Chapter3 Breathing_light

1. About PMM:

PWM, the English name Pulse Width Modulation, is an abbreviation for pulse width modulation. It modulates the width of a series of pulses, equivalent to the required waveform (including shape and amplitude), and digitally encodes the analog signal level. That is to say, by adjusting the change of duty cycle to adjust the change of signal, energy, etc., the duty ratio refers to the percentage of the whole signal period in which the signal is at a high level in one cycle.

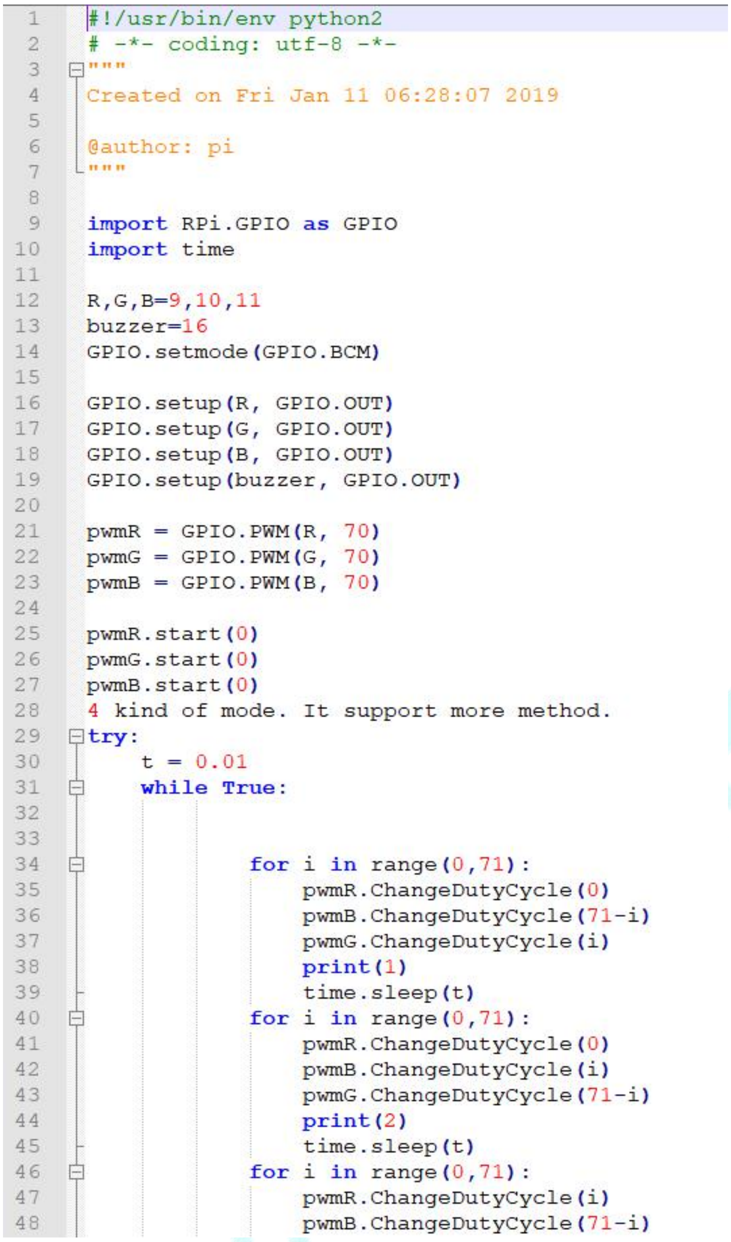
2. How to operate PWM by RPi.GPIO:

```
p = GPIO.PWM(channel, frequency)          //Create PWM instance
p.start(dc) # where dc is the duty cycle (0.0 <= dc <= 100.0)     //Start PWM
p.ChangeFrequency(freq)                   //Change Frequency
p.ChangeDutyCycle(dc)# where 0.0 <= dc <= 100.0    //ChangeDutyCycle
p.stop()                                                  // stop PWM
```

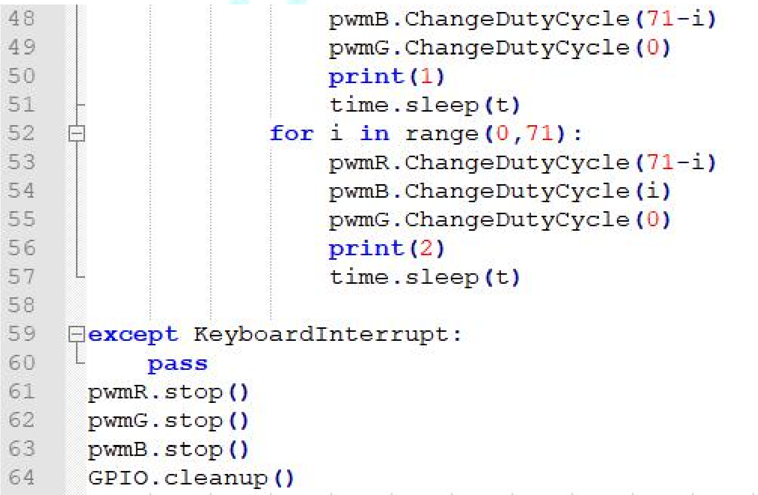
3. About code:

The code below provides two control modes for the colorful breathing light. Based on blue, alternating with red and green.

The code as shown in the figure below:
Method 1:

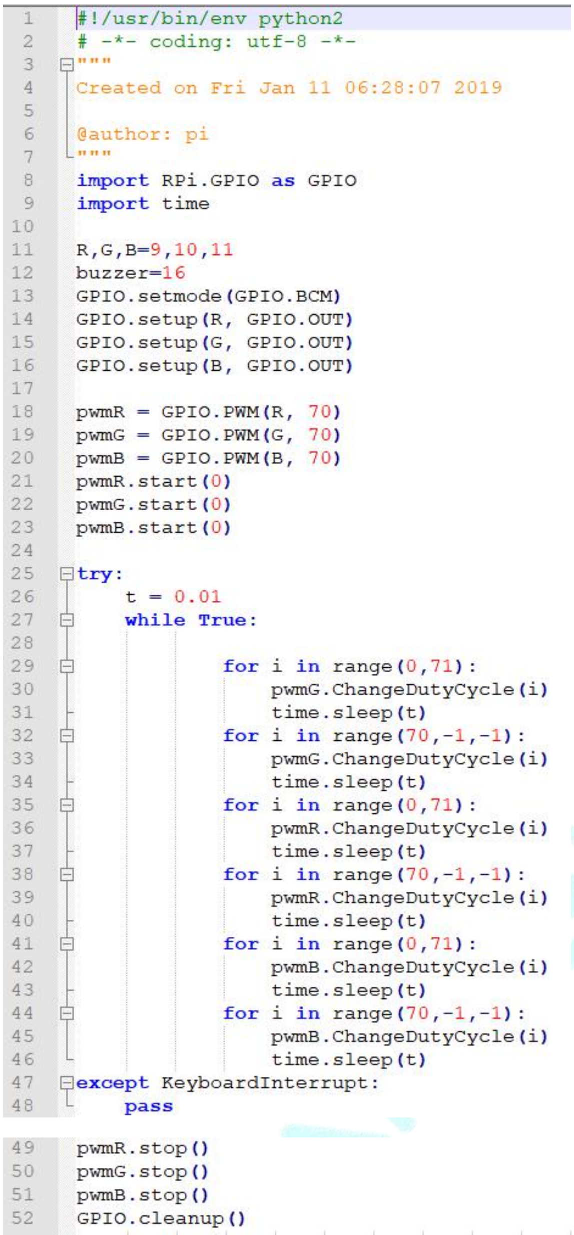


```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
"""
Created on Fri Jan 11 06:28:07 2019

@author: pi
"""

import RPi.GPIO as GPIO
import time

R,G,B=9,10,11
buzzer=16
GPIO.setmode(GPIO.BCM)

GPIO.setup(R, GPIO.OUT)
GPIO.setup(G, GPIO.OUT)
GPIO.setup(B, GPIO.OUT)
GPIO.setup(buzzer, GPIO.OUT)

pwmR = GPIO.PWM(R, 70)
pwmG = GPIO.PWM(G, 70)
pwmB = GPIO.PWM(B, 70)

pwmR.start(0)
pwmG.start(0)
pwmB.start(0)
4 kind of mode. It support more method.
try:
    t = 0.01
    while True:


            for i in range(0,71):
                pwmR.ChangeDutyCycle(0)
                pwmB.ChangeDutyCycle(71-i)
                pwmG.ChangeDutyCycle(i)
                print(1)
                time.sleep(t)
            for i in range(0,71):
                pwmR.ChangeDutyCycle(0)
                pwmB.ChangeDutyCycle(i)
                pwmG.ChangeDutyCycle(71-i)
                print(2)
                time.sleep(t)
            for i in range(0,71):
                pwmR.ChangeDutyCycle(i)
                pwmB.ChangeDutyCycle(71-i)
```



```
                pwmB.ChangeDutyCycle(71-i)
                pwmG.ChangeDutyCycle(0)
                print(1)
                time.sleep(t)
            for i in range(0,71):
                pwmR.ChangeDutyCycle(71-i)
                pwmB.ChangeDutyCycle(i)
                pwmG.ChangeDutyCycle(0)
                print(2)
                time.sleep(t)

except KeyboardInterrupt:
    pass
pwmR.stop()
pwmG.stop()
pwmB.stop()
GPIO.cleanup()
```

Method 2：



```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
"""
Created on Fri Jan 11 06:28:07 2019

@author: pi
"""
import RPi.GPIO as GPIO
import time

R,G,B=9,10,11
buzzer=16
GPIO.setmode(GPIO.BCM)
GPIO.setup(R, GPIO.OUT)
GPIO.setup(G, GPIO.OUT)
GPIO.setup(B, GPIO.OUT)

pwmR = GPIO.PWM(R, 70)
pwmG = GPIO.PWM(G, 70)
pwmB = GPIO.PWM(B, 70)
pwmR.start(0)
pwmG.start(0)
pwmB.start(0)

try:
    t = 0.01
    while True:

            for i in range(0,71):
                pwmG.ChangeDutyCycle(i)
                time.sleep(t)
            for i in range(70,-1,-1):
                pwmG.ChangeDutyCycle(i)
                time.sleep(t)
            for i in range(0,71):
                pwmR.ChangeDutyCycle(i)
                time.sleep(t)
            for i in range(70,-1,-1):
                pwmR.ChangeDutyCycle(i)
                time.sleep(t)
            for i in range(0,71):
                pwmB.ChangeDutyCycle(i)
                time.sleep(t)
            for i in range(70,-1,-1):
                pwmB.ChangeDutyCycle(i)
                time.sleep(t)
except KeyboardInterrupt:
    pass

pwmR.stop()
pwmG.stop()
pwmB.stop()
GPIO.cleanup()
```