Handle control micro:bit car

Note:

1> Before download code, we need to import the tinybit Python library file into the micro:bit board.

More detail, please read [Preparation before class] -- [how to import Python library].

2> Download the microbit-car-code.py into the micro: bit board of micro:bit car. Download the handle-code.py into the micro: bit board of the Handle.

3> After the program download is complete, open the handle and car normally, they will be automatically paired, and they can be controlled within the range of signal transmission.

1. Learning goals

In this lesson, we will learn to use handle 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.

Handle-code

```
1 # -*- coding: utf-8-*-# Encoding cook
2 from microbit import display, Image
3 import ghandle
4 import radio
```

First, we need to import the library needed for this lesson from micro:bit, ghandle library is dedicated to handle; radio for micro:bit wireless communication function.

```
6 display.show(0)
7 radio.on()
8 radio.config(group=1)
```

display.show(0: Display the 0 on the micro:bit matrix;

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:

    if ghandle.rocker(ghandle.up):
        radio.send('A')
        display.show(Image.ARROW_N)
    elif ghandle.rocker(ghandle.down):
        radio.send('B')
        display.show(Image.ARROW_S)
    elif ghandle.rocker(ghandle.left):
        radio.send('D')
        display.show(Image.ARROW_W)
    elif ghandle.rocker(ghandle.right):
        radio.send('C')
        display.show(Image.ARROW_E)
    elif ghandle.rocker(ghandle.pressed):
        radio.send('0')
        display.show(Image.NO)
    else:
        radio.send('0')
        display.clear()
```

...

if ghandle.rocker (ghandle.up) is True, it means that the rocker of the handle is pushed up, the wireless send the ‘up’ command and display an up icon on micro:bit matrix;

if the ghandle.rocker (ghandle.down) is True, it means that the rocker of the handle is pushed up, the wireless send the ‘down’ command and display a down icon on micro:bit matrix;

if the detection of ghandle.rocker(ghandle.left) is True, it means that the rocker of the handle is pushed left, the wireless send the ‘left’ command and display an left icon on micro:bit matrix;

if ghandle.rocker(ghandle.right) is detected as True, it means that the rocker of the handle is pushed right, the wireless send the 'right' command and display an right icon on micro:bit matrix;

if the detection of ghandle.rocker(ghandle.pressed) is True, it means that the rokcer of the handle is pressed, the wireless send 'pressed' command, and display 'X' icon on micro:bit matrix;

If there is no operation on the handle, send 'stop' and clear the display;

```
31      if ghandle.B1_is_pressed():
32          radio.send('E')
33          display.show("R")
34      if ghandle.B2_is_pressed():
35          radio.send('F')
36          display.show("G")
37      if ghandle.B3_is_pressed():
38          radio.send('G')
39          display.show("B")
40      if ghandle.B4_is_pressed():
41          radio.send('I')
42          display.show("Y")
```

if ghandle.B1_is_pressed(): is True, it means that the B1(red button) is pressed, the wireless send the 'R' command and display "R" on micro:bit matrix;
if ghandle.B2_is_pressed(): is True, it means that the B2(green button) is pressed, the wireless send the 'G' command and display "G" on micro:bit matrix;
if ghandle.B3_is_pressed(): is True, it means that the B3(blue button) is pressed, the wireless send the 'B' command and display "B" on micro:bit matrix;
if ghandle.B4_is_pressed(): is True, it means that the B4(yellow button) is pressed, the wireless send the 'Y' command and display "Y" on micro:bit matrix;

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.

```
from microbit import *
import ustruct
import math
import radio
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 '0' , the car stop.

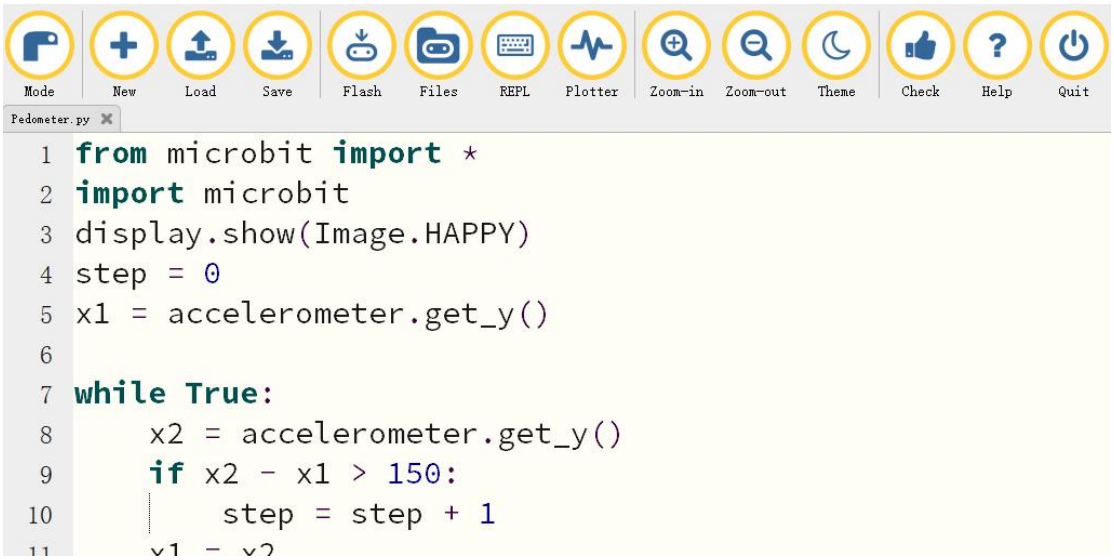
If incoming is 'E', the RGB lights become red;
If incoming is 'F', the RGB lights become green;
If incoming is 'G', the RGB lights become blue;
If incoming is 'I', the RGB lights will be closed;

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

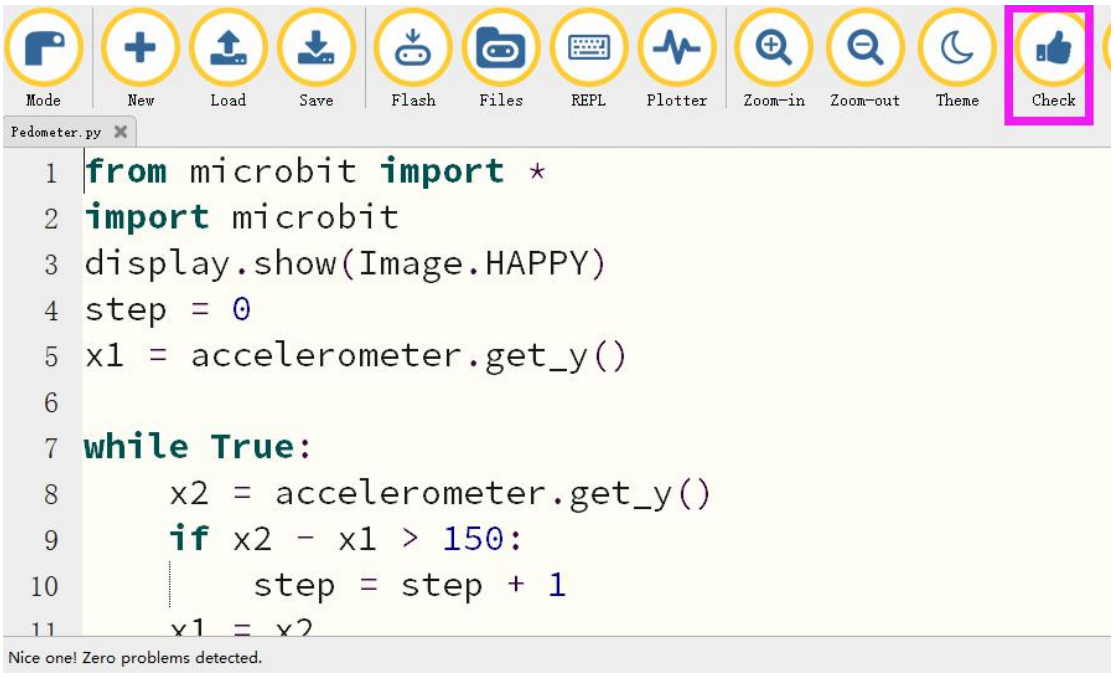
4. Programming and downloading

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

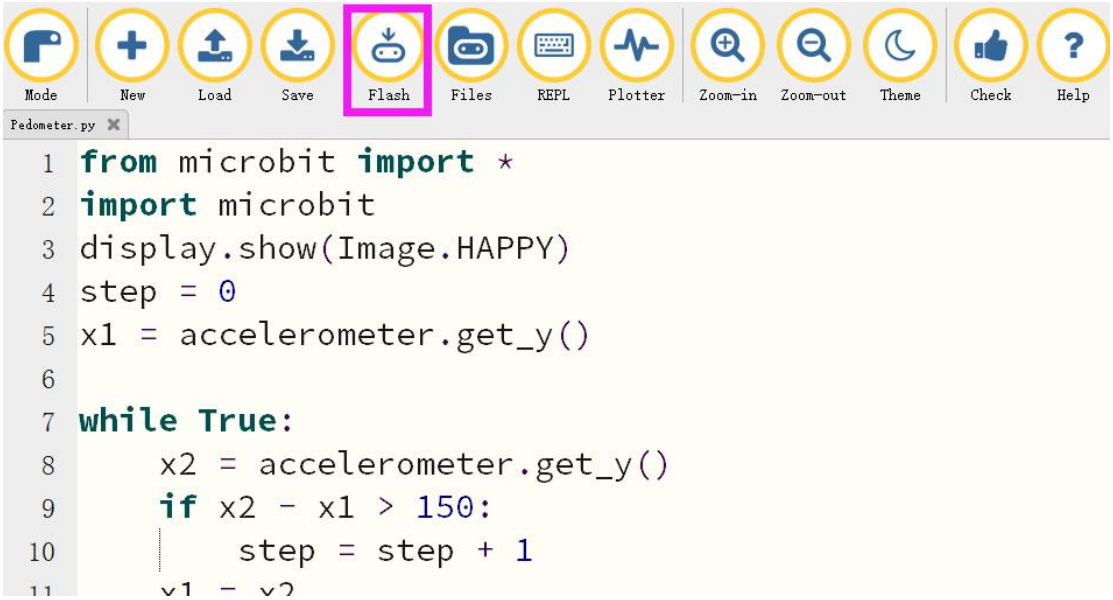
Note! All English and symbols should be entered in English, and the last line must be a space.



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.



4.4 If the download failed, please confirm whether the micro:bit is connected to the computer through the micro USB data cable, and confirm whether the Tiny:bit Python library has been imported.
More detail, please read [Preparation before class] -- [how to import Python library].

5.Experimental phenomena

We need to download the microbit-car-code.py into the micro: bit board of micro: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 handle-code.py into the micro: bit board of the Handle Open the power switch of the wrist:bit, we can see that the micro:bit dot matrix will initially display 0.

The function of the handle is shown below.