# Wristbit control Tumble:bit car

**Tips:**

**1) Before downloading the program, we need to import the superbit library file to the micro:bit board of the car.**

**More detail, please check [1.Preparation before class]--[How to import python library file].**

**2) After the program is downloaded, open the power switch of wrist:bit 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 Tumble:bit control micro:bit car by Python programming.

## 2.Wireless communication principle

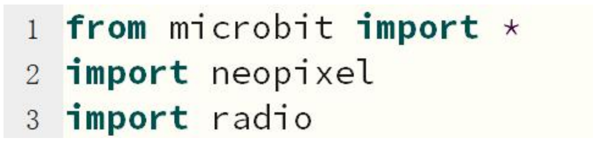
Through the micro:bit radio module, different devices can work together through a simple wireless network. When the radio function is turned on, a simple wireless local area network is generated. The micro:bit board with the radio function turned on can set parameters To communicate within the effective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



```python
from microbit import *
import neopixel
import radio
```

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

```python
Red = (255, 0, 0)
Orange = (255, 165, 0)
Yellow = (255, 255, 0)
Green = (0, 255, 0)
Blue = (0, 0, 255)
Violet = (148, 0, 211)
White = (255, 255, 255)
Black = (0, 0, 0)
color_lib = {'Red': Red, 'Orange': Orange, 'Yellow': Yellow, 'Green': Green,
             'Blue': Blue, 'Violet': Violet, 'White': White, 'Black': Black}

def RGBLight_more_show(first, num, color):
    global np
    np.clear()
    for i in range(first, first + num):
        np[i] = color_lib[color]
    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.

```python
display.show(Image.HEART)
np = neopixel.NeoPixel(pin1, 1)
radio.on()
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;

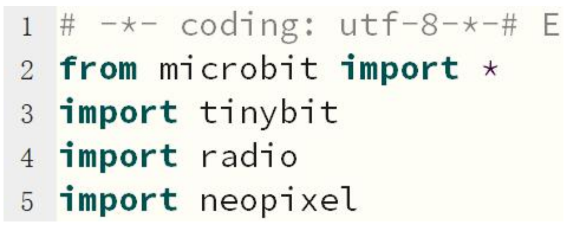
```python
def send_control():
    gesture = accelerometer.current_gesture()
    if gesture == "face up":
        radio.send('0')
```

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';

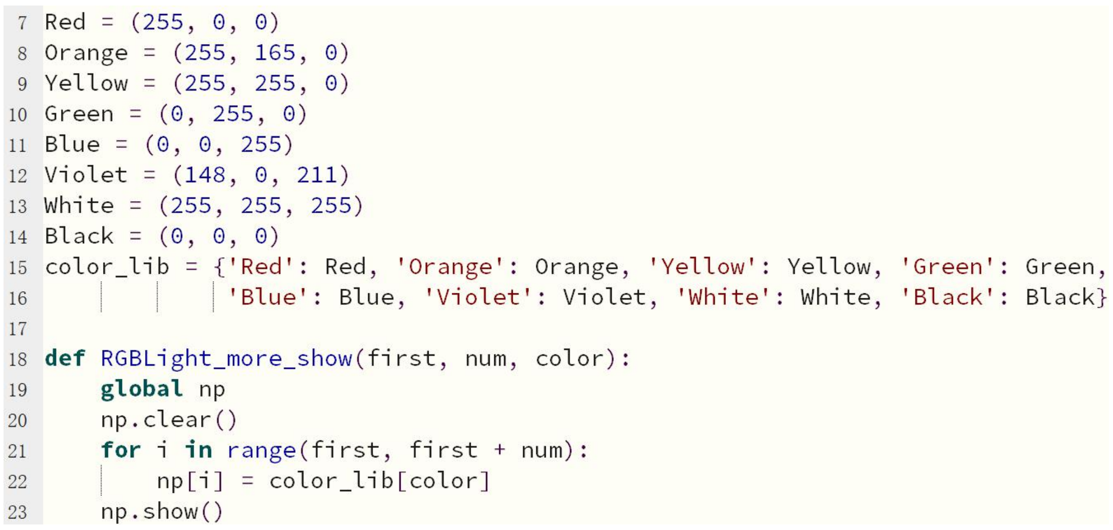
If the micro:bit logo up, the micro:bit dot matrix displays the arrow points to the south and sends the string 'D';
if the micro:bit logo up, the micro:bit dot matrix displays the arrow points to the north and sends the string 'C';
if the micro:bit tilt left, the micro:bit dot matrix display arrow points 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'.

Tumble:bit car code:



```python
# -*- coding: utf-8-*-# E
from microbit import *
import tinybit
import radio
import neopixel
```

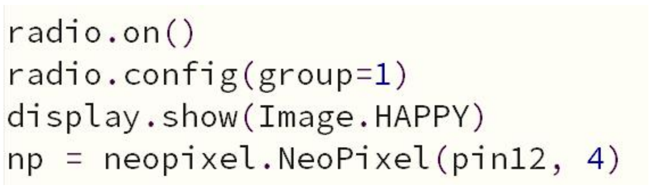
First, we need to import the library needed for this lesson from micro:bit, tinybit library is dedicated to tinybit car; neopixel is used to control RGB lights;radio for micro:bit wireless communication function.



```python
Red = (255, 0, 0)
Orange = (255, 165, 0)
Yellow = (255, 255, 0)
Green = (0, 255, 0)
Blue = (0, 0, 255)
Violet = (148, 0, 211)
White = (255, 255, 255)
Black = (0, 0, 0)
color_lib = {'Red': Red, 'Orange': Orange, 'Yellow': Yellow, 'Green': Green,
             'Blue': Blue, 'Violet': Violet, 'White': White, 'Black': Black}

def RGBLight_more_show(first, num, color):
    global np
    np.clear()
    for i in range(first, first + num):
        np[i] = color_lib[color]
    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.



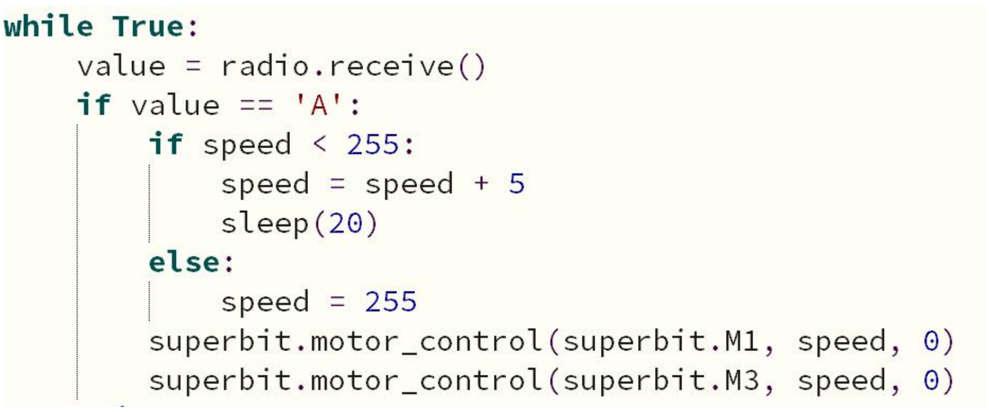
```
radio.on()
radio.config(group=1)
display.show(Image.HAPPY)
np = neopixel.NeoPixel(pin12, 4)
```

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

np = neopixel.NeoPixel (pin12, 4): RGB lamp initialization settings, a total of 4 RGB lamps, connected to the P12 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':
        if speed < 255:
            speed = speed + 5
            sleep(20)
        else:
            speed = 255
        superbit.motor_control(superbit.M1, speed, 0)
        superbit.motor_control(superbit.M3, speed, 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 and the "cannonball" will be shooted;
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;

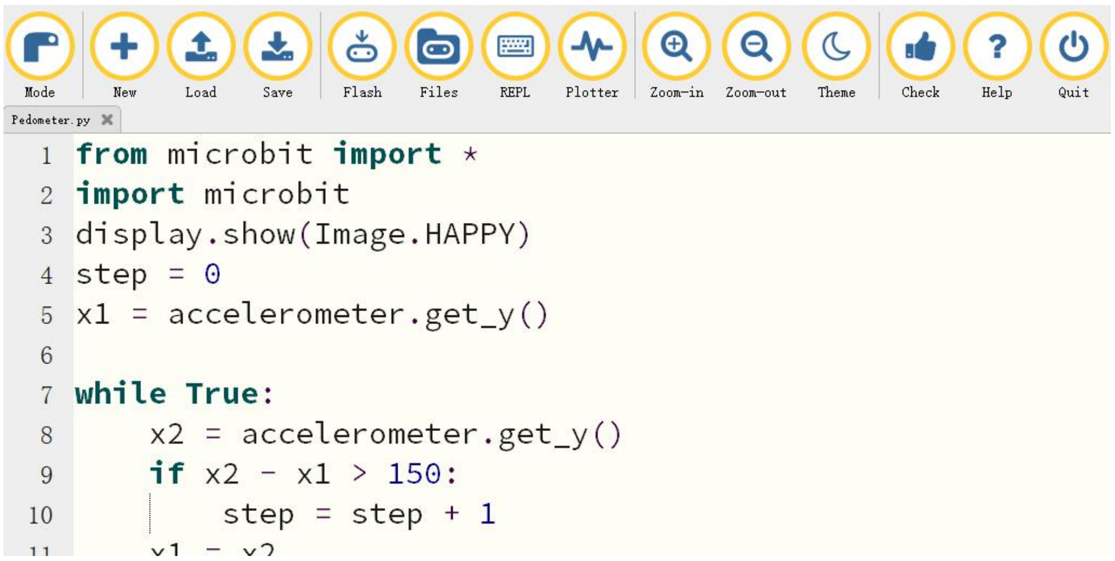
! Note:
The incoming value needs to correspond to the value sent by the wrist:bit. Only the

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 .

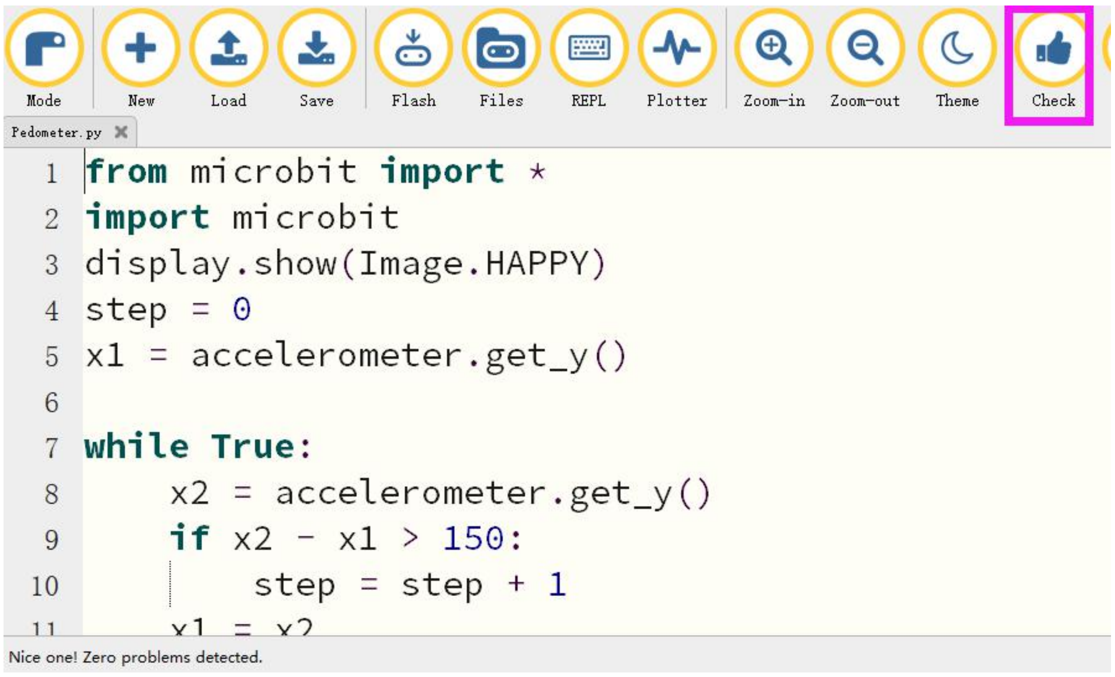
**Note! All English and symbols should be entered in English, and the last line must be a space.**



```
from microbit import *
import microbit
display.show(Image.HAPPY)
step = 0
x1 = accelerometer.get_y()

while True:
    x2 = accelerometer.get_y()
    if x2 - x1 > 150:
        step = step + 1
    x1 = x2
```

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.

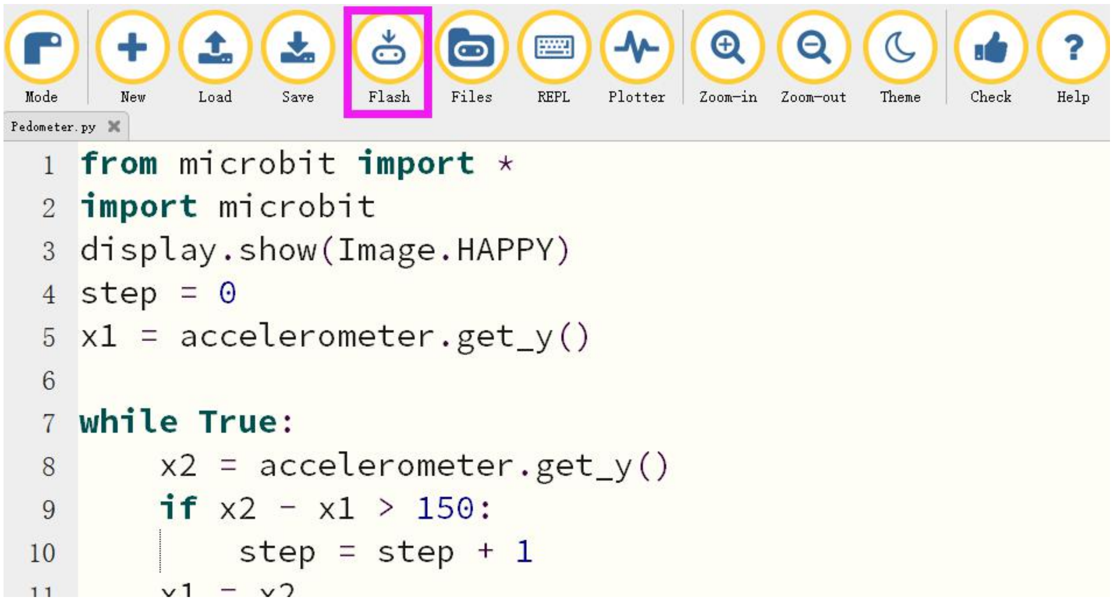


```
from microbit import *
import microbit
display.show(Image.HAPPY)
step = 0
x1 = accelerometer.get_y()

while True:
    x2 = accelerometer.get_y()
    if x2 - x1 > 150:
        step = step + 1
    x1 = x2
```

Nice one! Zero problems detected.

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 as shown in figure.

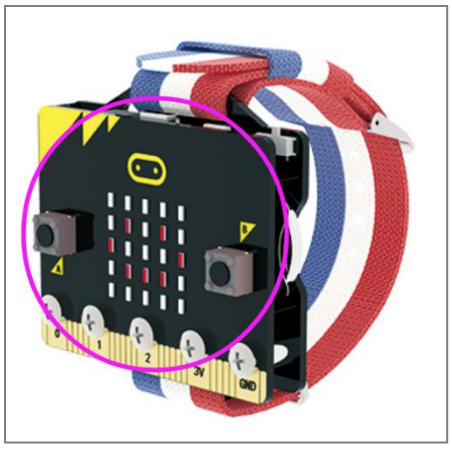


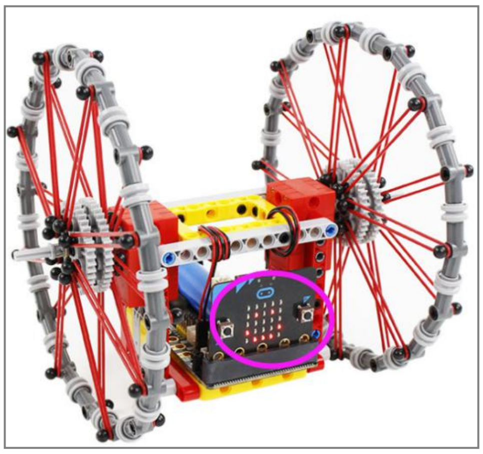
```
from microbit import *
import microbit
display.show(Image.HAPPY)
step = 0
x1 = accelerometer.get_y()

while True:
    x2 = accelerometer.get_y()
    if x2 - x1 > 150:
        step = step + 1
    x1 = x2
```

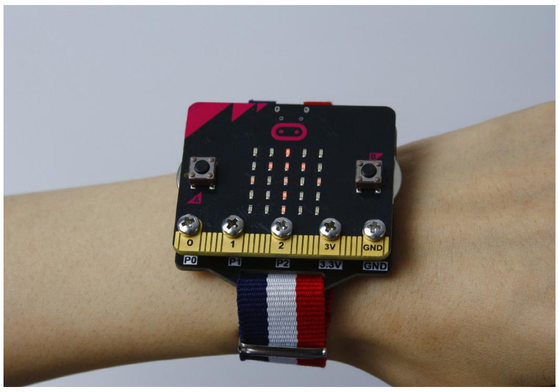
4.4 If the download failed, please confirm whether the micro:bit is connected to the computer through the micro USB data cable, and confirm whether the super:bit Python library has been imported.

## 5.Experimental phenomena





We need to download the Tumble-bit-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.