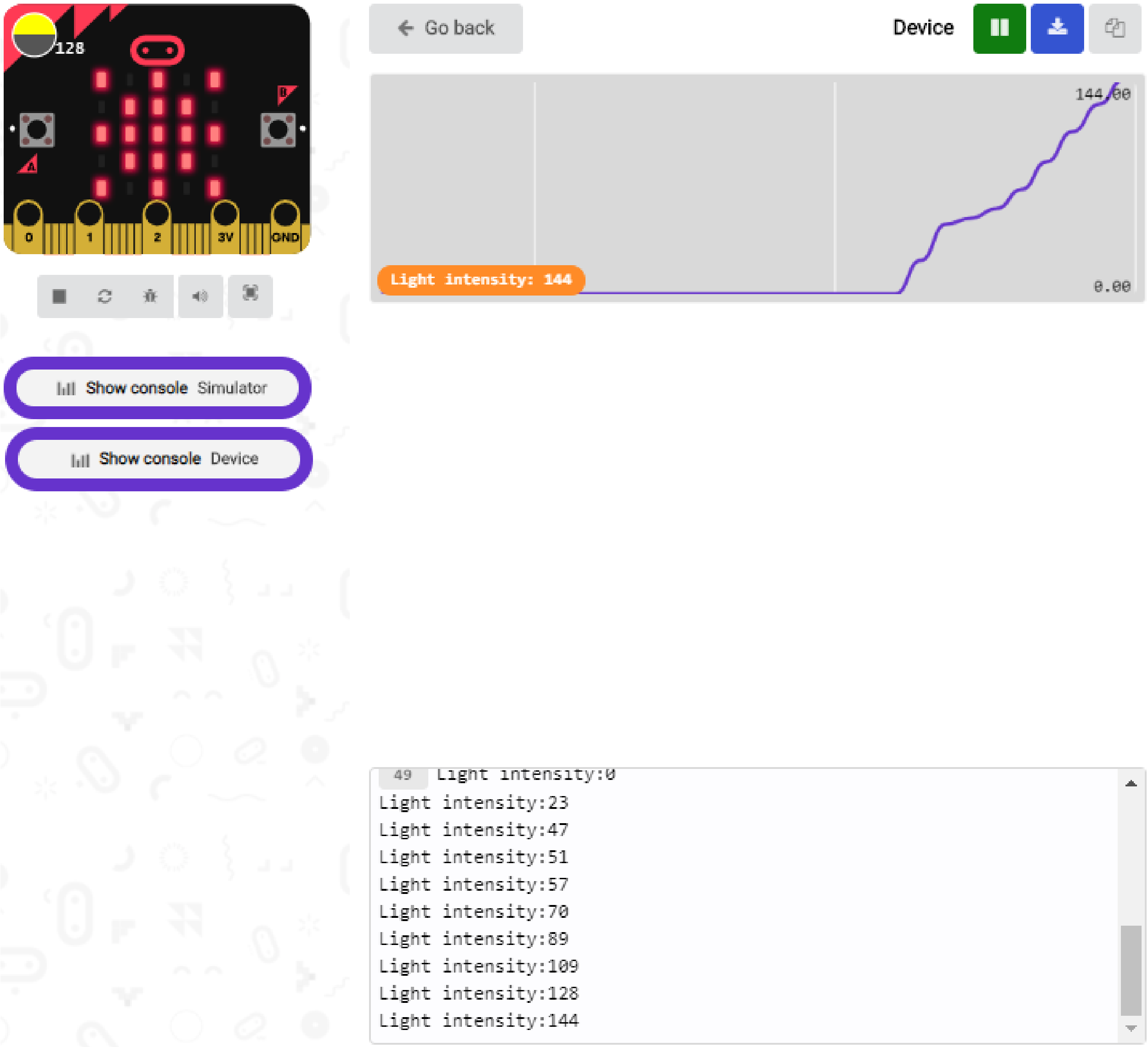
```
serial.redirect_to_usb()

def on_forever():
    serial.write_value("Light intensity", input.light_level())
    basic.pause(100)
basic.forever(on_forever)
```

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.

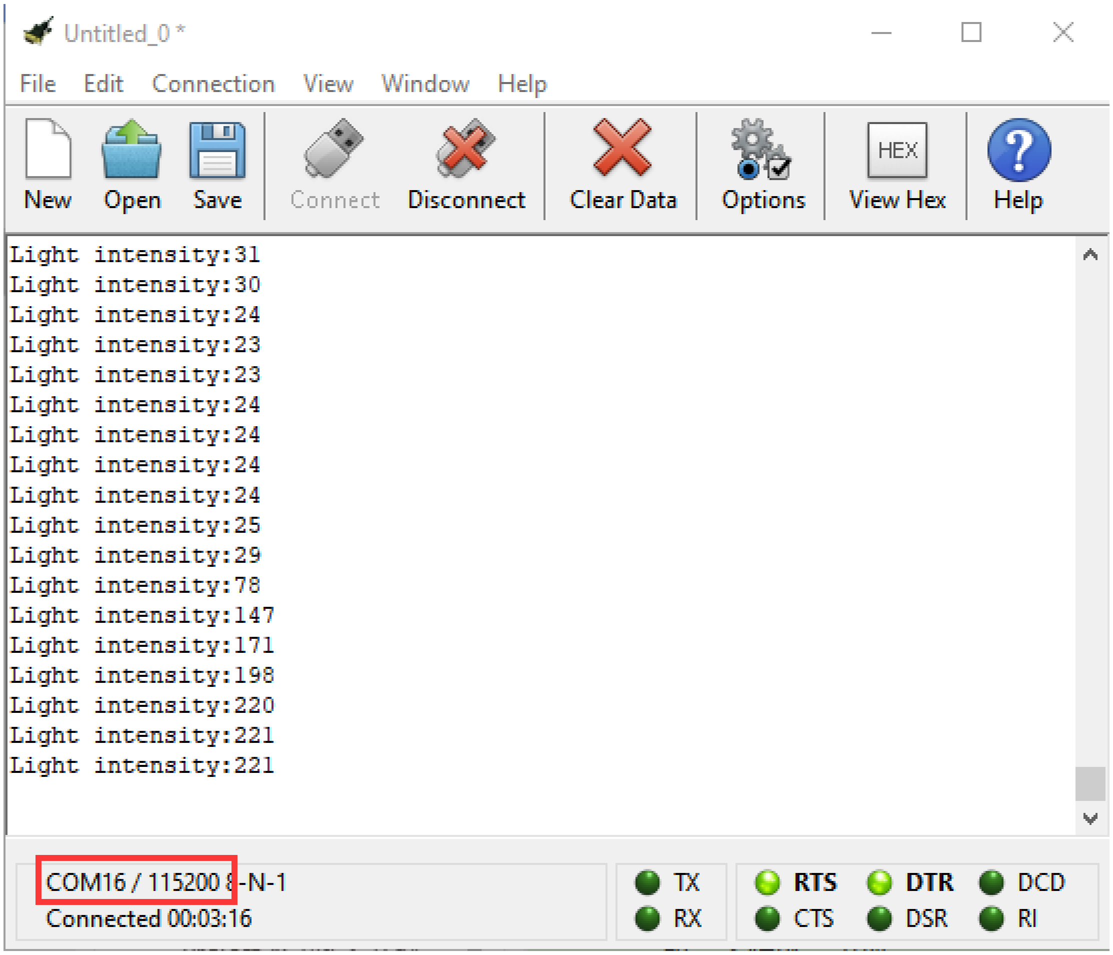


```
49  Light intensity:0
Light intensity:23
Light intensity:47
Light intensity:51
Light intensity:57
Light intensity:70
Light intensity:89
Light intensity:109
Light intensity:128
Light intensity:144
```

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.

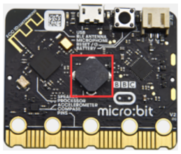
You could open CoolTerm software, click Options, select SerialPort, set COM port and baud rate to 115200 (after testing, the baud rate of USB SerialPort communication on Micro: Bit main board V2 is 115200), click OK, and Connect. The CoolTerm serial monitor shows the value of light intensity , as shown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



```
Light intensity:31
Light intensity:30
Light intensity:24
Light intensity:23
Light intensity:23
Light intensity:24
Light intensity:24
Light intensity:24
Light intensity:24
Light intensity:25
Light intensity:29
Light intensity:78
Light intensity:147
Light intensity:171
Light intensity:198
Light intensity:220
Light intensity:221
Light intensity:221
```

COM16 / 115200 8-N-1
Connected 00:03:16

7.9 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:

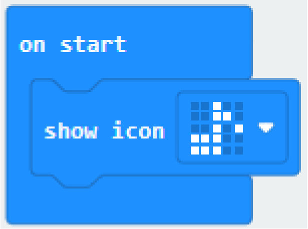
- Micro:bit main board V2 *1
- Micro USB cable*1

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 “”



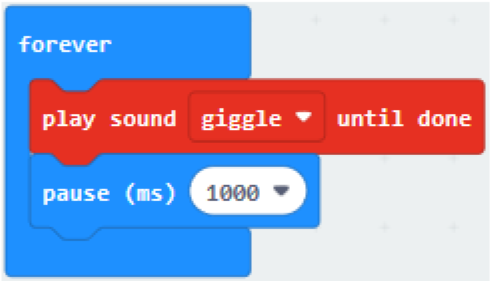
```
on start
  show icon
```

****

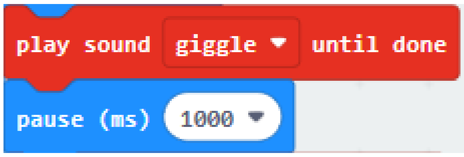
(2) Enter“Music”module to find and drug“play sound giggle until done”into“forever”block;

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

Change 100 into 1000;



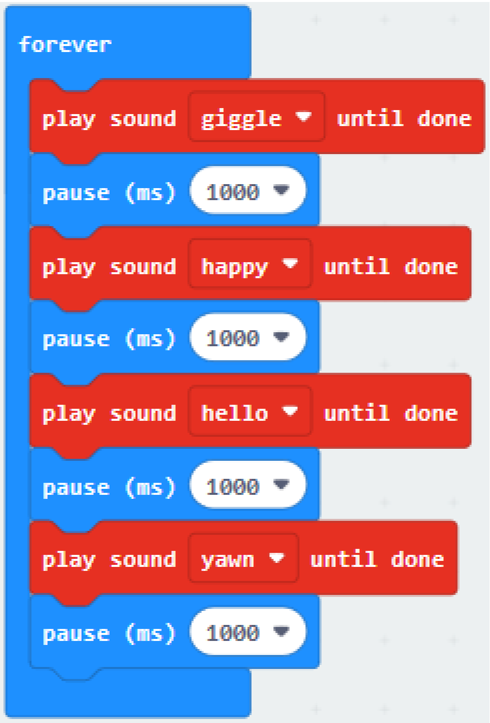
```
forever
  play sound giggle until done
  pause (ms) 1000
```



```
play sound giggle until done
pause (ms) 1000
```

(3) Copy three times and place it into “forever” block ;

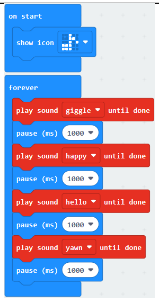
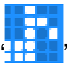
Click the little triangle to select“happy”,”hello”,”yawn”;



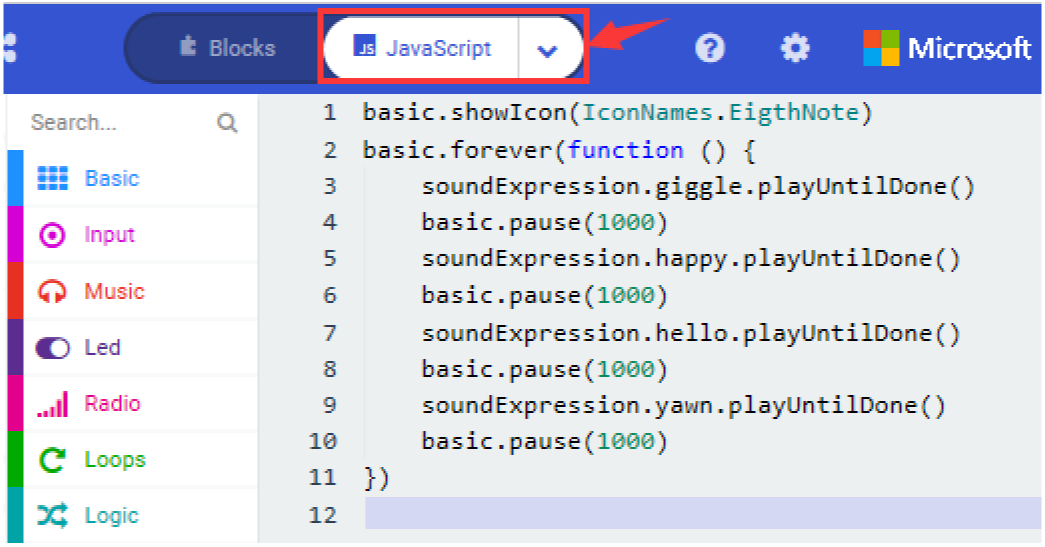
```
forever
    play sound giggle ▾ until done
    pause (ms) 1000 ▾
    play sound happy ▾ until done
    pause (ms) 1000 ▾
    play sound hello ▾ until done
    pause (ms) 1000 ▾
    play sound yawn ▾ until done
    pause (ms) 1000 ▾
```

****

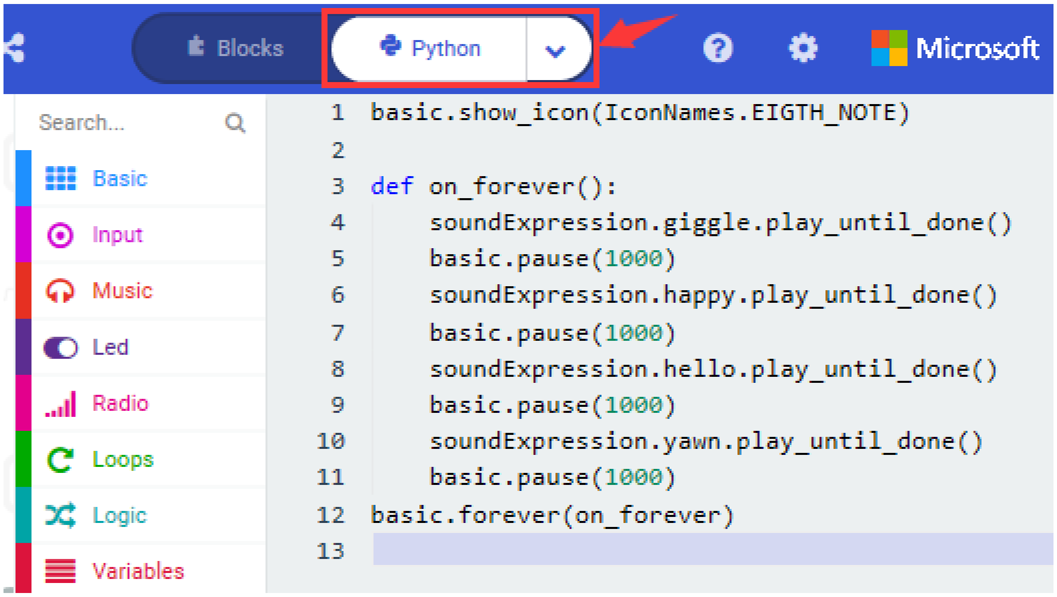
Complete Program

 on start show icon ▾ forever play sound giggle ▾ until done pause (ms) 1000 ▾ play sound happy ▾ until done pause (ms) 1000 ▾ play sound hello ▾ until done pause (ms) 1000 ▾ play sound yawn ▾ until done pause (ms) 1000 ▾	In “on start”the program only runs once;LED dot matrix displays pattern“”;In“forever”the program runs cyclically;The buzzer makes the sound“giggle”;Delay in 1000ms;The buzzer makes the sound“happy”;Delay in 1000ms;The buzzer makes the sound“hello”;Delay in 1000ms;The buzzer makes the sound“yawn”;Delay in 1000ms;

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



```
basic.showIcon(IconNames.EigthNote)
basic.forever(function () {
    soundExpression.giggle.playUntilDone()
    basic.pause(1000)
    soundExpression.happy.playUntilDone()
    basic.pause(1000)
    soundExpression.hello.playUntilDone()
    basic.pause(1000)
    soundExpression.yawn.playUntilDone()
    basic.pause(1000)
})
```

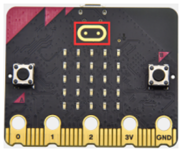


```
basic.show_icon(IconNames.EIGTH_NOTE)

def on_forever():
    soundExpression.giggle.play_until_done()
    basic.pause(1000)
    soundExpression.happy.play_until_done()
    basic.pause(1000)
    soundExpression.hello.play_until_done()
    basic.pause(1000)
    soundExpression.yawn.play_until_done()
    basic.pause(1000)
basic.forever(on_forever)
```

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.

7.10 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:

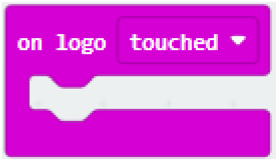
- Micro:bit main board V2 *1
- Micro USB cable*1

3. Test Code:

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

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

(2)Enter“Input”module to find and drag“on logo pressed” ; Click the little triangle to find “touched”';

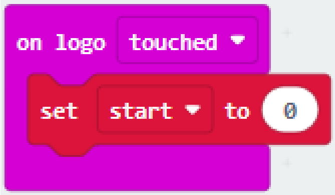


```
on logo touched ▾
```

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

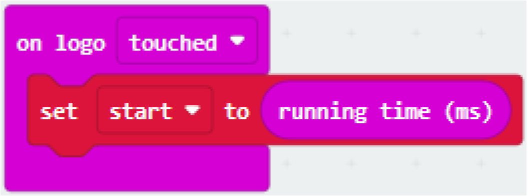
The variable“start”is established;

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



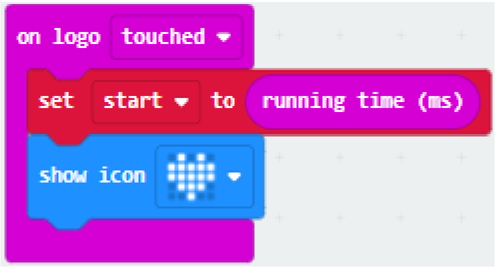
```
on logo touched ▾
  set start ▾ to 0
```

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



```
on logo touched ▾
  set start ▾ to running time (ms)
```

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

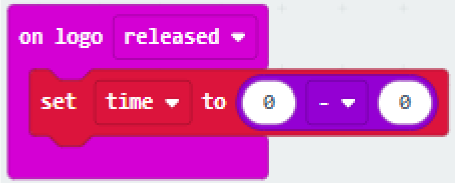


```
on logo touched ▾
  set start ▾ to running time (ms)
  show icon ▾
```

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

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

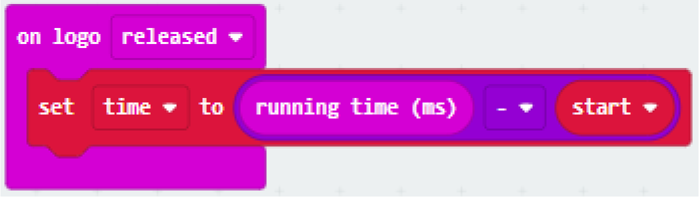
(8)Enter“Math”module to find and drag “0-0”into the “0”of“set start to 0”block;



```
on logo released ▾
    set time ▾ to 0 - ▾ 0
```

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

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

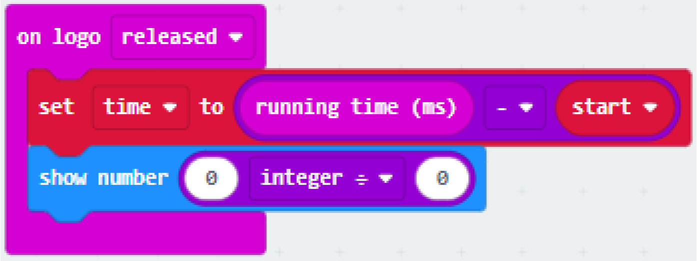


```
on logo released ▾
    set time ▾ to running time (ms) - ▾ start ▾
```

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

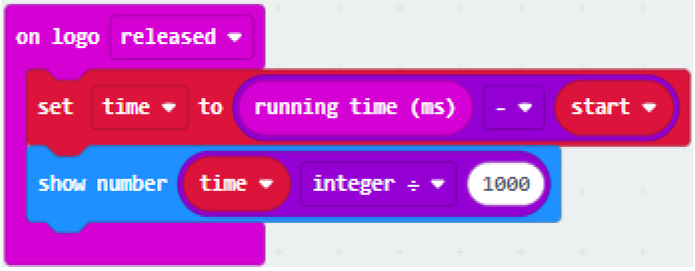
(12)Enter“Math”module to find and drag“square root 0” into “0”;

(13)Click the little triangle to find”integer÷”;



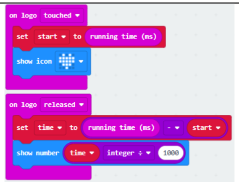
```
on logo released ▾
    set time ▾ to running time (ms) - ▾ start ▾
    show number 0 integer ÷ ▾ 0
```

(14)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”;

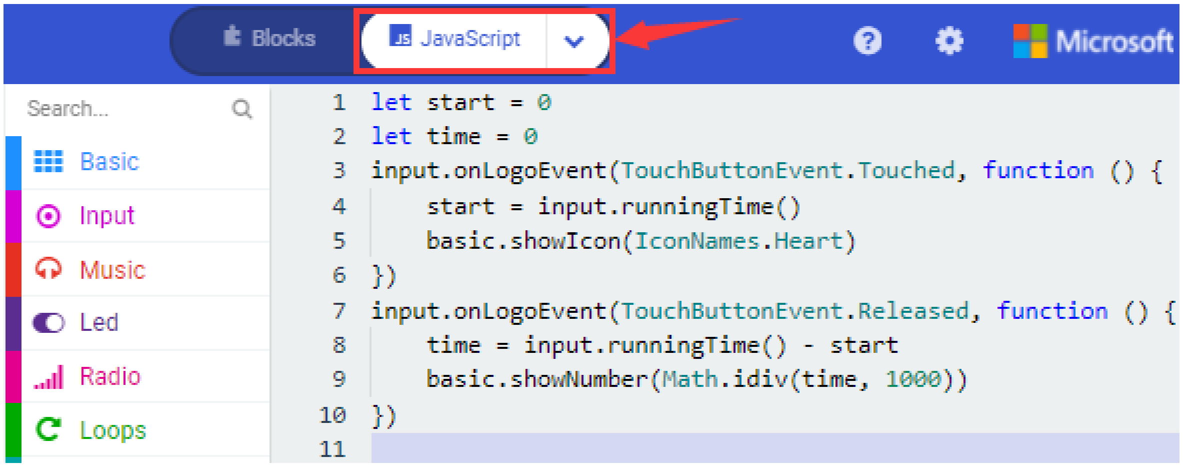


```
on logo released ▾
    set time ▾ to running time (ms) - ▾ start ▾
    show number time ▾ integer ÷ ▾ 1000
```

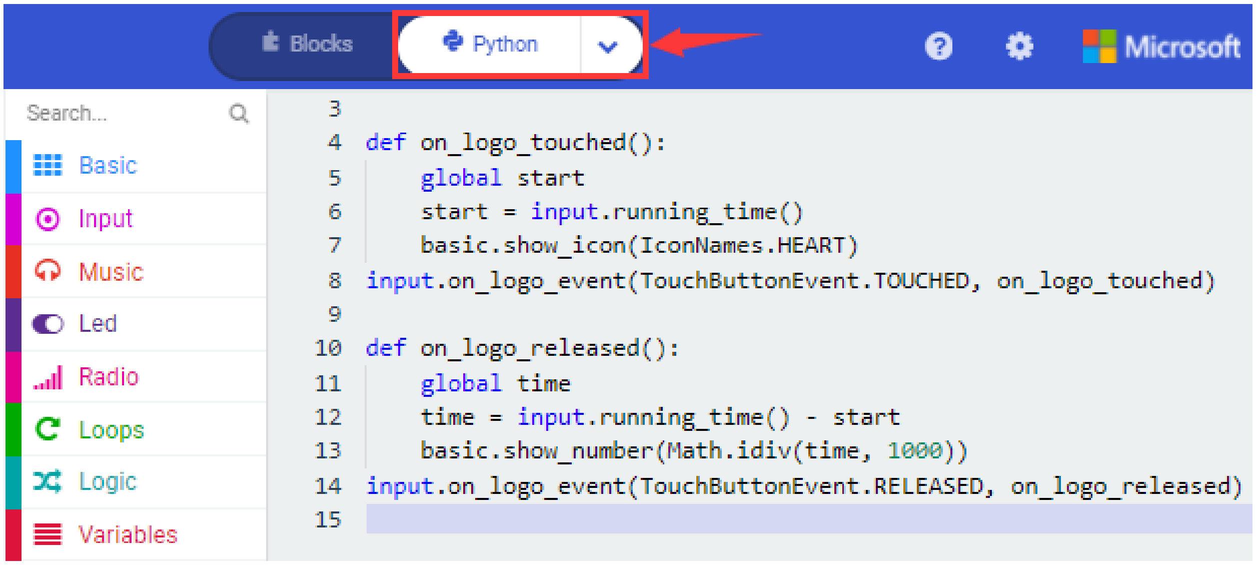
Complete Program

 on logo touched ▾ set start ▾ to running time (ms) show icon ♥ ▾ on logo released ▾ set time ▾ to running time (ms) - ▾ start ▾ show number time ▾ integer ÷ ▾ 1000	Touch the logo on the micro:bit with hand;Assign “running time” to variable”start”;LED dot matrix displays pattern“♥”;Put your hand away from the logo;Assign “running time” to variable “time”LED dot matrix displays the integer of variable “time”divided by 1000.

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



```
let start = 0
let time = 0
input.onLogoEvent(TouchButtonEvent.Touched, function () {
    start = input.runningTime()
    basic.showIcon(IconNames.Heart)
})
input.onLogoEvent(TouchButtonEvent.Released, function () {
    time = input.runningTime() - start
    basic.showNumber(Math.idiv(time, 1000))
})
```



```

def on_logo_touched():
    global start
    start = input.running_time()
    basic.show_icon(IconNames.HEART)
input.on_logo_event(TouchButtonEvent.TOUCHED, on_logo_touched)

def on_logo_released():
    global time
    time = input.running_time() - start
    basic.show_number(Math.idiv(time, 1000))
input.on_logo_event(TouchButtonEvent.RELEASED, on_logo_released)
```

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.

7.11 Project 11: Microphone

1. Project Description:

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

2. Components Needed:

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

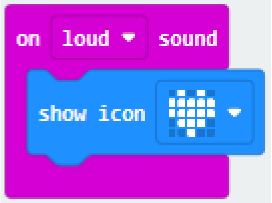
3. Test Code 1:

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

Delete block“on start”and“forever”;

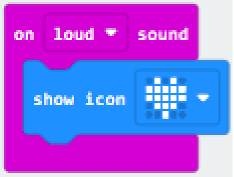
Enter“Input”module to find and drag“on loud sound”;

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



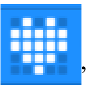
```
on loud ▾ sound
  show icon ▾
```

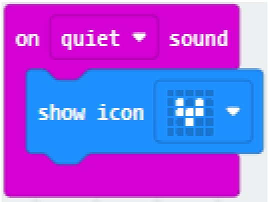
****

Copy  once;

```
on loud ▾ sound
  show icon ▾
```

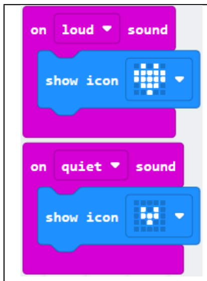
Click the little triangle of “lond” to choose”quiet”;

Click the little triangle of “” to choose” ”;

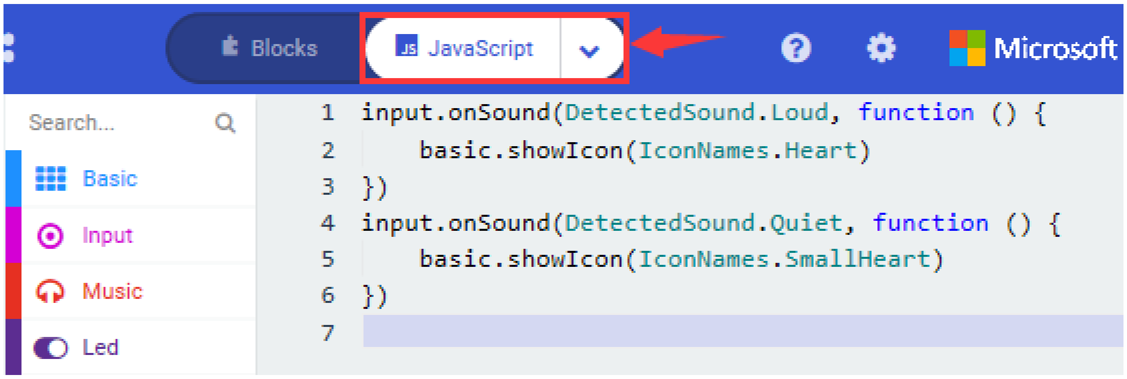


```
on quiet ▾ sound
    show icon ▾
```

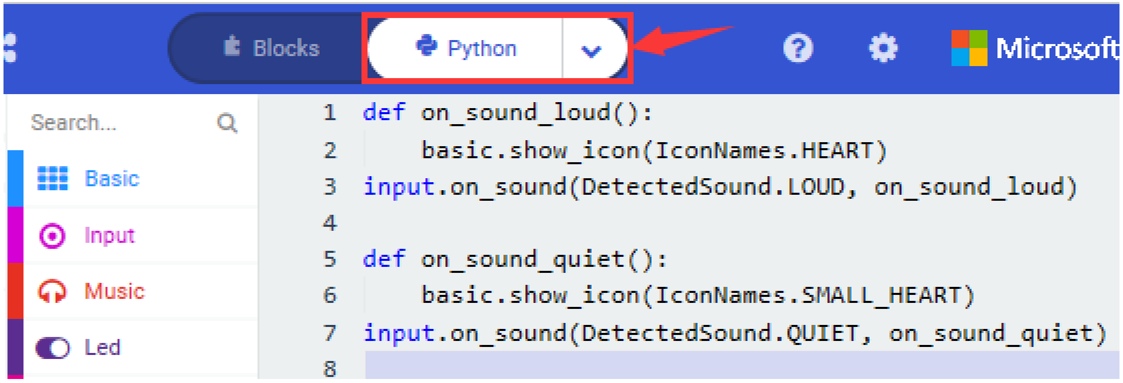
Complete Program

 on loud ▾ sound show icon ▾ on quiet ▾ sound show icon ▾	The microphone on the micro:bit detects sound;LED dot matrix displays pattern“ ”;The microphone on the micro:bit detects no sound;LED dot matrix displays pattern“ ”.

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

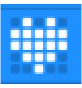


```
input.onSound(DetectedSound.Loud, function () {
    basic.showIcon(IconNames.Heart)
})
input.onSound(DetectedSound.Quiet, function () {
    basic.showIcon(IconNames.SmallHeart)
})
```



```
def on_sound_loud():
    basic.show_icon(IconNames.HEART)
input.on_sound(DetectedSound.LOUD, on_sound_loud)

def on_sound_quiet():
    basic.show_icon(IconNames.SMALL_HEART)
input.on_sound(DetectedSound.QUIET, on_sound_quiet)
```

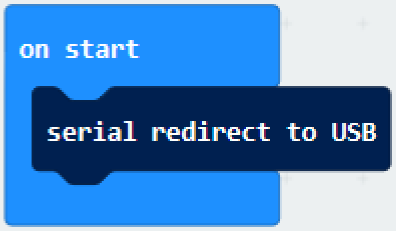
4. Test Results 1:

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 claps and pattern when it is quiet around.

5. Test Code 2:

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

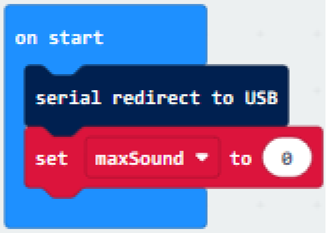
Enter“Advanced”module→ choose“Serial”to find and drag“serial redirect to USB”into “on start”block ;



```
on start
    serial redirect to USB
```

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

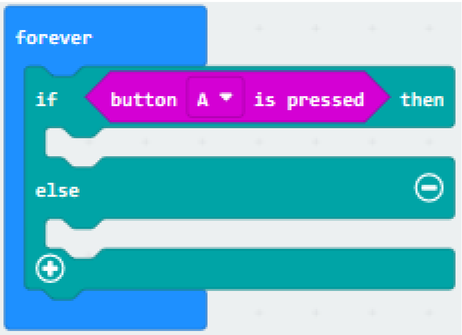
Enter“Variables”module to find and drag“set maxSound to 0”into “on start”block ;



```
on start
    serial redirect to USB
    set maxSound ▾ to 0
```

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

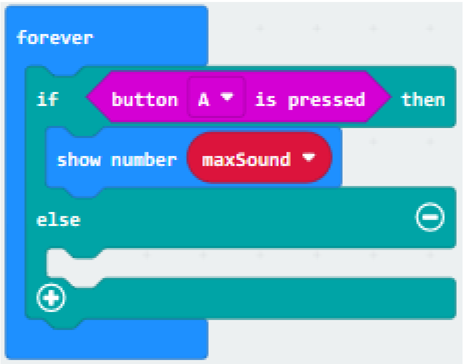
Enter“Input”module to find and dragbutton A is pressed”into“then” ;



```
forever
    if button A ▾ is pressed then

    else ⊖

    ⊕
```

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

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

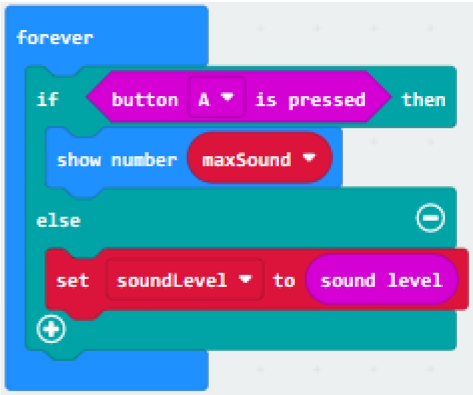


```
forever
  if button A is pressed then
    show number maxSound
  else

```

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”;

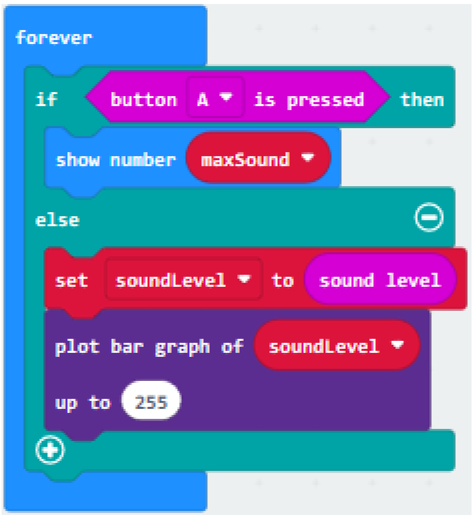


```
forever
  if button A is pressed then
    show number maxSound
  else
    set soundLevel to sound level
```

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”;



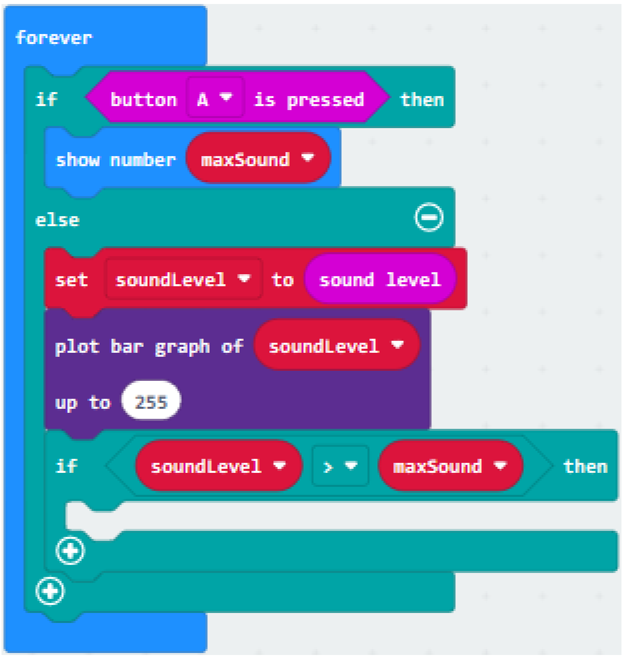
```
forever
  if button A is pressed then
    show number maxSound
  else
    set soundLevel to sound level
    plot bar graph of soundLevel
    up to 255
```

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” ;

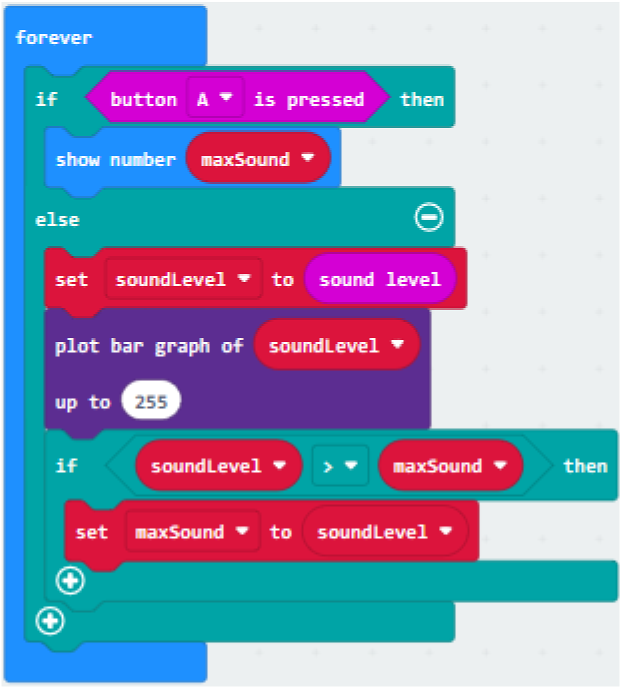
Enter“Variables”module to find and drag“maxSound”into“0”on the right side;



```
forever
  if button A is pressed then
    show number maxSound
  else
    set soundLevel to sound level
    plot bar graph of soundLevel
    up to 255
    if soundLevel > maxSound then
```

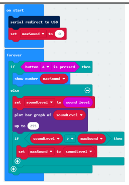
Enter“Variables”module to find and drag“set maxSound to 0”into the second “then”;

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

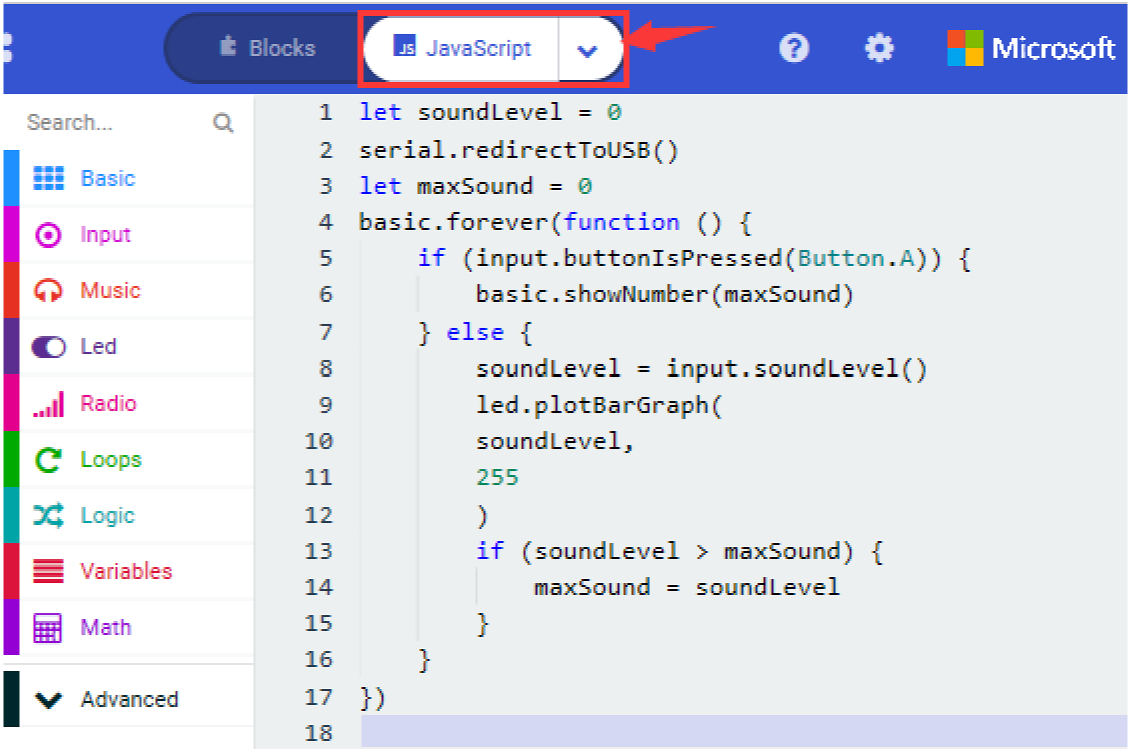


```
forever
  if button A is pressed then
    show number maxSound
  else
    set soundLevel to sound level
    plot bar graph of soundLevel
    up to 255
    if soundLevel > maxSound then
      set maxSound to soundLevel
```

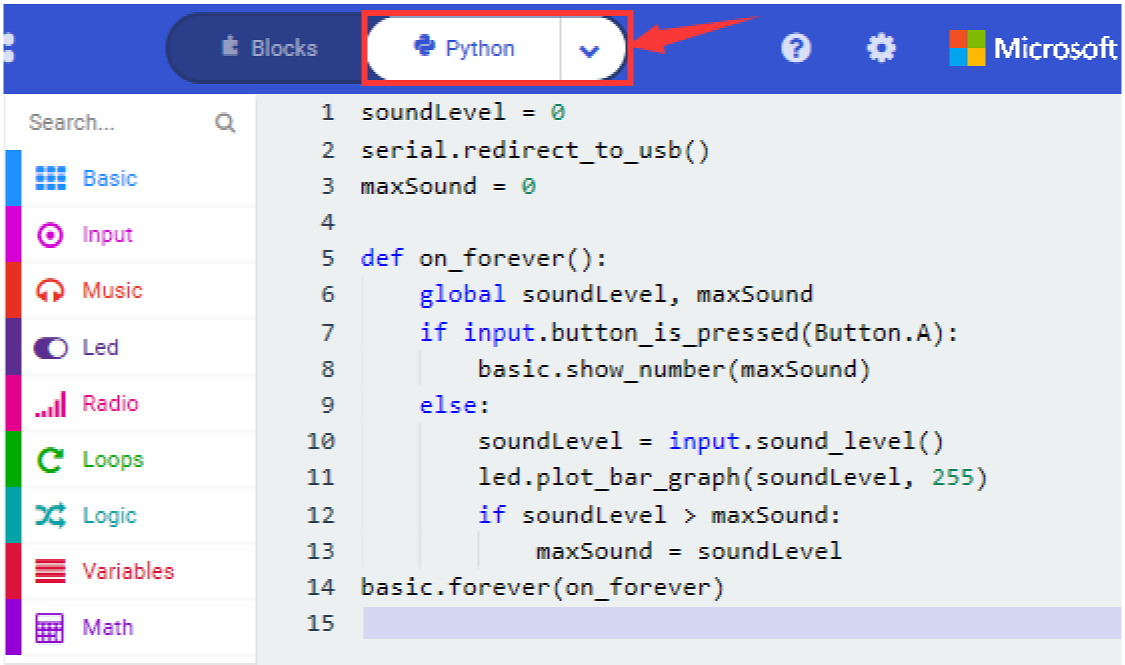
Complete Program

 on start serial redirect to USB set maxSound to 0 forever if button A is pressed then show number maxSound else set soundLevel to sound level plot bar graph of soundLevel up to 255 if soundLevel > maxSound then set maxSound to soundLevel	In “on start” the program only runs once;Redirect serial to USB;Set the initial value of variable “maxSound” to 0;In “forever” the program runs cyclically;When button A is pressed, the program in “then” runs;LED dot matrix displays the loudest sound value detected by the microphone sensor;When above conditions are not true, the program in “else” runs;Assign sound value to variable”soundLevel”;LED dot matrix displays the brightness of LED;and the brightest value is 25;If the sound value detected is bigger than the loudest one in ambient environment Then assign the sound value detected to variable”soundLevel”.

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



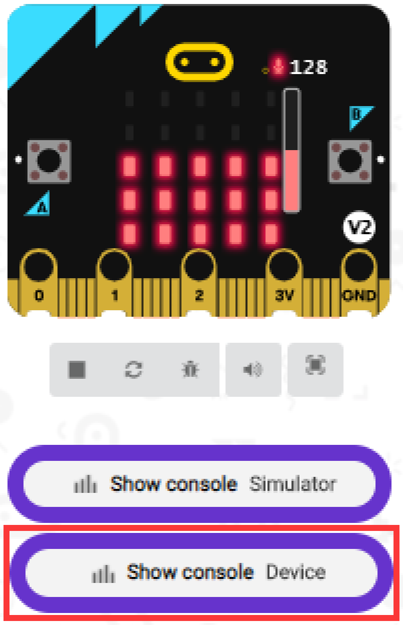
```
let soundLevel = 0
serial.redirectToUSB()
let maxSound = 0
basic.forever(function () {
    if (input.buttonIsPressed(Button.A)) {
        basic.showNumber(maxSound)
    } else {
        soundLevel = input.soundLevel()
        led.plotBarGraph(
        soundLevel,
        255
        )
        if (soundLevel > maxSound) {
            maxSound = soundLevel
        }
    }
})
```



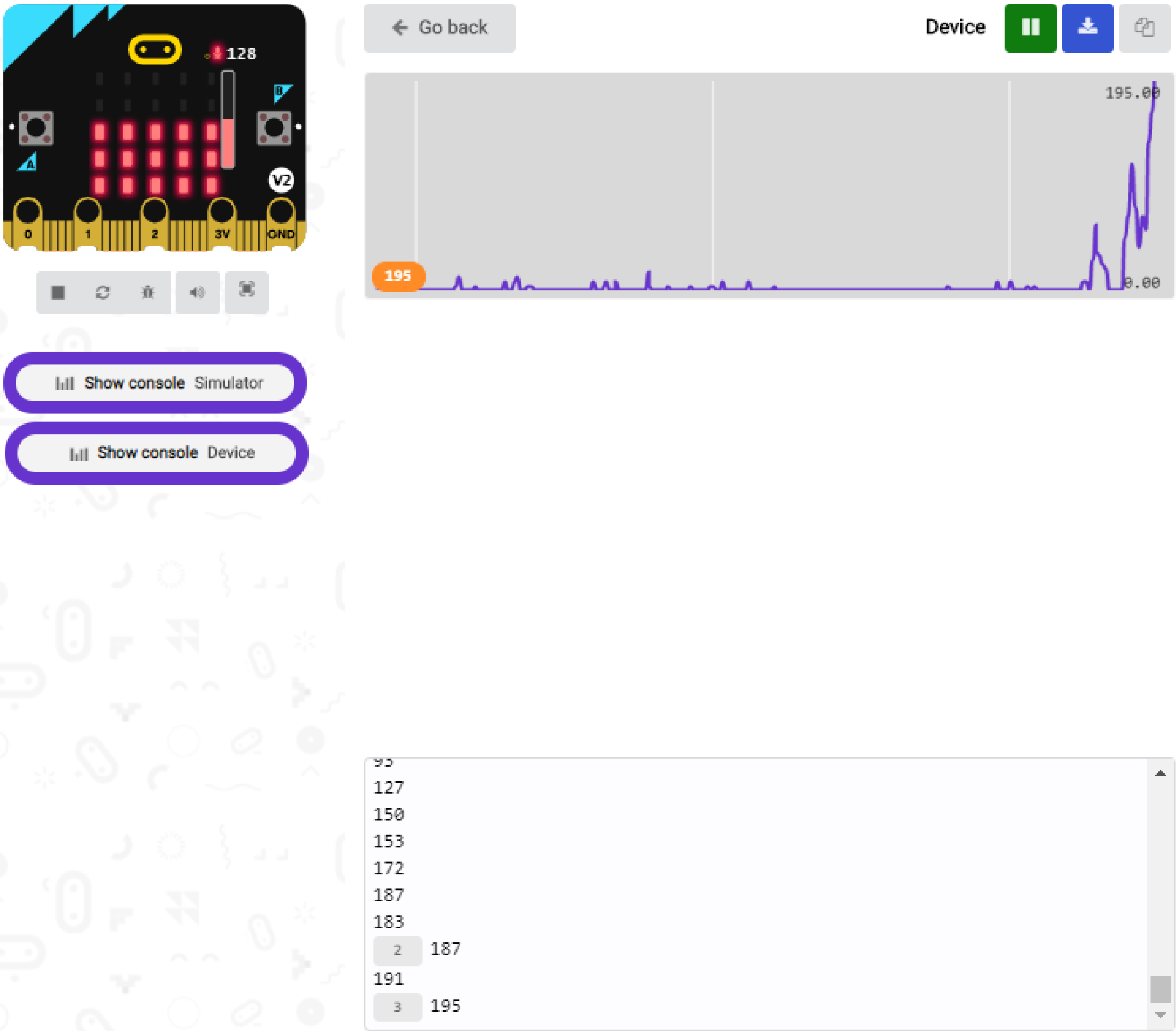
```python
soundLevel = 0
serial.redirect_to_usb()
maxSound = 0

def on_forever():
    global soundLevel, maxSound
    if input.button_is_pressed(Button.A):
        basic.show_number(maxSound)
    else:
        soundLevel = input.sound_level()
        led.plot_bar_graph(soundLevel, 255)
        if soundLevel > maxSound:
            maxSound = soundLevel
basic.forever(on_forever)
```

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.



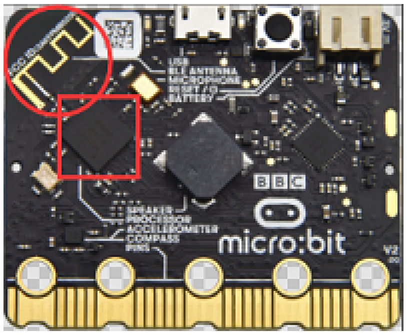
When the sound is louder around, the sound value shows in the serial port is bigger as shown below.



```
93
127
150
153
172
187
183
  2  187
191
  3  195
```

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

7.12 Project 12: Bluetooth Wireless Communication



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 Bluetooth devices, including smart phones and tablets.

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

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

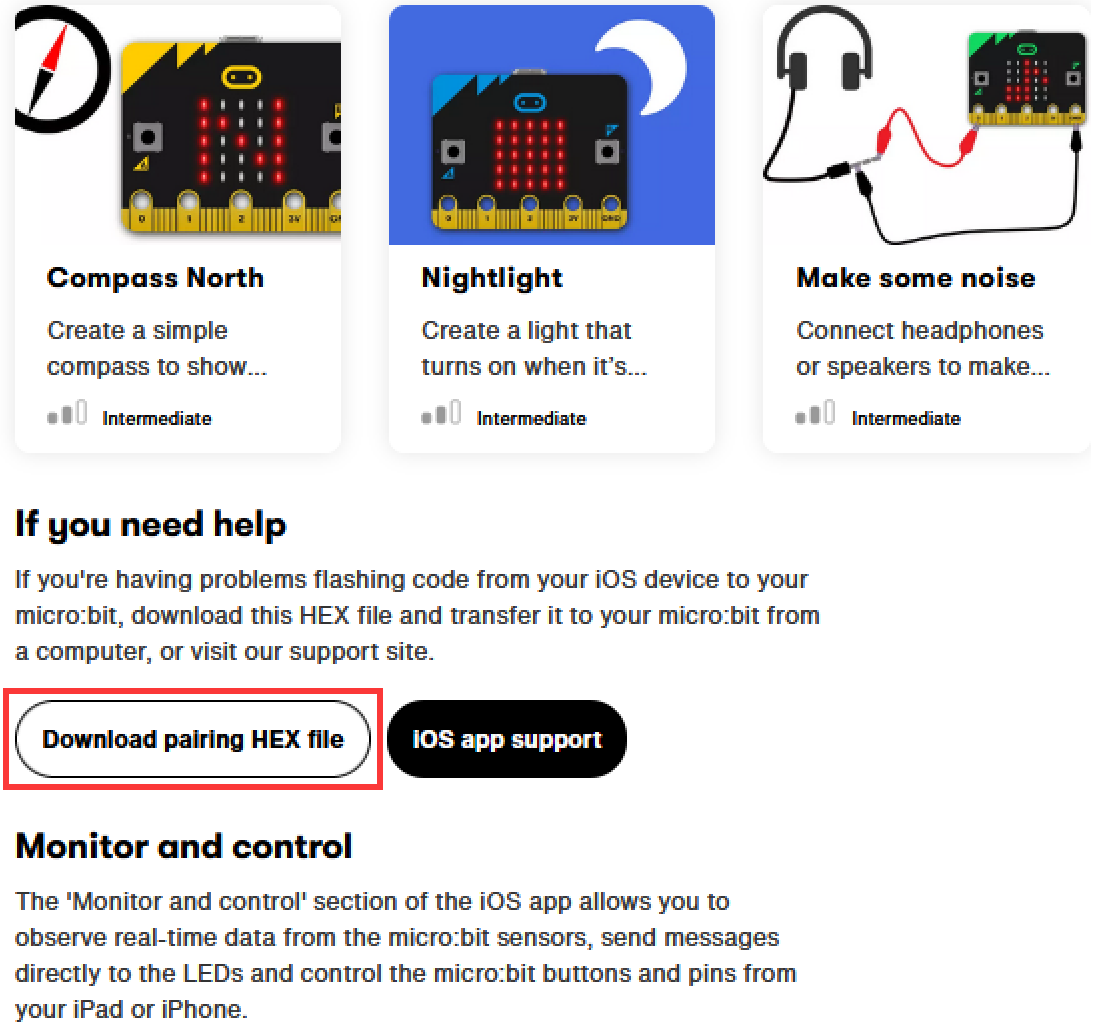
2. Preparation:

*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.



Compass North

Create a simple compass to show...

Intermediate

Nightlight

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

Intermediate

Make some noise

Connect headphones or speakers to make...

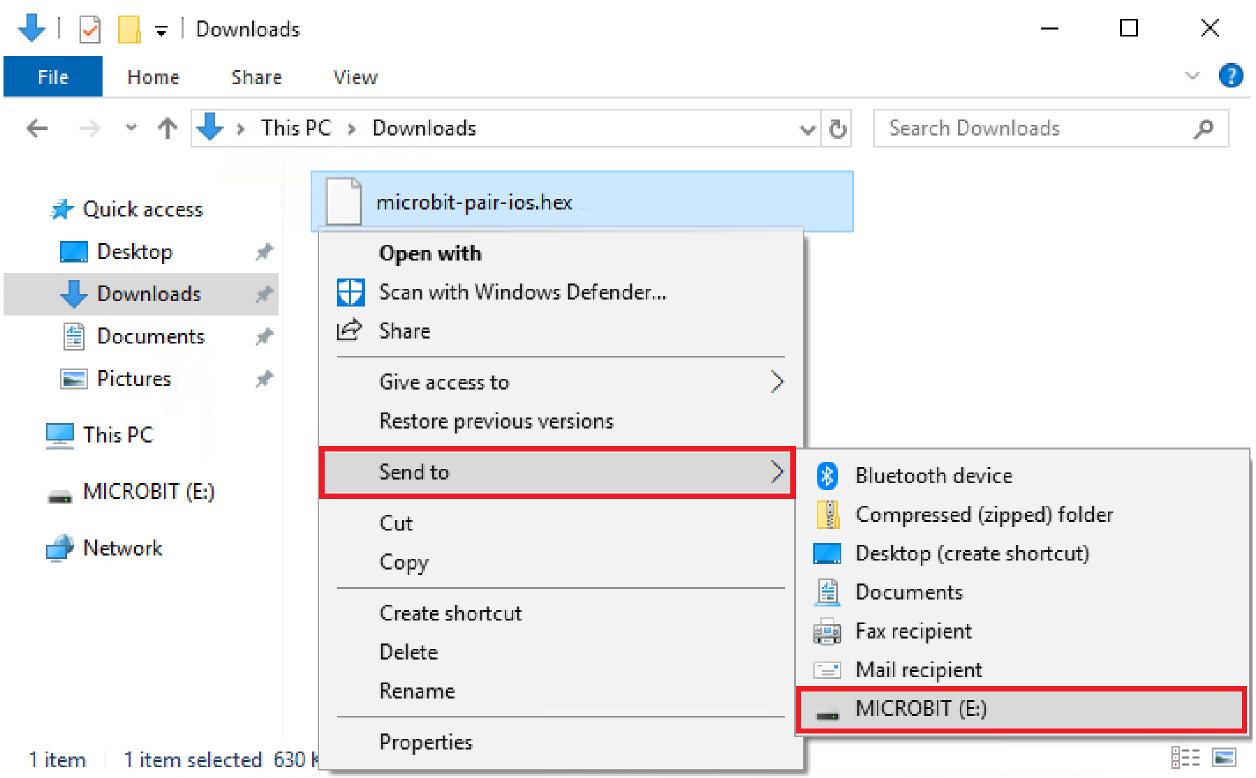
Intermediate

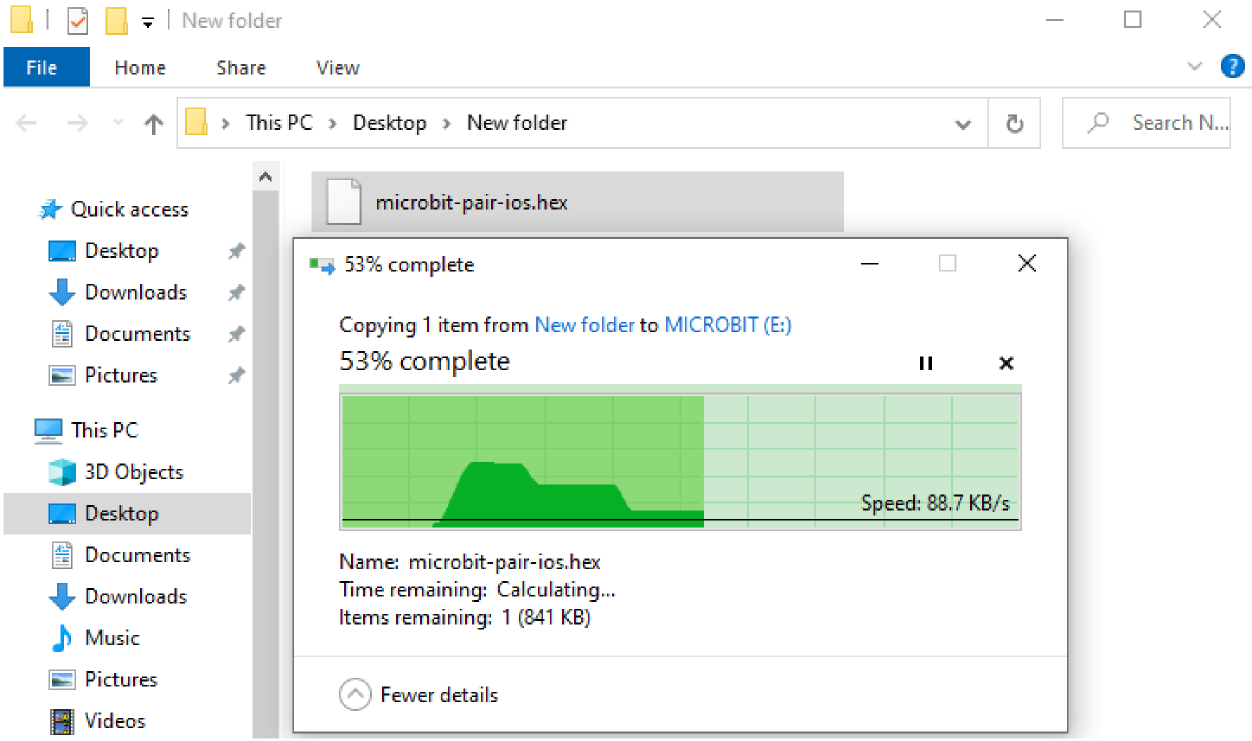
If you need help

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

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.



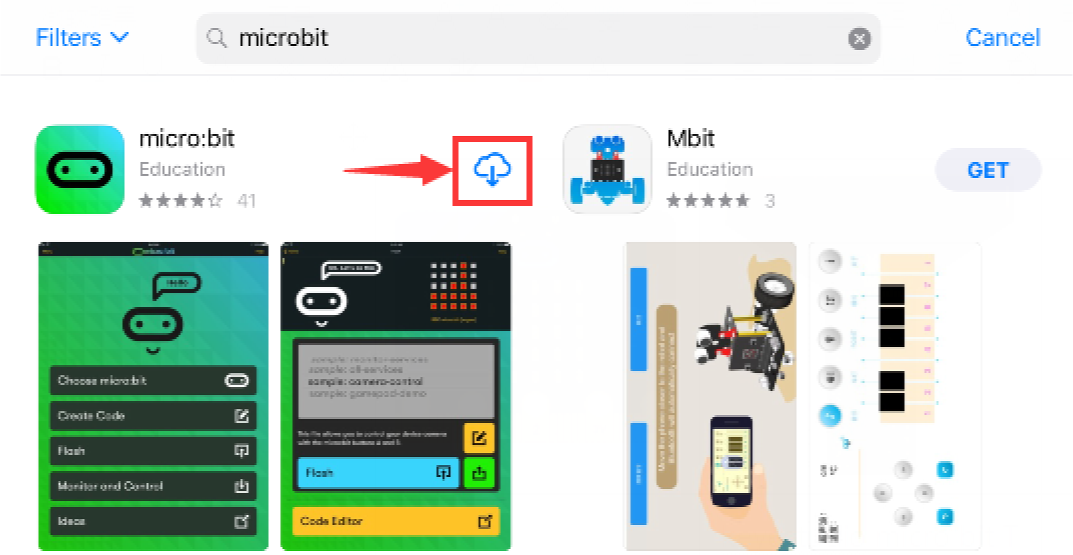


Copying 1 item from New folder to MICROBIT (E:)

53% complete

Name: microbit-pair-ios.hex
Time remaining: Calculating...
Items remaining: 1 (841 KB)

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

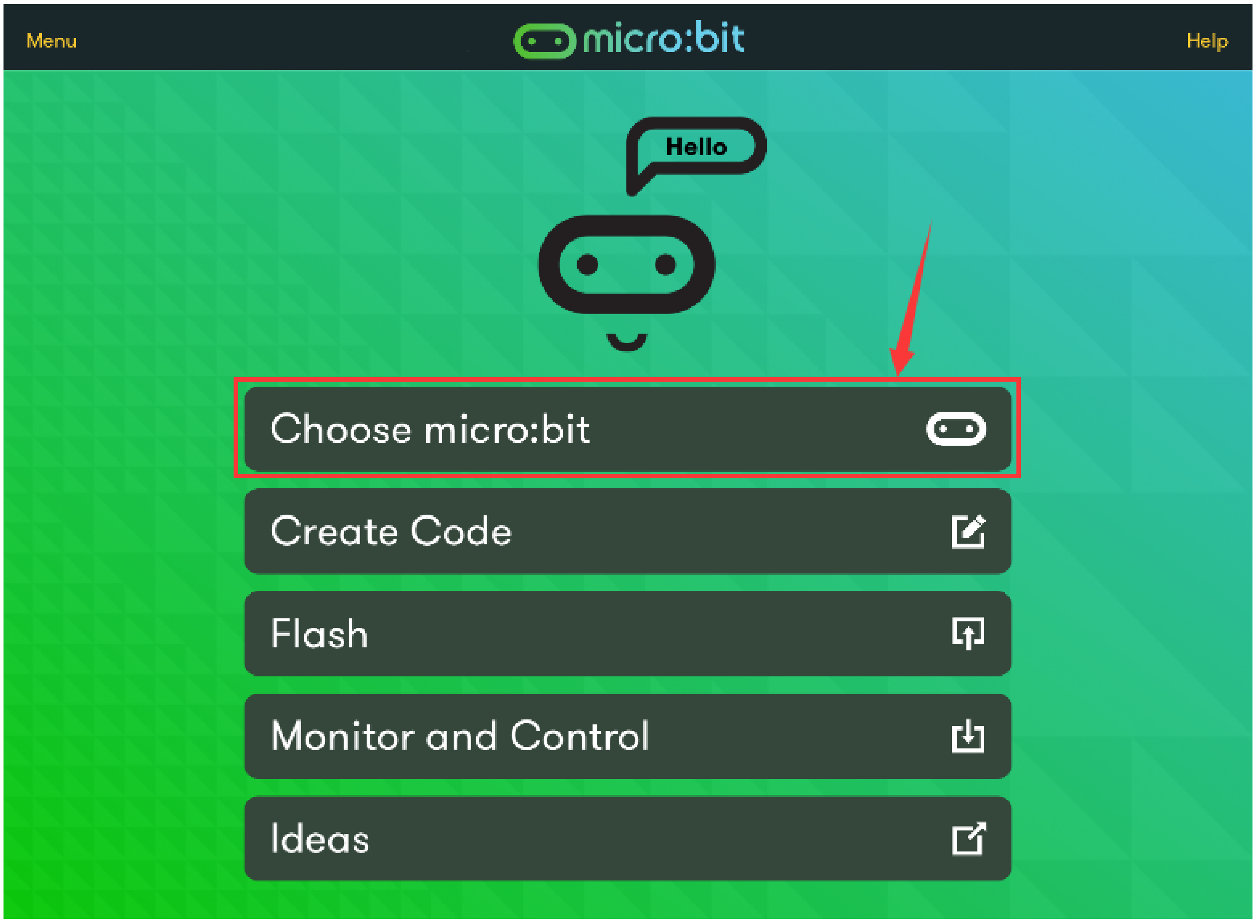


Wirelessly connect your iOS device to the micro: bit board V2:

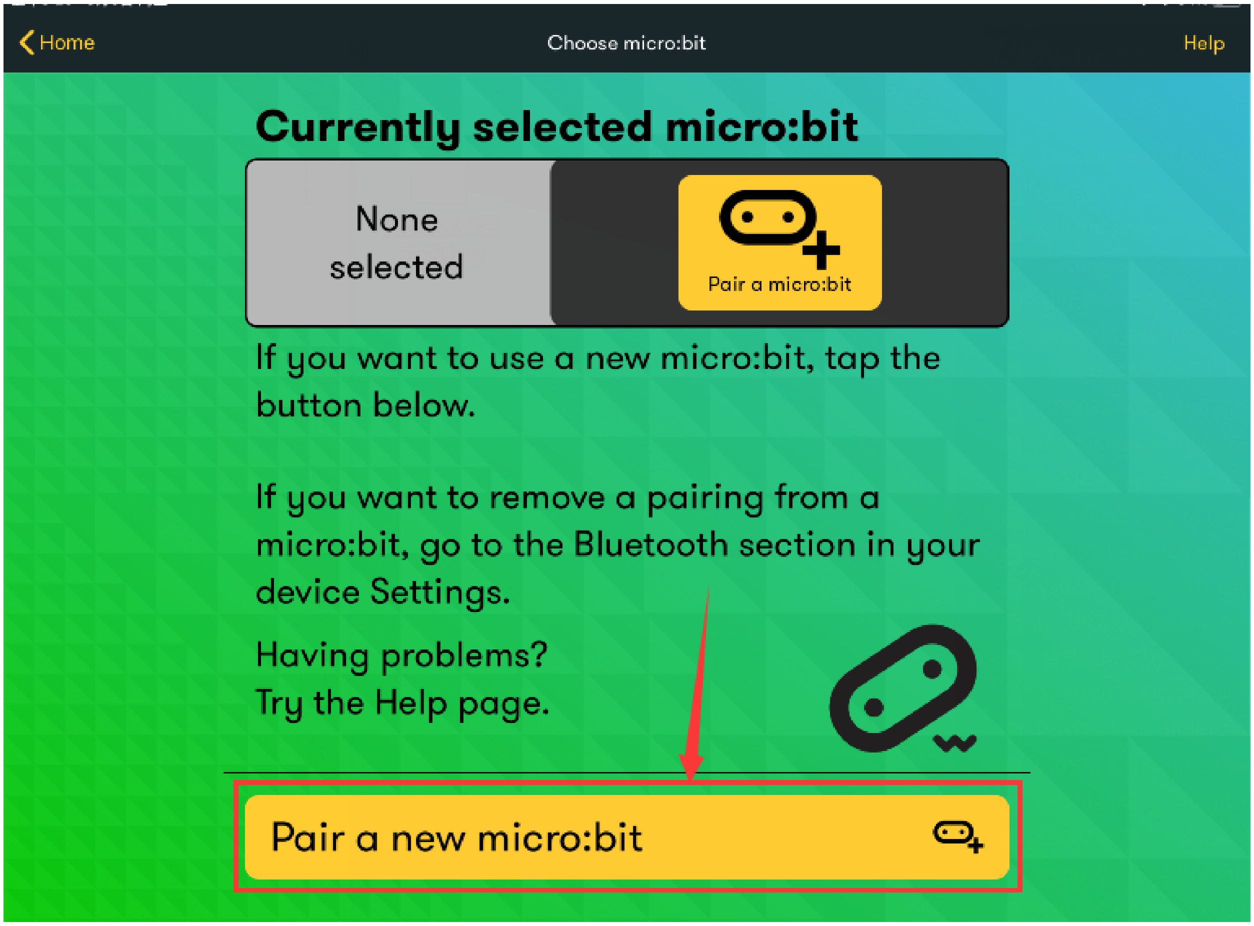
Firstly, turn on the Bluetooth of your iOS device and open

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

Select"Choose micro:bit"to start pairing Bluetooth.



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



Currently selected micro:bit

None selected

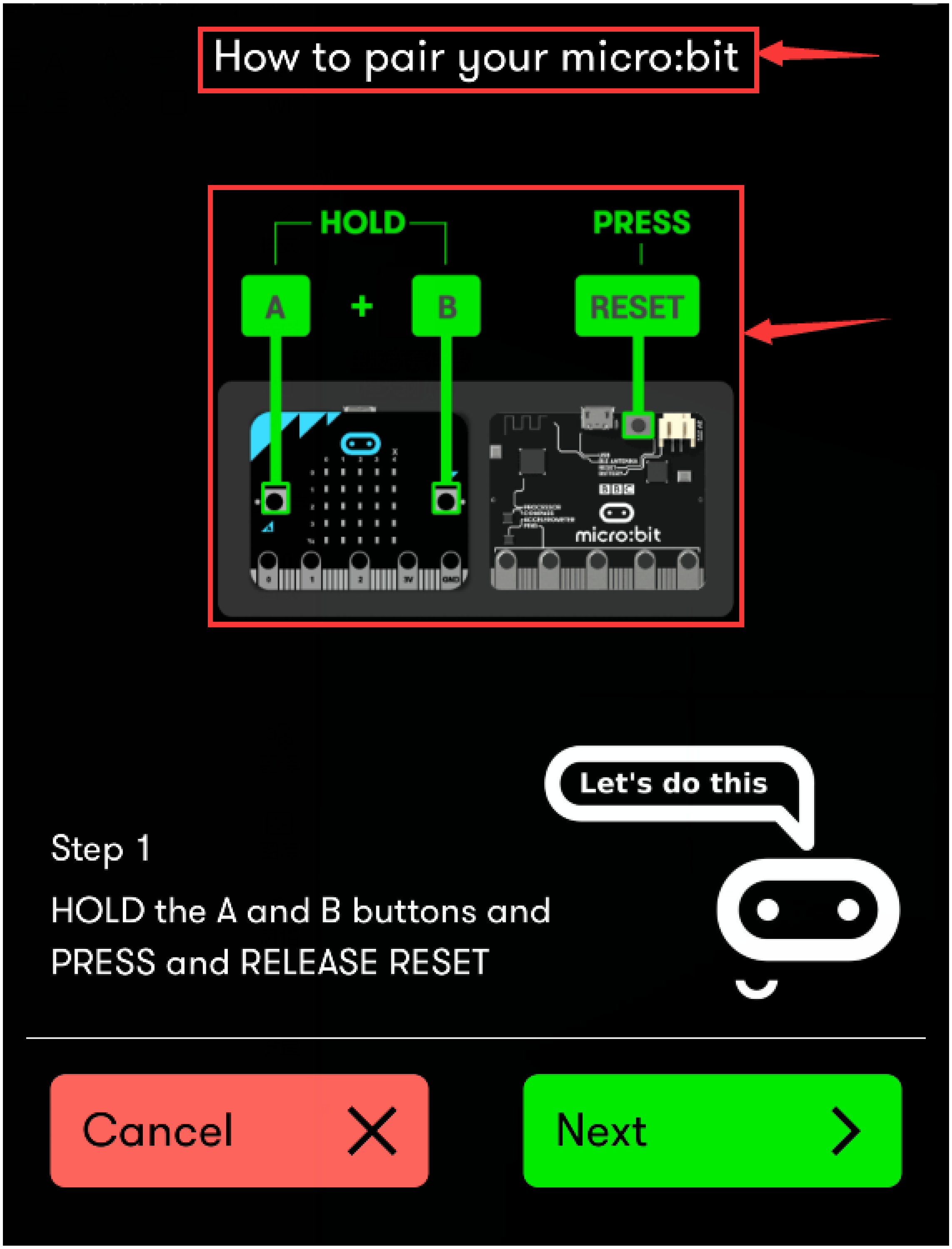
If you want to use a new micro:bit, tap the button below.

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

Having problems?
Try the Help page.

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.

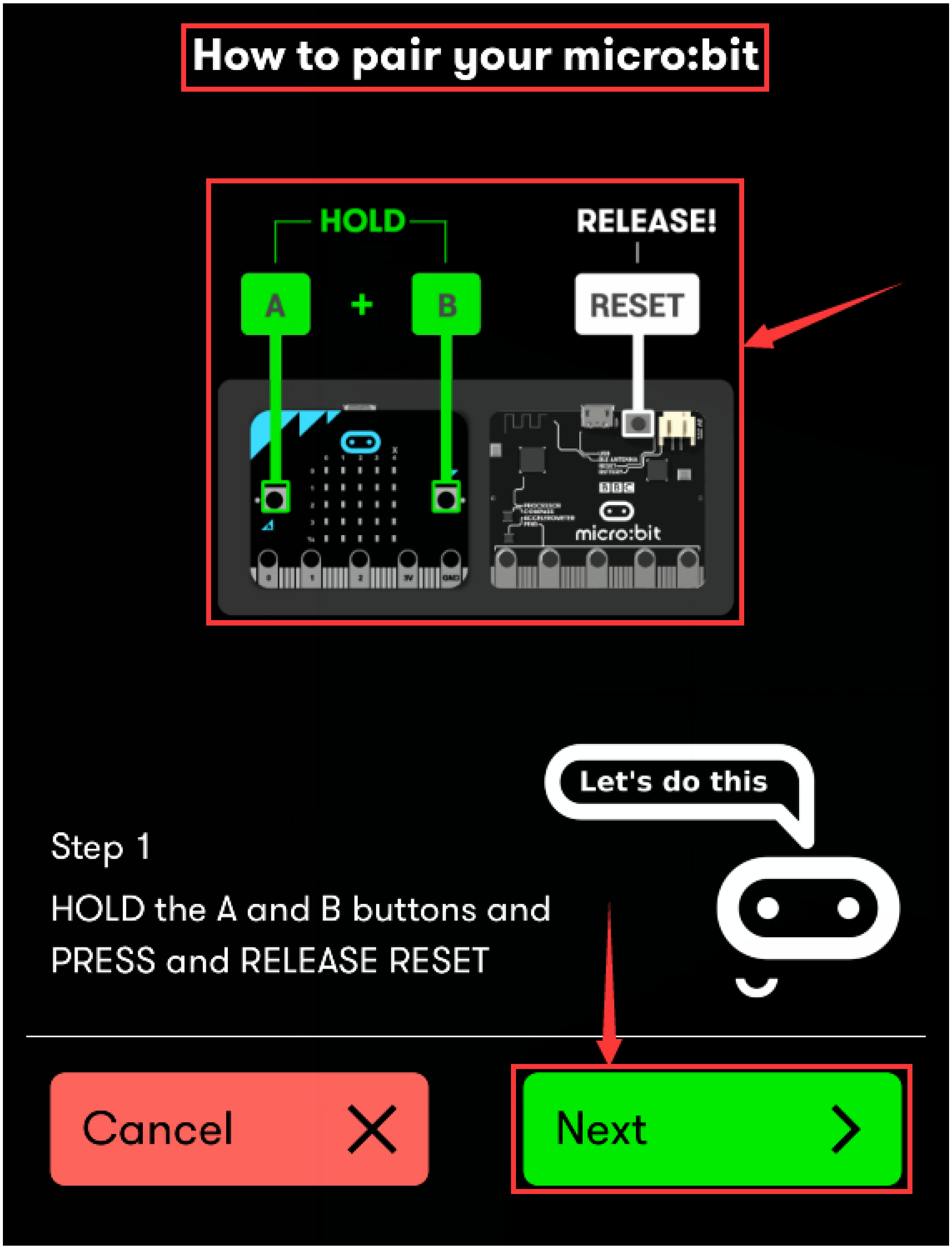
Release the Reset & Power button, you will see a password pattern shows on the LED dot matrix. Now , release buttons A and B and click Next.



How to pair your micro:bit

Step 1

HOLD the A and B buttons and PRESS and RELEASE RESET

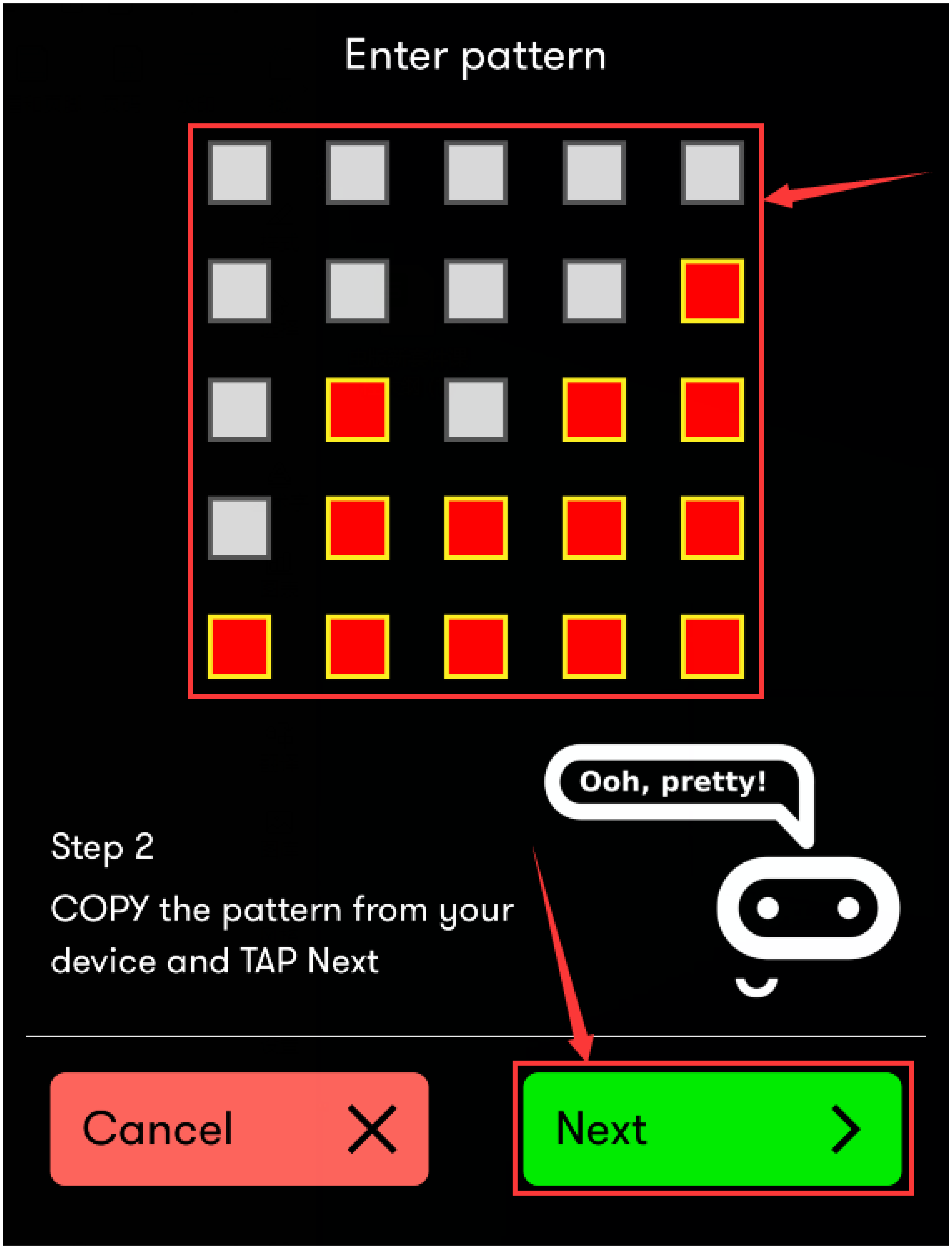


How to pair your micro:bit

Step 1

HOLD the A and B buttons and PRESS and RELEASE RESET

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

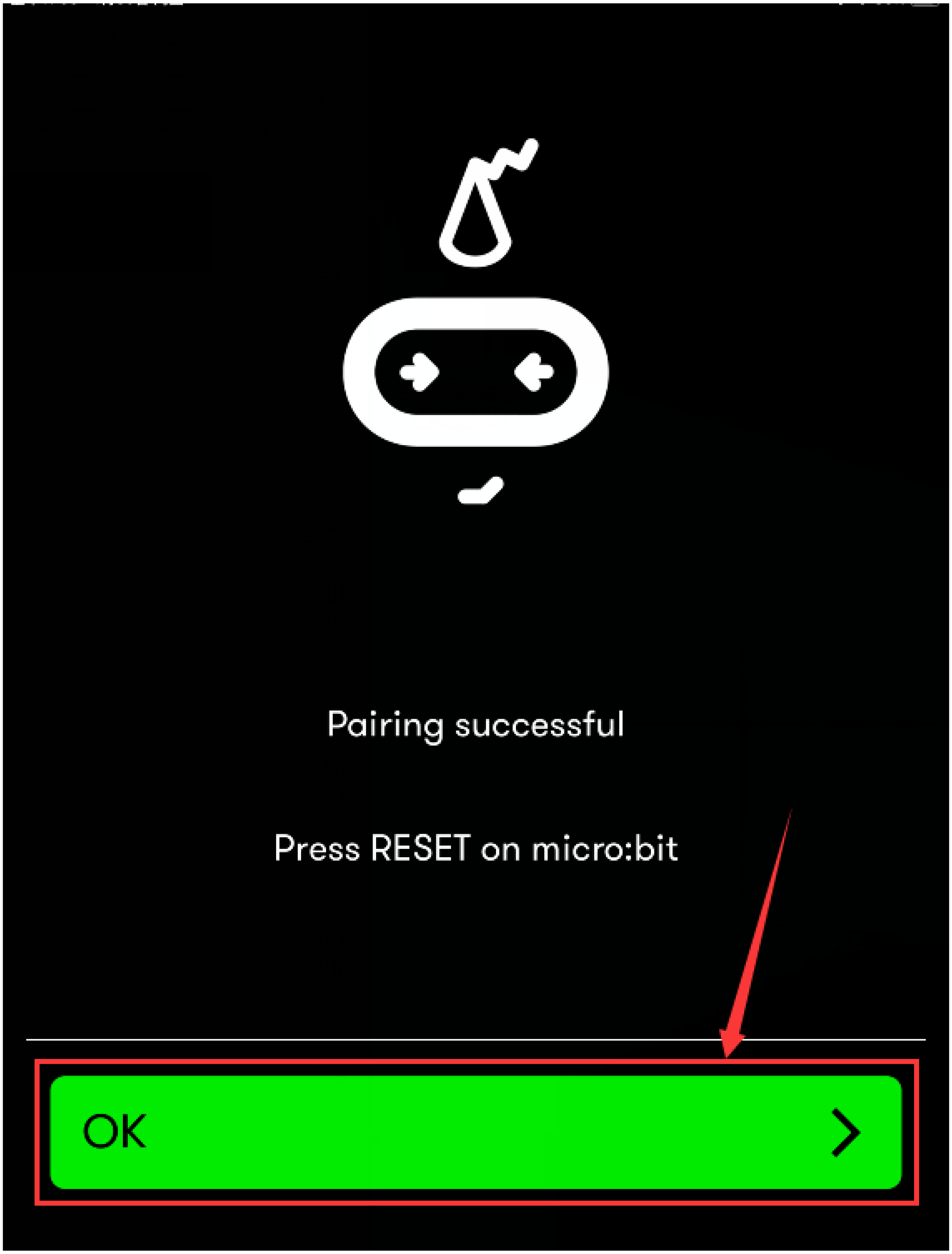


Enter pattern

Step 2

COPY the pattern from your device and TAP Next

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

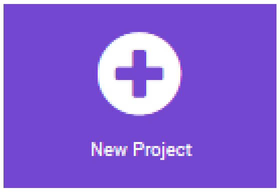
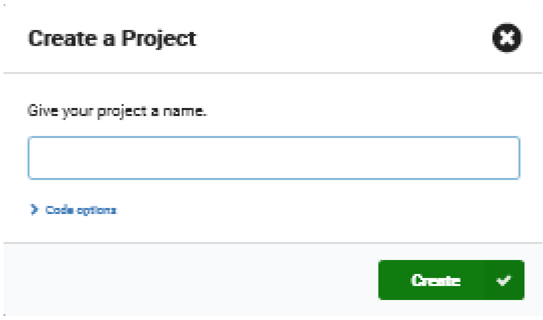


Pairing successful

Press RESET on micro:bit

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

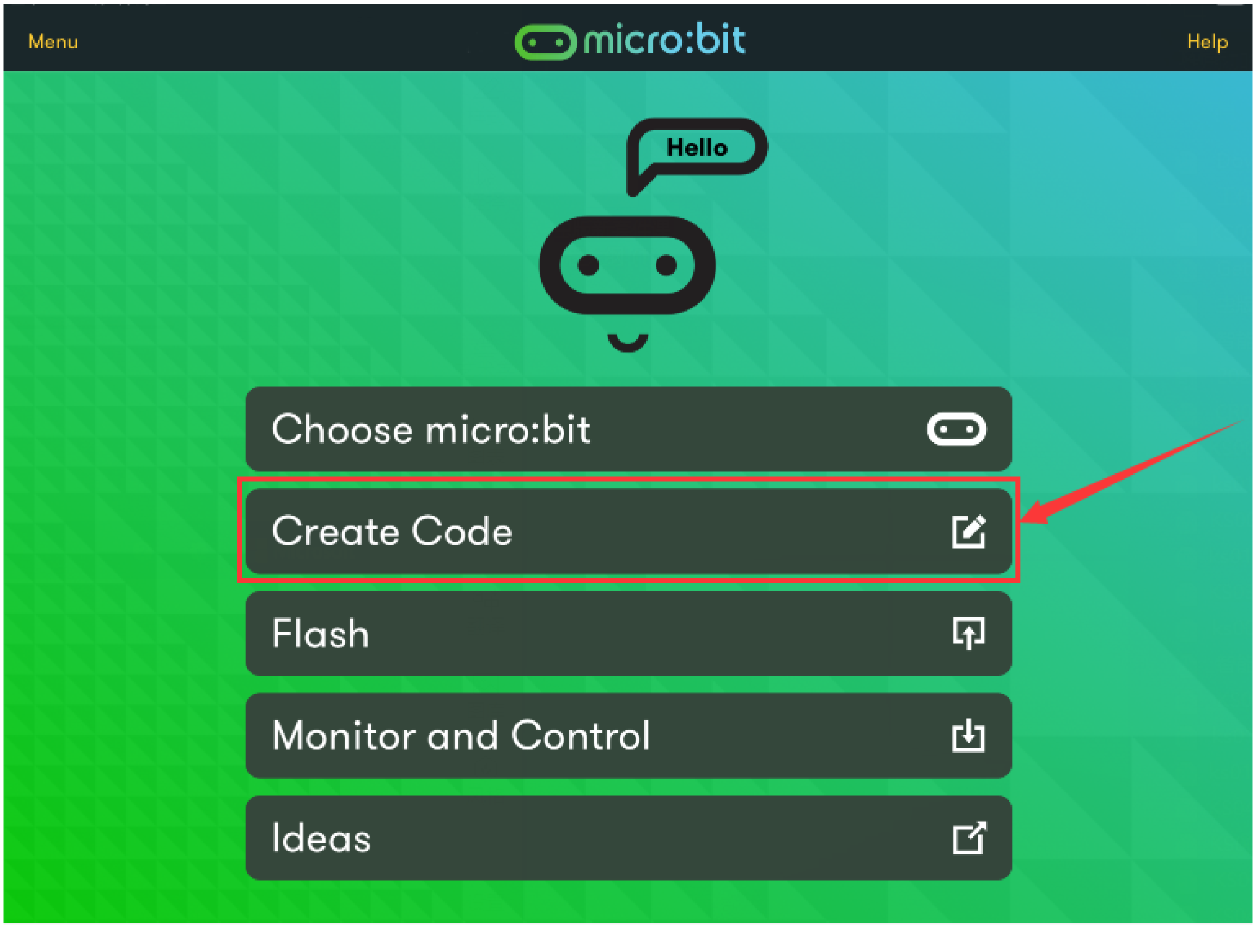
Click "Create Code" to enter the programming page and write code.

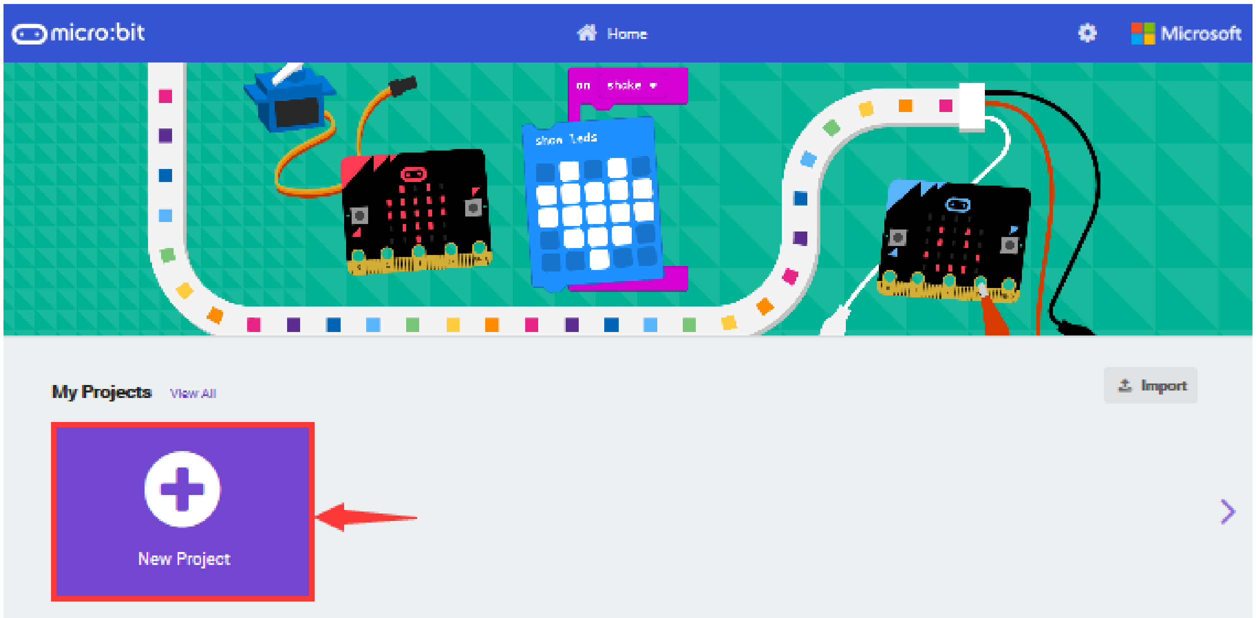
Click  and the box 

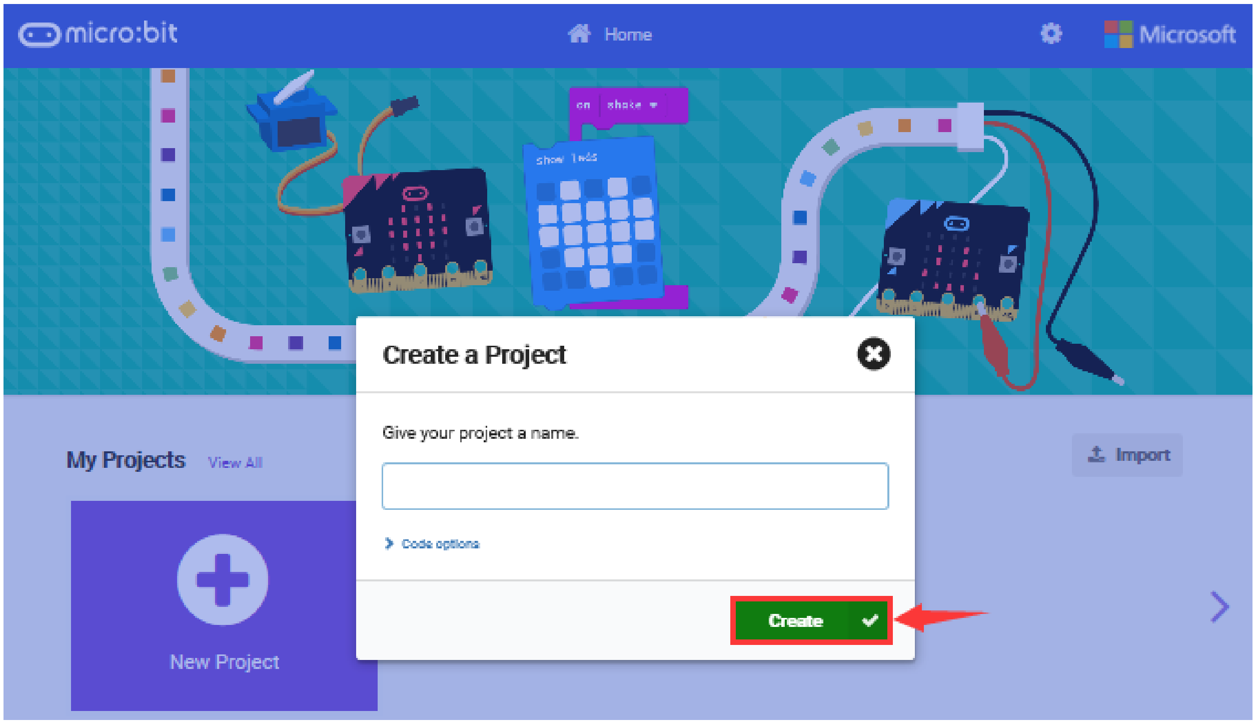
Create a Project

Give your project a name.

appears, and then select "Create ".





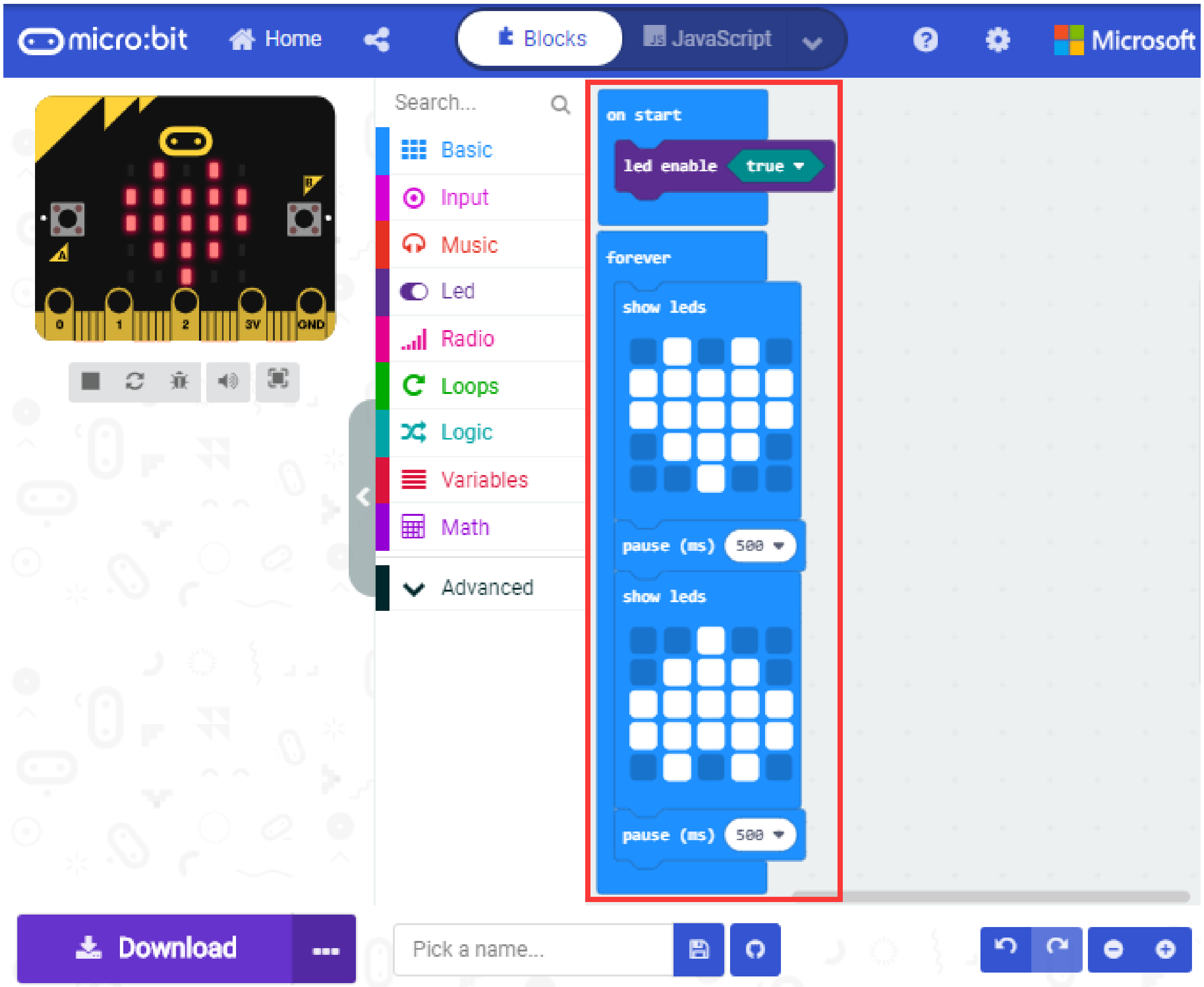


Create a Project

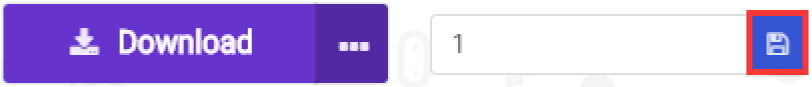
Give your project a name.

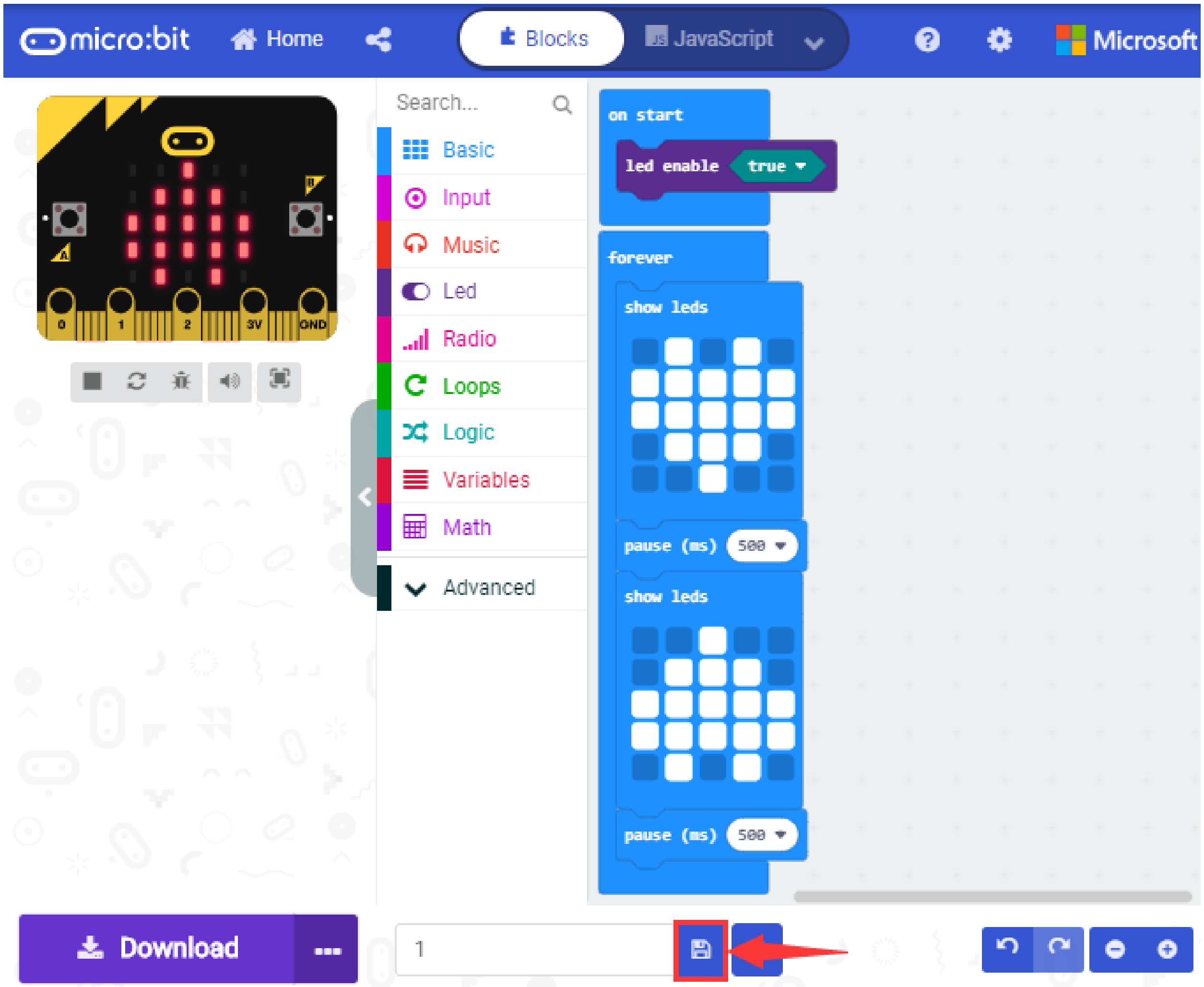
Code options

Create ✓



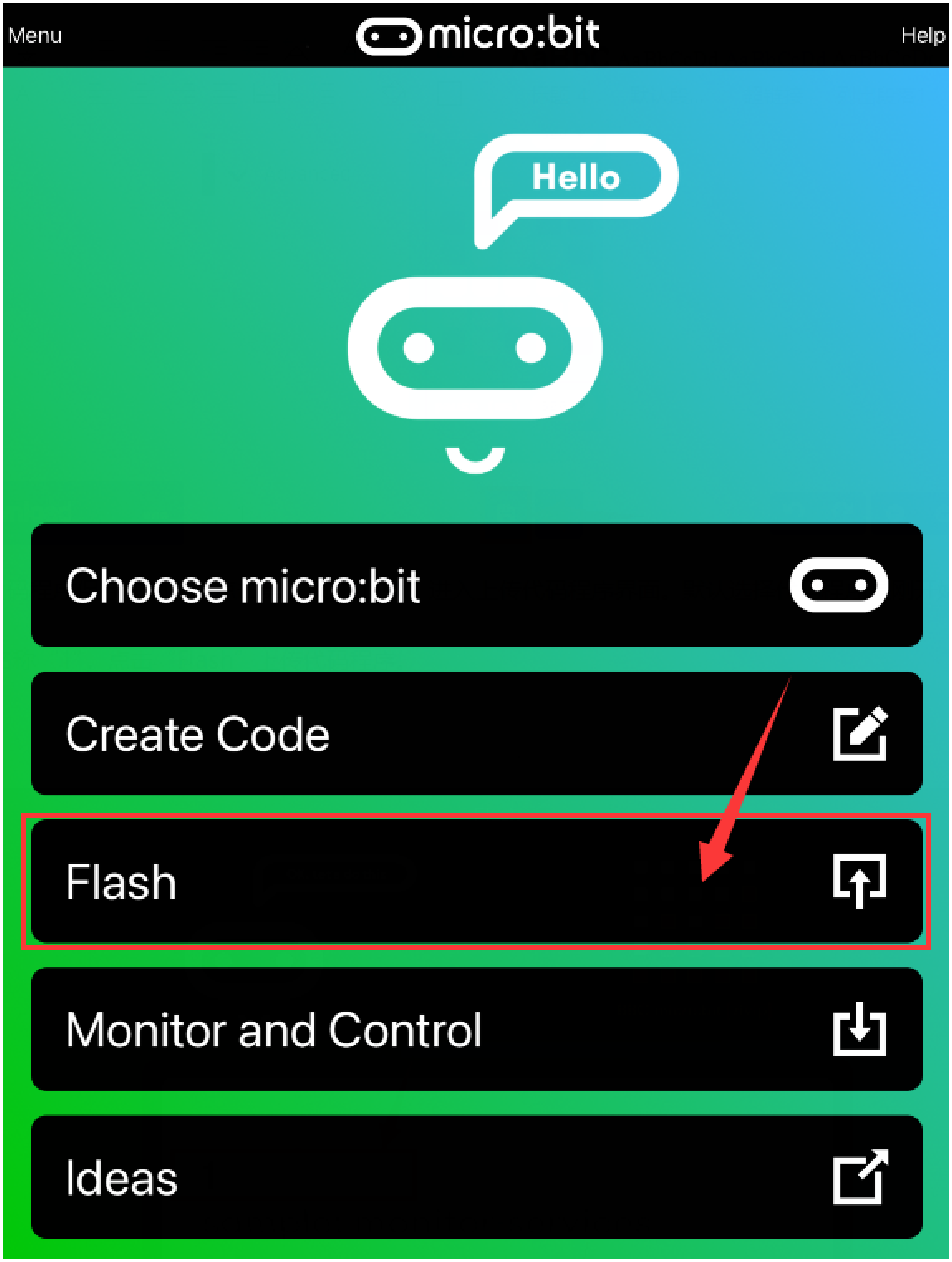
```
on start
    led enable true
forever
    show leds
        . # . # .
        # # # # #
        # # # # #
        . # # # .
        . . # . .
    pause (ms) 500
    show leds
        . . # . .
        . # # # .
        # # # # #
        # # # # #
        . # . # .
    pause (ms) 500
```

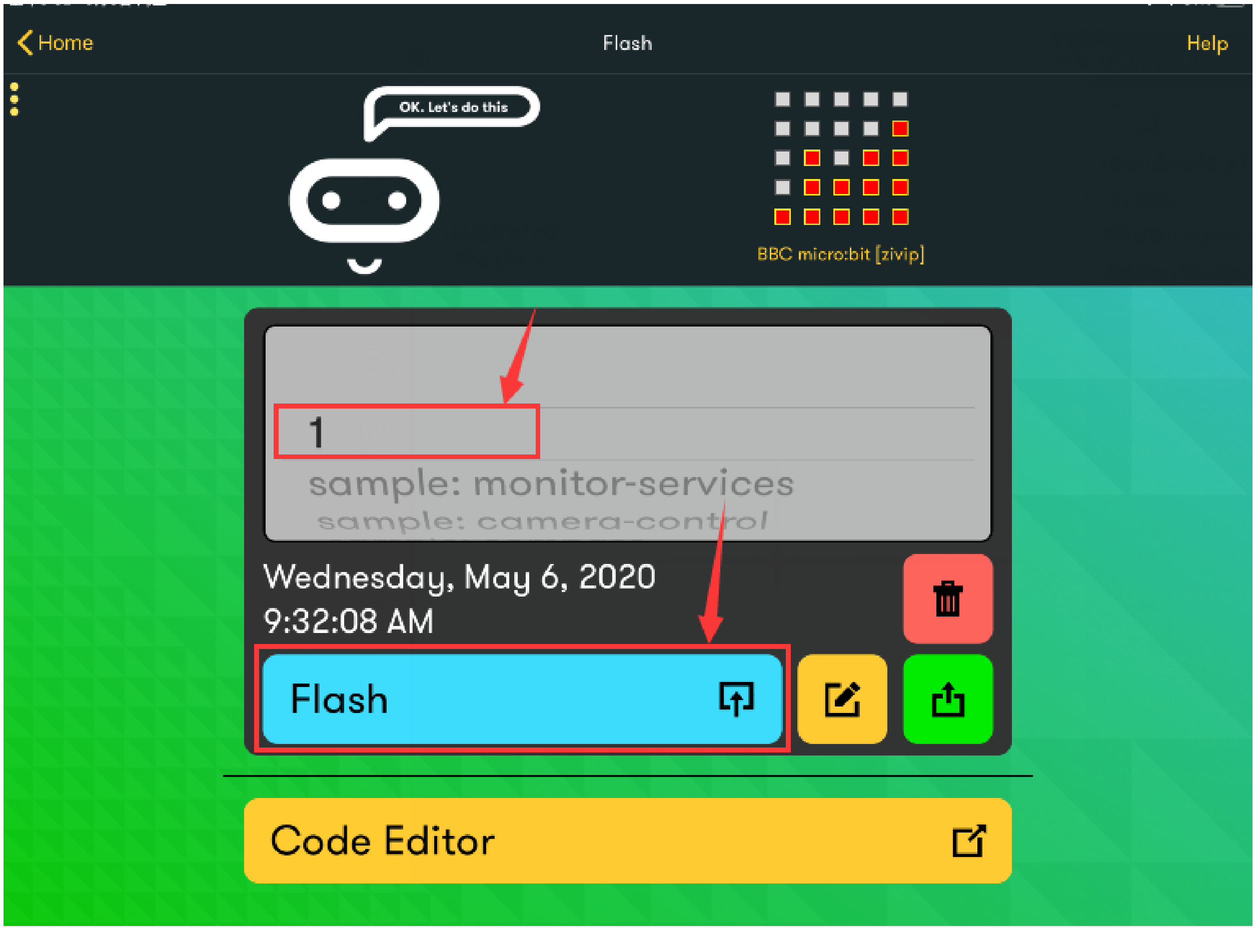
Name the code as "1 "and click  to save it.

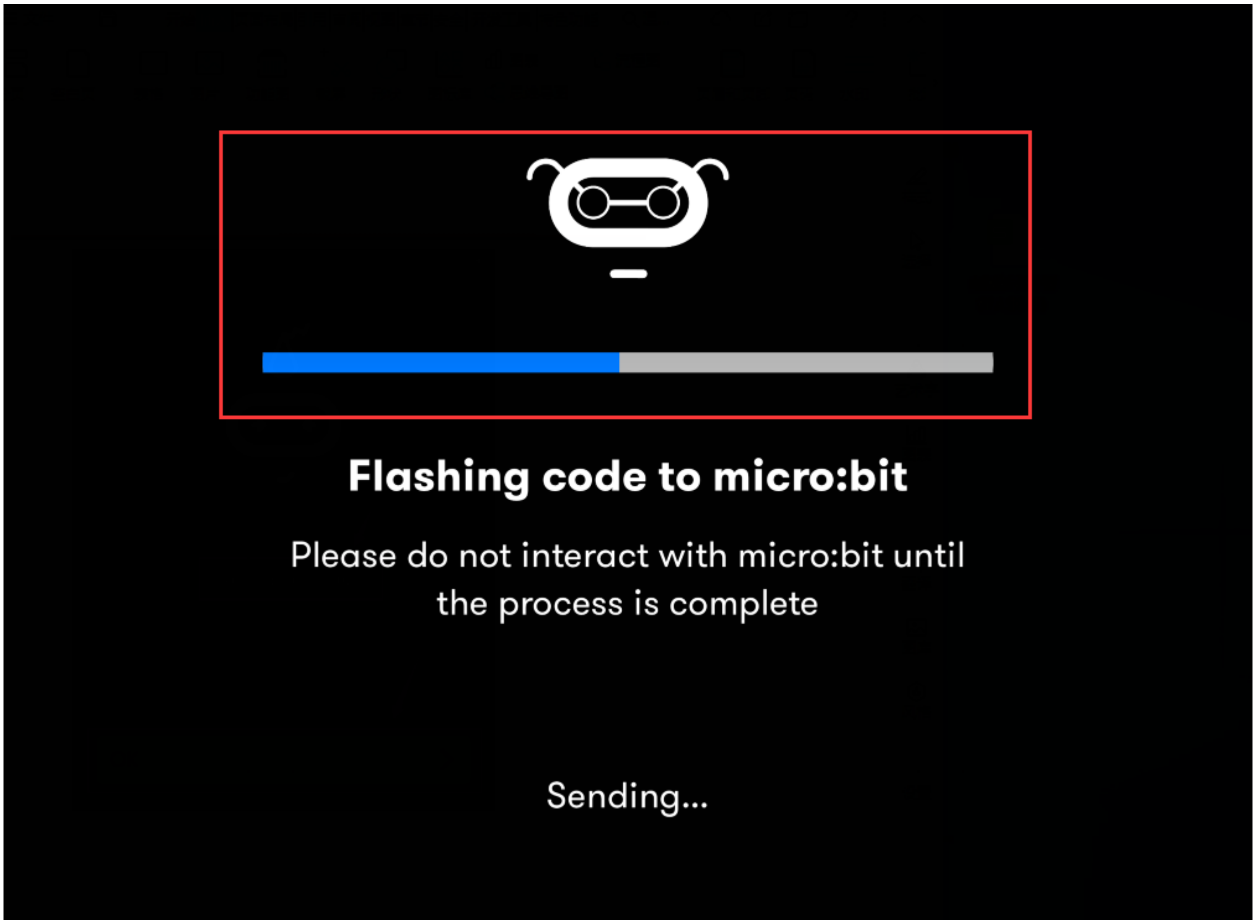


```
on start
    led enable true
forever
    show leds
    pause (ms) 500
    show leds
    pause (ms) 500
```

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”.


Menu
micro:bit
Help
Hello
Choose micro:bit
Create Code
Flash
Monitor and Control
Ideas


Home
Flash
Help
OK. Let's do this
BBC micro:bit [zivip]
1
sample: monitor-services
sample: camera-control
Wednesday, May 6, 2020
9:32:08 AM
Flash
Code Editor

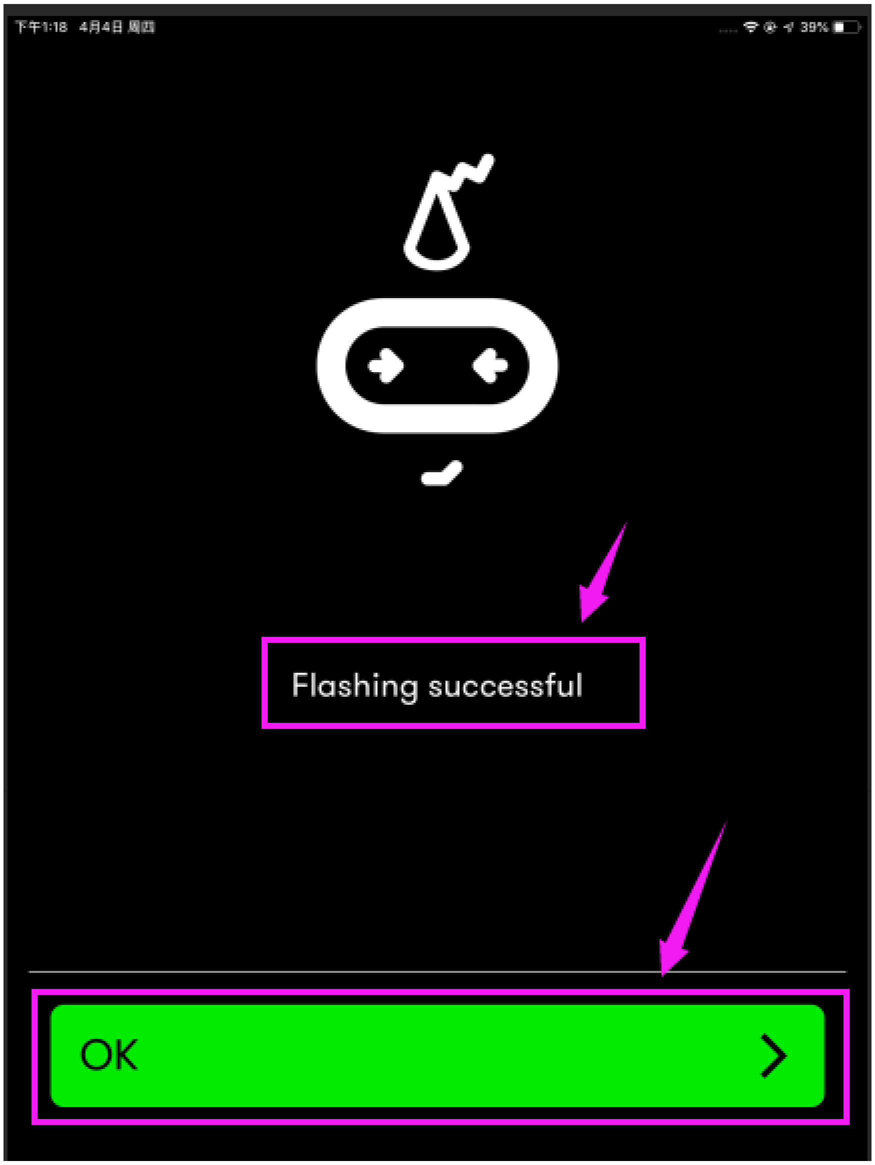


Flashing code to micro:bit

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

Sending...

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



Projects 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.

Attentionto avoid burning the the Micro:bit main board V2, please remove the USB cable and the external power from the board before fix it with a T-shaped shield; likewise, the USB cable and the external power should be cut from the main board before disconnect the shield from the board.)

7.13 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

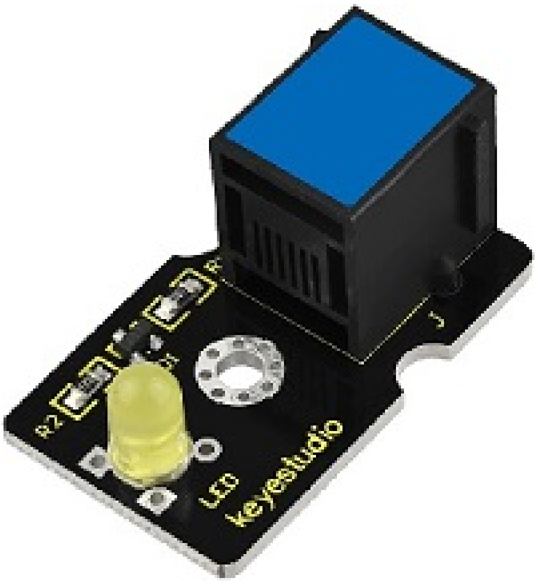
EASY Plug Yellow LED Module*1

RJ11 Cable*1

1-Slot AA Battery Holder*1

1.5V AA Battery*6

3. EASY Plug Yelow 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.

Specification

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

4. EASY Plug Shield for micro bit V1.1:

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

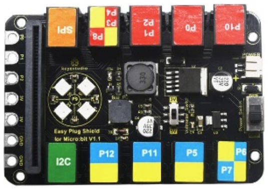
The golden finger interfaces ,as well as 10 pcs easy plug ports (RJ11 6P6C interfaces)could be connected to other modules and sensors, therefore, you don't need to worry about wiring up components incorrectly.

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.

Specification



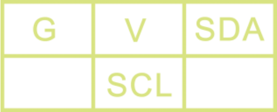
- Power supply: DC 6-10V
- Output current: 1.5A
- Interface: RJ11 6P6C interface and golden finger interface
- Size: 98*65*17mm

Interface Description

GGND

V: Voltmeter_Switch controldial to 5V end5Vdial to 3V end, 3.3V

I2C Communication Port

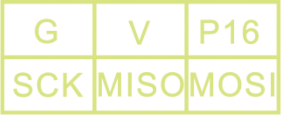


G	V	SDA
	SCL	

SDAP20

SCLP19

SPI Communication Port



G	V	P16
SCK	MISO	MOSI

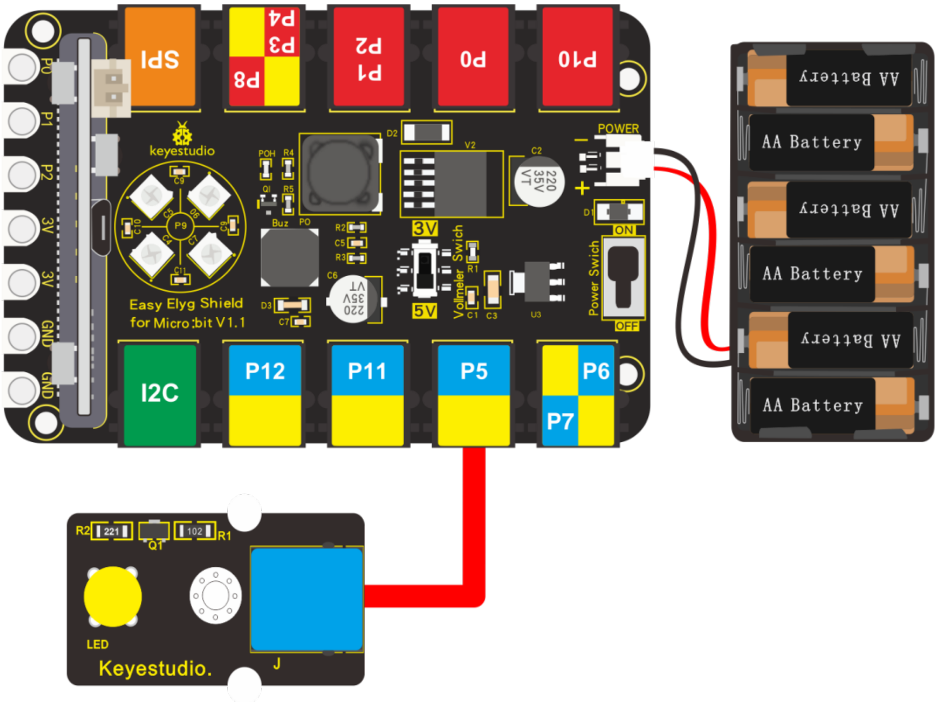
MOSIP15

MISOP14

SCKP13

5. Wiring Up:

Insert micro:bit onto EASY Plug shield, link white LED module with P5 port of shield and plug in power.

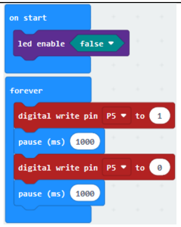


Note: Dial Voltmeter_Switch to 3V end.

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 led enable false forever digital write pin P5 to 1 pause (ms) 1000 digital write pin P5 to 0 pause (ms) 1000	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitThe program under the block "forever" runs cyclically.Set P5 to high level(1), turn on LEDDelay in 1000msSet P5 to low level(0), turn off LEDDelay in 1000ms

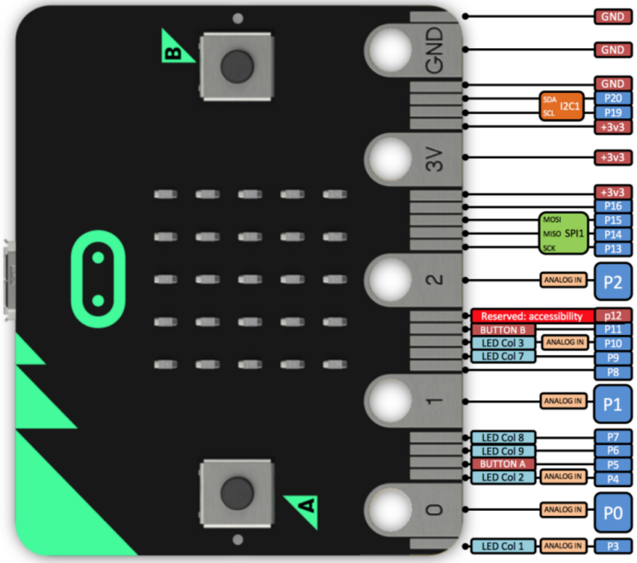
7. Test Results:

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

7.14 Project 14: Breath

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.



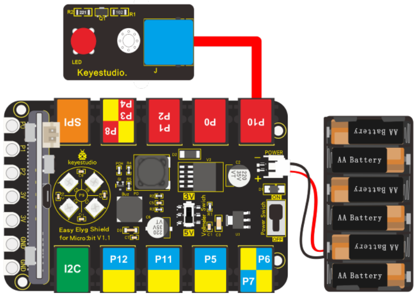
This lesson you will learn how to control the brightness of LED on the module, gradually becoming brighter and dimming, just like the LED is breathing.

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-Slot 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.

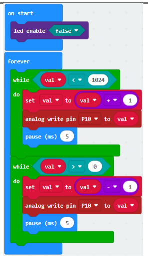


Note: Dial Voltmeter_Switch to 3V end

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 led enable false forever 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	“on start”: command block only runs once to start program.Turn off dot matrix on micro:bitThe program under the block “forever” runs cyclically.When val<1024, run the program in the do blockSet val to val+1Set analog value of P10 to valDelay in 5msWhen val>0, run the program in the do blockSet val to val-1Set 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.

7.15 Project 15: Blink and Breath

1. Description:

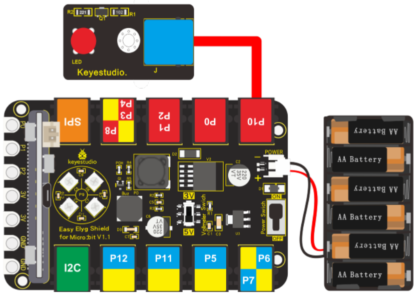
In this project, we will combine LED flash and breathing effect together.

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-Slot AA Battery Holder*11.5V AA Battery*6

3. Wiring Up:

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

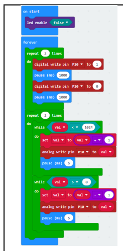


Note: Dial Voltmeter_Switch to 3V end

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”: command block only runs once to start program.Turn off dot matrix on micro:bitThe program under the block “forever” runs cyclically.Repeat the program in the do block twiceSet P10 to high level1, turn on LEDDelay in 1000msSet P10 to low level0, turn off LED Delay in 1000msRepeat the program in the do block twiceWhen val<1024, run the program in the do blockSet val to val+1Set the analog value of P10 to valDelay in 5msWhen val>0, execute the program in the do blockSet val to val-1Set the analog value of P10 to valDelay in 5ms

```
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
```

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.

7.16 Project 16: RGB

1. Description:

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

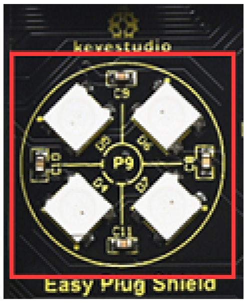
In this project, we will demonstrate how to play music with passive buzzer. Easy Plug shield comes with one. Let’s get started. (Passive buzzer is connected to P9 on Easy Plug shield)

2. What You Need:

- Micro:bit Board*1

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

3. 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.

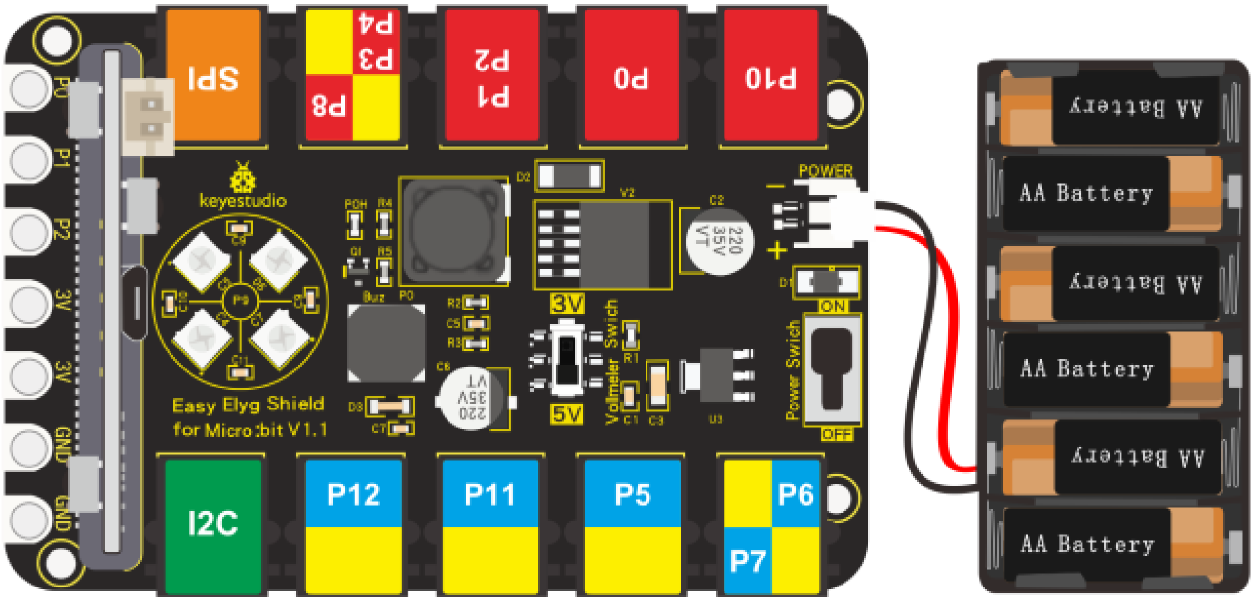
The data protocol adopts the single-line return-to-zero code communication mode. After power-on and reset the pixel point, the S pin receives the data transmitted from the controller. And the 24-bit data are extracted by the first pixel and then sent to the data latch inside the pixel point.

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.

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:



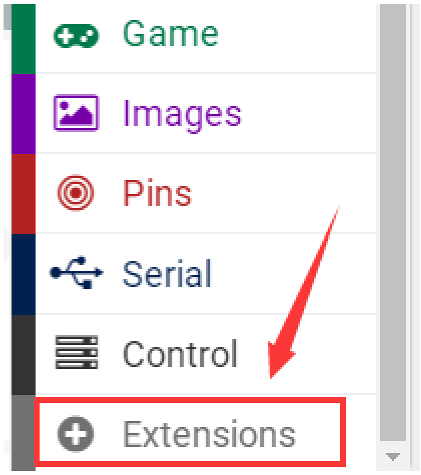
Note: Dial Voltmeter_Switch to 3V end

5. Test Code:

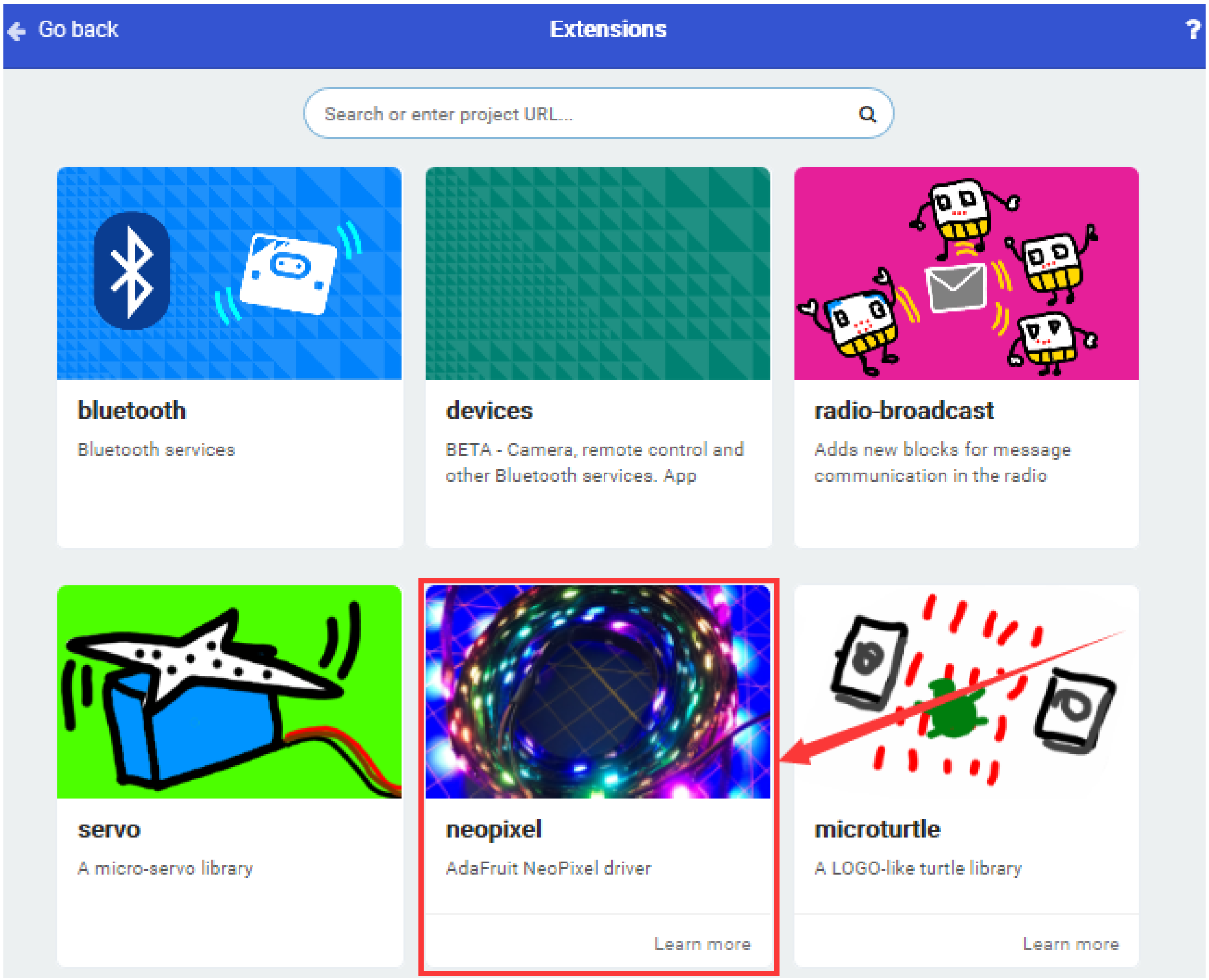
You could navigate https://makecode.micro:bit.org/reference to have access to more details.

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

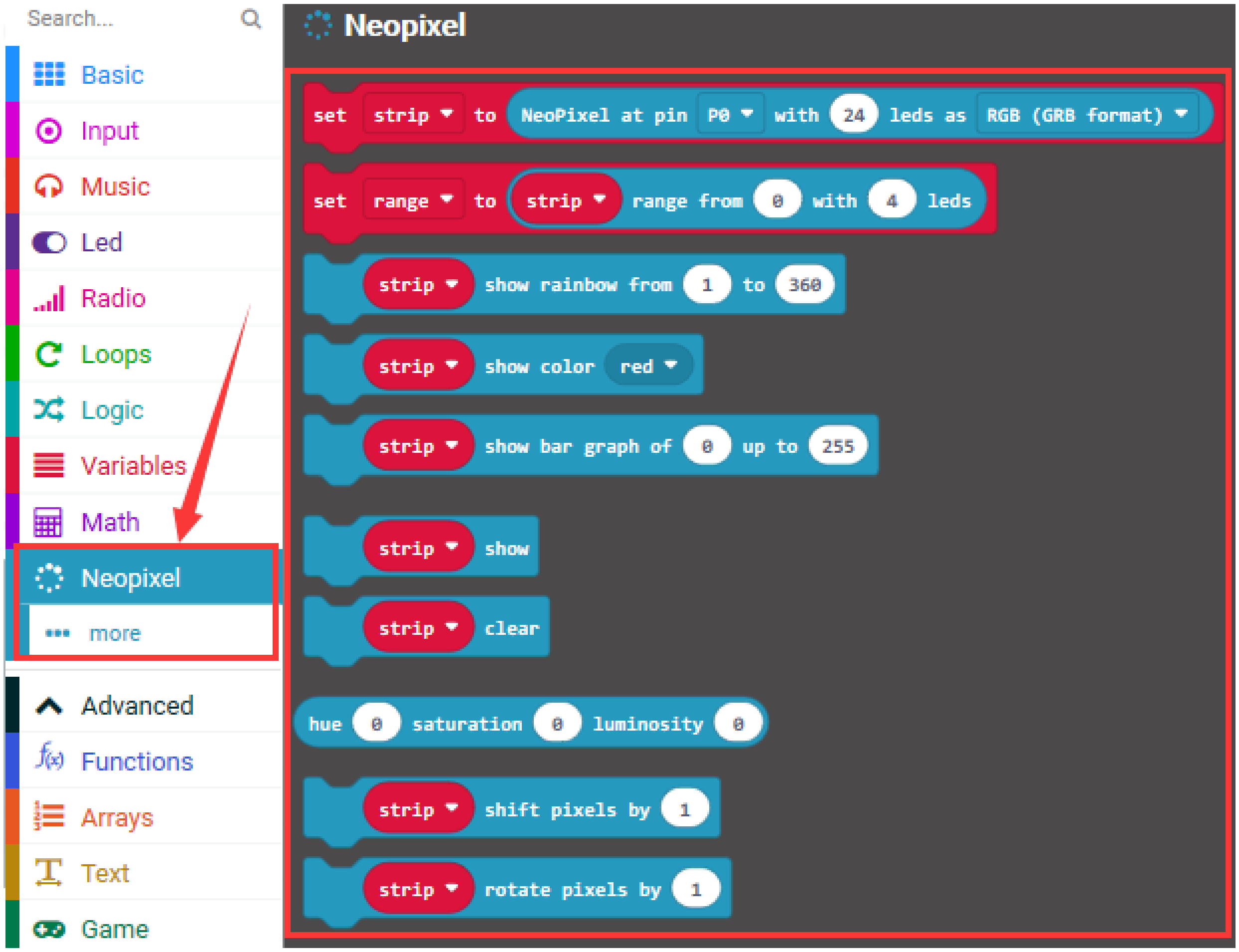
We need to set test code in library file, and add the library of "neopixe".



Click"Extensions"→"neopixel"click to download

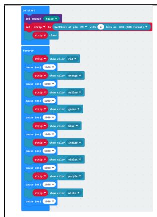

Go back
Extensions
?
Search or enter project URL...
bluetooth
Bluetooth services
devices
BETA - Camera, remote control and other Bluetooth services. App
radio-broadcast
Adds new blocks for message communication in the radio
servo
A micro-servo library
neopixel
AdaFruit NeoPixel driver
Learn more
microturtle
A LOGO-like turtle library
Learn more

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

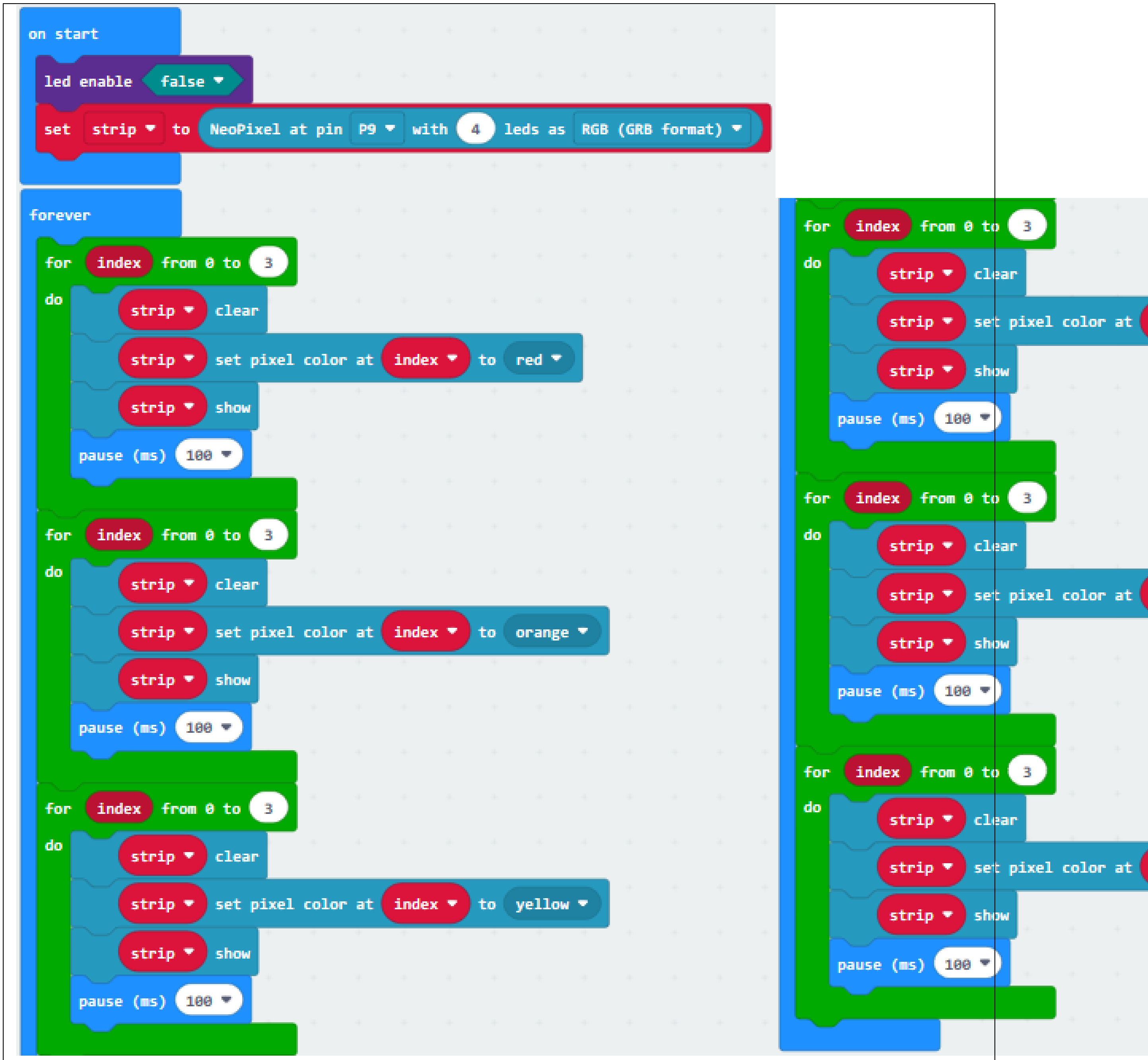


```
set strip to NeoPixel at pin P0 with 24 leds as RGB (GRB format)
set range to strip range from 0 with 4 leds
strip show rainbow from 1 to 360
strip show color red
strip show bar graph of 0 up to 255
strip show
strip clear
hue 0 saturation 0 luminosity 0
strip shift pixels by 1
strip rotate pixels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

Code 1:

	“on start”: command block only runs once to start program.Turn off dot matrix on micro:bitSet strip to NeoPixel at pin P9 with 4 leds as RGBTurn off 4 pcs WS2812 RGB lightsThe program under “forever” runs cyclically.make all RGB show red colorDelay in 1000msmake all RGB show orange colorDelay in 1000msmake all RGB show yellow colorDelay in 1000msmake all RGB show green colorDelay in 1000msAll of RGB show blue colorDelay in 1000msmake all RGB show indigo colorDelay in 1000msmake all RGB show violet colorDelay in 1000msmake all RGB show purple colorDelay in 1000msmake all RGB show white colorDelay in 1000ms

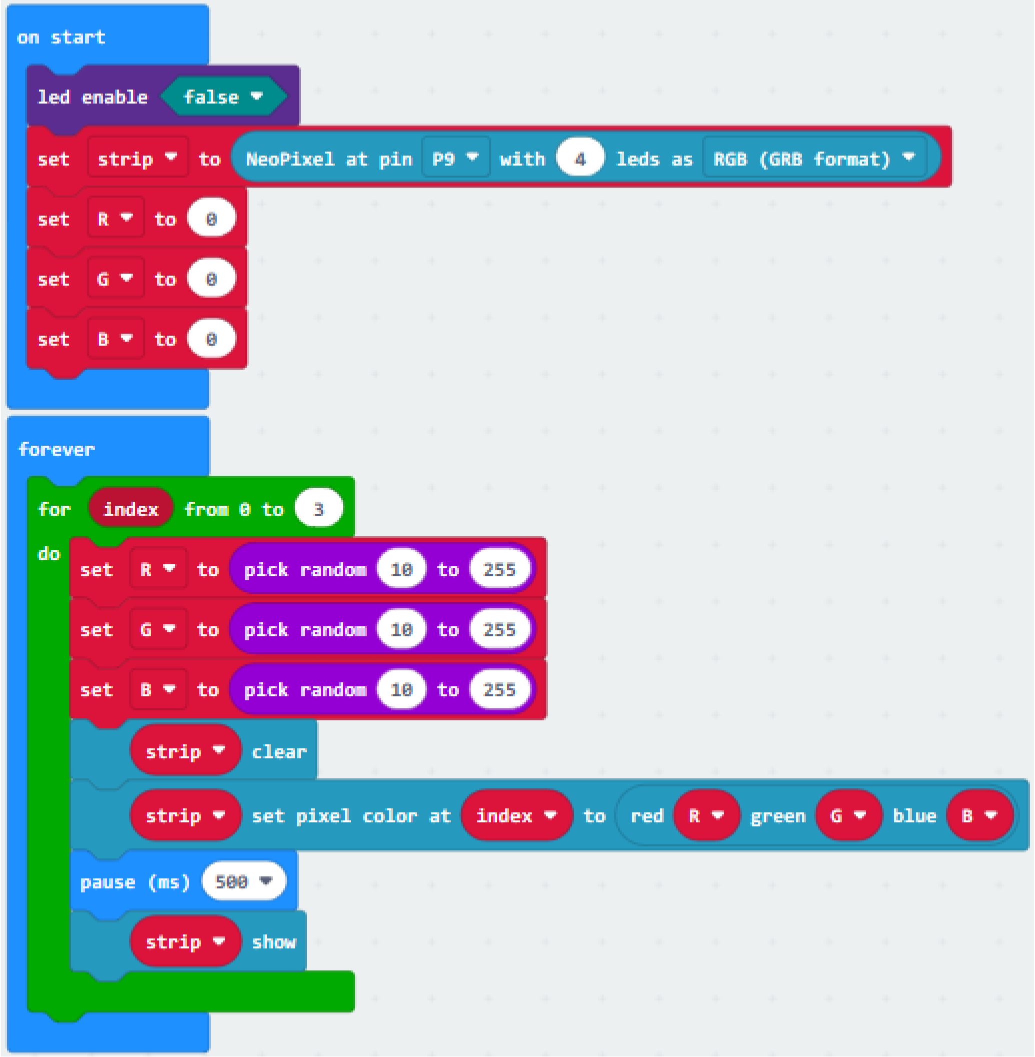
Code 2



```
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

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

"on start": command block only runs once to start program.Turn off dot matrix on micro:bitSet strip to NeoPixel at pin P9 with 4 leds as RGBThe program under "forever" runs cyclically.For index from 0 to 3, execute the program under do blockTurn off 4 pcs WS2812 RGB lightsSet pixel color of 4 pcs WS2812 RGB lights to red colorStrip showDelay in 100msFor index from 0 to 3, execute the program under do blockTurn off 4 pcs WS2812 RGB lightsSet pixel color of 4 pcs WS2812 RGB lights to orange colorStrip showDelay in 100msFor index from 0 to 3, execute the program under do blockTurn off 4pcs WS2812 RGB lightsSet pixel color of 4 pcs WS2812 RGB lights to yellow colorStrip showDelay in 100msFor index from 0 to 3, execute the program under do blockTurn off 4pcs WS2812 RGB lightsSet pixel color of 4 pcs WS2812 RGB lights to green colorStrip showDelay in 100msFor index from 0 to 3, execute the program under do blockTurn off 4 pcs WS2812 RGB lightsSet pixel color of 4 pcs WS2812 RGB lights to blue colorStrip showDelay in 100msFor index from 0 to 3, execute the program under do blockTurn off 4 pcs WS2812 RGB Set pixel color of 4 pcs WS2812 RGB lights to indigo colorStrip showDelay in 100msFor index from 0 to 3, execute the program under do blockTurn off 4 pcs WS2812 RGB lightsSet pixel color of 4 pcs WS2812 RGB lights to violet colorStrip showDelay in 100msFor index from 0 to 3, execute the program under do blockTurn off 4 pcs WS2812 RGBSet pixel color of 4 pcs WS2812 RGB lights to violet colorStrip refreshes to displayDelay in 100msFor index from 0 to 3, execute the program under do blockTurn off 4 pcs WS2812 RGBTurn off 4 pcs Set pixel color of 4 pcs WS2812 RGB lights to white colorStrip showDelay 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
```

"on start": command block only runs once to start program.Turn off micro:bit LED dot matrixSet strip to NeoPixel at pin P9 with 4 leds as RGBSet strip to initialization oSet variable R, G, B to 0The program under the block "forever" runs cyclically.When value of variable index is in 0-3, execute the program in the do blockSet variable R, G, B to random number in 10~255Turn off all RGB lights on the stripSet pixel color of 4 pcs WS2812 RGB lights to RGBDelay in 500msstrip 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.

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.

7.17 Project 17: Play Music

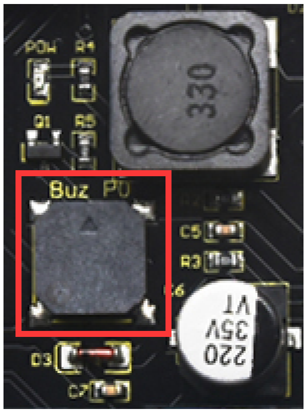
1. Description:

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

3. Passive Buzzer Module:



The buzzer includes active buzzer and passive buzzer. The difference between them a built-in vibration source, therefore, it will make a sound when power is plugged in.

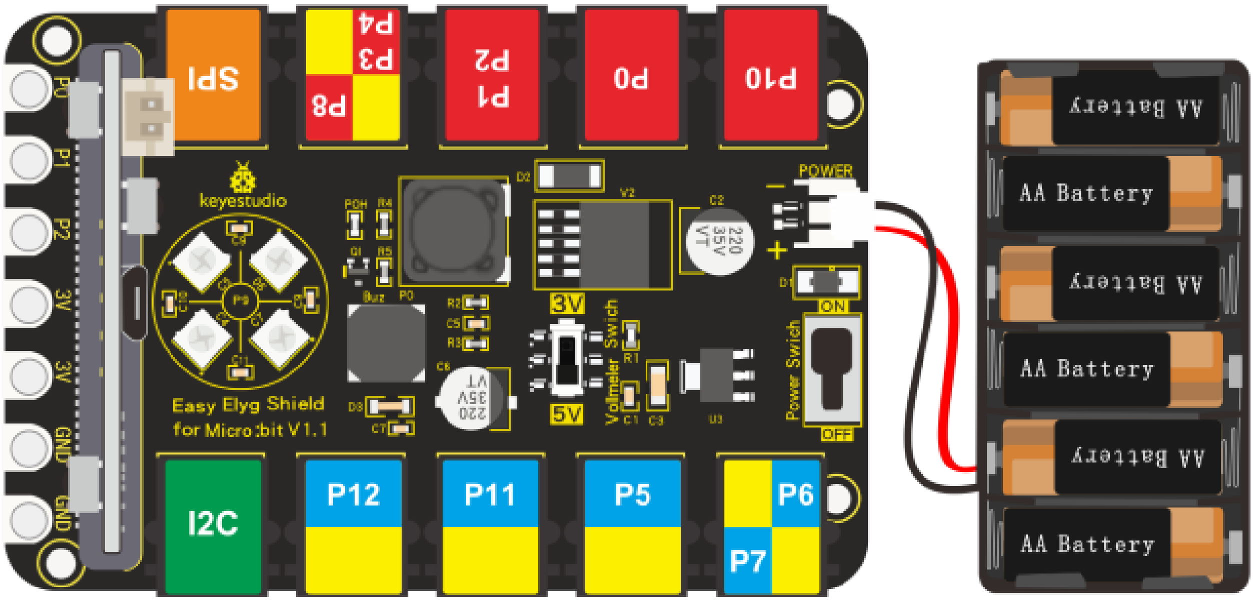
We need 2K-5K square wave to drive passive buzzer because the buzzer on EASY Plug Shield doesn't come with this kind of source.

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

Specification

- Working voltage: 3.3-5V
- Interface type: Digital

4. Wiring Up:



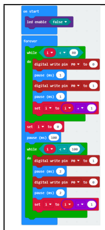
Note: Dial Voltmeter_Switch to 3V end.

5. Test Code:

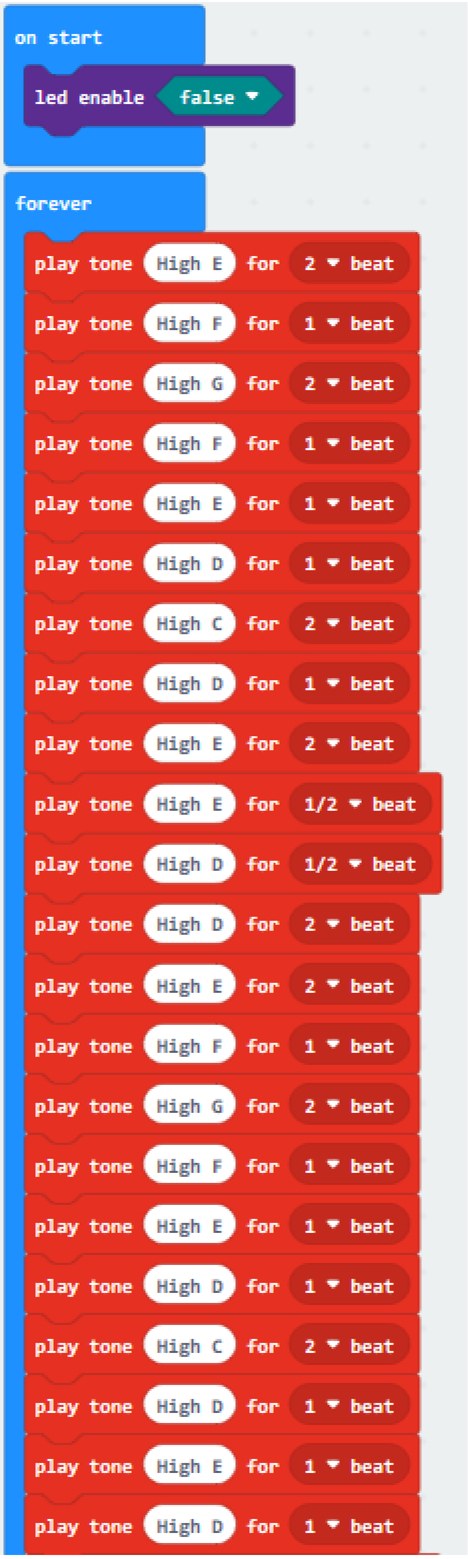
You could navigate https://makecode.micro:bit.org/reference to have access to more details.

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

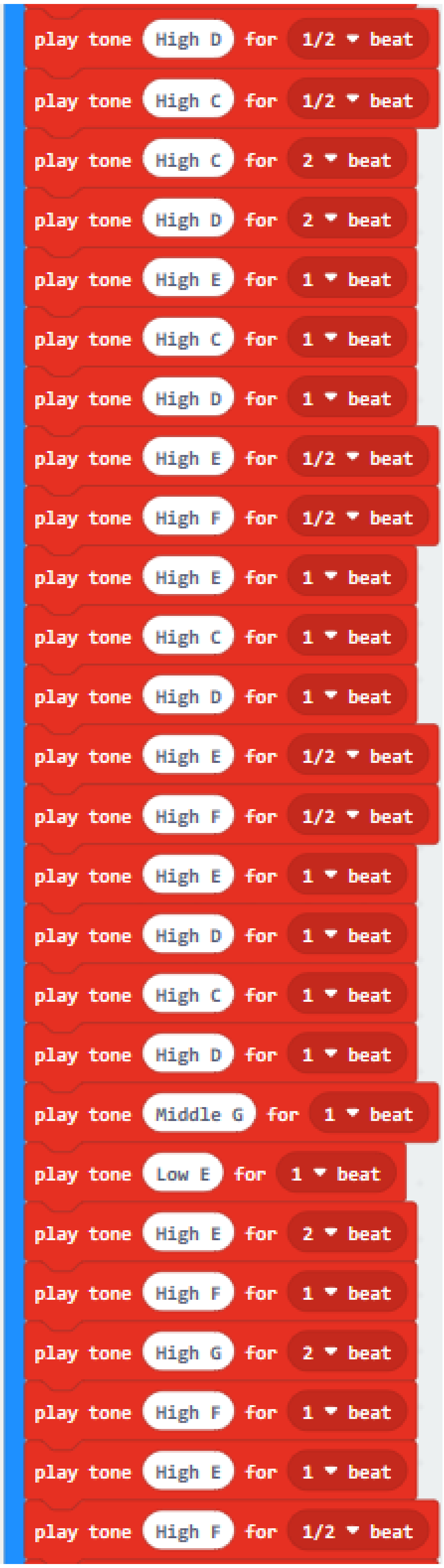
Code 1:

 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 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	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitThe program under the block "forever" runs cyclically.When i<80, run the program in the blockSet P0 to low level(0) to make passive buzzer silentDelay in 1msSet P0 to high level(1) to make passive buzzer emit soundDelay in 1msSet variable i to i+1Set variable i to 0Delay in 100msWhen i<100, run the program in the do block Set P0 to high level(1) to make passive buzzer emit soundDelay in 2msSet P0 to low level(0) to make passive buzzer silentDelay in 2msSet variable i to i+1

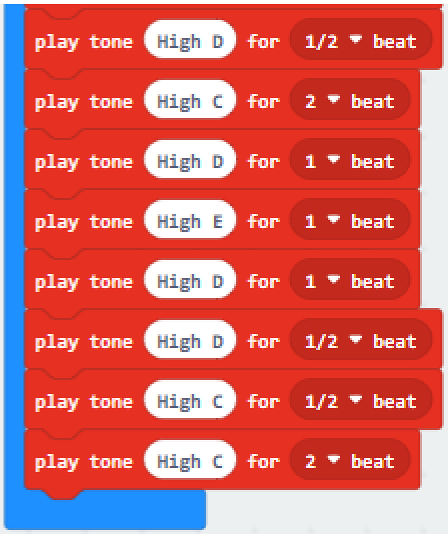
Code 2:



```
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 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
```

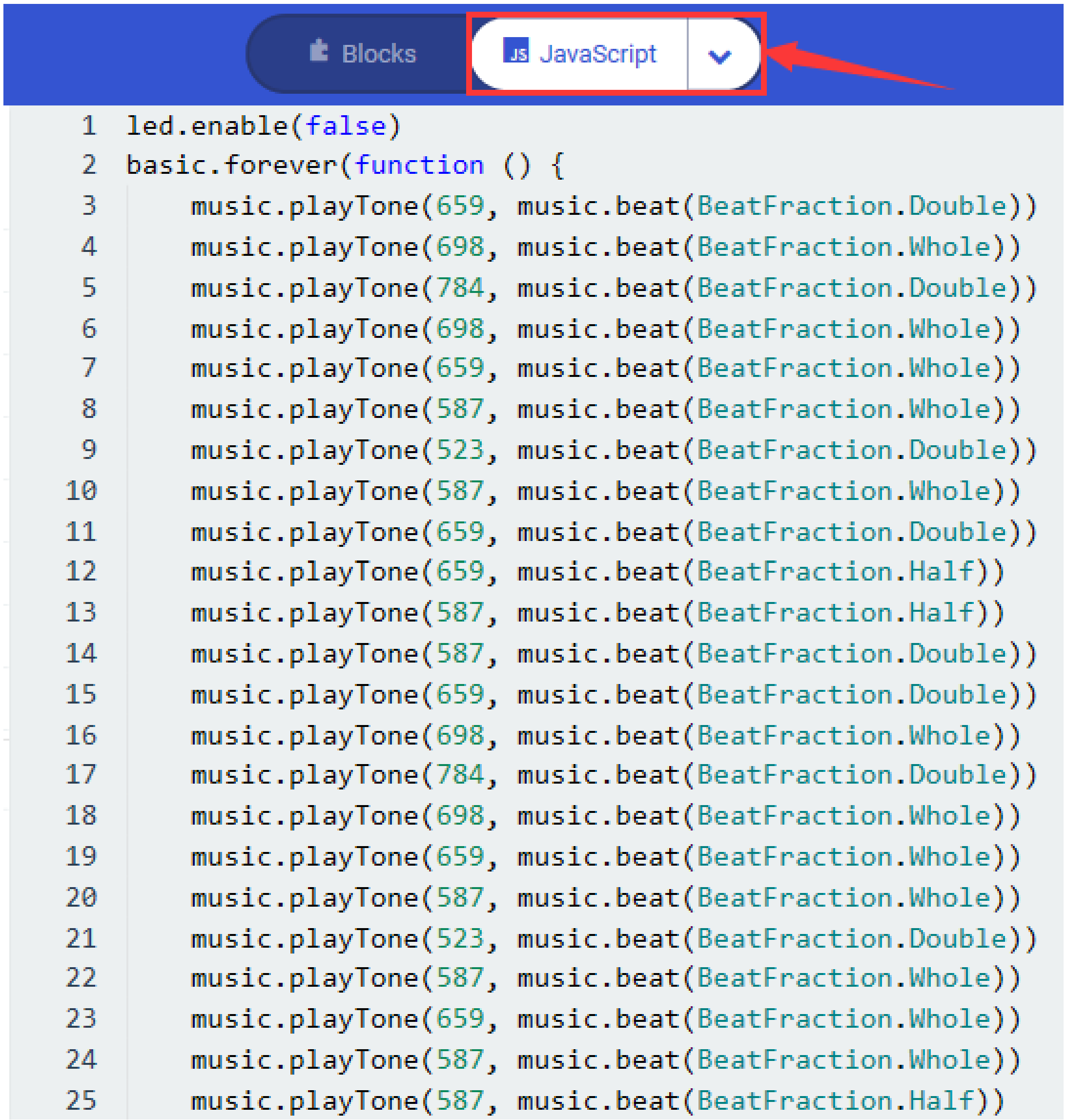


```
play tone High D for 1/2 ▾ beat
play tone High C for 1/2 ▾ beat
play tone High C for 2 ▾ beat
play tone High D for 2 ▾ beat
play tone High E for 1 ▾ beat
play tone High C for 1 ▾ beat
play tone High D for 1 ▾ beat
play tone High E for 1/2 ▾ beat
play tone High F for 1/2 ▾ beat
play tone High E for 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 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 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
play tone High D for 1/2 ▾ beat
play tone High C for 1/2 ▾ beat
play tone High C for 2 ▾ beat
```

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



```
led.enable(false)
basic.forever(function () {
    music.playTone(659, music.beat(BeatFraction.Double))
    music.playTone(698, music.beat(BeatFraction.Whole))
    music.playTone(784, music.beat(BeatFraction.Double))
    music.playTone(698, music.beat(BeatFraction.Whole))
    music.playTone(659, music.beat(BeatFraction.Whole))
    music.playTone(587, music.beat(BeatFraction.Whole))
    music.playTone(523, music.beat(BeatFraction.Double))
    music.playTone(587, music.beat(BeatFraction.Whole))
    music.playTone(659, music.beat(BeatFraction.Double))
    music.playTone(659, music.beat(BeatFraction.Half))
    music.playTone(587, music.beat(BeatFraction.Half))
    music.playTone(587, music.beat(BeatFraction.Double))
    music.playTone(659, music.beat(BeatFraction.Double))
    music.playTone(698, music.beat(BeatFraction.Whole))
    music.playTone(784, music.beat(BeatFraction.Double))
    music.playTone(698, music.beat(BeatFraction.Whole))
    music.playTone(659, music.beat(BeatFraction.Whole))
    music.playTone(587, music.beat(BeatFraction.Whole))
    music.playTone(523, music.beat(BeatFraction.Double))
    music.playTone(587, music.beat(BeatFraction.Whole))
    music.playTone(659, music.beat(BeatFraction.Whole))
    music.playTone(587, music.beat(BeatFraction.Whole))
    music.playTone(587, music.beat(BeatFraction.Half))
```

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 1 to micro:bit, you will hear the buzzer emit two kind of sounds; if download code 2 to micro:bit, the song “Ode to Joy” will be played.

7.18 Project 18: Knock Sensor

1. Description:

Sensor can detect the data, sense the signal and control the devices. We are familiar with light sensor, temperature, humidity and sound sensors. A great deal of experiments could be finished with these sensors and modules.

We will control the LED by knock sensor in the experiment.

2. What You Need:

- Micro:bit Board*1

- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Knock Sensor*1
- EASY Plug Red LED Module*1
- RJ11 Cable*2
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug Knock Sensor:

The knock sensor is mainly composed of SW-280 vibration switch, which is an inductive proximity switch.

It is an electronic switch that transmits the sensing result to the circuit device and induces the circuit to start working when the vibration force is induced.

The module comes with a positioning hole for you to fix it to other devices.

You can make full use of it with creative thinking, like electronic drum, and so on.

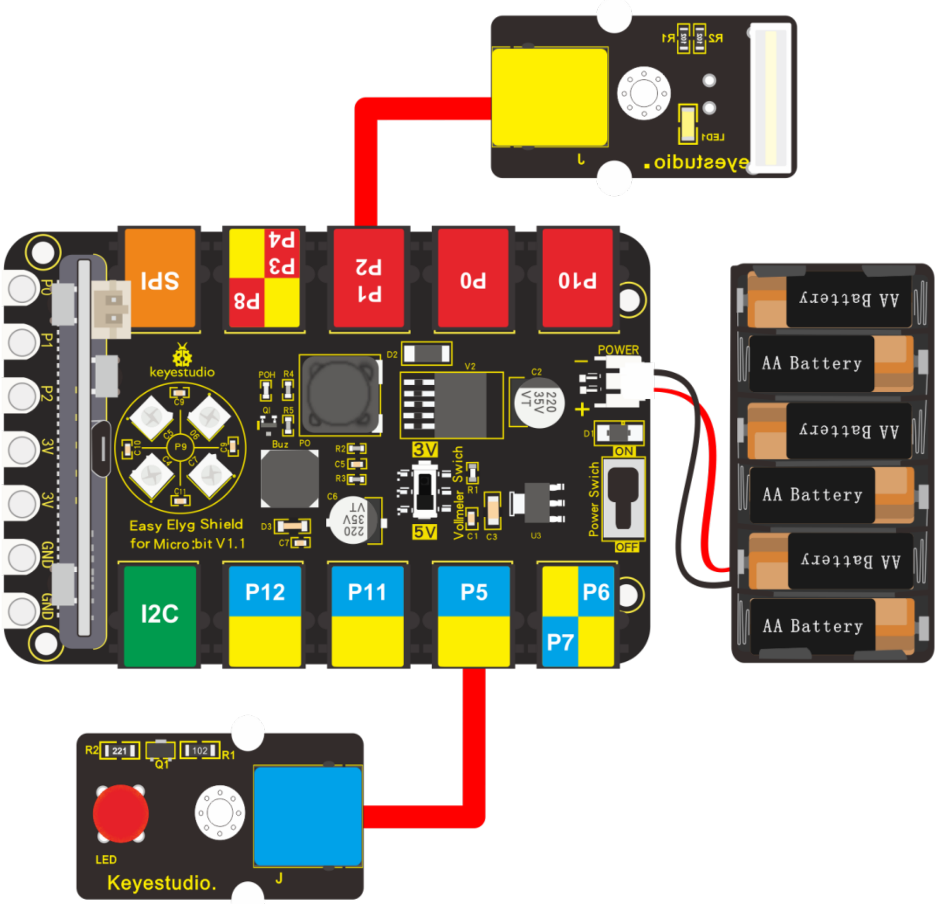
This module should be used together with EASY plug control board.

Specification:

- Interface: Digital
- Working voltage: 5V
- Sensor type: Easy plug

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect knock sensor and red LED module to P1 and P5 port of shield with two RJ11 cables.

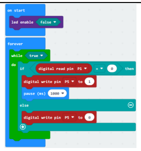


Note: Dial Voltmeter_Switch to 3V end.

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.

 on start led enable false forever while true do if digital read pin P1 = 0 then digital write pin P5 to 1 pause (ms) 1000 else digital write pin P5 to 0	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitThe program under the block "forever" runs cyclically.If it is true, execute the program under do blockIf the digital signal of P1 is 0, module is knocked, execute the program under then blockSet P5 to high level(1), turn on LEDDelay in 1000msIf the digital signal of P1 is 1, module is not knocked, execute the program under else blockSet P5 to low level(0) to turn off LED

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.

When the module is knocked, LED turns on for 1s; if not, LED is off.

7.19 Project 19:Someone Comes

1. Description:

In this experiment, we connect EASY Plug PIR motion sensor to micro:bit and detect the object moving , the detected digital signals will be displayed on 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 PIR Motion Sensor*1
- EASY Plug Red LED Module*1
- RJ11 Cable*2
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug PIR Motion Sensor:

PIR stands for Pyroelectric Infrared (many times, they are also referred as Passive Infrared sensors). This is because their principle of operation is based on the detection of infrared energy emitted by a moving body. The PIR sensor can detect infrared signals from a moving person or moving animal, outputting switching signals.

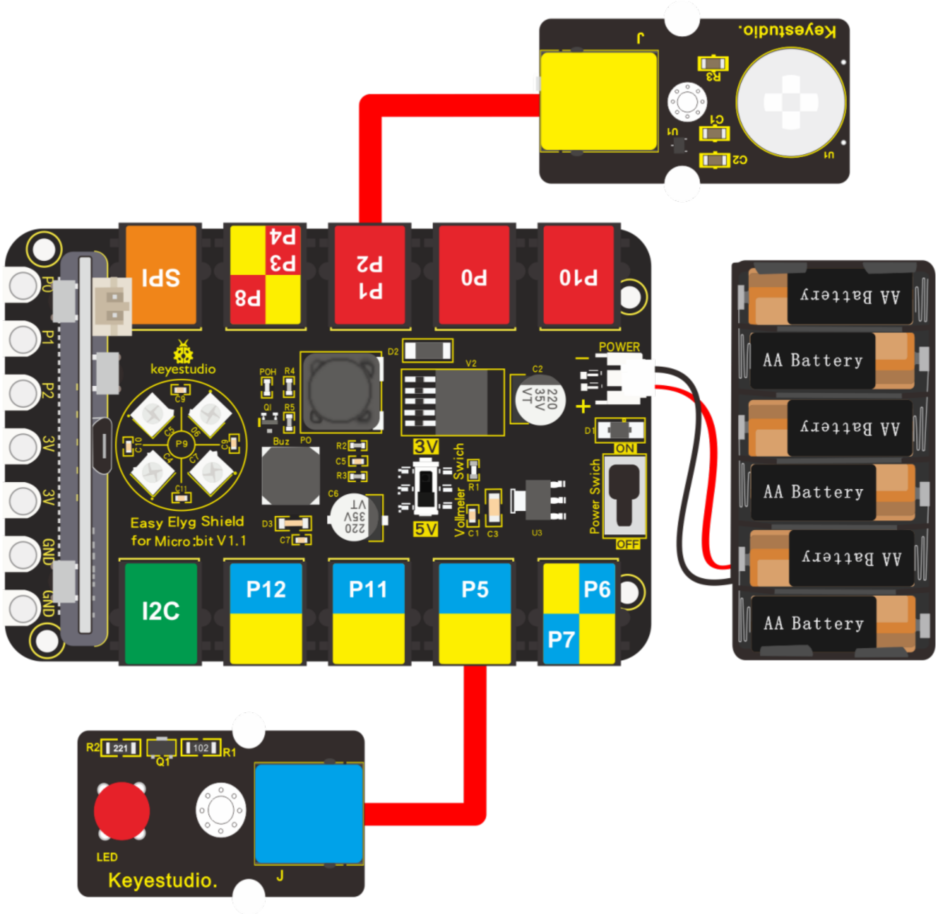
One important thing to mention is that when motion is detected, the output will stay high for 2.3 to 3 seconds after the motion stops. Regarding the power supply, it can work with voltages of both 3.3V and 5V. The device has a detection range of 7 meters and a detection angle of 100º.

Specification:

- Connector: Easy plug
- Input Voltage: 3.3 ~ 5V, Maximum 6V
- Working Current: 15uA
- Working Temperature: -20 ~ 85℃
- Output Voltage: High 3V, Low 0V
- Output Delay Time (High Level): about 2.3 to 3 Seconds
- Detection angle: 100°
- Detection distance: 7 meters
- Output Indicator LED (When output HIGH, it will be ON)
- Pin limit current: 100mA

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect PIR motion sensor and red LED to P1 and P5 port of shield.

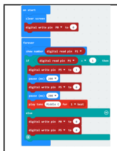


Note: Dial Voltmeter_Switch to 3V end.

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 digital write pin P0 to 0 forever show number digital read pin P1 if digital read pin P1 = 1 then digital write pin P5 to 1 pause (ms) 200 digital write pin P5 to 0 pause (ms) 200 play tone Middle C for 1 beat else digital write pin P0 to 0 digital write pin P5 to 0	“on start”: command block only runs once to start program.Clear LED dot matrixSet P0 to low level(0), turn off passive buzzerThe program under the block“forever”runs cyclically.Micro:bit shows the digital signal read by PIR motion sensor.If digital signal read by P1, detect people’s motion, execute the program under then block.Set P5 to high level(1), LED turns onDelay in 200msSet P5 to low level(0), LED is offDelay in 200msPlay tone C for 1 beat, passive buzzer emits sound.When digital signal read by P1 is 0, nobody is detected, execute the program under else blockSet P0 to low level(0), passive buzzer doesn’t soundSet P5 to low level(0), LED is 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.

When PIR motion sensor detects the movement of PIR motion sensor. LED flashes, passive buzzer emits sound and micro:bit shows high level(1); by contrast, micro:bit shows low level(0), LED is off and buzzer doesn't emit sound.

7.20 Project 20: Capacitive Touch

1. Description:

Are you tired of mechanical buttons? Try the capacitive touch module.

In this lesson, we will replace button switch with capacitive touch module and demonstrate how to control passive buzzer with capacitive buzzer.

2. What You Need:

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

3. EASY Plug Capacitive Touch Module:

Based on touch detection chip, the touch area of capacitive touch module is applied widely on a plenty of touch sensors. This touch sensor features the one-key button function and adopts the most popular capacitive sensing technology. In addition, it can "feel" people and metal touch and feedback a high/low voltage level. Even isolated by some cloth and paper, it can still feel the touch. The sensitivity will decrease as the isolation getting thick.

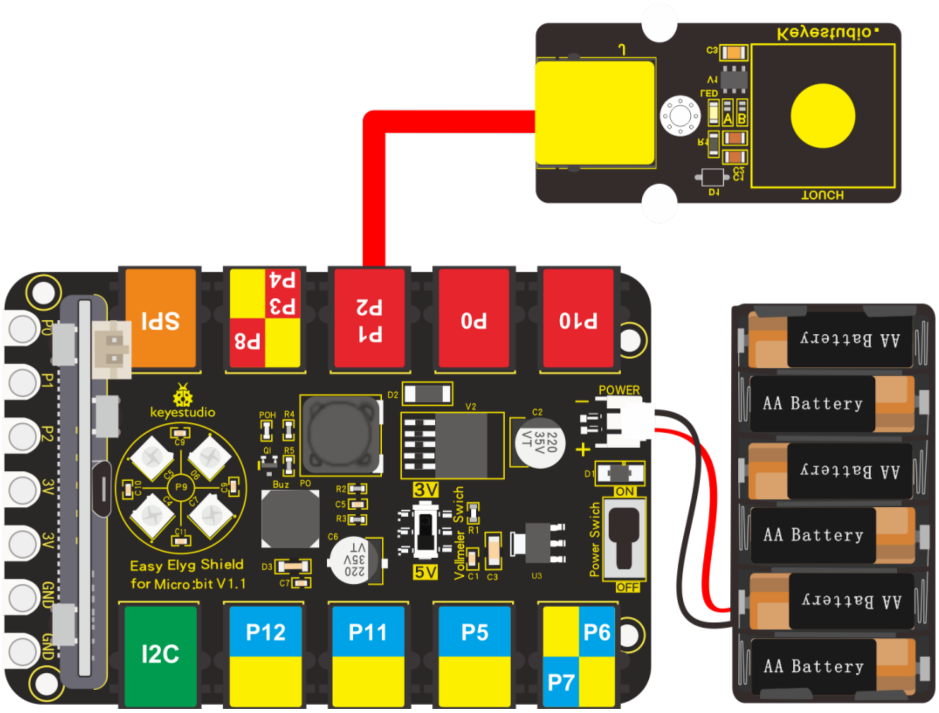
This sensor can tackle the traditional button problems, with the characteristics of low consumption and wide working voltage.

Specification:

- Jog type: the initial state is low, high touch, do not touch is low (similar touch of a button feature);
- Low power consumption;
- Power supply for 3.3 ~ 5V DC;
- Smooth touch surface

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect capacitive touch module to P1 port of shield.

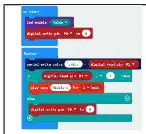


Note: Dial Voltmeter_Switch to 3V end.

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.

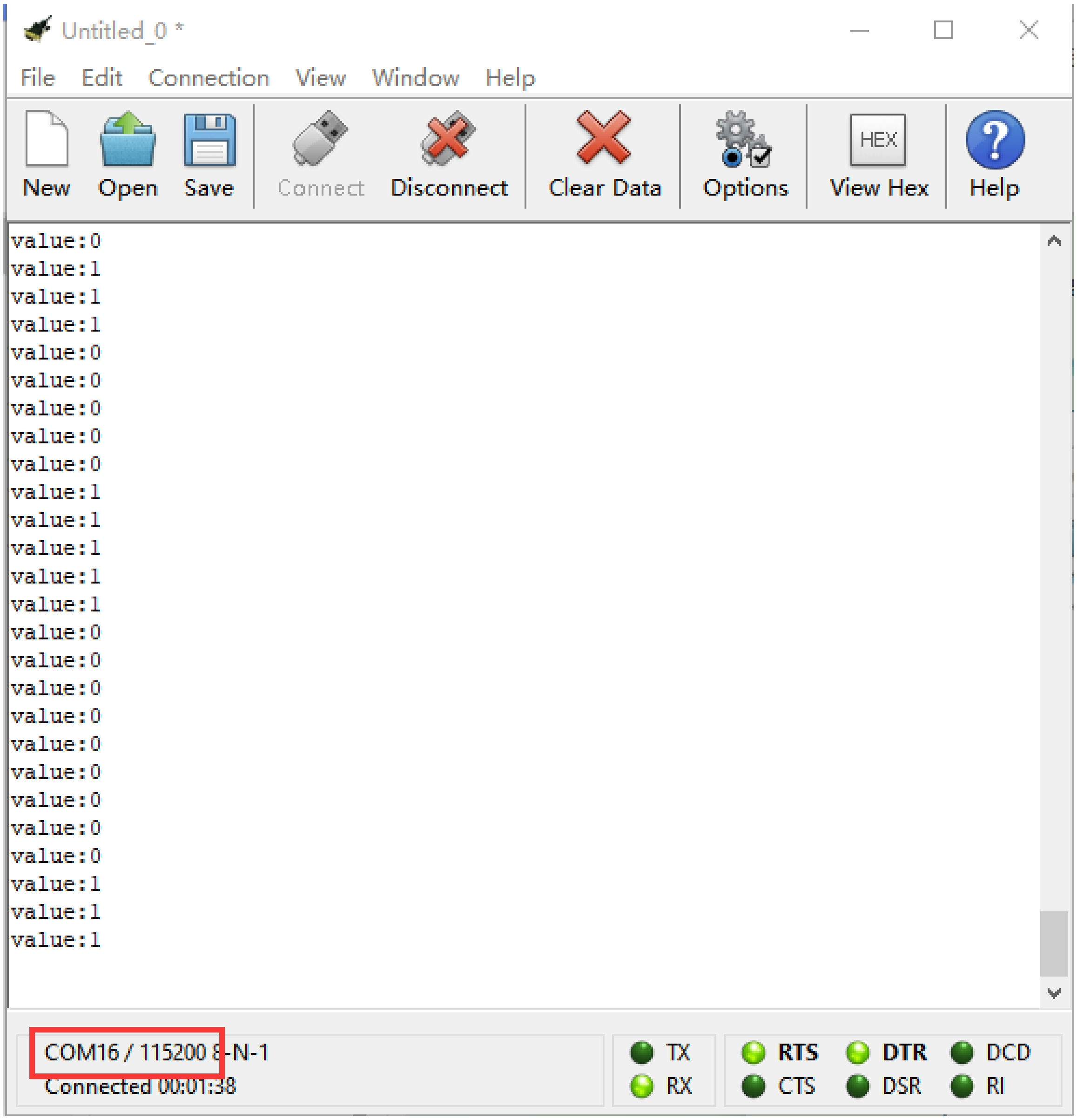
 on start led enable false digital write pin P0 to 0 forever serial write value "value" = digital read pin P1 if digital read pin P1 = 1 then play tone Middle C for 1 beat else digital write pin P0 to 0	“on start”: command block only runs once to start program.Turn off dot matrix on micro:bitSet P0 to low level(0), turn off passive buzzerThe program under the block“forever”runs cyclically.Serial writes the digital signals read by sensorIf the digital signal read by P1 is 1, touch sensor, execute the program under then blockPlay tone C for 1 beat, make passive buzzer emit soundIf the digital signal read by P1 is 0, sensor is not touched, execute the program under else blockSet P0 to low level (0), turn off passive

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.

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

When touch area is touched, CoolTerm monitor shows 1 and passive buzzer emits sound; otherwise, 0 is shown on CoolTerm monitor and passive buzzer doesn’t emit sound, as shown below:



```
value:0
value:1
value:1
value:1
value:0
value:0
value:0
value:0
value:0
value:1
value:1
value:1
value:1
value:1
value:0
value:0
value:0
value:0
value:0
value:0
value:0
value:0
value:0
value:1
value:1
value:1
```

COM16 / 115200 8-N-1
Connected 00:01:38

7.21 Project 21: Obstacle Avoidance

1. Description:

Have you ever seen a smart car avoid the obstacle itself?

Do you know why? Let me explain to you in this chapter.

We will use a passive buzzer, red LED module and an obstacle avoidance sensor in the experiment.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1

- EASY Plug Obstacle Avoidance Sensor*1
- EASY Plug Red LED Module*1
- RJ11 Cable*2
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. Obstacle Avoidance Sensor:

Infrared obstacle avoidance sensor is equipped with distance adjustment function and is especially designed for wheeled robots.

This sensor has strong adaptability to ambient light and is of high precision. It has a pair of infrared transmitting and receiving tube.

When the infrared ray launched by the transmitting tube encounters an obstacle (its reflector), the infrared ray is reflected to the receiving tube, after a comparator circuit processing, the indicator will light up.

You can adjust the detection distance by rotating the potentiometer knob, the effective distance range of 2~40cm.

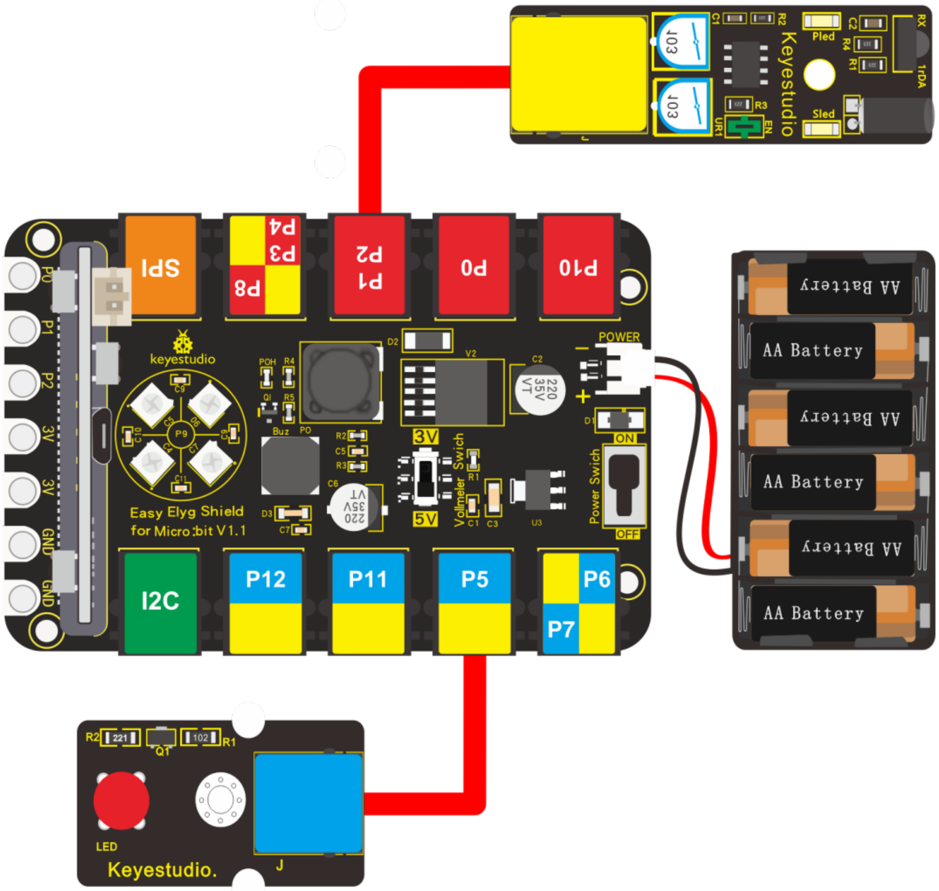
They can be widely used in robot obstacle avoidance, avoidance car, line count, and black and white line tracking and many other occasions.

Specification:

- Working voltage: DC 3.3V-5V
- Working current: 20mA
- Working temperature: -10℃ to50℃
- IO interface: EASY Plug(G(-)/V(+)/out/EN)
- Signal output: TTL voltage
- Detection distance: 2-40cm

4. Wiring Up:

Insert micro:bit onto slot of EASY Plug shield, connect obstacle avoidance sensor and red LED module to P1 and P5 of Easy Plug shield.

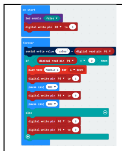


Note: Dial Voltmeter_Switch to 3V end.

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.

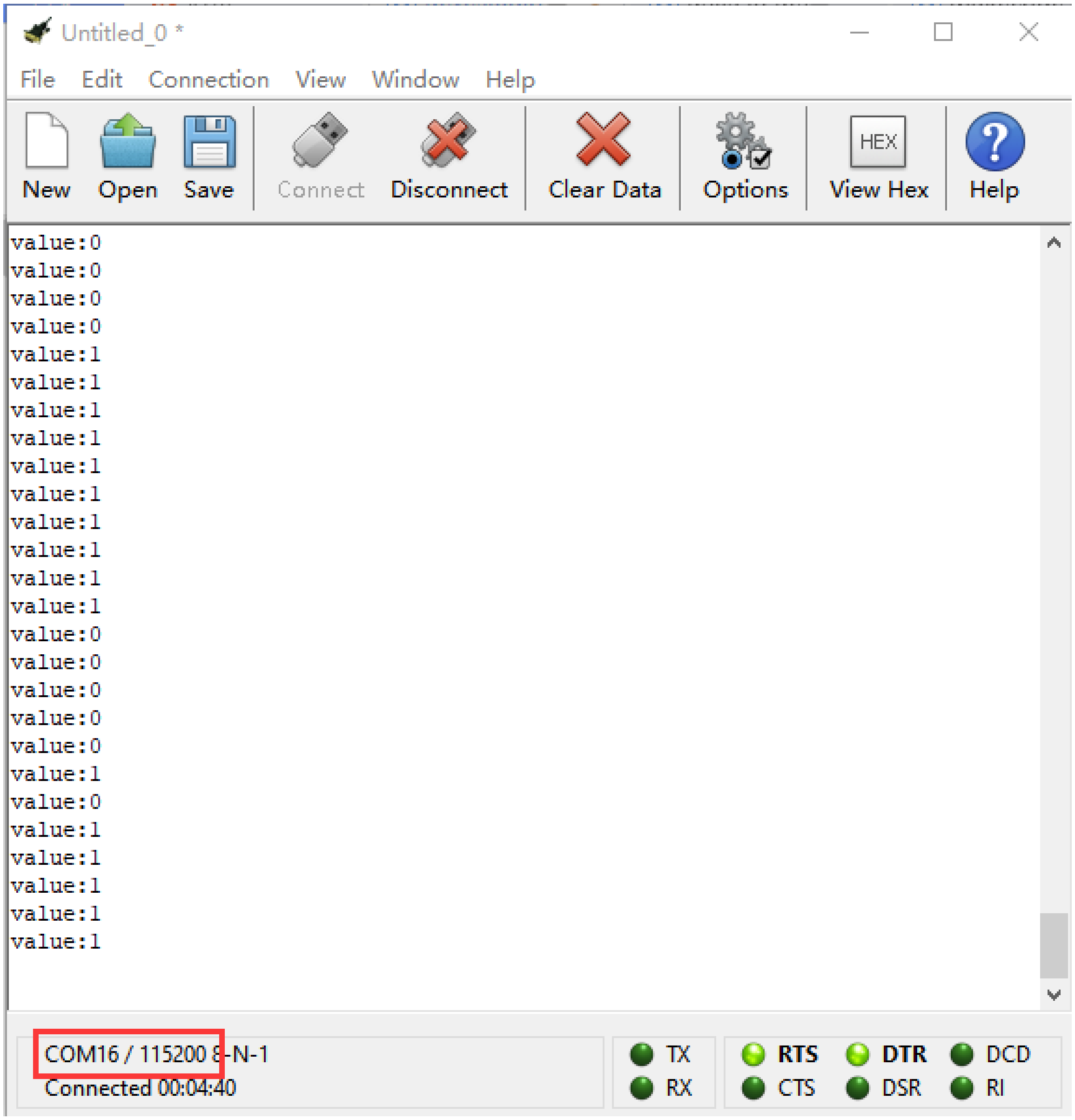
 on start led enable false digital write pin P0 to 0 forever serial write value "value" = digital read pin P1 if digital read pin P1 = 0 then play tone Middle C for 1 beat digital write pin P5 to 1 pause (ms) 100 digital write pin P5 to 0 pause (ms) 100 else digital write pin P0 to 0 digital write pin P5 to 0	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitSet P0 to low level(0), turn off passive buzzerThe program under the block"forever"runs cyclically.Serial writes the digital signal read by obstacle avoidance sensorIf the digital signal=0, detect the obstacle, execute the program under then blockPlay tone C for 1 beat to make passive buzzer emit soundSet P5 to high level(1), turn on LEDdelay in 100msSet P5 to low level(0), turn off LEDdelay in 100msIf the digital signal=1, detect no obstacle, execute the program under else blockSet P0 to low level(0), passive buzzer doesn't emit soundSet P5 to low level(0), turn off LED

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.

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

When the obstacle is detected, CoolTerm monitor shows 0 and passive buzzer emits sound and LED flashes ceaselessly; by contrast, CoolTerm monitor displays 1, LED is off, nor the passive buzzer does emit sound.



```
value:0
value:0
value:0
value:0
value:1
value:1
value:1
value:1
value:1
value:1
value:1
value:1
value:1
value:1
value:0
value:0
value:0
value:0
value:0
value:1
value:0
value:1
value:1
value:1
value:1
value:1
```

COM16 / 115200 8-N-1
Connected 00:04:40

7.22 Project 22: Servo

1. Description:

In this chapter, we will illustrate the principle and application of servo.

2. What You Need:

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

3. EASY Plug Servo Module:

If you want to use the Micro Servo and EASY PLUG control board to do some experiments, you need to use the EASY Plug Servo extension module.

The EASY Plug Servo module is extended into Registered Jack, so you can connect it to EASY PLUG control board using only a RJ11 cable.

The Servo module also comes with 3pins of 2.54mm pin pitch, fully compatible with servo pins.

4. Servo:

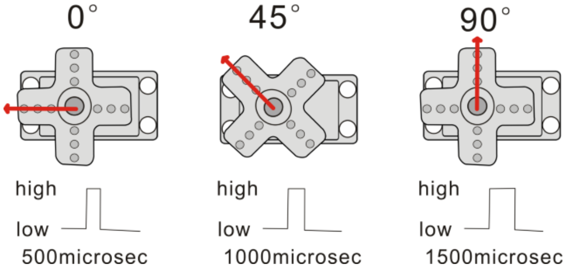
Servo motor comes with many specifications. But all of them have three connection wires, distinguished by brown, red, orange colors. Brown one is for ground, red one for power positive, orange one for signal line.

Included with your Micro Servo you will find a variety of white motor mounts that connect to the shaft of your servo.

You may choose to attach any mount you wish for the circuit. It will serve as a visual aid, making it easier to see the servo spin.

The rotation angle of servo is controlled by regulating the duty cycle of the PWM(Pulse-Width Modulation) signal.

The standard cycle of the PWM signal is fixed at 20ms (50 Hz), and the pulse width is distributed between 1ms-2ms. The pulse width corresponds to the rotation angle (0°90°) of servo.

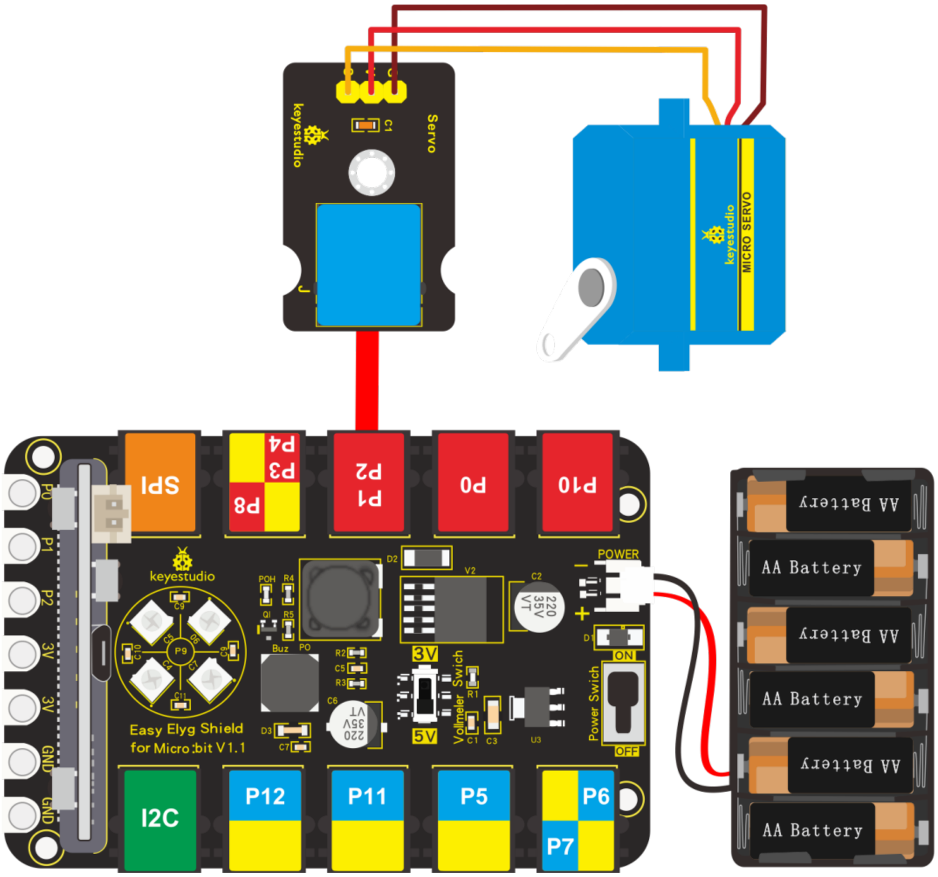


Specification:

- Operating voltage: DC 4.8V6V
- Angle range: about 180°(in 500→2500sec)
- Pulsewidth range: 500→2500sec
- No-load speed: 0.12±0.01 sec/60DC 4.8V; 0.1±0.01 sec/60DC 6V
- No-load current: 200±20mADC 4.8V; 220±20mADC 6V
- Stop torque: 1.3±0.01kg/cmDC 4.8V; 1.5±0.1kg/cmDC 6V
- Stop current: 850mADC 4.8V; 1000mADC 6V
- Standby current: 3±1mADC 4.8V; 4±1mADC 6V
- Operation temperature: -10℃50℃
- Save temperature: -20℃60℃
- Motor wire length: 250 ± 5 mm
- Dimensions: 22.9mm*12.2mm*30mm
- Weight: 9± 1 g (without servo mounts)

5. Wiring Up:

Insert micro:bit onto EASY Plug shield, link servo with servo module. Brown line is connected to G, red line is linked with V and orange line is connected to S. Connect servo module to P1 port of shield.

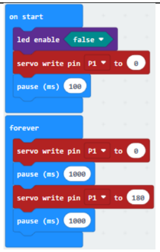


Note:Dial Voltmeter_Switch to 5V end.

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 led enable false servo write pin P1 to 0 pause (ms) 100 forever servo write pin P1 to 0 pause (ms) 1000 servo write pin P1 to 180 pause (ms) 1000	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitRotate P1 to 0°Delay in 100msThe program under "forever" runs cyclically.Rotate P1 to 0°Delay in 1000msRotate P1 to 180°Delay in 1000ms

7. Test Results:

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 view the servo rotate from 0 ° to 180 °

7.23 Project 23: Fan Module

1. Description:

We will make fan module turn clockwise, anticlockwise and stop.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY PlugL9110 Fan Module * 1
- RJ11 Cable*1
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug Fan Module:

EASY PlugL9110 fan module cooperates GND, VCC, INA and INB pin. Pin INA and INB can control the speed and direction of fan.

This fan control module adopts L9110 motor control chip. It can control the rotation direction of the motor, hence the fan. The module is designed with mounting hole, compatible with servo motor control.

The module is of high efficiency, with the high quality fan, it can easily blow out flame of a light in 20cm distance.

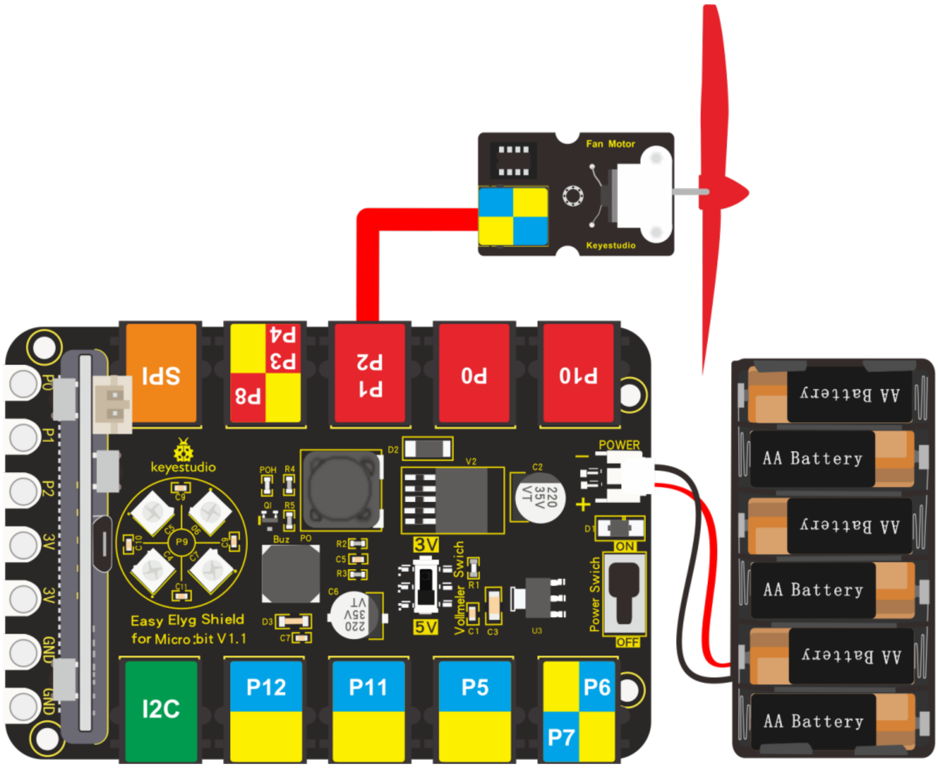
It is widely applied in air propeller, cooling system and spinning frame.

Specification:

- Fan blade diameter: 75mm
- Interface type: dual analog I/O port interface
- Working voltage: 5V

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect fan module to P1-P2 port of shield.

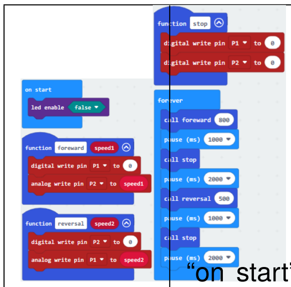


Note: dial Voltmeter_Switch to 5V end.

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.

	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitSet subfucntion foreward speed1Set P1 to low level(0)Set the analog value of P2 to speed 1Set subfucntion reversal speed 2Set P2 to low level(0)Set analog value of P1 to speed 2Subfunction stopSet P1 and P2 to low level(0)The program under "forever" runs cyclically.Call subfunction foreward, set speed to 800Delay in 1000msCall subfunction stop, stopDelay in 2000msCall subfunction reversal, set speed to 500Delay in 1000msSet subfunction stop, stop Call the subDelay in 2000ms

```
on start
    led enable false

function foreward speed1
    digital write pin P1 to 0
    analog write pin P2 to speed1

function reversal speed2
    digital write pin P2 to 0
    analog write pin P1 to speed2

function stop
    digital write pin P1 to 0
    digital write pin P2 to 0

forever
    call foreward 800
    pause (ms) 1000
    call stop
    pause (ms) 2000
    call reversal 500
    pause (ms) 1000
    call stop
    pause (ms) 2000
```

6. Test Results:

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 view that fan module rotate clockwise for 1s, and stop for 2s, then anticlockwise for 1s, and stop for 2s.

7.24 Project 24: Fire Alarm

1. Description:

The violent fire will cause huge economic and human loss if without any effective measures.

You only need a flame sensor that can alarm if sensing the fire.

In this program, we will imitate the fire alarm system with flame sensor, passive buzzer and red LED module.

2. What You Need:

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

3. EASY Plug Flame Sensor:

The flame sensor can be used to detect fire or other wavelength at 760nm~1100nm light.

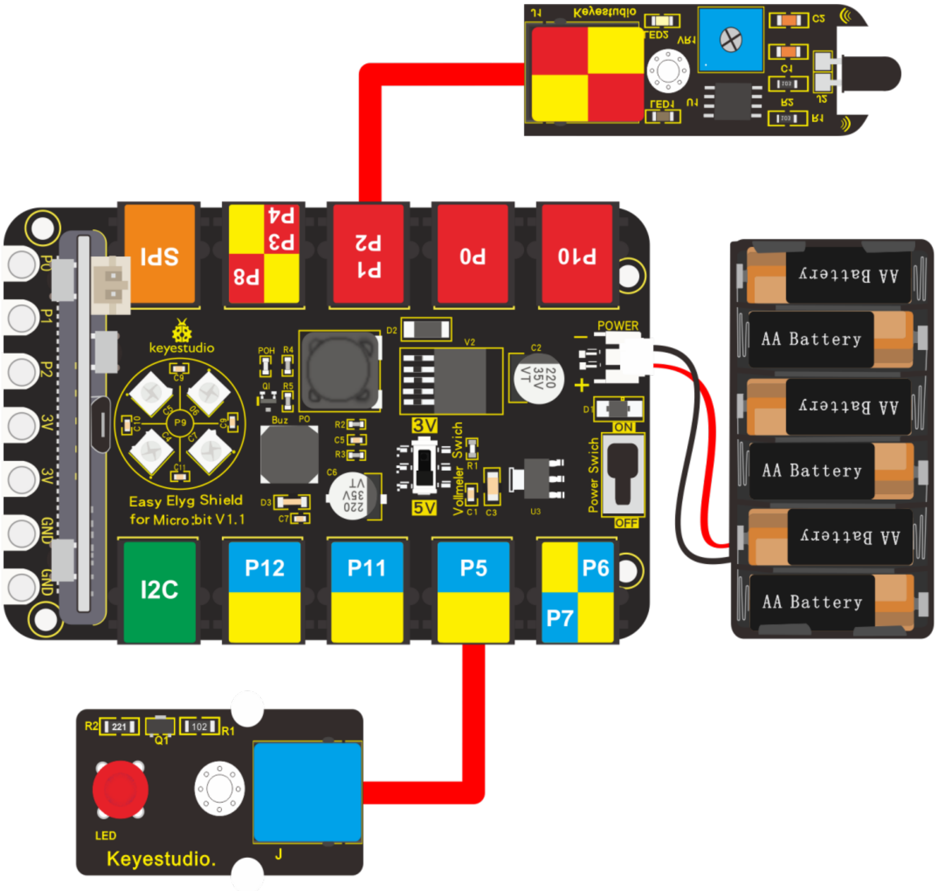
In the fire-fighting robot game, the flame plays an important role in the probe, which can be used as the robot's eyes to find fire source. The potentiometer on the flame sensor can be used to adjust the sensitivity.

Specification:

- Working voltage: 3.3V to 5V
- Detection angle: about 60°
- Detection range: 20cm (4.8V) ~ 100cm (1V)
- Spectral bandwidth: 760nm to 1100nm
- Working temperature: -25℃ to 85℃
- Sensor type: Easy plug
- Interface: digital

4. Wiring Up:

Insert micro:bit onto Easy Plug shield , connect flame sensor and red LED module to P1 and P5 port of shield with RJ11 cables.

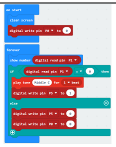


Note: Dial Voltmeter_Switch to 3V end.

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.

 on start clear screen digital write pin P0 to 0 forever show number digital read pin P1 if digital read pin P1 = 0 then play tone Middle C for 1 beat digital write pin P5 to 1 else digital write pin P5 to 0 digital write pin P0 to 0	“on start”: command block only runs once to start program.Turn off dot matrixSet P0 to low level(0), turn off passive buzzerThe program under the block“forever”runs cyclically.Micro:bit shows digital signalsIf the digital signal is 0, there is flame, execute the program under then blockPlay tone C for 1 beat, passive buzzer emits soundSet P5 to high level(1), turn on LEDIf the digital signal is 1, there is no flame, execute the program under else blockSet P5 to low level(0), turn off passive buzzerSet P0 to low level(0), turn off passive buzzer

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 show 0 (low level), LED is on and passive buzzer emits sound; by contrast.

1 appears on micro:bit, LED is off and buzzer doesn’t emit sound.

7.25 Project 25: Flammable Gas in the Air

1. Description:

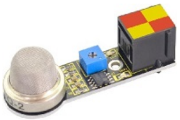
This gas sensor is used in gas leakage detecting equipment in consumer electronics and industrial markets.

This sensor is suitable for detecting LPG, I-butane, propane, methane, alcohol, Hydrogen and smoke. It has high sensitivity and quick response. We will show you how to detect the flammable gas in the air with gas sensor.

2. What You Need:

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

3. EASY Plug Analog Gas Sensor:



This is a robust Gas sensor suitable for sensing **LPG**, **Smoke**, **Alcohol**, **Propane**, **Hydrogen**, **Methane** and **Carbon Monoxide** concentrations in the air. If you are planning on creating an indoor air quality monitoring system; breath checker or early fire detection system, Easy Plug gas Sensor Module is a great choice.

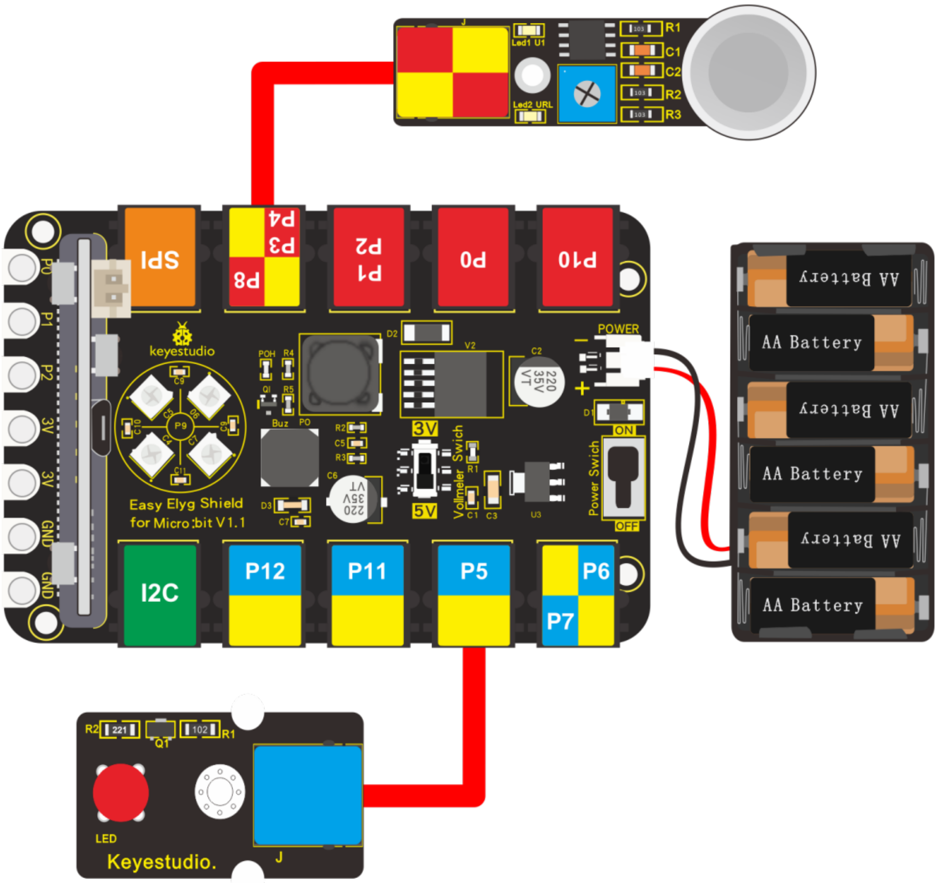
It has two signal terminals-pin A0 and D0. The value of A0 will rise up as the concentration of flammable gas.

Specification:

- Port: Easy plug
- Working Voltage: 5V
- Interface Type: Digital and Analog
- Wide detecting scope
- Simple drive circuit
- Stable and long lifespan
- Quick response and High sensitivity

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect analog gas sensor and red LED module to P4 and P5 port of shield with RJ11 cables.



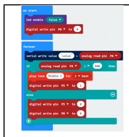
Note: dial Voltmeter_Switch to 5V end.

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.

Notethe gas analog value could be adjusted

 on start led enable false digital write pin P0 to 0 forever serial write value "value" = analog read pin P4 if analog read pin P4 > 600 then play tone Middle C for 1 beat digital write pin P5 to 1 else digital write pin P5 to 0 digital write pin P0 to 0	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitSet P0 to low level(0), turn off passive buzzerThe program under the block"forever"runs cyclically.Serial writes analog signals read by analog gas sensorIf the analog signal read by P4 is greater than 600,execute the program under then blockPlay C tone for 1 beat, passive buzzer emits soundSet P5 to high level(0), turn on LEDIf the analog signals read by P4 is less than and equivalent to 600, execute the program under else blockSet P5 to low level(0), turn off LEDSet P0 to low level(0), turn off passive buzzer

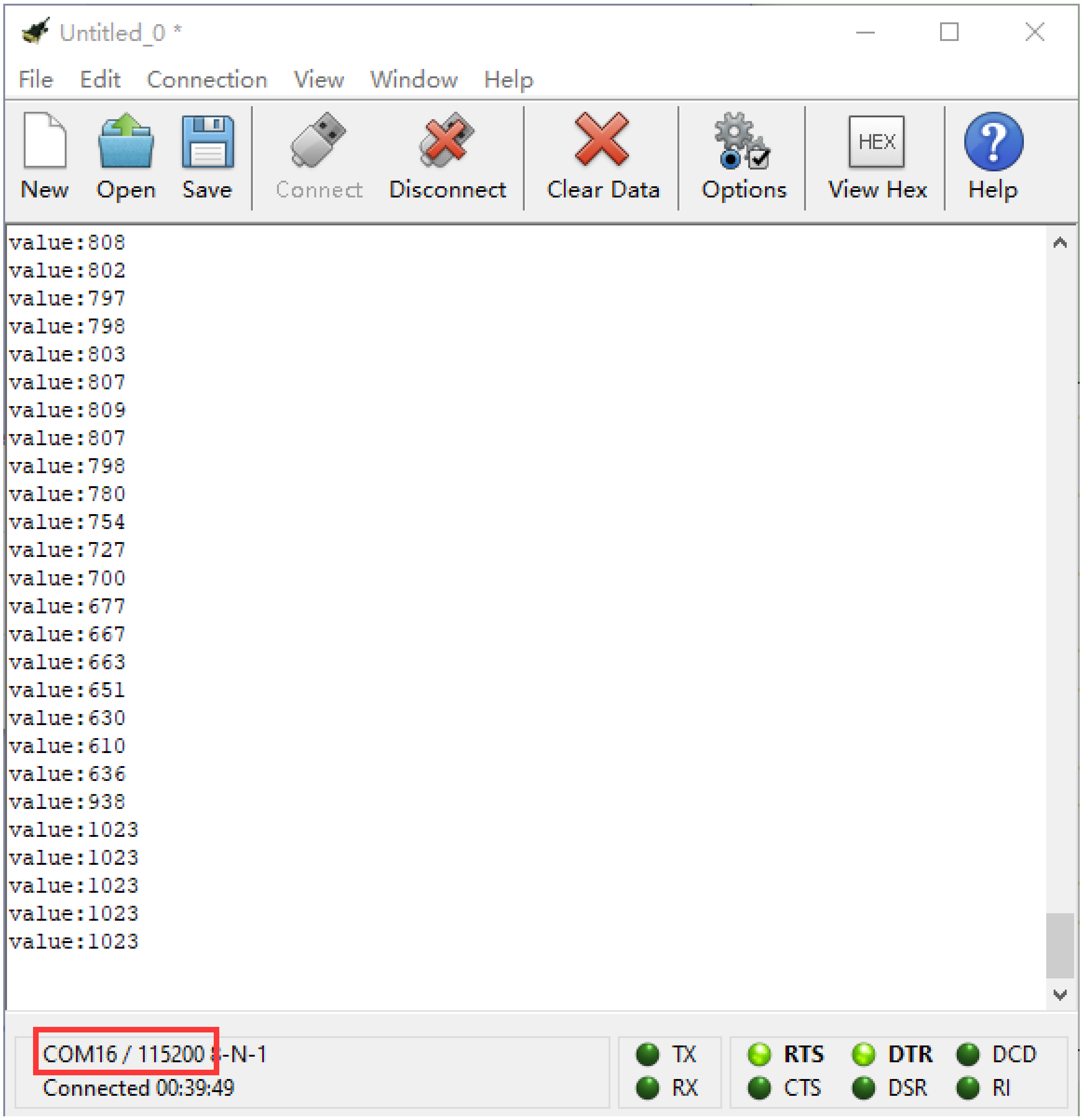
6. Test Results:

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 on gas sensor will be on, you could adjust the sensitivity(make LED at on-and-off state) with blue potentiometer.

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

Putt the firelighter close to analog gas sensor, the analog value on CoolTerm monitor will get larger and LED1 will be on. When sensor detects the analog value more than 600, buzzer will emit sound; otherwise, buzzer is silent and LED is off.



```
value:808
value:802
value:797
value:798
value:803
value:807
value:809
value:807
value:798
value:780
value:754
value:727
value:700
value:677
value:667
value:663
value:651
value:630
value:610
value:636
value:938
value:1023
value:1023
value:1023
value:1023
value:1023
```

COM16 / 115200 8-N-1
Connected 00:39:49

7.26 Project 26: Ambient Light

1. Description:

In this project, we will detect the ambient brightness with EASY PlugTEMT6000 light sensor.

2. What You Need:

- Micro:bit Board*1
- EASY PlugShield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug TEMT6000 Ambient Light Sensor*1
- RJ11 Cable*1
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug TEMT6000:

Ambient Light Sensor:

The TEMT6000 was designed as an ambient light detector for automatically controlling the backlight dimming of cell phones, laptops, car dashboards and similar items. It can be used in many applications where it is desirable to measure the relative brightness of the light falling on the sensor.

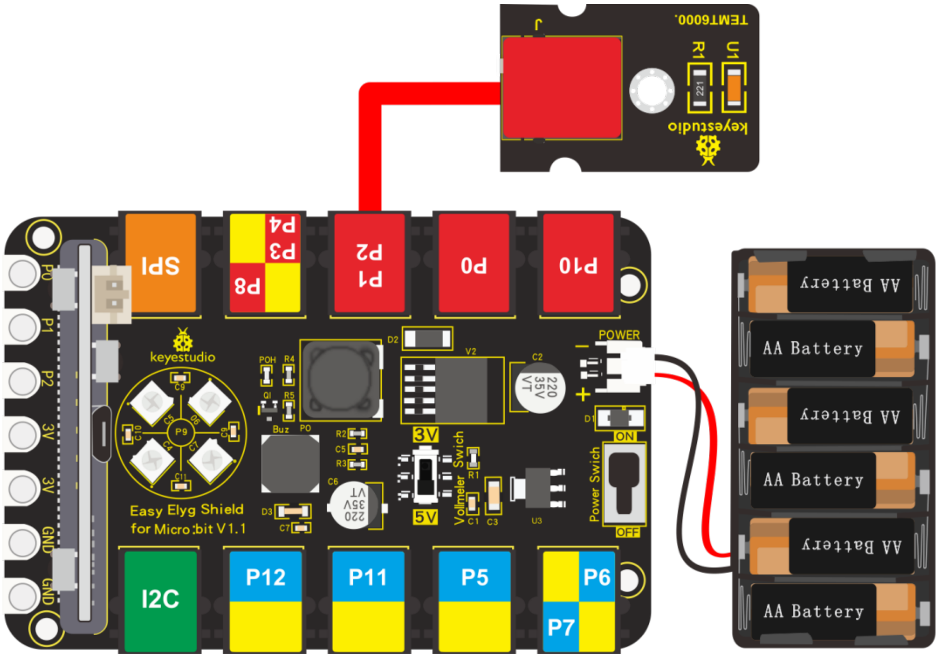
The sensor is designed to mainly detect the light spectrum visible to the human eye with peak sensitivity at 570nm which is in the green spectrum. The full range spans 440nm to 800nm.
It does react well to very small changes in a wide range of brightness, however, it does not react well to IR or UV light. The sensor can help you to to detect the light density.

Specification:

- Working voltage: DC +5v
- Communication interface: analog voltage
- Only sensitive to visible light, no need for additional filters
- Recognizable light intensity range: 1 – 1000 Lux
- Good linear output

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect EASY Plug TEMT6000 ambient sensor to P1 port of shield with a RJ11 cable.

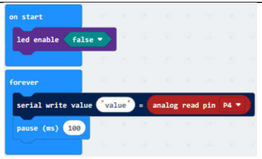


Note: Dial Voltmeter_Switch to 5V end.

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.

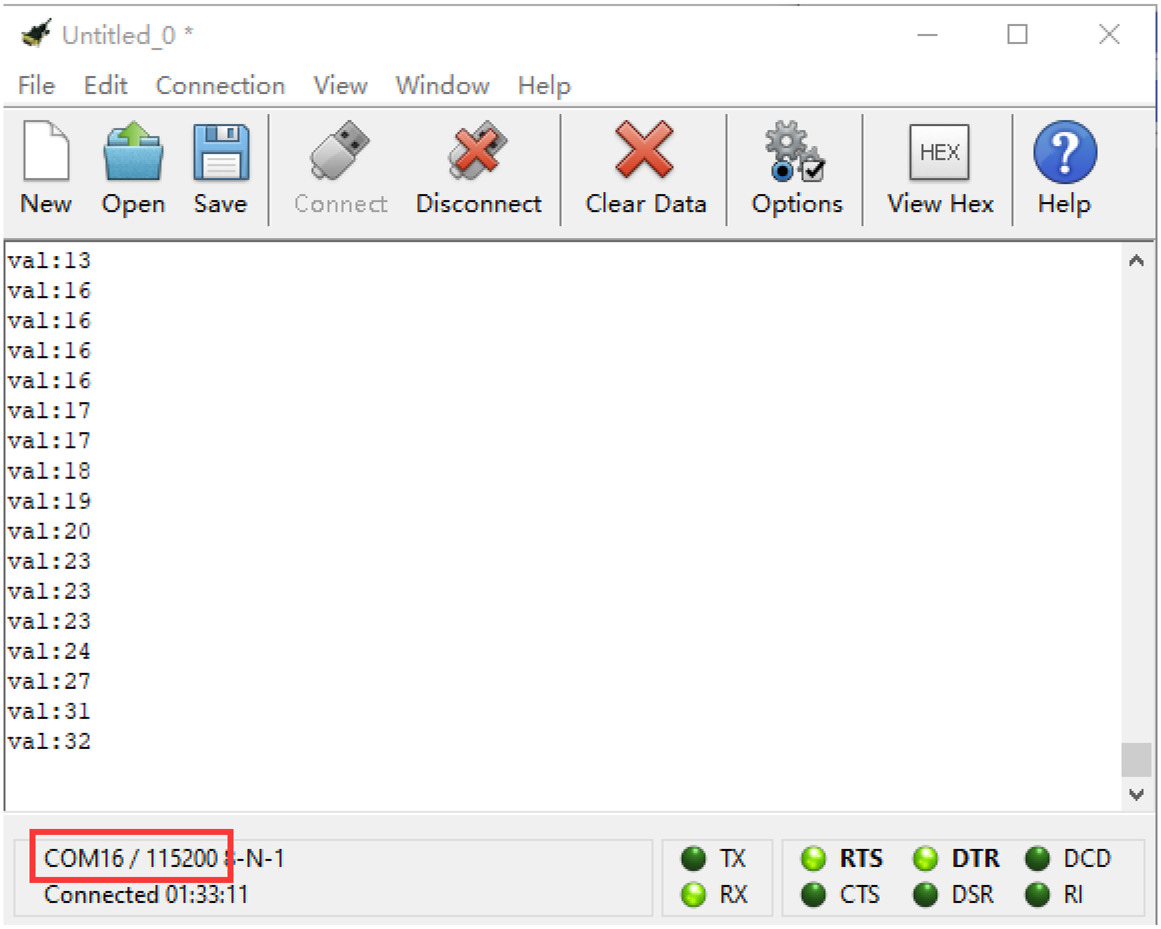
 on start led enable false forever serial write value "value" = analog read pin P4 pause (ms) 100	“on start”: command block only runs once to start program.Turn off dot matrix on micro:bitThe program under the block “forever” runs cyclically.Serial writes analog signals read by alcohol sensorDelay in 100ms

6. Test Results:

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.

The stronger the light is , the larger the analog value is; by contrast, the smaller the analog value is, as shown below:



```
val:13
val:16
val:16
val:16
val:16
val:17
val:17
val:18
val:19
val:20
val:23
val:23
val:23
val:24
val:27
val:31
val:32
```

COM16 / 115200 8-N-1
Connected 01:33:11

7.27 Project 27: Slide Position

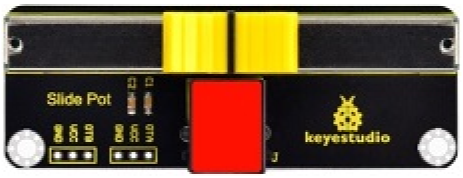
1. Description:

You will learn how to use slide potentiometer to control LED and servo.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Slide Potentiometer*1
- EASY Plug Red LED Module*1
- EASY Plug Servo*1
- Keyestudio Servo*1
- RJ11 Cable*3
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug Slide Potentiometer:



The EASY plug slide potentiometer uses high-quality sliding appliances for stable and reliable performance. It is a dual analog output that outputs a 0-VCC analog voltage signal.

The module pins are extended into Registered jack, so you can easily connect it to EASY Plug control board using a RJ11 cable. There are 6 pad interfaces on the module. So you can solder two 3pin headers with a pitch of 2.54mm on the module.

It can be used to connect with other MCUs. The signal terminal outputs two analog values. The sum of the two analog values is 1023.

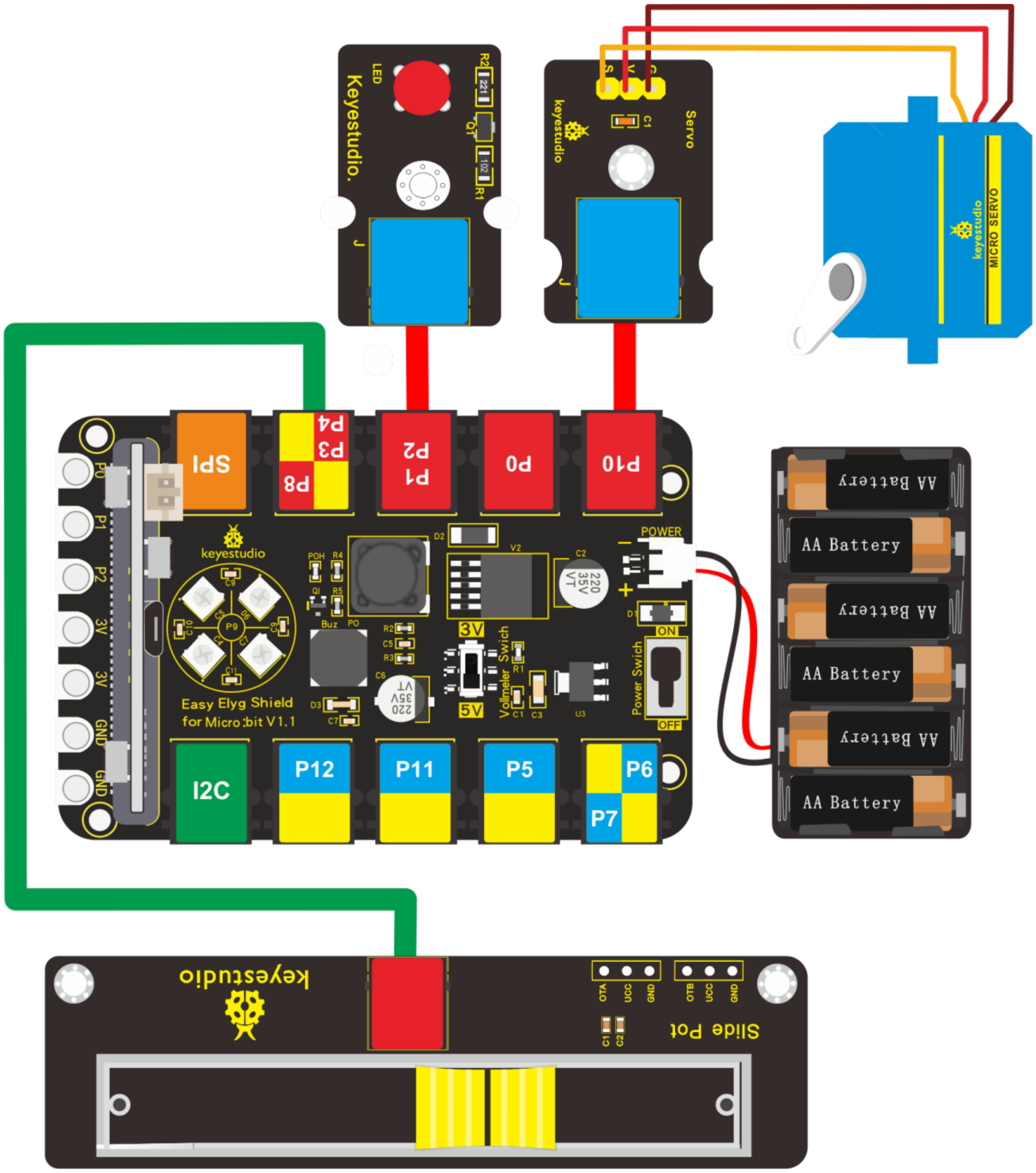
Specification:

- Working voltage: DC 3.3V-5V
- Resistance: 10K
- Interface: analog interface
- Attribute: ROHS

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect slide potentiometer, red LED module and servo module to P3-P4, P1 and P10 port of shield with RJ11 cables.

The brown line, red line and orange line of servo are respectively connected to G, V and S.

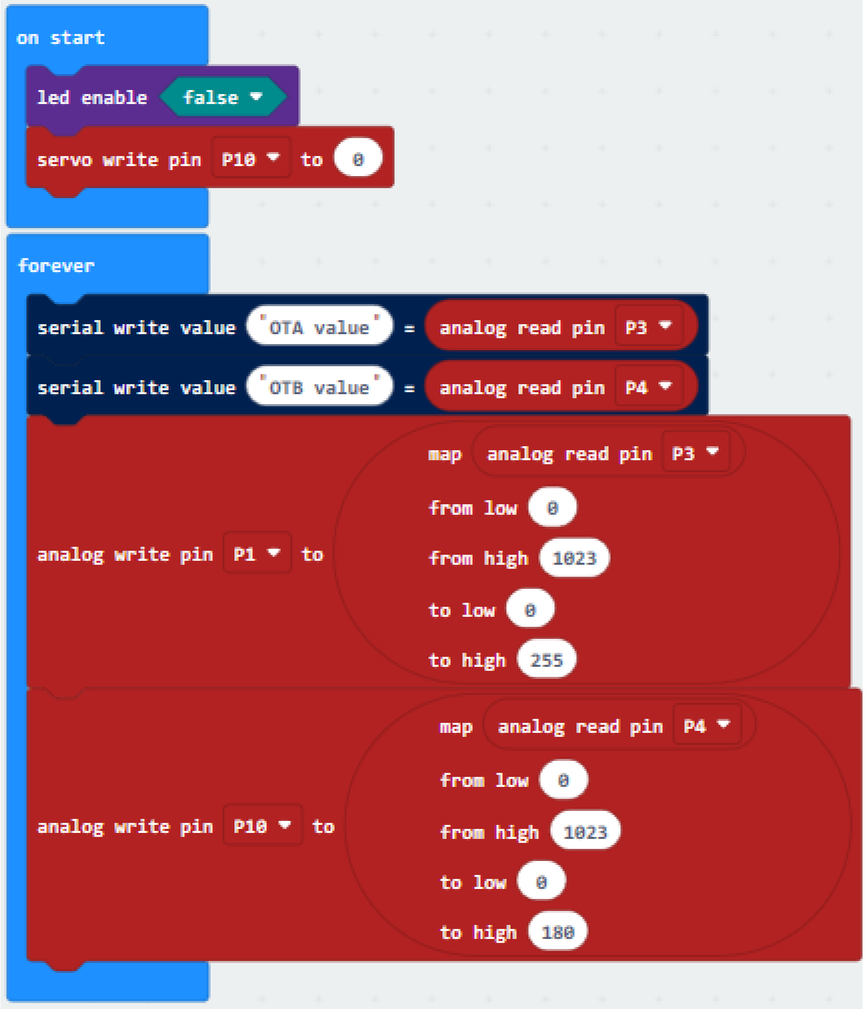


Note: Dial Voltmeter_Switch to 5V end.

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.



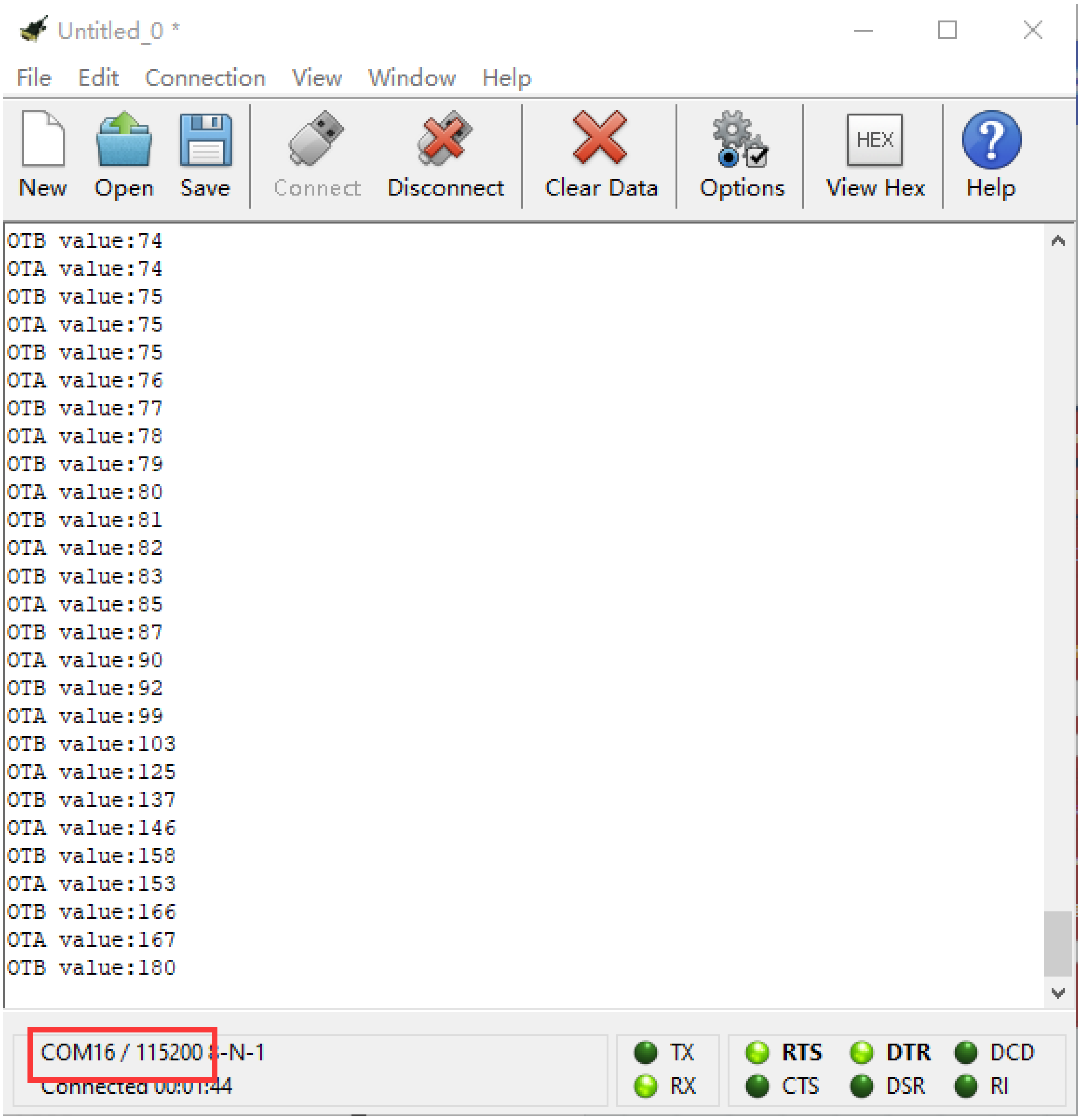
```
on start
    led enable false
    servo write pin P10 to 0
forever
    serial write value "OTA value" = analog read pin P3
    serial write value "OTB value" = analog read pin P4
    analog write pin P1 to map analog read pin P3
        from low 0
        from high 1023
        to low 0
        to high 255
    analog write pin P10 to map analog read pin P4
        from low 0
        from high 1023
        to low 0
        to high 180
```

6.Test Results:

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 displays analog value of pin P3 and P4, if slide the potentiometer, value will alter in 0-1023, and servo will rotate in the range of 0°-180°, as shown below:



```
OTB value:74
OTA value:74
OTB value:75
OTA value:75
OTB value:75
OTA value:76
OTB value:77
OTA value:78
OTB value:79
OTA value:80
OTB value:81
OTA value:82
OTB value:83
OTA value:85
OTB value:87
OTA value:90
OTB value:92
OTA value:99
OTB value:103
OTA value:125
OTB value:137
OTA value:146
OTB value:158
OTA value:153
OTB value:166
OTA value:167
OTB value:180
```

COM16 / 115200 8-N-1
Connected 00:01:44

7.28 Project 28: Light Brightness

1. Description:

Sensors are everywhere in our life, street lights will turn on at night but off at day, why? In fact, it is because the photosensitive element that can sense the ambient light brightness. In this program, we will control the light brightness by photoresistor module.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Photoresistor*1

- EASY Plug Red LED Module*1
- RJ11 Cable*2
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug Photoresistor:

A photoresistor or light-dependent resistor (LDR) or photocell is a light-controlled variable resistor. The principal is very simple. The resistance of photoresistor varies with incident light intensity. If the incident light intensity is high, the resistance decreases; if the light intensity is low, the resistance increases.

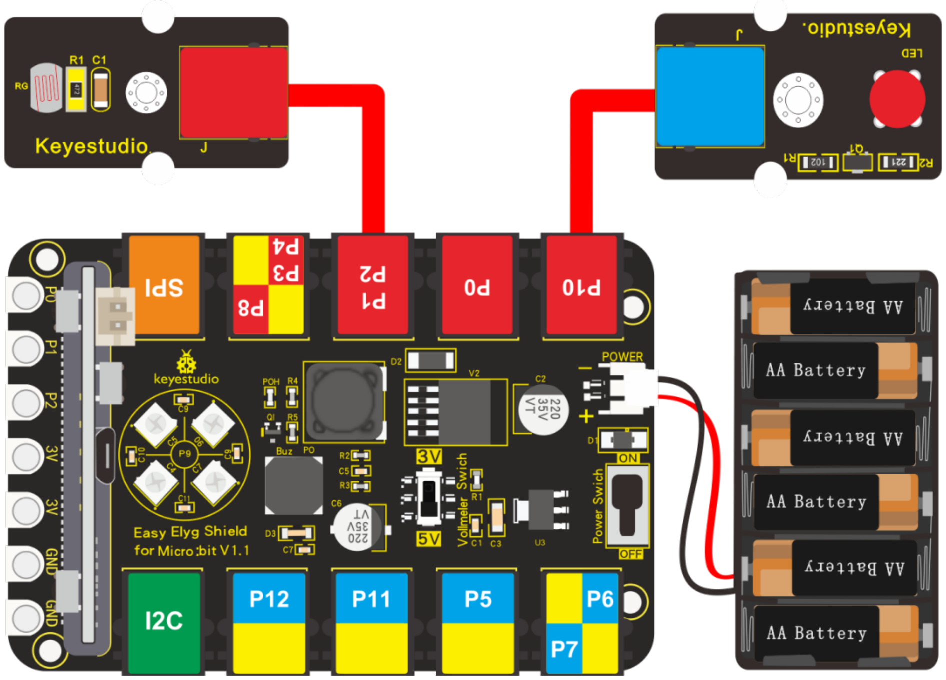
keyestudio EASY Plug photocell sensor is a semiconductor, integrated with a photoresistor, easy to use and very convenient for wiring.

It has features of high sensitivity, quick response, spectral characteristic and R-value consistence.

It can be applied in light-sensitive detector circuits, intelligent switch design and light- and dark-activated switching circuits.

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect photoresistor and red LED module to P1 and P10 port of shield with RJ11 cables.

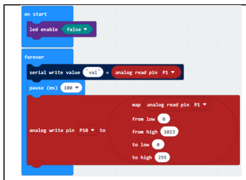


Note: Dial Voltmeter_Switch to 3V end.

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.

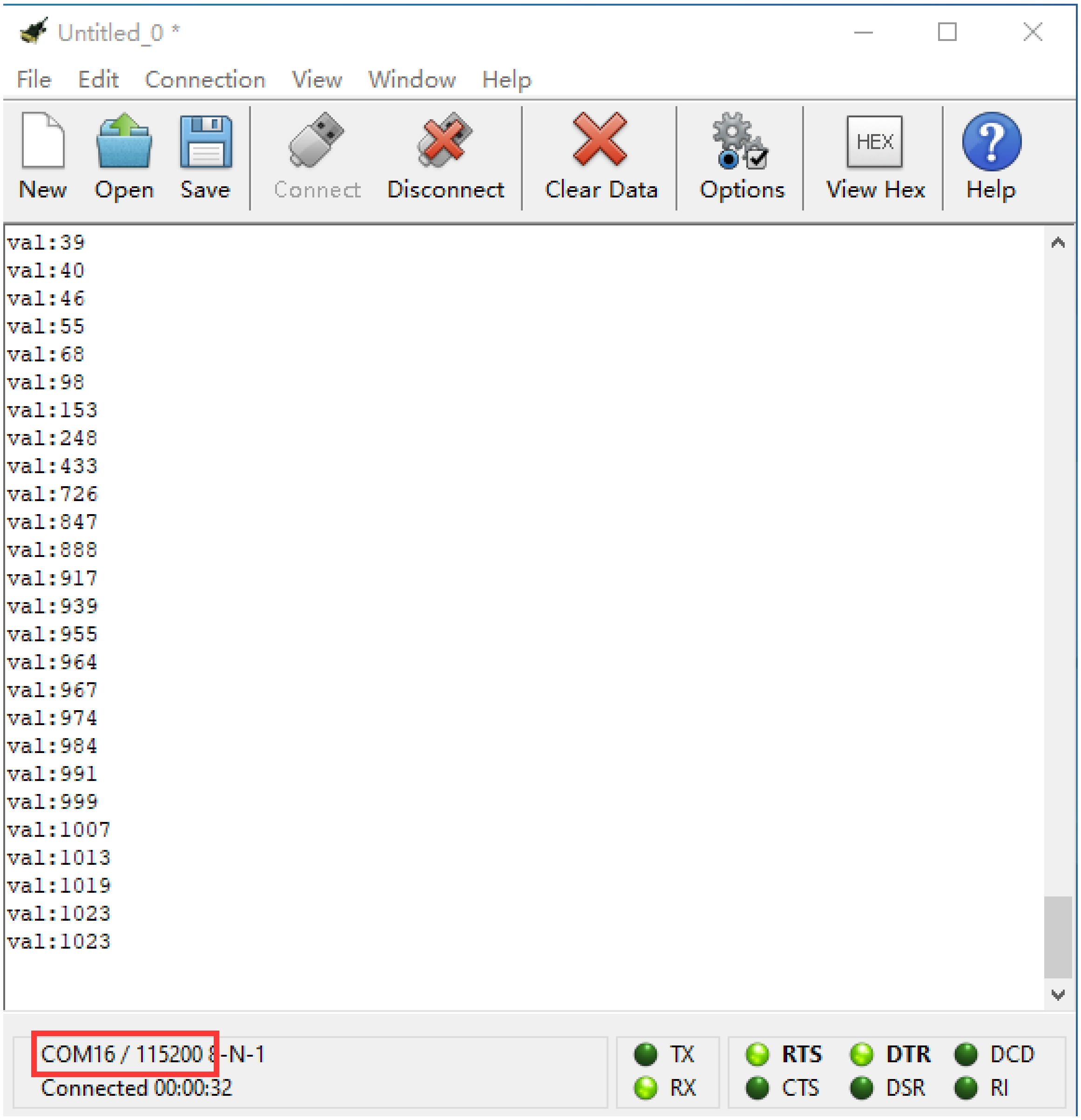
 on start led enable false forever serial write value "val" = analog read pin P1 pause (ms) 100 analog write pin P10 to map analog read pin P1 from low 0 from high 1023 to low 0 to high 255	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitThe program under the block "forever" runs cyclically.Serial writes the analog signals detected by photoresistorDelay in 100msSet the detected analog signal(0-1023) by photoresistor and map the analog value of LED connected P10.

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.

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

As the ambient light brightness increases, so does the analog value on CoolTerm monitor, red LED gradually gets brighter; by contrast, analog value reduces and LED gets dimmer.



```
val:39
val:40
val:46
val:55
val:68
val:98
val:153
val:248
val:433
val:726
val:847
val:888
val:917
val:939
val:955
val:964
val:967
val:974
val:984
val:991
val:999
val:1007
val:1013
val:1019
val:1023
val:1023
```

COM16 / 115200 8-N-1
Connected 00:00:32

7.29 Project 29: OLED Display

1. Description:

OLED module is applied widely in mobile devices, in this project, we will demonstrate how to display numbers, context and pictures with OLED module.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug OLED Module*1

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

3. EASY Plug OLED Module:

OLED is short for organic light emitting diode. On the microscopic level, an OLED display is a matrix of organic LEDs that light up when they emit energy.

Our EASY Plug OLED displays are perfect when you need a small display with vivid, high-contrast color.

The visible portion of the OLED measures 0.96” diagonal and contains 128 x 64 pixels.

An OLED display works without a backlight. Thus, it can display deep black levels and can be thinner and lighter than a liquid crystal display (LCD).

In low ambient light conditions such as a dark room an OLED screen can achieve a higher contrast ratio than an LCD.

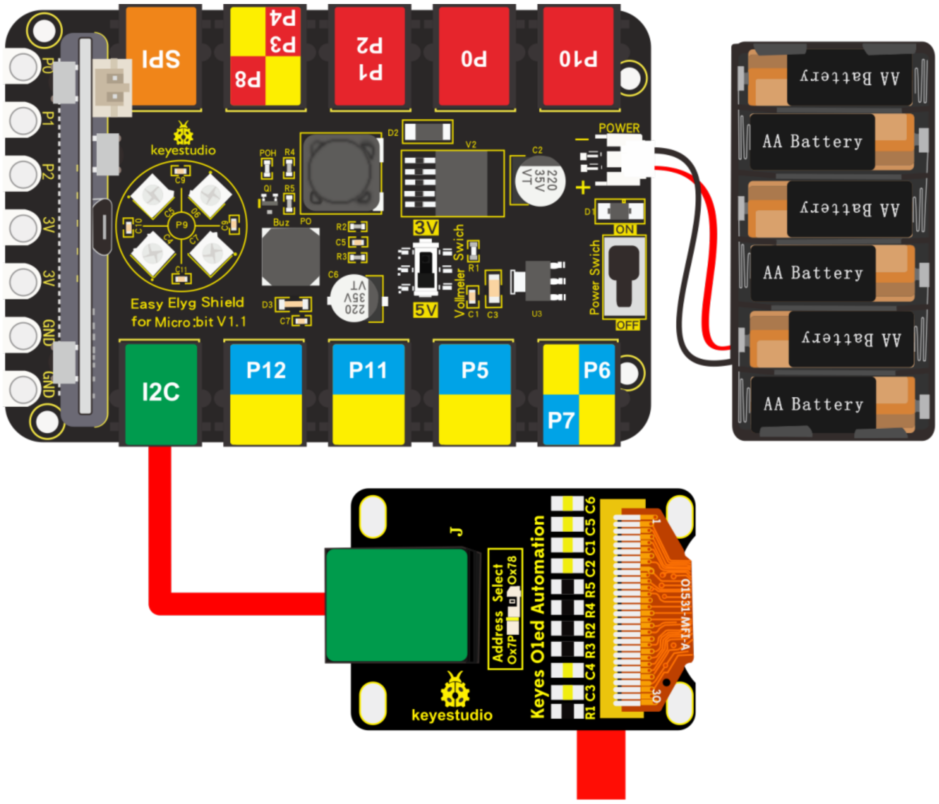
OLED technology is used in commercial applications such as displays for mobile phones and portable digital media players, car radios and digital cameras among others.

Specification:

- 0.96” diagonal OLED
- Pixels: 128 × 64
- Color Depth: Monochrome (White)
- 5V power
- Brightness (cd/m2): 100 (Typ)

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect OLED module to I2C port of shield with a RJ11cable.



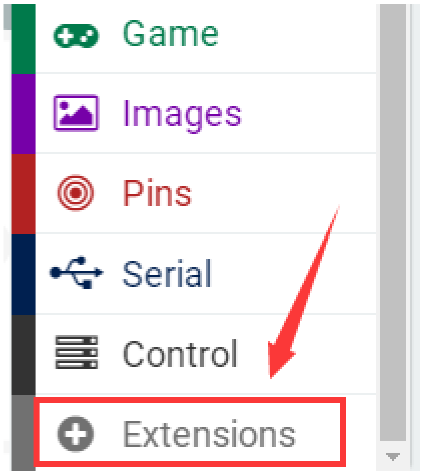
Note: Dial Voltmeter_Switch to 5V end.

5. Test Code:

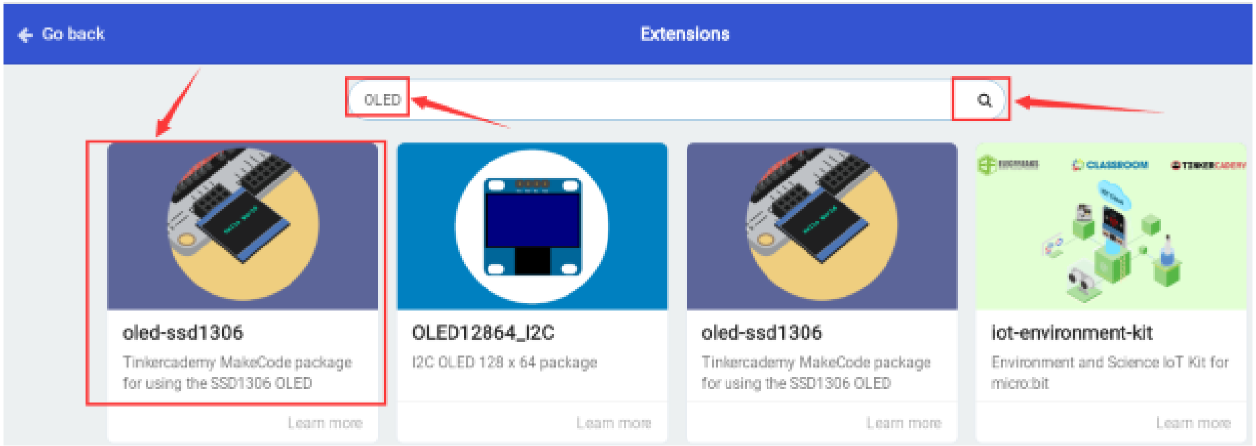
You could navigate https://makecode.micro:bit.org/reference to have access to more details.

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

Next, we need to add the library file of OLED module:



Search “OLED”, as shown below, select the first “oled -ssd1306” library file, and install library.



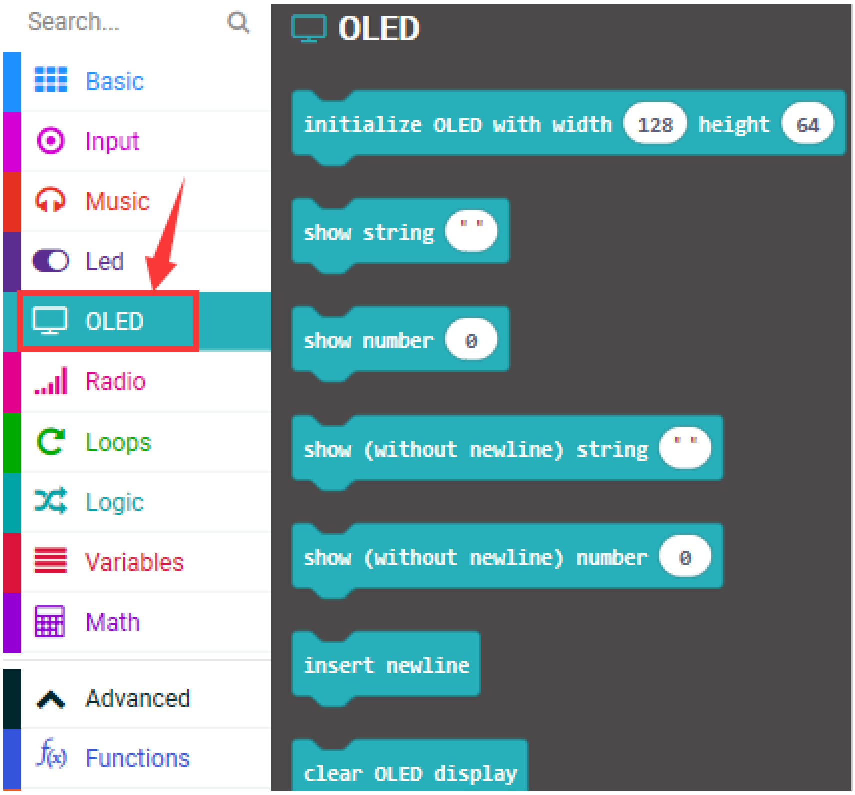
oled-ssd1306
Tinkercademy MakeCode package for using the SSD1306 OLED

OLED12864_I2C
I2C OLED 128 x 64 package

oled-ssd1306
Tinkercademy MakeCode package for using the SSD1306 OLED

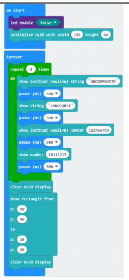
iot-environment-kit
Environment and Science IoT Kit for micro:bit

After the installation, the library of OLED module will be displayed in the listed blocks on Makecode editor.



```
initialize OLED with width 128 height 64
show string " "
show number 0
show (without newline) string " "
show (without newline) number 0
insert newline
clear OLED display
```

Complete Code:

 on start led enable false initialize OLED with width 128 height 64 forever repeat 4 times do show (without newline) string "ABCDEFGHIJK" pause (ms) 500 show string "LMNOPQRST" pause (ms) 500 show (without newline) number 123456789 pause (ms) 500 show number 10111213 pause (ms) 500 clear OLED display draw rectangle from: x: 90 y: 90 to x: 20 y: 20 clear OLED display	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitOLED initializes, 128 in width, 64 in heightThe program under "forever" runs cyclically.Repeat 4 times OLED shows character string "ABCDEFGHIJK"Delay in 500msOLED shows character string "LMNOPQRST"Delay in 500msOLED displays "123456789"Delay in 500msOLED shows "10111213"Delay in 500msClear OLED displayDraw rectangular from x:90 y:90 to x:20 y:20Clear OLED display

6. Test Results:

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.

OLED display will show the corresponding characters and pictures.

7.30 Project 30: Water Your Plant

1. Description:

This is a simple soil humidity sensor aims to detect the soil humidity.

If the soil is in lack of water, the analog value output by the sensor will decrease; otherwise, it will increase. If you use this sensor to make an automatic watering device, it can detect whether your botany is thirsty to prevent it from withering when you go out.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug Soil Humidity Sensor*1
- EASY Plug OLED Module*1
- RJ11 Cable*2
- Slot AA Battery Holder*11.5V AA Battery*6

3. EASY Plug Soil Humidity Sensor:

The EASY Plug soil moisture sensor can read the amount of moisture present in the soil surrounding it. It's an ideal for monitoring an urban garden, or your pet plant's water level.

This soil moisture sensor uses the two probes to pass current through the soil, and then it reads that resistance to get the moisture level.

More water makes the soil conduct electricity more easily (less resistance), while dry soil conducts electricity poorly (more resistance).

If you use this sensor to make an automatic watering device, it will be helpful to remind you to water your indoor plants or to monitor the soil moisture in your garden.

Specification:

- Working voltage: 3.3V-5V
- Working current: 20mA
- Output voltage: 0-2.3V (The greater the humidity, the higher the output voltage.)
- Sensor type: Analog output

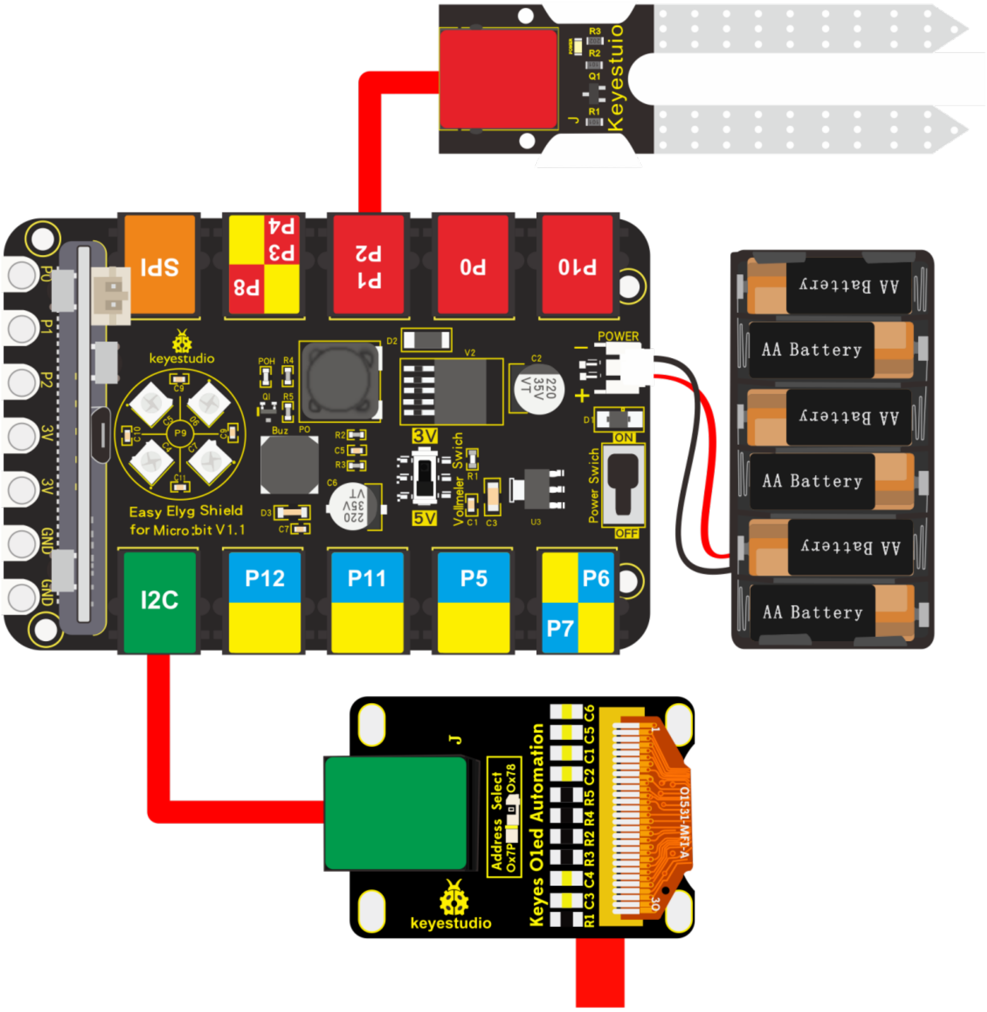
Automotive Watering System:

We connect a soil humidity sensor and a relay module to MCU.

Separately link water pump and power on NO end of relay module. When the soil lacks the water, MCU controls relay module(NO is connected) to drive water pump to water plant; if the soil is wet enough, NO is cut off, water pump stops watering plant.

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect soil humidity sensor and OLED module to P1 and I2C port of shield.

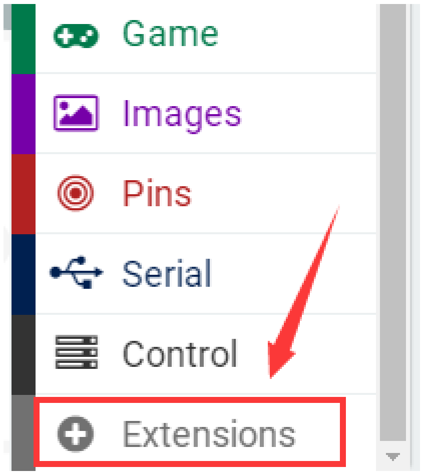


Note: Dial Voltmeter_Switch to 5V end.

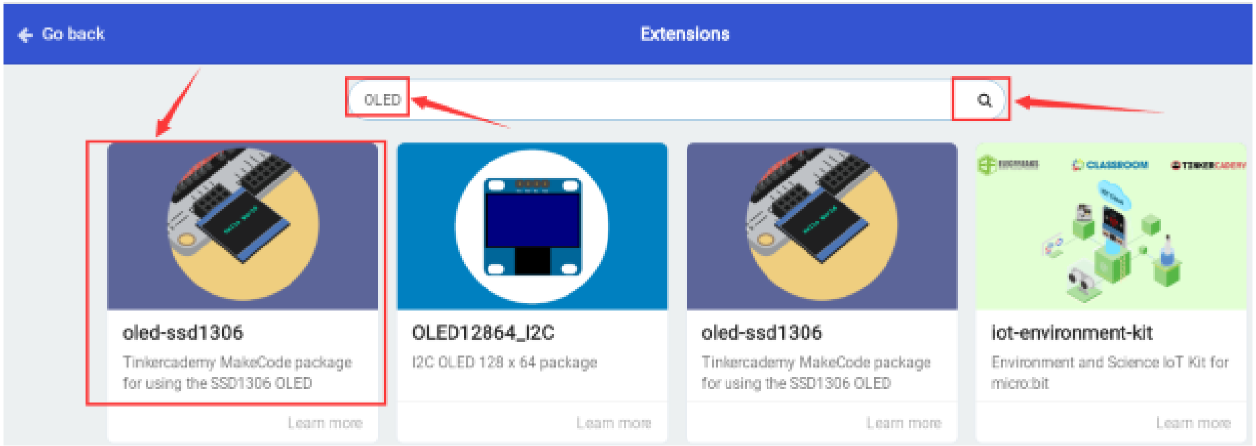
5. Test Code:

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.



Search “OLED”, as shown below, select “oled -ssd1306” library and install it.



Go back

Extensions

OLED

oled-ssd1306

Tinkercademy MakeCode package for using the SSD1306 OLED

Learn more

OLED12864_I2C

I2C OLED 128 x 64 package

Learn more

oled-ssd1306

Tinkercademy MakeCode package for using the SSD1306 OLED

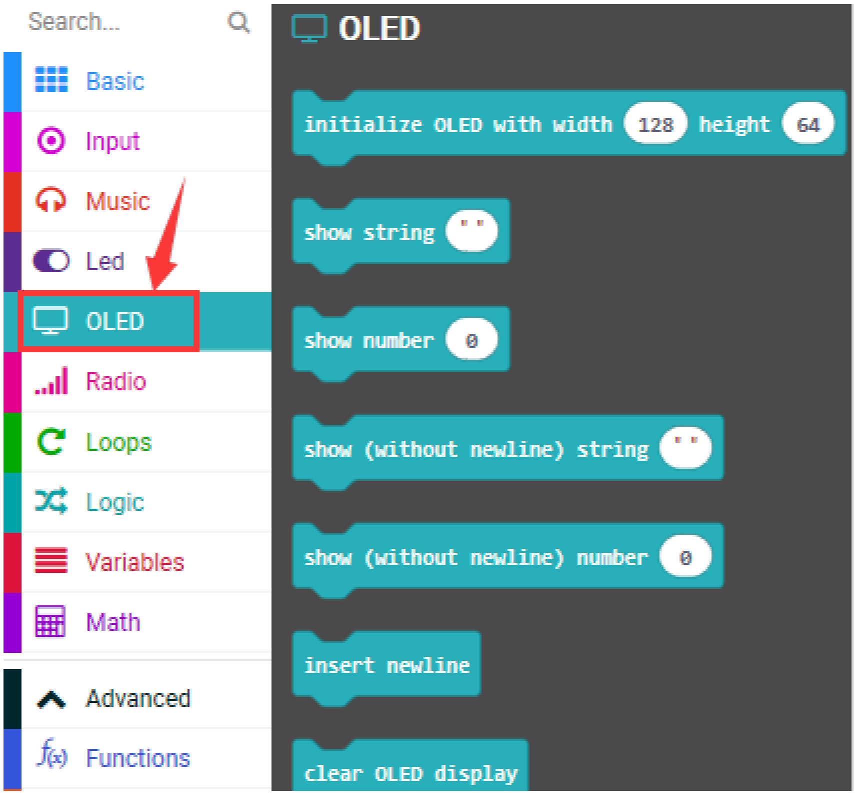
Learn more

iot-environment-kit

Environment and Science IoT Kit for micro:bit

Learn more

After the installation, you will view the OLED in the listed blocks.



Search...

Basic

Input

Music

Led

OLED

Radio

Loops

Logic

Variables

Math

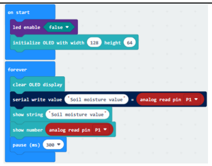
Advanced

Functions

OLED

```
initialize OLED with width 128 height 64

show string " "

show number 0

show (without newline) string " "

show (without newline) number 0

insert newline

clear OLED display
```

Complete Code

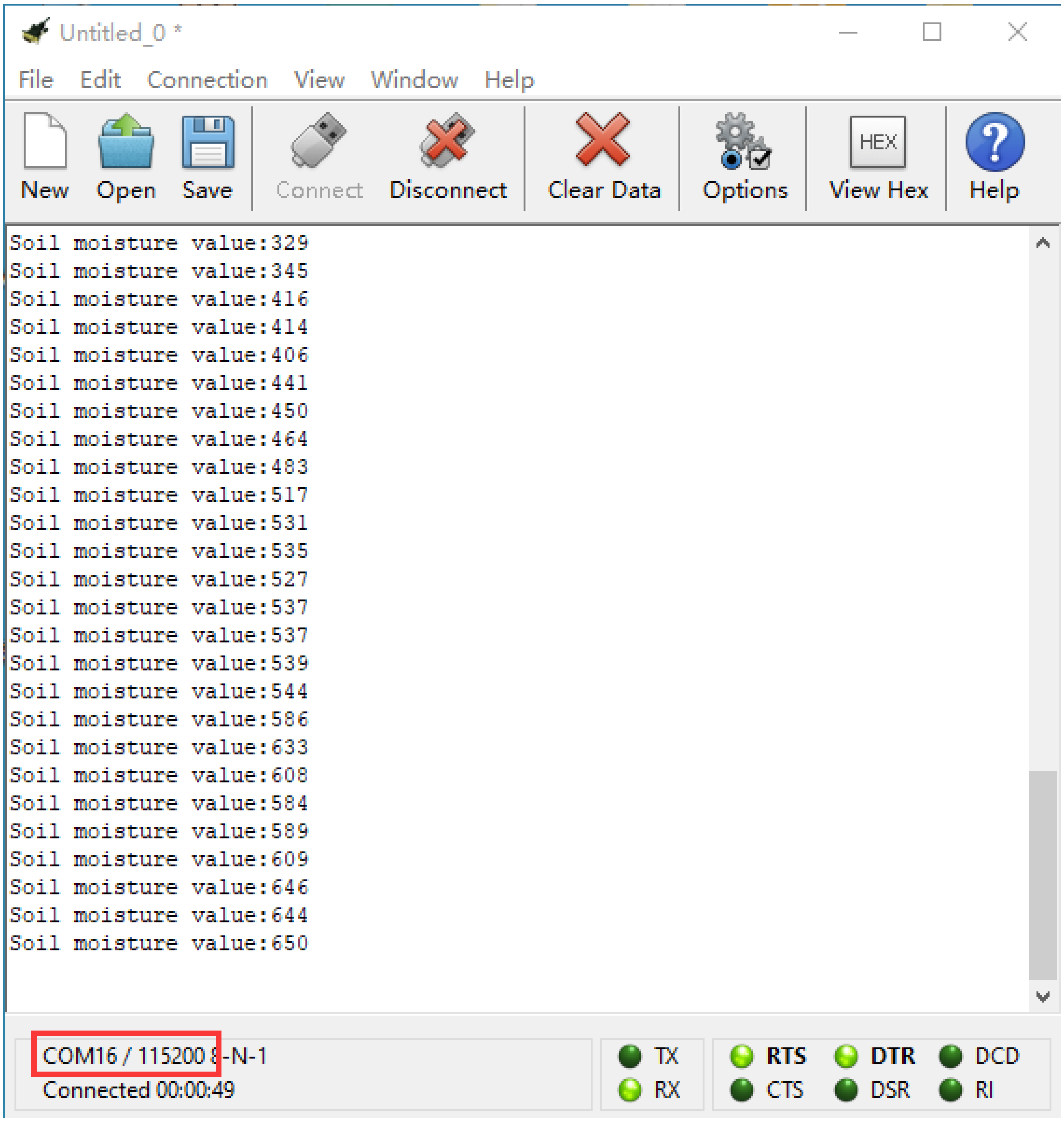
 on start led enable false initialize OLED with width 128 height 64 forever clear OLED display serial write value "Soil moisture value" = analog read pin P1 show string "Soil moisture value" show number analog read pin P1 pause (ms) 300	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitOLED initializes, 128 in width, 64 in heightThe program under "forever" runs cyclically.Clear OLED displaySerial writes the soil humidity analog signals OLED shows "Soil moisture value" OLED shows the soil humidity analog signals Delay in 300ms

6. Test Results:

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.

Insert the soil humidity sensor into soil, CoolTerm monitor and OLED module will display the analog value of soil humidity. As the humidity rises up, so does the analog value; otherwise, the analog sol humidity value will gradually decreases.



Untitled_0 *
File Edit Connection View Window Help
New Open Save Connect Disconnect Clear Data Options View Hex Help
HEX
TX RX RTS DTR DCD CTS DSR RI

```
Soil moisture value:329
Soil moisture value:345
Soil moisture value:416
Soil moisture value:414
Soil moisture value:406
Soil moisture value:441
Soil moisture value:450
Soil moisture value:464
Soil moisture value:483
Soil moisture value:517
Soil moisture value:531
Soil moisture value:535
Soil moisture value:527
Soil moisture value:537
Soil moisture value:537
Soil moisture value:539
Soil moisture value:544
Soil moisture value:586
Soil moisture value:633
Soil moisture value:608
Soil moisture value:584
Soil moisture value:589
Soil moisture value:609
Soil moisture value:646
Soil moisture value:644
Soil moisture value:650
```

COM16 / 115200 8-N-1
Connected 00:00:49

7.31 Project 31: Ultrasonic Ranging

1. Description:

Ultrasonic sensor can detect the distance. In this program, we will conduct you how to detect the distance with it.

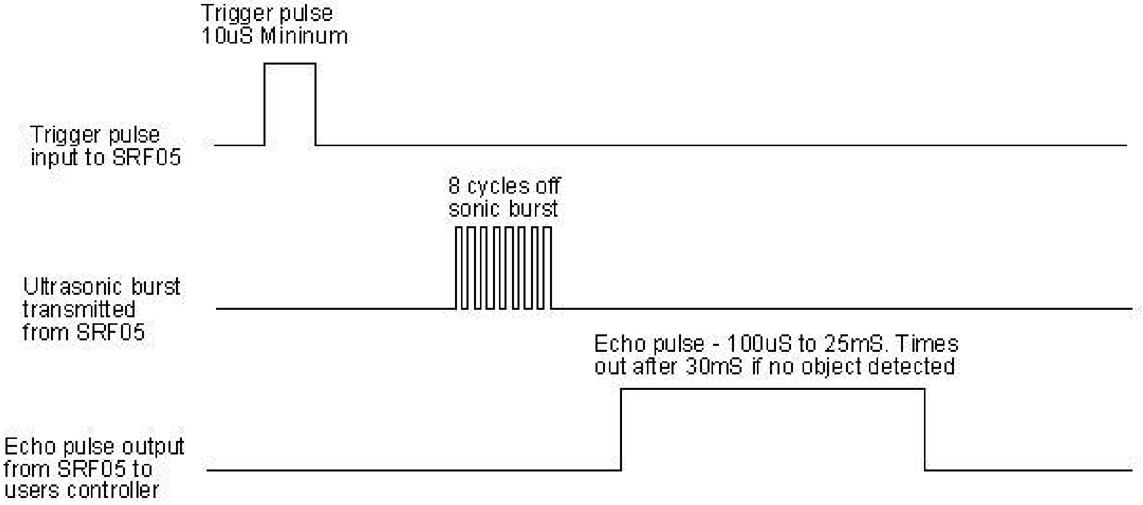
2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- MicroUSB Cable*1
- EASY Plug Ultrasonic Sensor*1
- RJ11 Cable*1

- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug Ultrasonic Sensor:

The ultrasonic module will emit the ultrasonic waves after trigger signal. When the ultrasonic waves encounter the object and are reflected back, the module outputs an echo signal, so it can determine the distance of object from the time difference between trigger signal and echo signal.


Trigger pulse
10uS Mininum
Trigger pulse input to SRF05
8 cycles off sonic burst
Ultrasonic burst transmitted from SRF05
Echo pulse - 100uS to 25mS. Times out after 30mS if no object detected
Echo pulse output from SRF05 to users controller

Use method and timing chart of ultrasonic module:

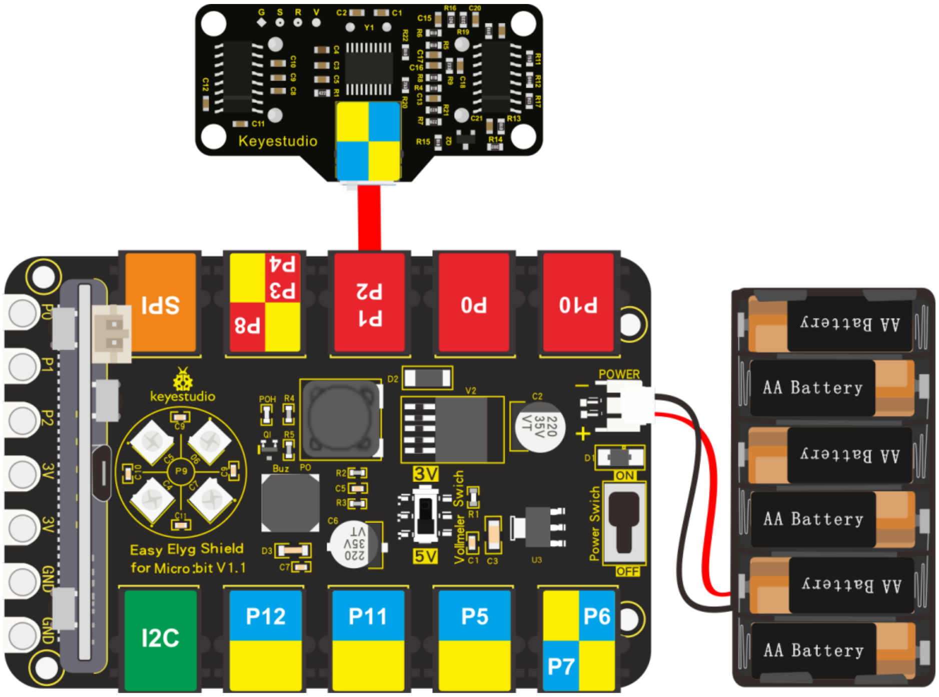
1. Setting the delay time of Trig pin of SR04 to 10s at least, which can trigger it to detect distance.
2. After triggering, the module will automatically send eight 40KHz ultrasonic pulses and detect whether there is a signal return. This step will be completed automatically by the module.
3. If the signal returns, the Echo pin will output a high level, and the duration of the high level is the time from the transmission of the ultrasonic wave to the return.
4. the distance measured by ultrasonic wave = (speed * time) / 2 .
5. Specification:

- Power Supply :+5V DC
- Quiescent Current : <2mA
- Working Current: 15mA
- Effectual Angle: <15°
- Ranging Distance : 2cm – 400 cm
- Resolution : 0.3 cm
- Measuring Angle: 30 degree
- Trigger Input Pulse width: 10uS

- Output echo signal: output TTL level(high), proportional to distance
- Interfacedual digital I/O port

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect an ultrasonic sensor to P1-P2 of shield.



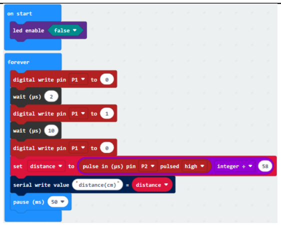
Note: Dial Voltmeter_Switch to 5V end.

5. Test Code:

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

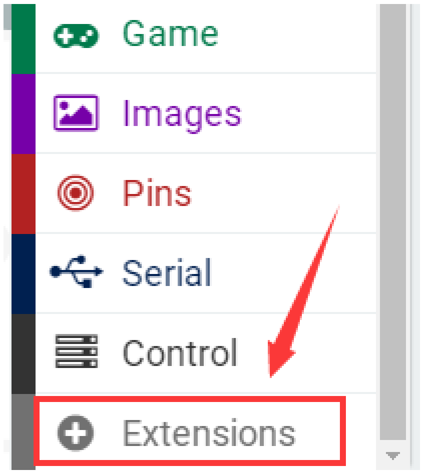
Browse link https://makecode.micro:bit.org/ to edit your program. The following test code is as for your reference.

Code 1

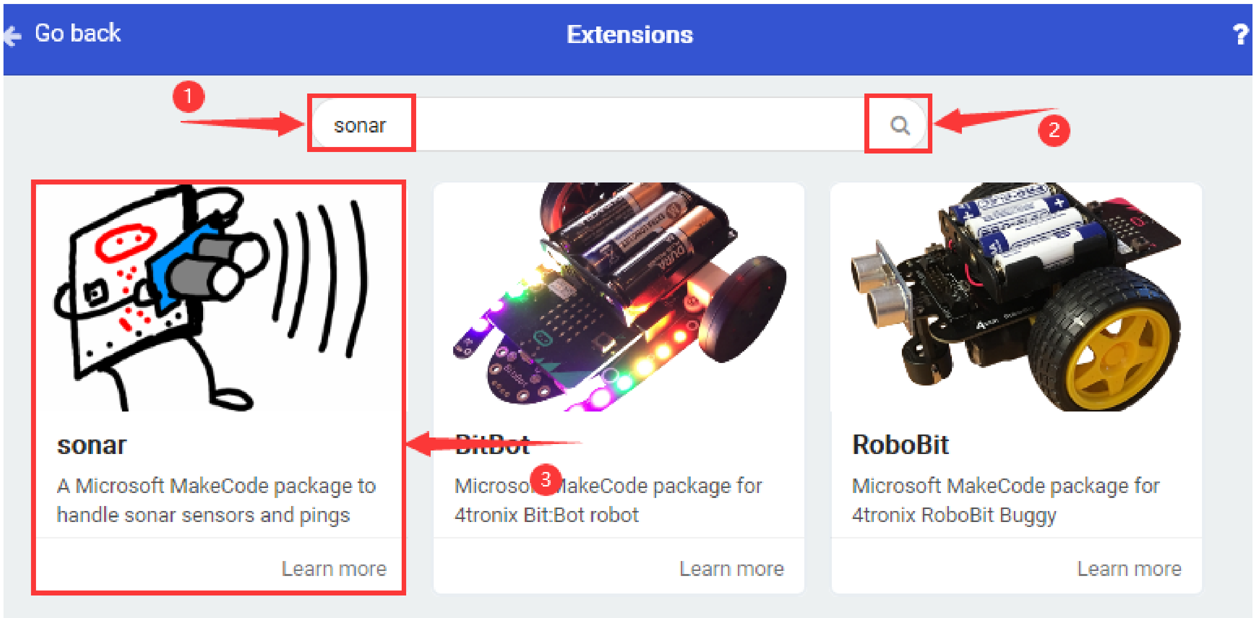
 on start led enable false forever digital write pin P1 to 0 wait (µs) 2 digital write pin P1 to 1 wait (µs) 10 digital write pin P1 to 0 set distance to pulse in (µs) pin P2 pulsed high integer ÷ 58 serial write value "distance(cm)" = distance pause (ms) 50	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitThe program under "forever" runs cyclically.Set P1 to low level(0)Delay in 2msSet P1 to high level(1)Delay in 10msSet P1 to low level(0)Serial writes distance(cm)=distanceDelay in 50ms

Code 2

Next to add the library of ultrasonic sensor, click "Extensions"



Search “sonar”and click sonar library to download it.



sonar

A Microsoft MakeCode package to handle sonar sensors and pings

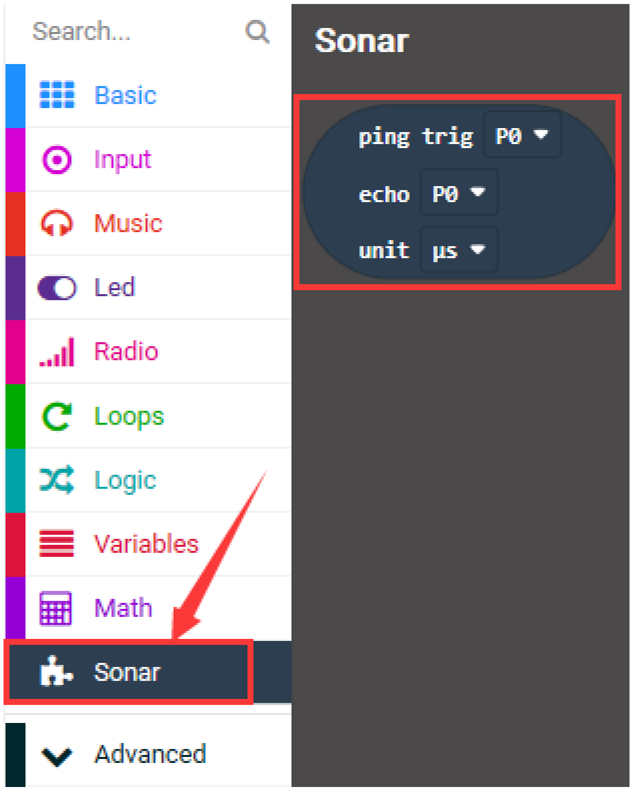
BitBot

Microsoft MakeCode package for 4tronix Bit:Bot robot

RoboBit

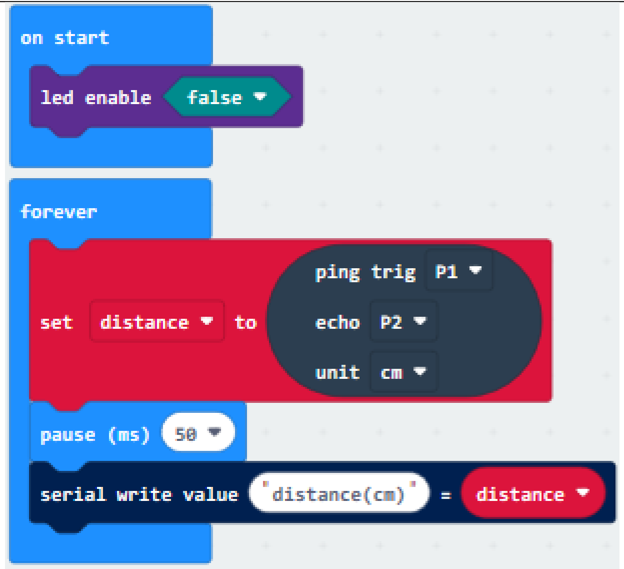
Microsoft MakeCode package for 4tronix RoboBit Buggy

After the installation, you will view it appear in the listed blocks on Makecode editor.



```
ping trig P0
echo P0
unit µs
```

Complete Code



```
on start
    led enable false
forever
    set distance to ping trig P1
                        echo P2
                        unit cm
    pause (ms) 50
    serial write value "distance(cm)" = distance
```

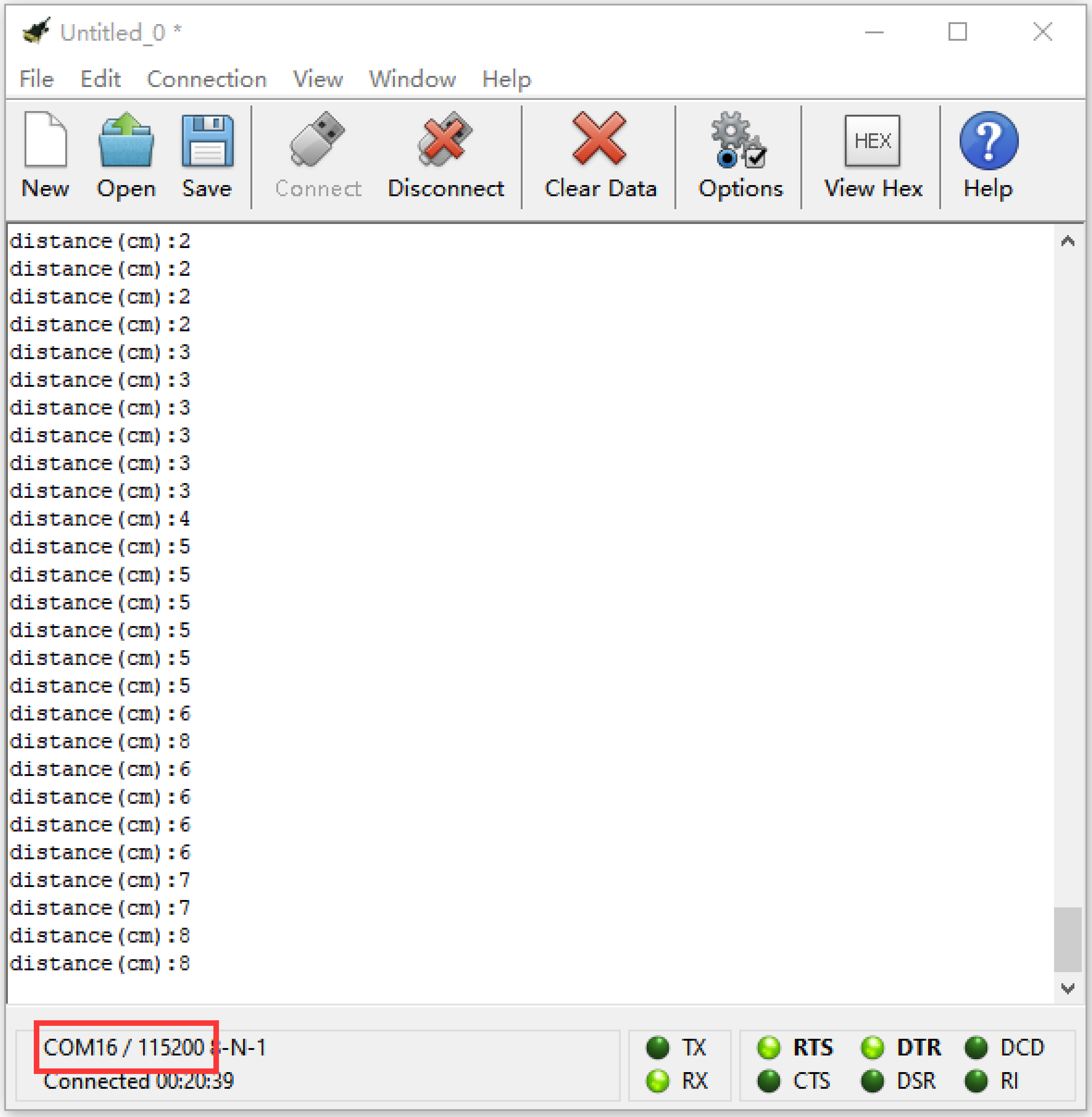
6. Test Results:

Wiring up, dial Voltmeter_Switch to 3V end, plug in external power and dial Power_Switch to ON end and upload code 1 and 2 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:

CoolTerm monitor will display the distance value.



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

COM16 / 115200 8-N-1
Connected 00:20:39

7.32 Project 32: IR Remote Control Decoding

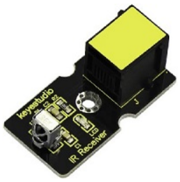
1. Description:

Every mad scientist's lab, or teenager's secret room, needs advanced protection against intrusion by rogue agents or siblings. If you are one of them, you should probably consider getting a Passive infrared (PIR) sensor for you. PIR sensors allow you to detect when someone is in your room when they shouldn't be.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug IR Receiver Module*1
- EASY Plug IR Remote Control*1
- RJ11 Cable*1
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. EASY Plug IR Receiver Module:



There is no doubt that infrared remote control is ubiquitous in daily life. It is used to control various household appliances, such as TVs, stereos, video recorders and satellite signal receivers. Infrared remote control is composed of infrared transmitting and infrared receiving systems, that is, an infrared remote control and infrared receiving module and a single-chip microcomputer capable of decoding.

The 38K infrared carrier signal emitted by remote controller is encoded by the encoding chip in the remote controller. It is composed of a section of pilot code, user code, user inverse code, data code, and data inverse code. The time interval of the pulse is used to distinguish whether it is a 0 or 1 signal and the encoding is made up of these 0, 1 signals.

The user code of the same remote control is unchanged. The data code can distinguish the key.

When the remote control button is pressed, the remote control sends out an infrared carrier signal. When the IR receiver receives the signal, the program will decode the carrier signal and determines which key is pressed. The MCU decodes the received 01 signal, thereby judging what key is pressed by the remote control.

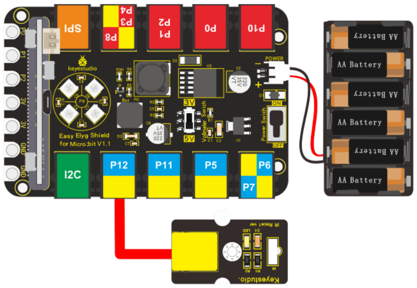
Infrared receiver we use is an infrared receiver module. Mainly composed of an infrared receiver head, it is a device that integrates reception, amplification, and demodulation. Its internal IC has completed demodulation, and can achieve from infrared reception to output and be compatible with TTL signals. Additionally, it is suitable for infrared remote control and infrared data transmission. The infrared receiving module made by the receiver has only three pins, signal line, VCC and GND. It is very convenient to communicate with arduino and other microcontrollers.

Specification

- Interface: Easy plug
- Working voltage: 5V Interface
- Type: DigitaL
- Modulation frequency: 38Khz

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect an IR receiver module toP12 port of shield.



Note: Dial Voltmeter_Switch to 5V.

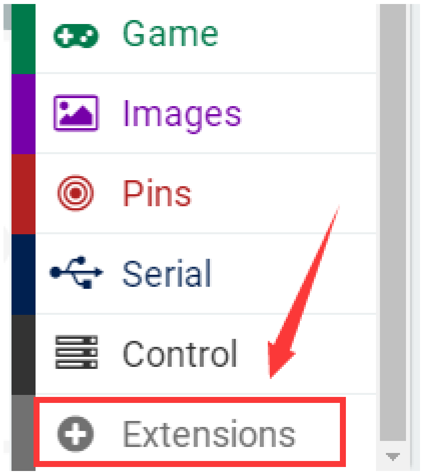
5. Test Code:

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

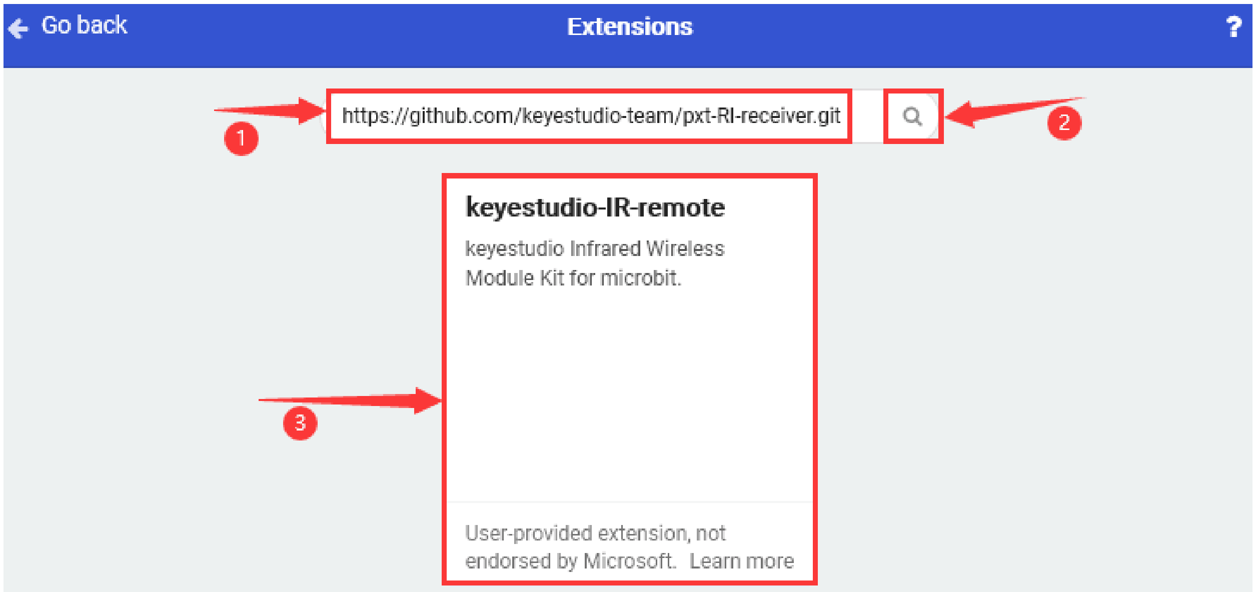
Browse link https://makecode.micro:bit.org/ to edit your program. The following test code is as for your reference.

Next, we need to add the library of IR receiver module.

Click “Extensions” on Makecode editor.



Enter the library link: Copy https://github.com/keyestudio-team/pxt-RI-receiver.git and search, and download “keyestudio-IR-remote” file.



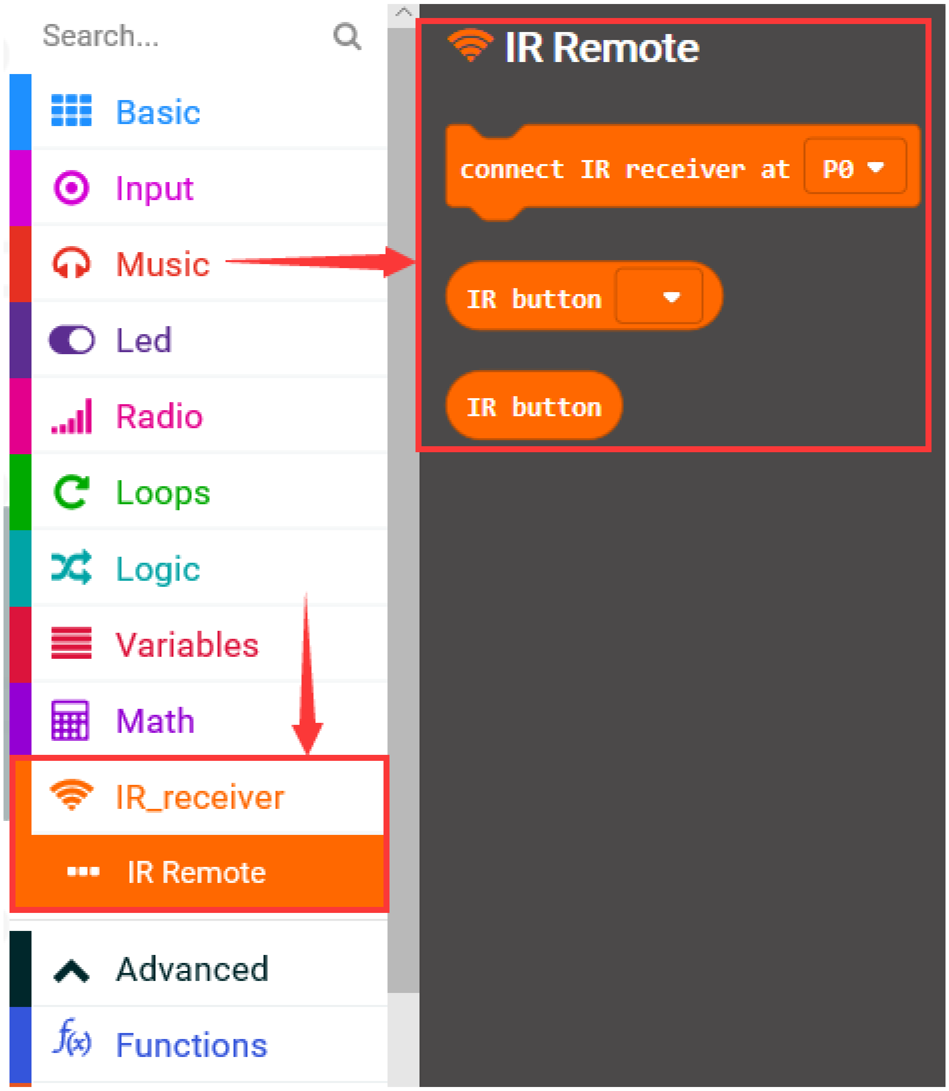
https://github.com/keyestudio-team/pxt-RI-receiver.git

keyestudio-IR-remote

keyestudio Infrared Wireless Module Kit for microbit.

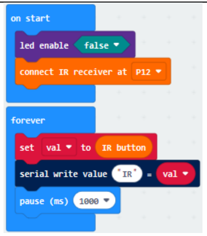
User-provided extension, not endorsed by Microsoft. Learn more

After the installation, you will see the "IR remote" in the listed blocks.



```
connect IR receiver at P0
IR button
IR button
```

Complete Code

 on start led enable false connect IR receiver at P12 forever set val to IR button serial write value "IR" = val pause (ms) 1000	"on start": command block only runs once to start program.Turn off LED matrixConnect IR receiver to P12The program under "forever" runs cyclically.Set val to IR buttonSerial writes IR=valDelay in 1000ms

Code explanation: serial monitor will refresh and show 0, when the key is not pressed; when it is pressed, CoolTerm will show the corresponding value.

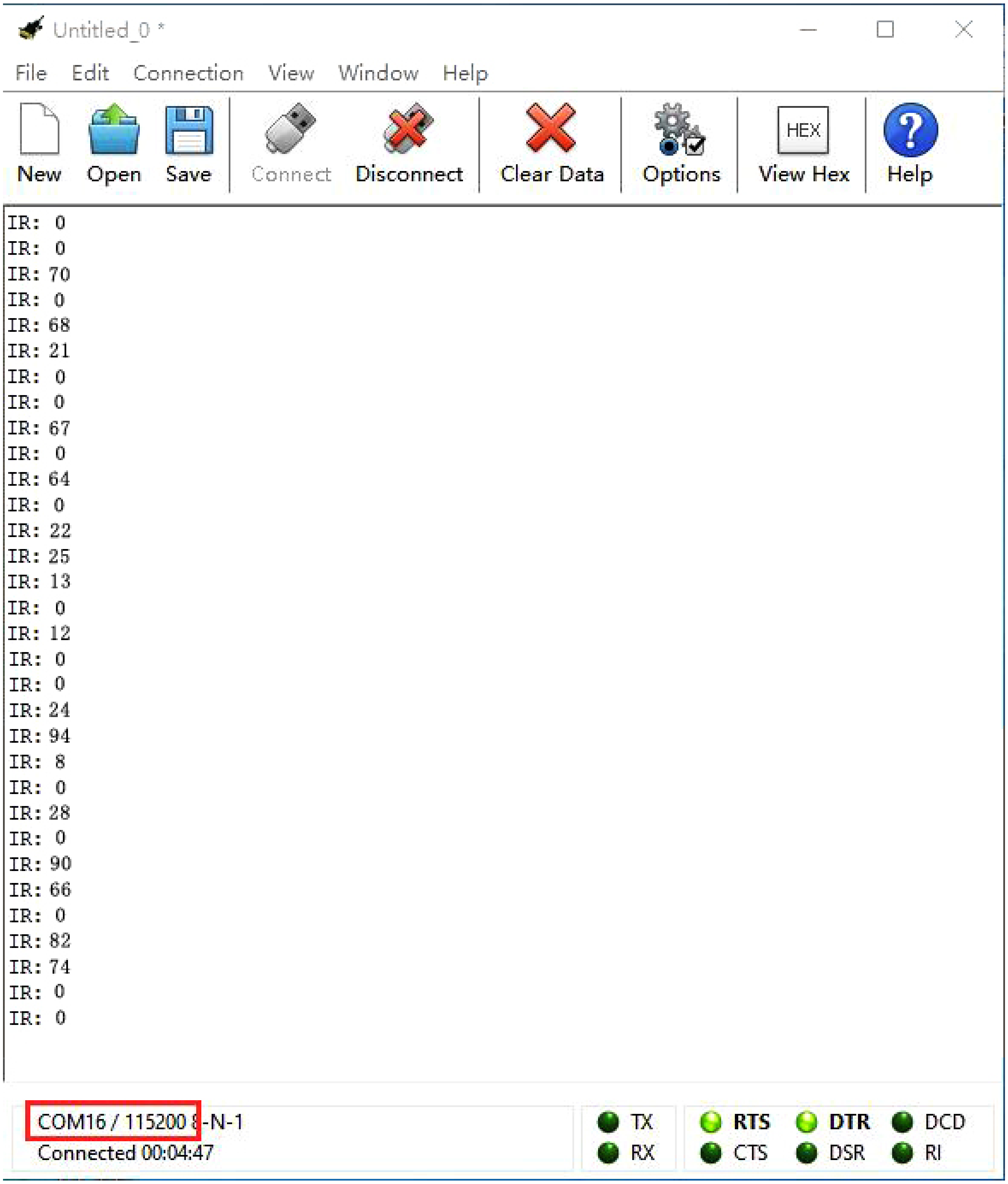
Note

1. Battery is not included, you need to make a purchase it (model: CR2025).
2. Make sure IR remote control is good, open camera on your cellphone, point IR remote control to camera and press button. It is good if you see purple light flashing.
3. Test Results

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.

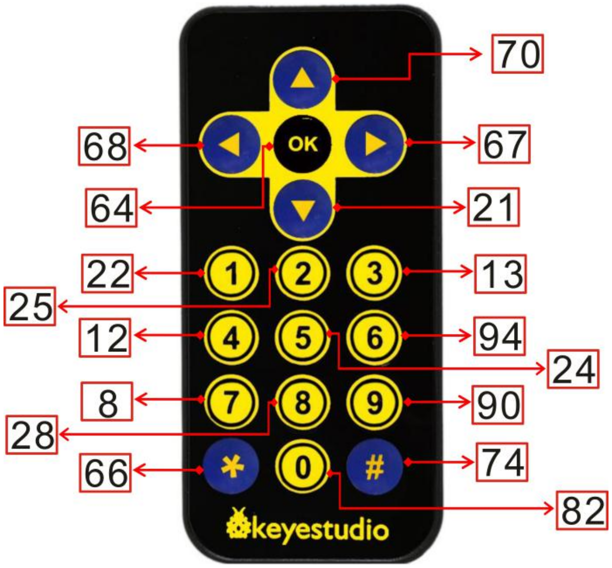
CoolTerm monitor will display the corresponding key values, as shown below:



```
IR:  0
IR:  0
IR: 70
IR:  0
IR: 68
IR: 21
IR:  0
IR:  0
IR: 67
IR:  0
IR: 64
IR:  0
IR: 22
IR: 25
IR: 13
IR:  0
IR: 12
IR:  0
IR:  0
IR: 24
IR: 94
IR:  8
IR:  0
IR: 28
IR:  0
IR: 90
IR: 66
IR:  0
IR: 82
IR: 74
IR:  0
IR:  0
```

COM16 / 115200 8-N-1
Connected 00:04:47

Below we have listed out each button value of keyestudio remote control. So you can keep it for reference.



7.33 Project 33: IR Remote Control RGB

1. Description:

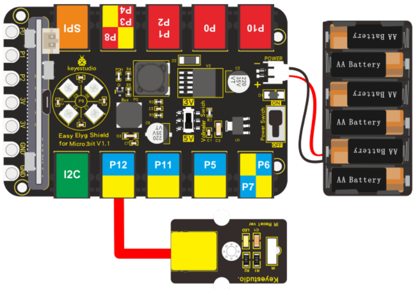
We decode the IR remote control with an IR receiver module and a micro:bit board before doing experiment. We could get key values which can control the external sensors/ modules after decoding IR remote control. In this experiment, we control 2812 2x2 full color RGB of shield with these key values.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug IR Receiver*1
- EASY Plug IR Remote Control*1
- RJ11 Cable*1
- 6-Slot AA Battery Holder*1
- 1.5V AA Battery*6

3. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect IR receiver to P12 port to shield with a RJ11 cable.



Note: Dial Voltmeter_Switch to 5V end.

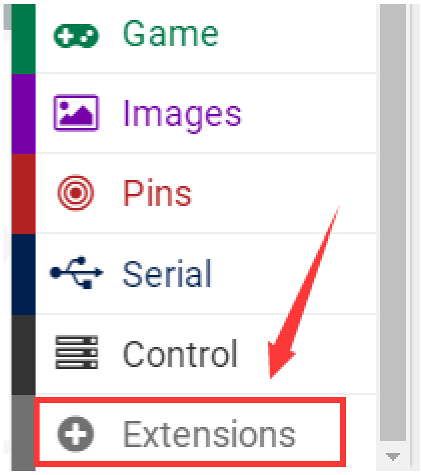
4. Test Code:

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

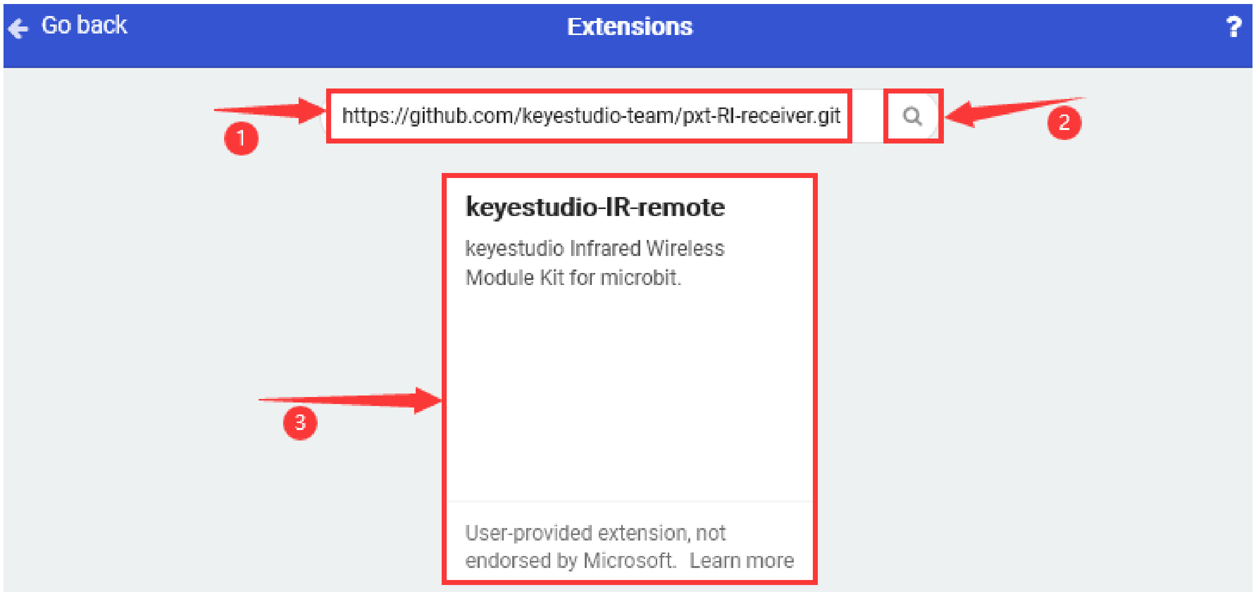
Browse link https://makecode.micro:bit.org/ to edit your program. The following test code is as for your reference.

Next, we need to add the library file of IR receiver module.

Click "Extensions" on Makecode.



Enter the library link https://github.com/keyestudio-team/pxt-RI-receiver.git and search, as shown belowdownload the file.



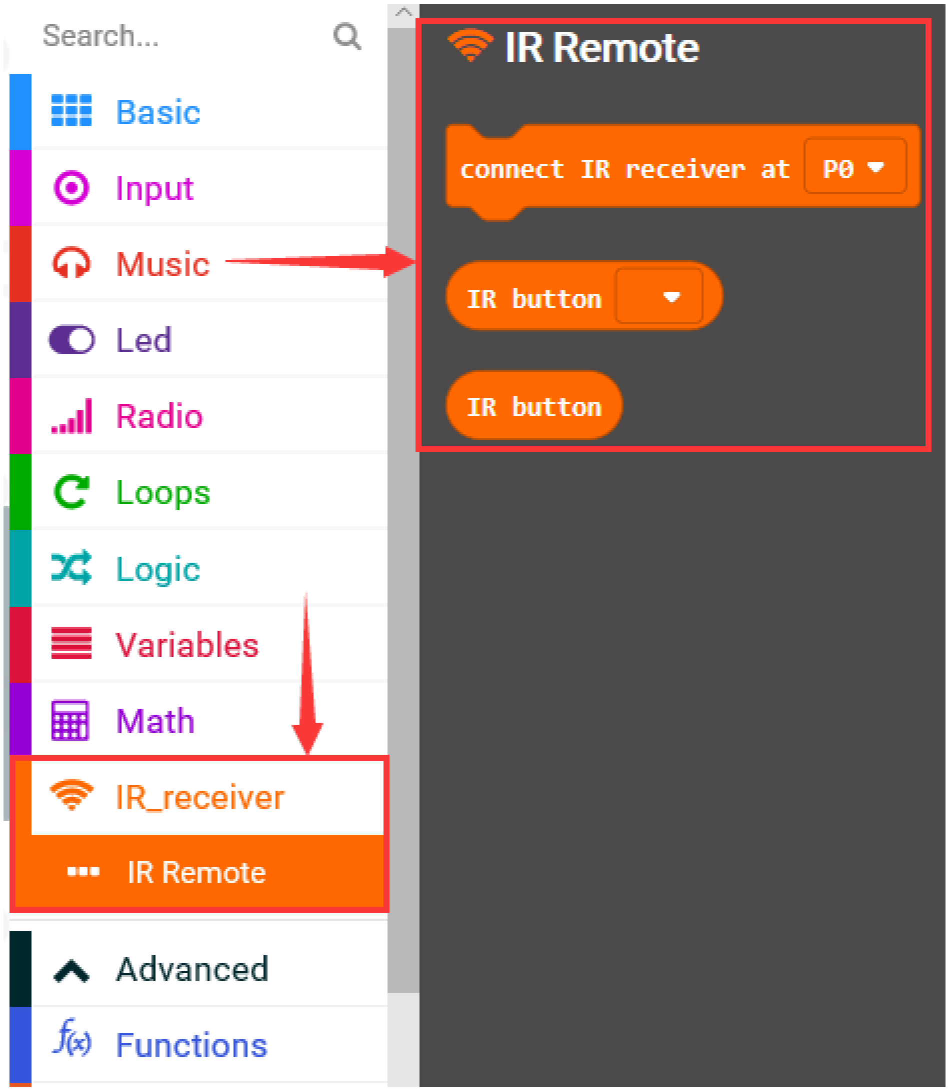
https://github.com/keyestudio-team/pxt-RI-receiver.git

keyestudio-IR-remote

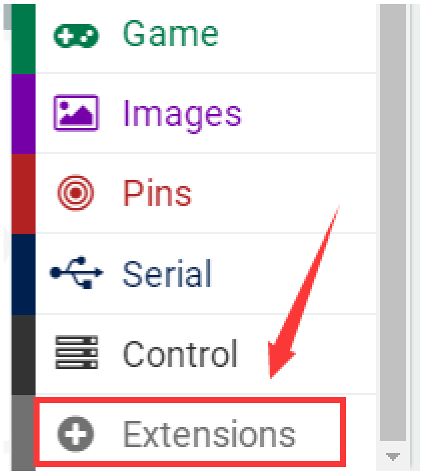
keyestudio Infrared Wireless Module Kit for microbit.

User-provided extension, not endorsed by Microsoft. Learn more

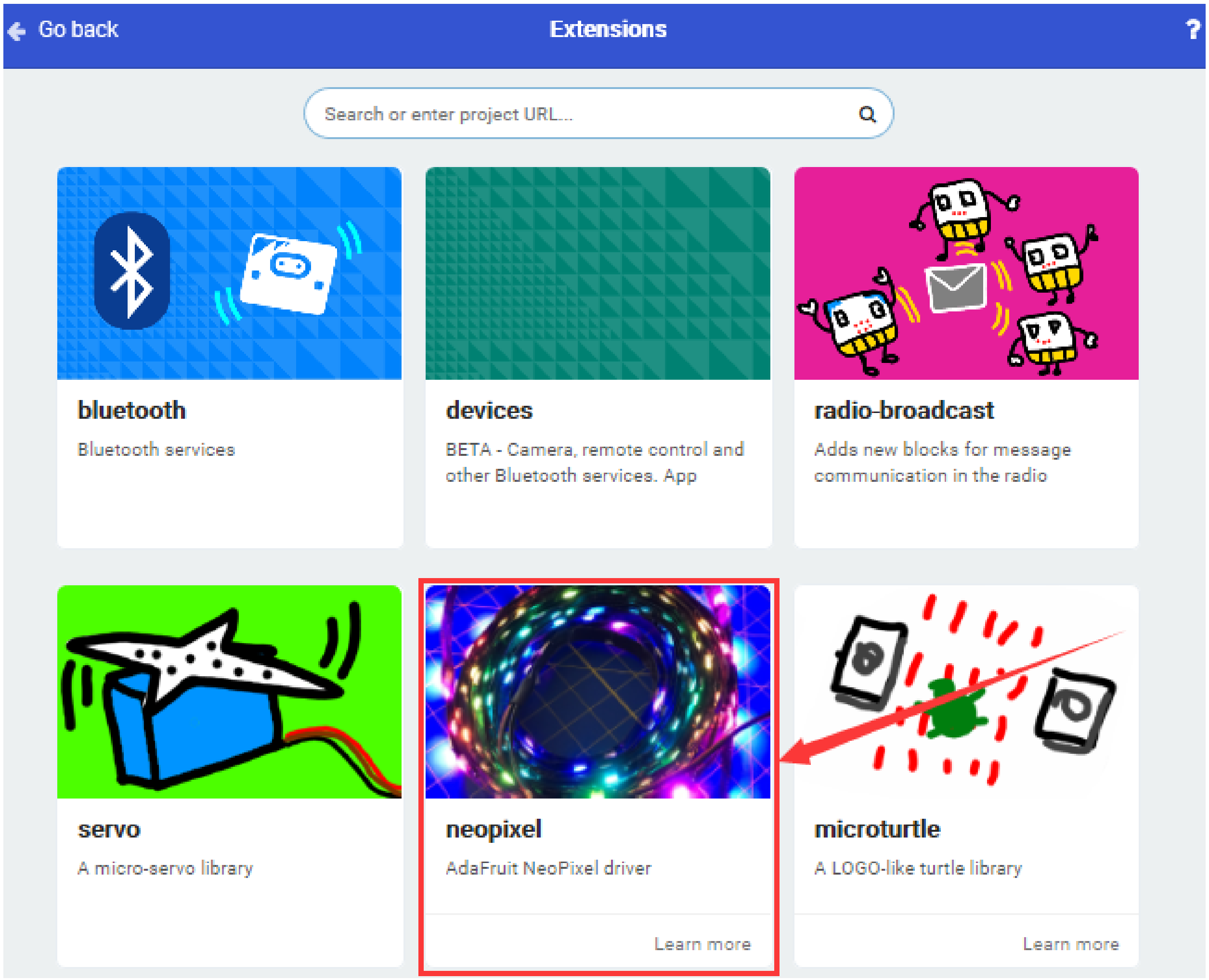
Then you could view the library of IR receiver module in the listed blocks.



Repeat the above steps to add the library of “neopixel”:



Click “Extensions” → “neopixel”, after second, install the library of “neopixel”.



bluetooth

Bluetooth services

devices

BETA - Camera, remote control and other Bluetooth services. App

radio-broadcast

Adds new blocks for message communication in the radio

servo

A micro-servo library

neopixel

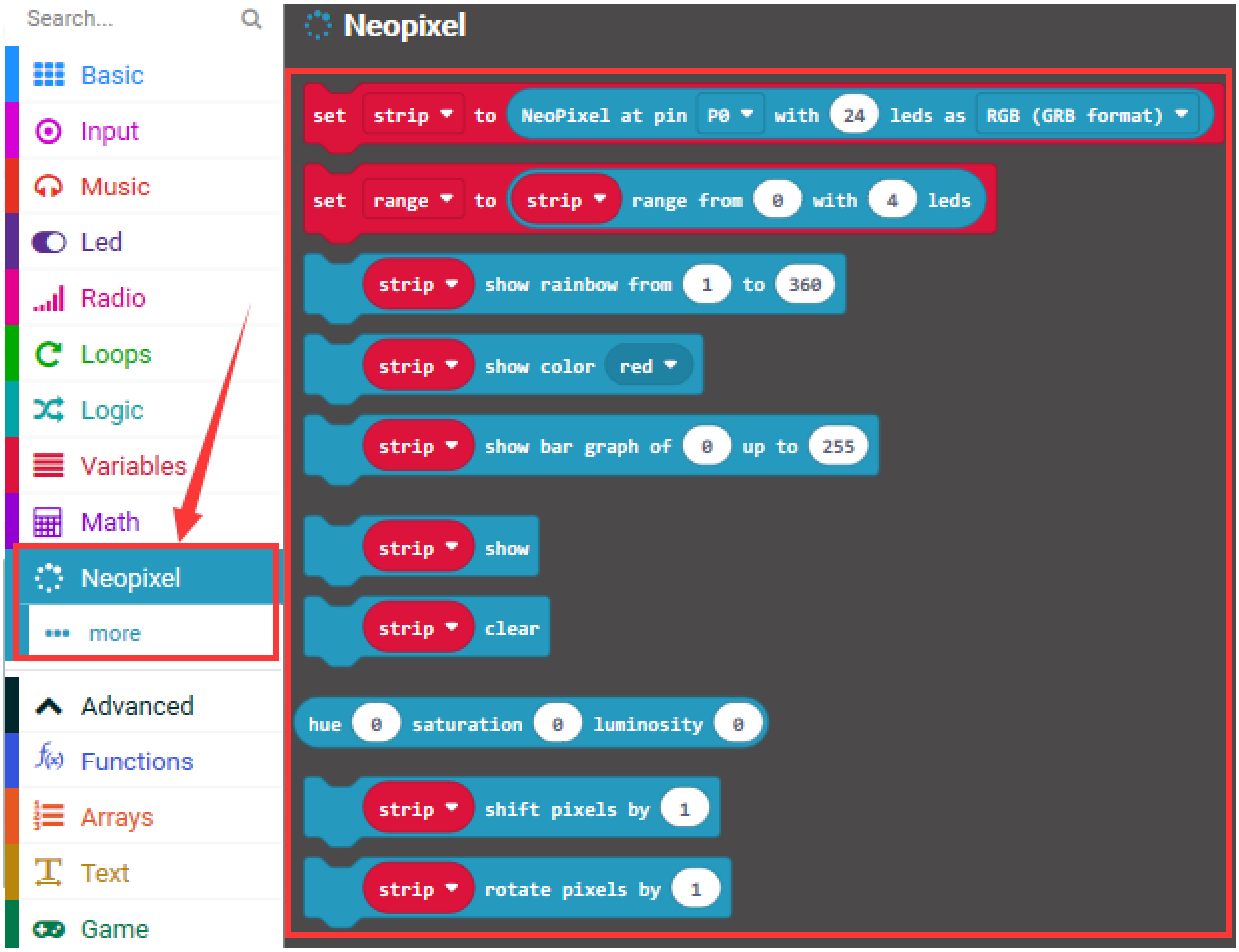
AdaFruit NeoPixel driver

Learn more

microturtle

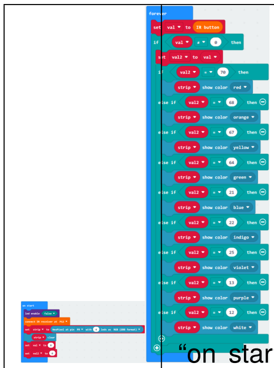
A LOGO-like turtle library

Learn more



```
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
```

Complete Code

 on start led enable false connect IR receiver at P12 set strip to NeoPixel at pin P9 with 4 leds as RGB (GRB format) strip clear set val to 0 set val2 to 0 forever set val to IR button if val ≠ 0 then set val2 to val if val2 = 70 then strip show color red else if val2 = 68 then strip show color orange else if val2 = 67 then strip show color yellow else if val2 = 64 then strip show color green else if val2 = 21 then strip show color blue else if val2 = 22 then strip show color indigo else if val2 = 25 then strip show color violet else if val2 = 13 then strip show color purple else if val2 = 12 then strip show color white	"on start": command block only runs once to start program.Turn off dot matrix on micro:bitConnect IR receiver to P12Set strip to Neopixel at pin p9 with 4 leds as RGBTurn off 4pcs WS2812 RGB lightsSet val to 0Set val2 to 0The program under the block"forever"runs cyclically.Set val to IR buttonWhen val0, execute the program under then blockSet val2 to valWhen val2=70, execute the program under then blockAll RGB display red colorWhen val2=68, execute the program under then blockAll RGB show orange colorWhen val2=67, execute the program under then blockAll RGB display yellow colorWhen val2=64, execute the program under then blockAll RGB display green colorWhen val2=21, execute the program under then blockAll RGB display blue colorWhen val2=22, execute the program under then blockAll RGB display indigo colorWhen val2=25, execute the program under then blockAll RGB display violet colorWhen val2=13, execute the program under then blockAll RGB display purple colorWhen val2=12, execute the program under then blockAll RGB display white color

5. Test Results:

Wire up, plug in power, dial Voltmeter_Switch to 5V end and upload code to micro:bit.

Point IR remote control to IR receiver and press keys.

Key	Status	2812 2x2 full-color RGB
▲	Press	Red Color
◀	Press	Orange Color
▶	Press	Yellow Color
OK	Press	Green Color
▼	Press	Blue Color
1	Press	Indigo Color
2	Press	Violet Color
3	Press	Purple Color
4	Press	White Color

7.34 Project 34: Joystick

1. Description:

Lots of robot projects need joystick. This module provides an affordable solution.

By simply connecting to two analog inputs, the robot is at your commands with X, Y control. It also has a switch that is connected to a digital pin.

2. What You Need:

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

3. EASY Plug Joystick Module:

This is a joystick very similar to the‘analog’ joysticks on PS2 (PlayStation 2) controllers. It is a self-centering spring loaded joystick, meaning when you release the joystick it will center itself. It also contains a comfortable cup-type knob/cap which gives the feel of a thumb-stick.

The goal of the joystick is to communicate motion in 3D (3-axis) to an Arduino. This is achieved by housing two independent 10K potentiometers (one per axis). These potentiometers are used as dual adjustable voltage dividers, providing x and y axis analog input in a control stick form.

This joystick also contains a switch which activates when you push down on the cap. The switch is the small black box on the rear of the joystick.

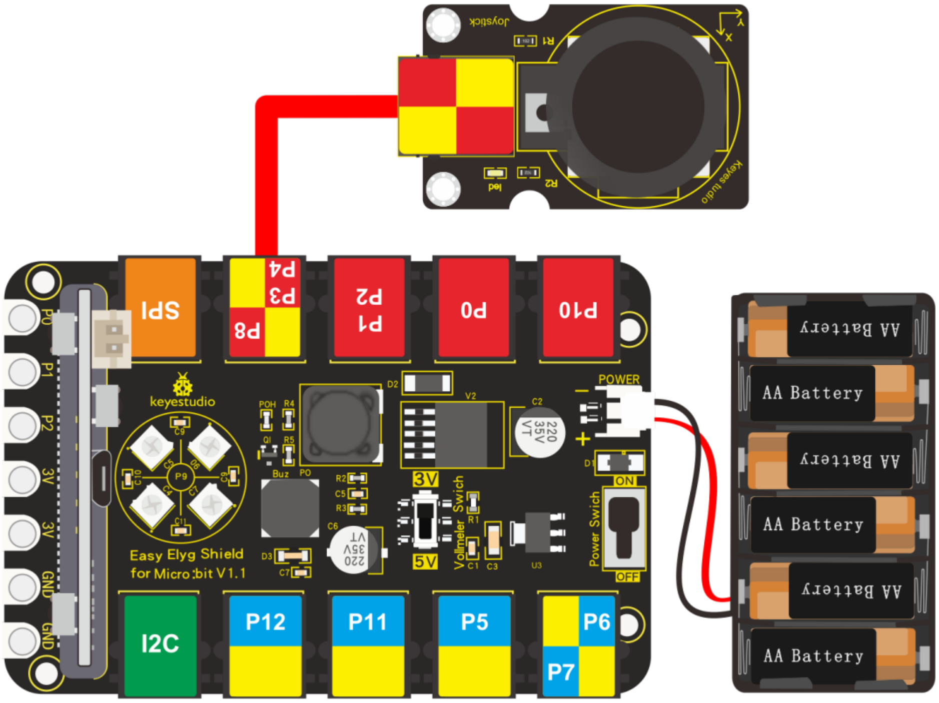
The basic idea of a joystick is to translate the stick’s position on three axes — the X-axis (left to right), the Y-axis(front and back) into electronic information an Arduino can process and the Z-axis(up and down).

Specification

- Interface: Easy plug
- Working voltage: 3.3V to 5V
- Interface type: analog port and digital port

4. Wiring Up:

Insert micro:bit onto EASY Plug shield and connect joystick module to P3-P4-P8 port of shield with a RJ11 cable.

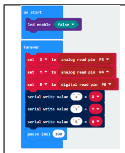


Note: Dial Voltmeter_Switch to 3V end.

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.

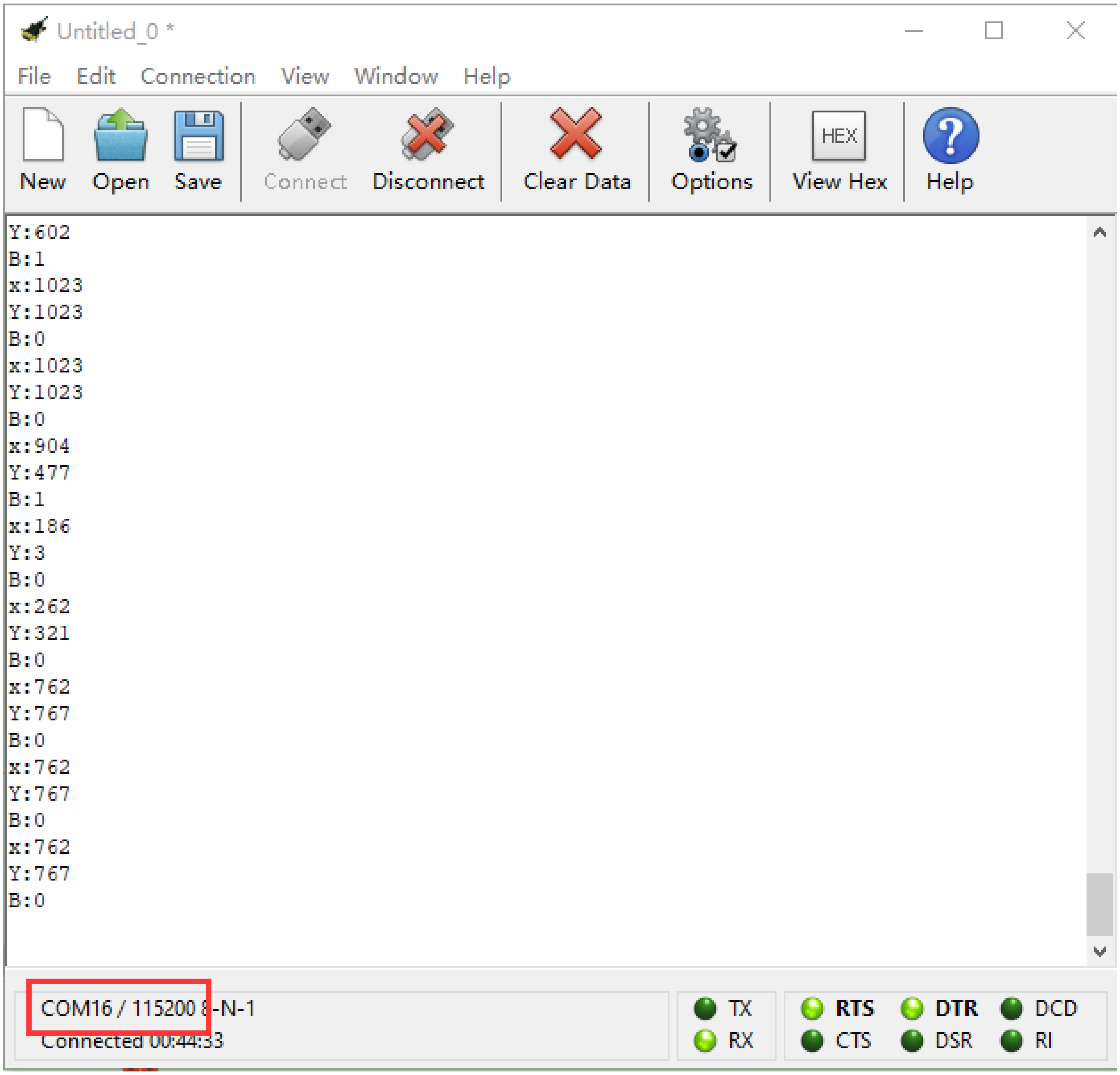
 on start led enable false forever set X to analog read pin P3 set Y to analog read pin P4 set B to digital read pin P8 serial write value "x" = X serial write value "Y" = Y serial write value "B" = B pause (ms) 100	“on start”: command block only runs once to start program.Turn off dot matrix on micro:bitThe program under “forever” runs cyclically.Set variable X to digital signals on X axis Set variable Y to digital signals on Y axisSet variable B to digital signals on Z axis Serial writes value X=P3 to read the analog signals on X axisSerial writes value Y=P4 to read the analog signals on Y axisSerial writes value B=P8 to read the digital signals on Z axis Delay in 100ms

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.

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 analog value on x, y axis and pin B.


Untitled_0 *
File Edit Connection View Window Help
New Open Save Connect Disconnect Clear Data Options View Hex Help
TX RX RTS CTS DTR DSR DCD RI

```
Y:602
B:1
x:1023
Y:1023
B:0
x:1023
Y:1023
B:0
x:904
Y:477
B:1
x:186
Y:3
B:0
x:262
Y:321
B:0
x:762
Y:767
B:0
x:762
Y:767
B:0
x:762
Y:767
B:0
```

COM16 / 115200 8-N-1
Connected 00:44:33

7.35 Project 35: What Time Is It?

1. Description:

In this program, we will demonstrate how to read time by combing DS3231 clock module with micro:bit. You will view the time on CoolTerm monitor.

2. What You Need:

- Micro:bit Board*1
- EASY Plug Shield for micro bit V1.1*1
- Micro USB Cable*1
- EASY Plug DS3231 Clock Module*1

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

3. EASY Plug DS3231 Clock Module:

The DS3231 is a low-cost, extremely accurate I2C real-time clock (RTC) with an integrated temperature-compensated crystal oscillator (TCXO) and crystal. A real-time clock (RTC) is a computer clock (most often in the form of an integrated circuit) that keeps track of the current time.

At the heart of the module is a low-cost, extremely accurate RTC chip from Maxim – **DS3231**. It manages all timekeeping functions and features a simple two-wire I2C interface which can be easily interfaced with any microcontroller of your choice.

The chip maintains seconds, minutes, hours, day, date, month, and year information. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including corrections for leap year (valid up to 2100).

The clock operates in either the 24-hour or 12-hour format with an AM/PM indicator. It also provides two programmable time-of-day alarms and one programmable square wave output.

The DS3231 incorporates a battery input, and maintains accurate timekeeping when main power to the device is interrupted.

The built-in power-sense circuit continuously monitors the status of VCC to detect power failures and automatically switches to the backup supply. So, you need not worry about power outages, your MCU can still keep track of time.

EASY Plug DS3231 module integrates pin GND, VCC, SDA and SCL for easily wiring up and can communicate with MCUs in synchronic and serial way.

SCL is a serial clock pin for I2C interface.

SDA is a serial data pin for I2C interface.

VCC pin supplies power for the module.

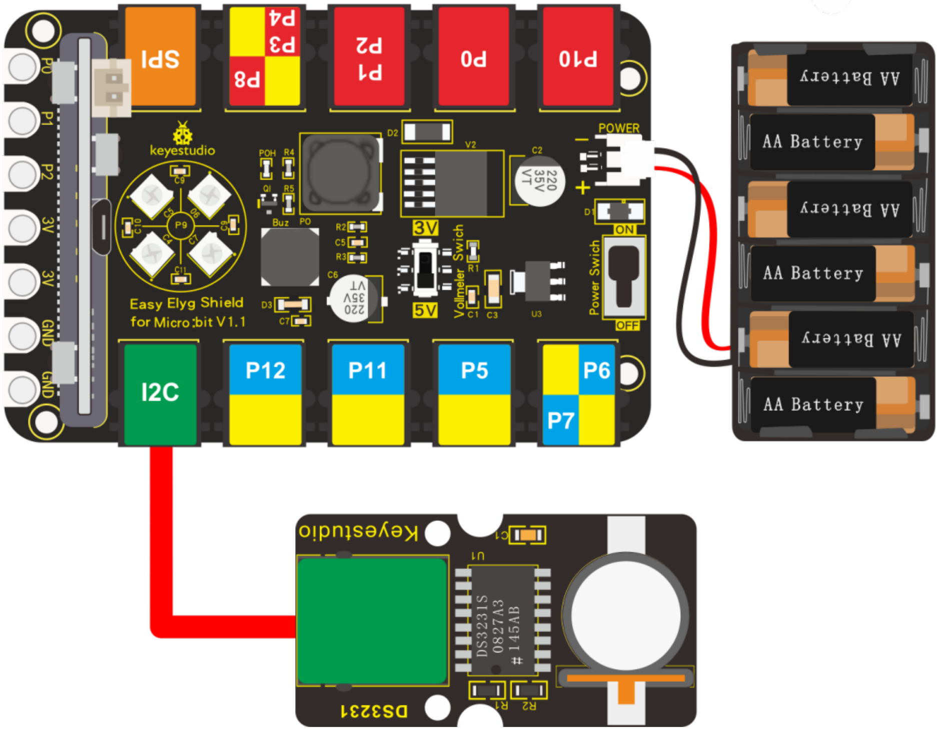
GND is a ground pin.

Specification

- Temperature range: -40℃ to +85℃
- Timing accuracy: about ± 5ppm
- Output: 1Hz and 32.768kHz
- High speed (400kHz), I2C serial bus
- Supply voltage: 3.3V to 5.5V
- Output Level: TTL level

4. Wiring Up:

Insert micro:bit onto EASY Plug shield, connect DS3231 clock module to I2C port of shield with a RJ11 cable.



Note: Dial Voltmeter_Switch to 5V end.

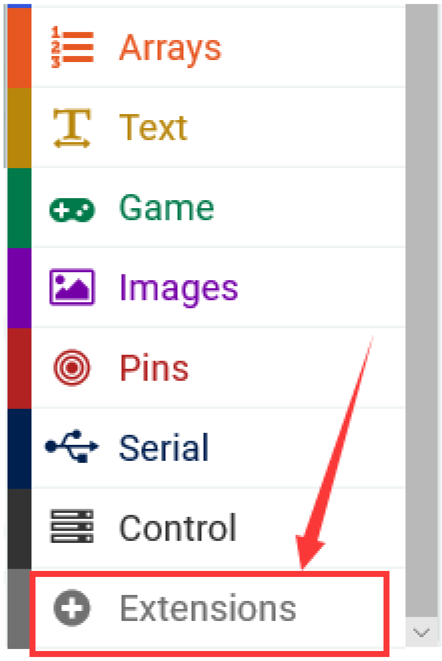
5. Test Code:

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

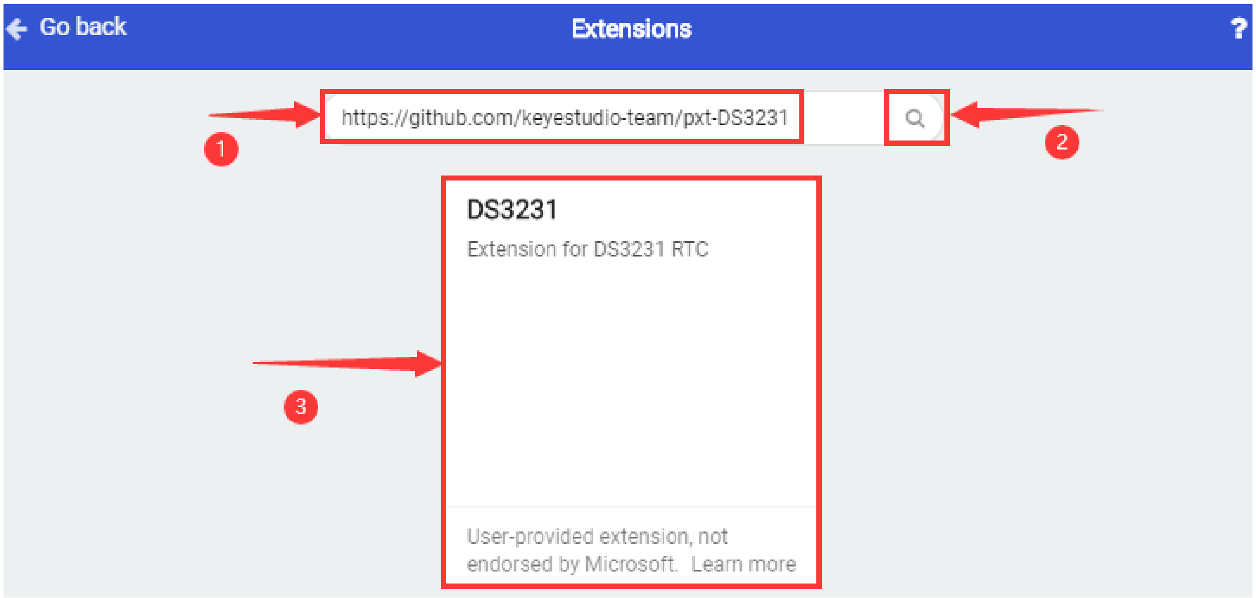
Browse link: https://makecode.micro:bit.org/ to edit your program. The following test code is as for your reference.

Need to set test code, therefore, let's add the library of DS3231 clock module.

Set code with library file, click "Extensions".



Enter library link: https://github.com/keyestudio-team/pxt-DS3231 and search it, download it as follows:

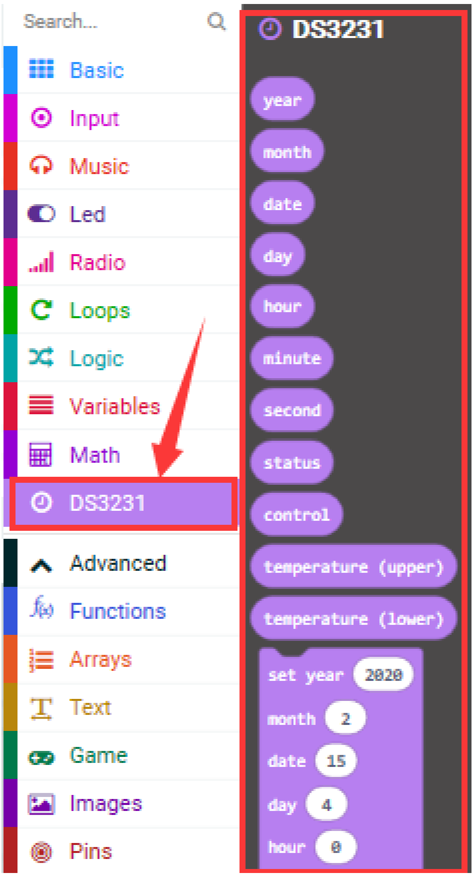


DS3231

Extension for DS3231 RTC

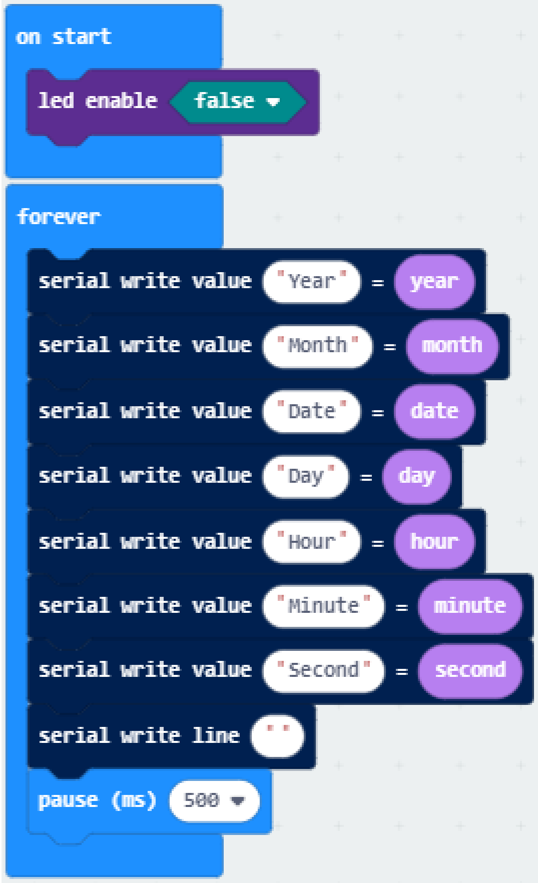
User-provided extension, not endorsed by Microsoft. Learn more

After the installation, you could see it in the listed blocks on Makecode editor.



```
set year 2020
month 2
date 15
day 4
hour 0
```

Complete Code



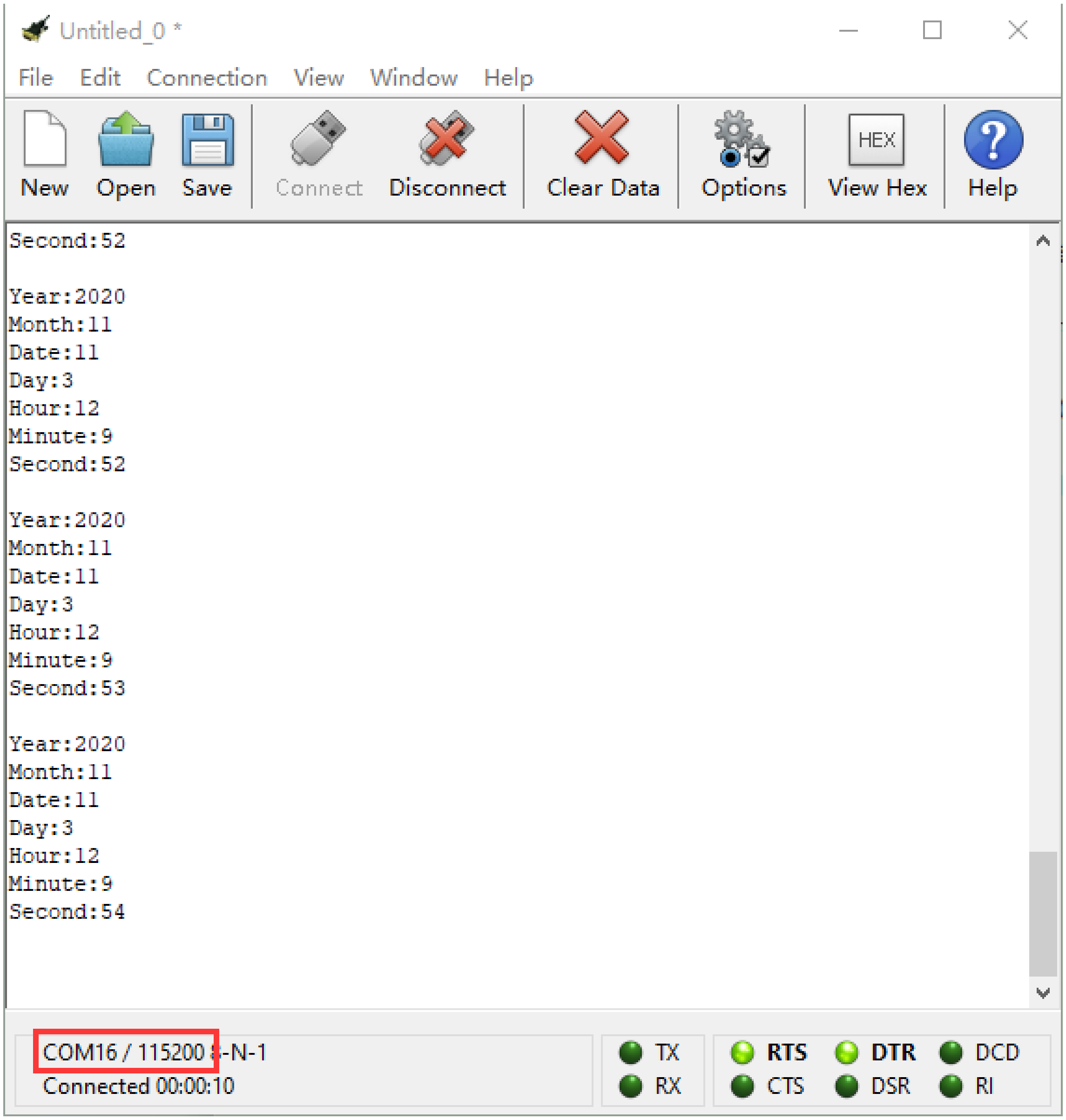
```
on start
    led enable false
forever
    serial write value "Year" = year
    serial write value "Month" = month
    serial write value "Date" = date
    serial write value "Day" = day
    serial write value "Hour" = hour
    serial write value "Minute" = minute
    serial write value "Second" = second
    serial write line ""
    pause (ms) 500
```

6. Test Results:

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 time(year/month/day/week/hour/minute/second), as shown below:



```
Second:52

Year:2020
Month:11
Date:11
Day:3
Hour:12
Minute:9
Second:52

Year:2020
Month:11
Date:11
Day:3
Hour:12
Minute:9
Second:53

Year:2020
Month:11
Date:11
Day:3
Hour:12
Minute:9
Second:54
```

COM16 / 115200 8-N-1
Connected 00:00:10