

## Wristbit control microbit car

### 1. Learning goals

In this lesson, we will learn to use Wrist:bit control micro:bit car by Python programming.

### 2. Wireless communication principle

Through the micro:bit radio module, different devices can work together through a simple wireless network. When the radio function is turned on, a simple wireless local area network is generated. The micro:bit board with the radio function turned on can set parameters To communicate within the effective range.

Wireless communication is divided into sending and receiving two program blocks, set the radio group of radio to the same group, two micro:bit boards can communicate.

### 3. Code and analysis

Please check the .py file for the detailed program of this course.

Wrist:bit code:

```
1 from microbit import *
2 import neopixel
3 import radio
```

First, we need to import the library needed for this lesson from micro:bit, superbite library is dedicated to super:bit expansion board; neopixel is used to control RGB lights; radio for micro:bit wireless communication function.

```
4 Red = (255, 0, 0)
5 Orange = (255, 165, 0)
6 Yellow = (255, 255, 0)
7 Green = (0, 255, 0)
8 Blue = (0, 0, 255)
9 Violet = (148, 0, 211)
10 White = (255, 255, 255)
11 Black = (0, 0, 0)
12 color_lib = {'Red': Red, 'Orange': Orange, 'Yellow': Yellow, 'Green': Green,
13             'Blue': Blue, 'Violet': Violet, 'White': White, 'Black': Black}
14
15 def RGBLight_more_show(first, num, color):
16     global np
17     np.clear()
18     for i in range(first, first + num):
19         np[i] = color_lib[color]
20     np.show()
```

This program is used to define RGB lights of different colors and define the function RGBLight\_more\_show to control the color of RGB lights.

This function will be called in the main loop.

```

60 display.show(Image.HEART)
61 np = neopixel.NeoPixel(pin1, 1)
62 radio.on()
63 radio.config(group=1)

```

**display.show(Image.HEART)**: Display the heart pattern on the micro:bit matrix;

**np = neopixel.NeoPixel (pin1, 1)**: RGB lamp initialization settings, a total of 1 RGB lamps, connected to the P1 pin of the micro:bit board;

**radio.on()**: Turn on the wireless function, because the wireless function consumes more power and occupies memory, so it be closed by default. We can also use **radio.off()** to turn off the wireless function;

**radio.config(group=1)**: configure wireless group=1, other micro:bit devices with wireless group=1 can communicate with each other, the default is 0. The selectable group is 0~255. The set group value needs to be consistent with the setting of the handle, otherwise it cannot communicate normally;

```

22 def send_control():
23     gesture = accelerometer.current_gesture()
24     if gesture == "face up":
25         radio.send('0')

```

Define **send\_control()** function be used to control car.

if the wrist:bit is facing upward, the micro:bit dot matrix displays X and sends the string '0';

if the wrist:bit detects vibration, the RGB light of wrist:bit turns off and sends the string 'I';

If the micro:bit logo up, the micro:bit dot matrix displays the arrow points to the south and sends the string 'D';

if the micro:bit logo up, the micro:bit dot matrix displays the arrow points to the north and sends the string 'C';

if the micro:bit tilt left, the micro:bit dot matrix display arrow points to the west and sends the string 'B';

if the micro:bit tilt right, the micro:bit dot matrix display arrow points to the east and sends the string 'A';

if the microbit button A is pressed, the RGB light of wrist:bit lights up red and sends the string 'E';

if the micro:bit button B is pressed,the RGB light of wrist:bit lights up green and sends the string 'F';

when the A and B buttons are pressed at the same time, the RGB light of wrist:bit lights up blue and sends the string 'G'.

Micro:bit car code:

!Note:

In this course, we will control motor movement through PCA9685 chip with I2C, send pwm through `pwm.set_pwm(a, 0, b)`, the first parameter is the pin number of PCA9685, the second parameter defaults to 0, the third The parameter is the duty cycle of PWM, and the range is 0-4095.

```
1 from microbit import *
2 import ustruct
3 import math
4 import radio
5 import neopixel
```

First, we need to import the library needed for this lesson from micro:bit, neopixel is used to control RGB lights;radio for micro:bit wireless communication function.

```
display.show(Image.HAPPY)
radio.on()
radio.config(group=1)
np = neopixel.NeoPixel(pin16, 3)
... ..
```

`display.show(Image.HAPPY)`: Display the smile pattern on the micro:bit matrix;

`np = neopixel.NeoPixel (pin16, 3)`: RGB lamp initialization settings, a total of 3 RGB lamps, connected to the P16 pin of the micro:bit board;

`radio.on()`: Turn on the wireless function, because the wireless function consumes more power and occupies memory, so it be closed by default. We can also use `radio.off()` to turn off the wireless function;

`radio.config(group=1)`: configure wireless group=1, other micro:bit devices with wireless group=1 can communicate with each other, the default is 0. The selectable group is 0~255. The set group value needs to be consistent with the setting of the handle, otherwise it cannot communicate normally;

```
while True:
    value = radio.receive()
    if value == "A":
        pwm.set_pwm(12, 0, 3000)
        pwm.set_pwm(13, 0, 0)
        pwm.set_pwm(15, 0, 3000)
        pwm.set_pwm(14, 0, 0)
```

...

`incoming = radio.receive()`: Receive the wirelessly transmitted data and save it in the incoming variable;

if incoming is 'A', the car move forward;  
 if incoming is 'B', the car move backward;  
 if incoming is 'C' , the car spin left;  
 if incoming is 'D' , the car spin right;  
 if incoming is 'O' , the car stop.

If incoming is 'E', the RGB lights become red and the "cannonball" will be shot;  
 If incoming is 'F', the RGB lights become green;  
 If incoming is 'G', the RGB lights become blue;  
 If incoming is 'I', the RGB lights will be closed;

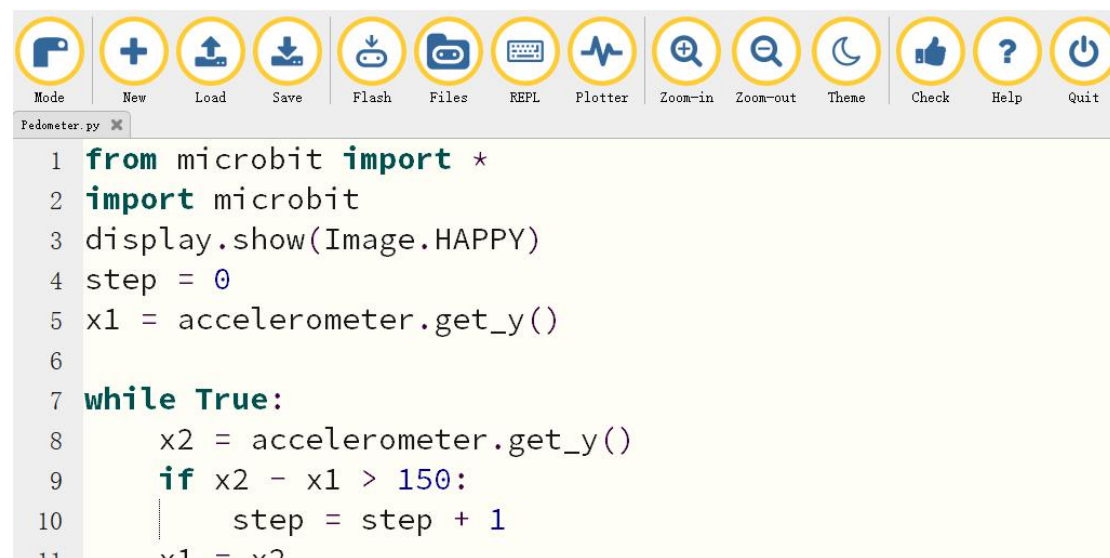
**! Note:**

The incoming value needs to correspond to the value sent by the handle. Only the same value can receive and execute commands.

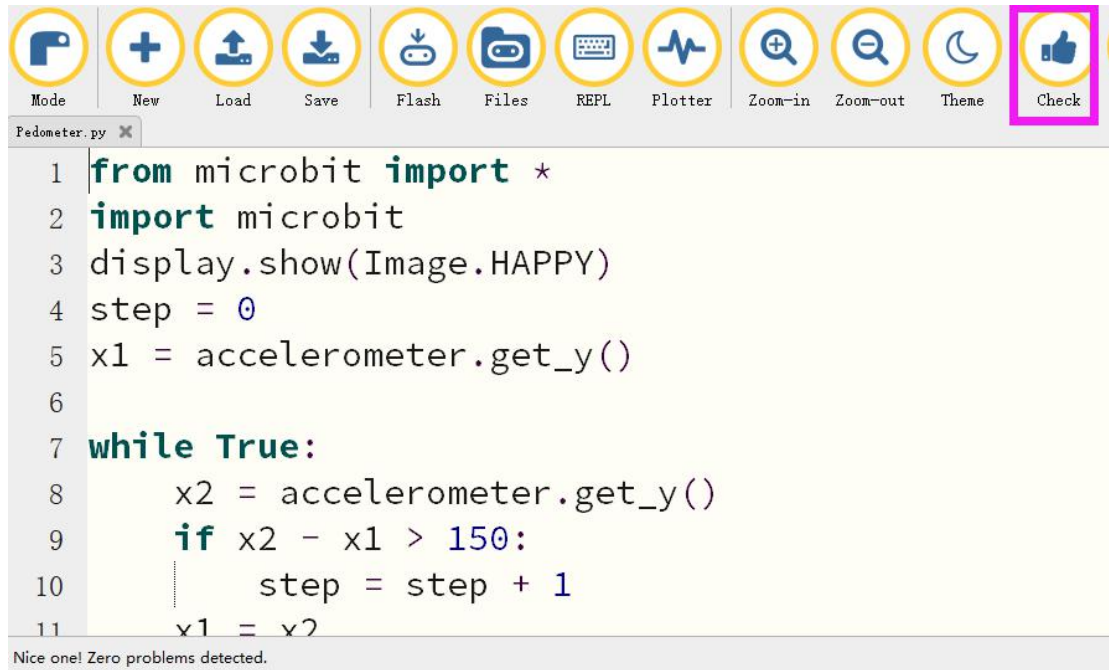
#### 4. Programming and downloading

4.1 You should open the Mu software, and enter the code in the edit window, , as shown in figure .

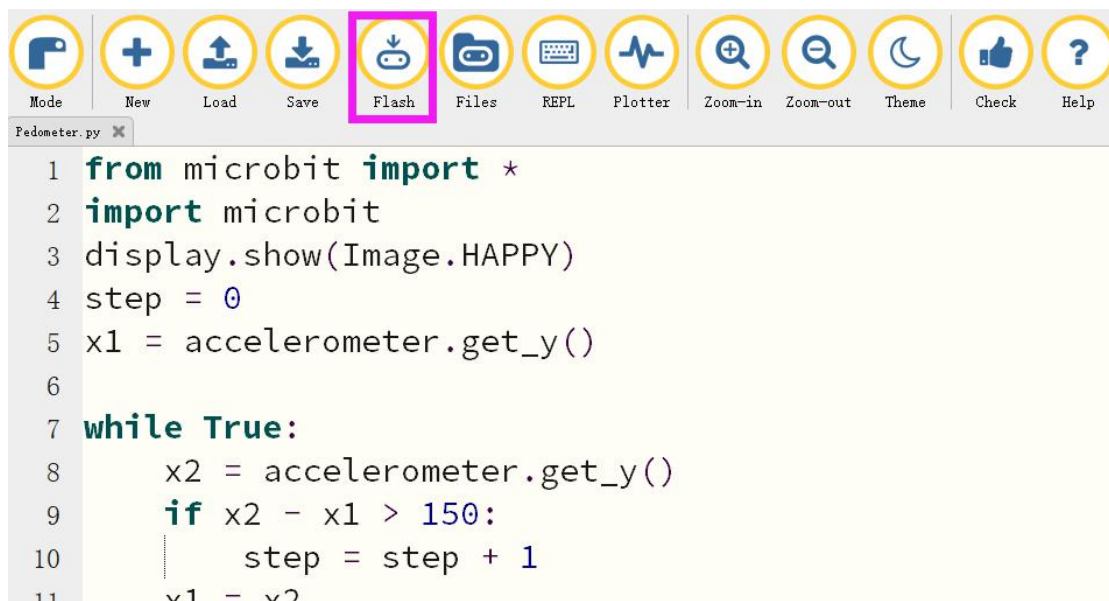
**Note! All English and symbols should be entered in English, and the last line must be a space.**



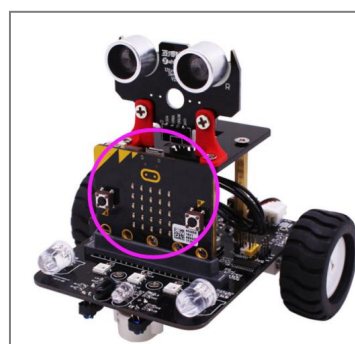
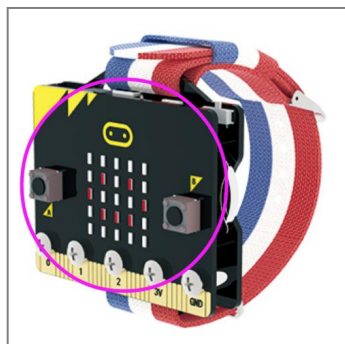
4.2 As shown in figure, you need to click the Check button to check if our code has an error. If a line appears with a cursor or an underscore, the program indicating this line is wrong.



4.3 You need to connect the micro data cable to micro:bit and the computer, then click the Flash button to download the program to micro:bit.



## 5.Experimental phenomena



We need to download the `microbit-car-code.py` into the micro: bit board of Tiny:bit car. Open the power switch of the car, we can see a smile pattern displayed on the micro: bit dot matrix;

We need to download the `Wristbit-code.py` into the micro: bit board of the Wrist:bit. Open the power switch of the wrist:bit, we can see that the micro: bit dot matrix will initially display a heart pattern.

Take wrist:bit on your left wrist as shown below.



They will automatically pairing, then, we can start remote control the car by Wrist:bit.

if the wrist:bit is facing upward, car will stop;

if the micro:bit tilt left, the micro:bit dot matrix display arrow points to west, car will back;

if the micro:bit tilt right, the micro:bit dot matrix display arrow points to the east, car will advance;

If the micro:bit logo up, micro:bit dot matrix display points to the south, car will spin left;

if the micro:bit logo down, micro:bit dot matrix display points to the north, car will spin right;

if we press the button A, the RGB lights of the Wrist:bit and the car will become red. When we press button B, the RGB lights of the Wrist:bit and the car will become green.

if we press the A and B buttons at the same time, the RGB lights of the Wrist:bit and the car will become blue.