Welcome!

Thank you for purchasing our *AZ-Delivery LED Ring 50mm module*. On the following pages, you will be introduced to how to use and set up this handy device.

Have fun!

AZ-Delivery

Table of Contents

Introduction

The LED Ring WS2812B 50mm is a ring with 12 RGB LEDs, also called pixels. It needs only one microcontroller pin to control all the pixels using a single-wire control protocol.

Each LED pixel consists of 3 LEDs and an integrated WS2812 control chip. All LEDs in the ring are individually addressable.

The state, brightness, and colour of all LEDs in each pixel can be controlled individually by a microcontroller.

The LED ring has two ports (Input and Output) so that multiple panels can be connected in series (daisy-chaining).

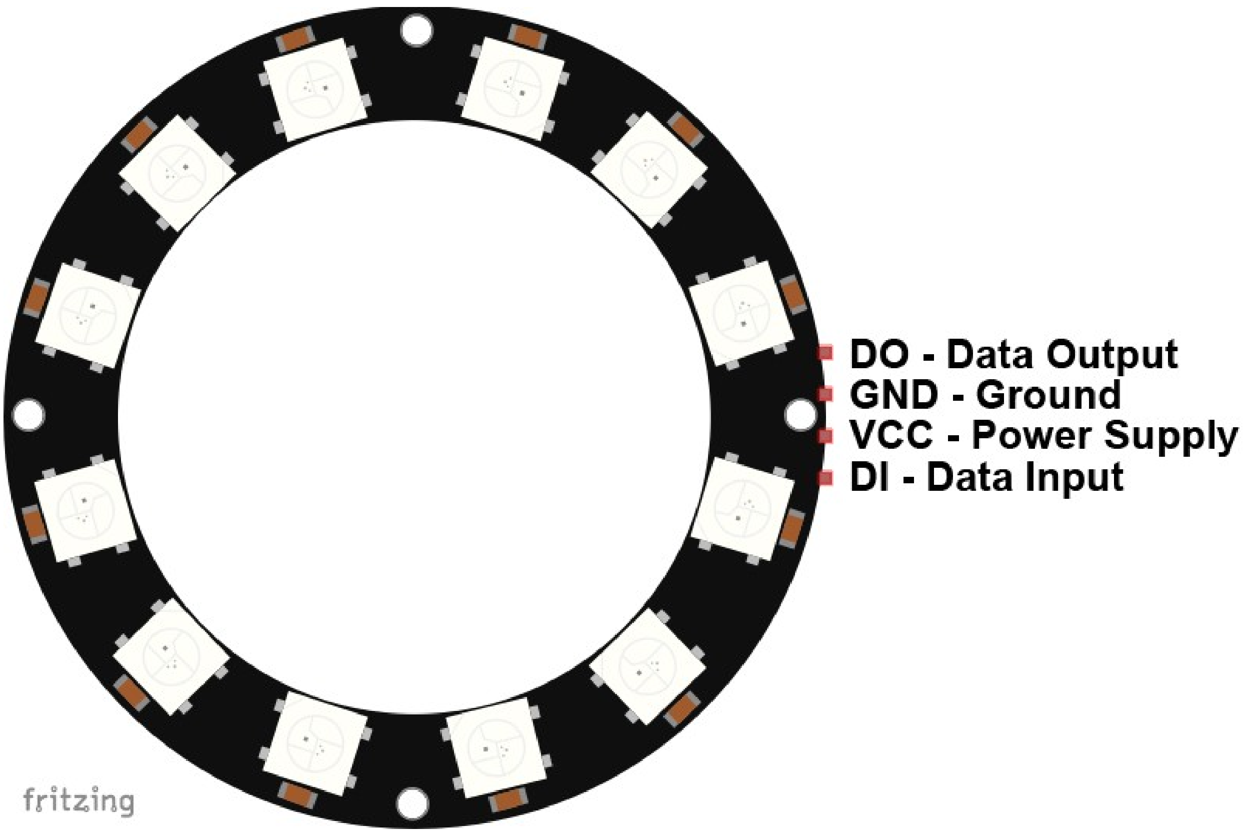
The LED Ring can be used for light painting, light effects, animations, etc. It is also used in a variety of applications such as LED decorative lighting, indoor/outdoor LED video irregular screens, etc.

Specifications

Operating voltage range	3.3 - 5V
LEDs	12 LEDs with 4mm diameter
Mounting holes	4 with 3mm diameter
Dimensions	Inside diameter: 36mm [1.41in] Outside diameter: 50mm [1.97in]

The pinout

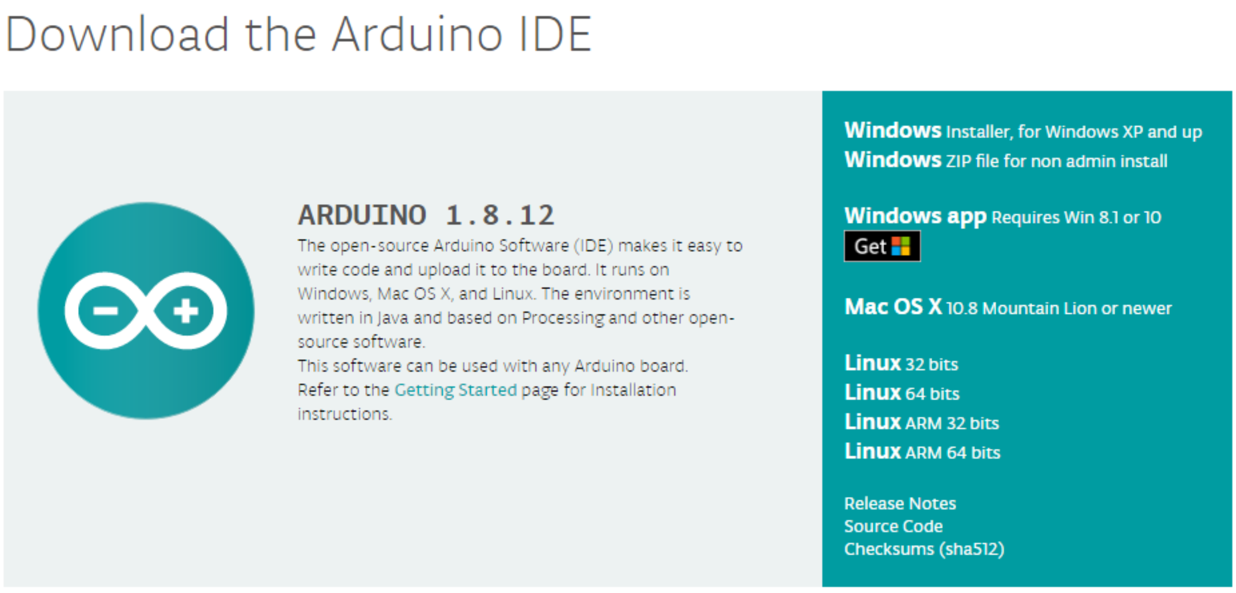
The LED Ring 50mm module has 4 pins. The pinout is shown in the following image:



Note: Before connecting the ring to any live (working) power supply, make sure that ground is always connected first, before any other wire.

How to set-up Arduino IDE

If the Arduino IDE is not installed, follow the *link* and download the installation file for the operating system of choice. The Arduino IDE version used for this eBook is **1.8.12.**


Download the Arduino IDE

ARDUINO 1.8.12

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board.
Refer to the Getting Started page for Installation instructions.

Windows Installer, for Windows XP and up
Windows ZIP file for non admin install
Windows app Requires Win 8.1 or 10
Get
Mac OS X 10.8 Mountain Lion or newer
Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits
Release Notes
Source Code
Checksums (sha512)

For *`Windows`* users, double click on the downloaded *`.exe`* file and follow the instructions in the installation window.

For *Linux* users, download a file with the extension *.tar.xz*, which has to be extracted. When it is extracted, go to the extracted directory and open the terminal in that directory. Two *.sh* scripts have to be executed, the first called *arduino-linux-setup.sh* and the second called *install.sh*.

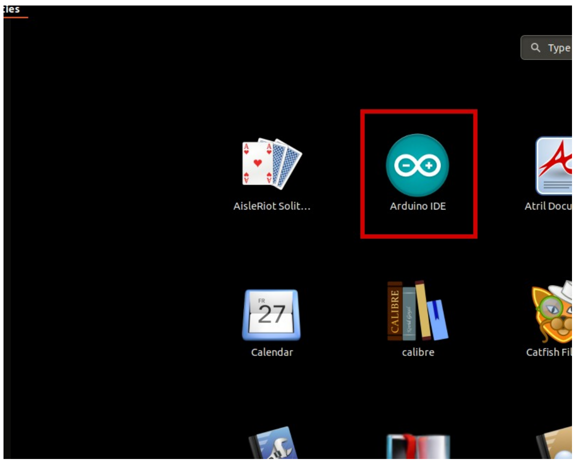
To run the first script in the terminal, open the terminal in the extracted directory and run the following command:

```
sh arduino-linux-setup.sh user_name
```

user_name - is the name of a superuser in Linux operating system. A password for the superuser has to be entered when the command is started. Wait for a few minutes for the script to complete everything.

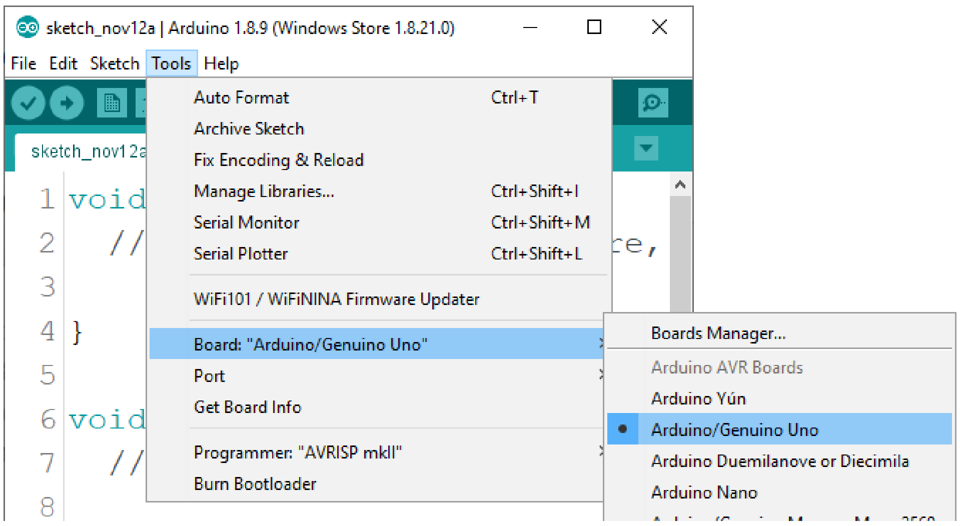
The second script, called *install.sh*, has to be used after the installation of the first script. Run the following command in the terminal (extracted directory): **`sh install.sh`**

After the installation of these scripts, go to the *All Apps*, where the *Arduino IDE* is installed.



Almost all operating systems come with a text editor preinstalled (for example, *`Windows`* comes with *`Notepad`*, *`Linux Ubuntu`* comes with *`Gedit`*, *`Linux Raspbian`* comes with *`Leafpad`*, etc.). All of these text editors are perfectly fine for the purpose of the eBook.

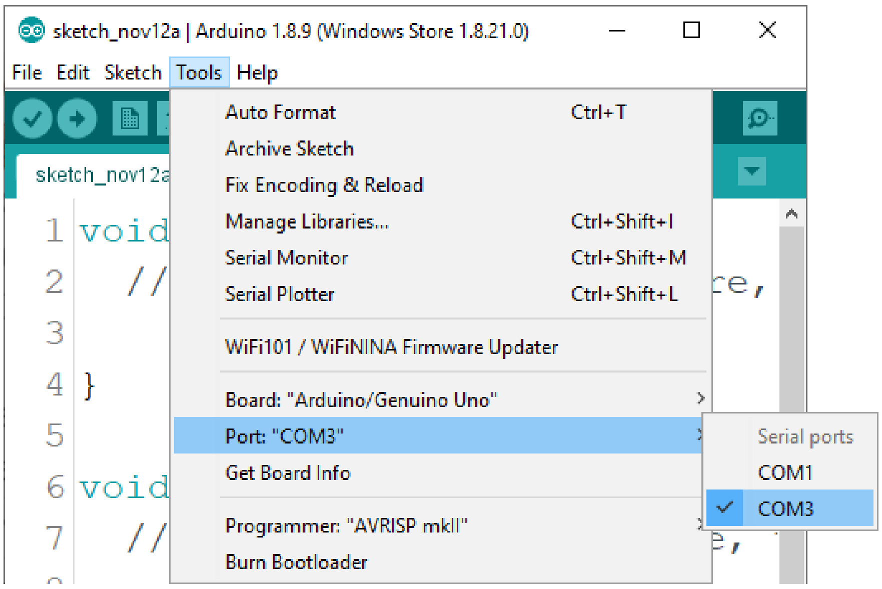
Next thing is to check if your PC can detect an Arduino board. Open freshly installed Arduino IDE, and go to:
Tools > Board > {your board name here}
{your board name here} should be the *`Arduino/Genuino Uno`*, as it can be seen on the following image:



sketch_nov12a | Arduino 1.8.9 (Windows Store 1.8.21.0)
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino/Genuino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Yún
Arduino/Genuino Uno
Arduino Duemilanove or Diecimila
Arduino Nano

The port to which the Arduino board is connected has to be selected. Go to:
Tools > Port > {port name goes here}
and when the Arduino board is connected to the USB port, the port name can be seen in the drop-down menu on the previous image.

If the Arduino IDE is used on Windows, port names are as follows:


sketch_nov12a | Arduino 1.8.9 (Windows Store 1.8.21.0)
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Get Board Info
Burn Bootloader
Serial ports
COM1
COM3

Board: "Arduino/Genuino Uno"
Port: "COM3"
Programmer: "AVRISP mkII"

For *`Linux`* users, for example, port name is *`/dev/ttyUSBx`*, where *`x`* represents integer number between *0* and *9*.

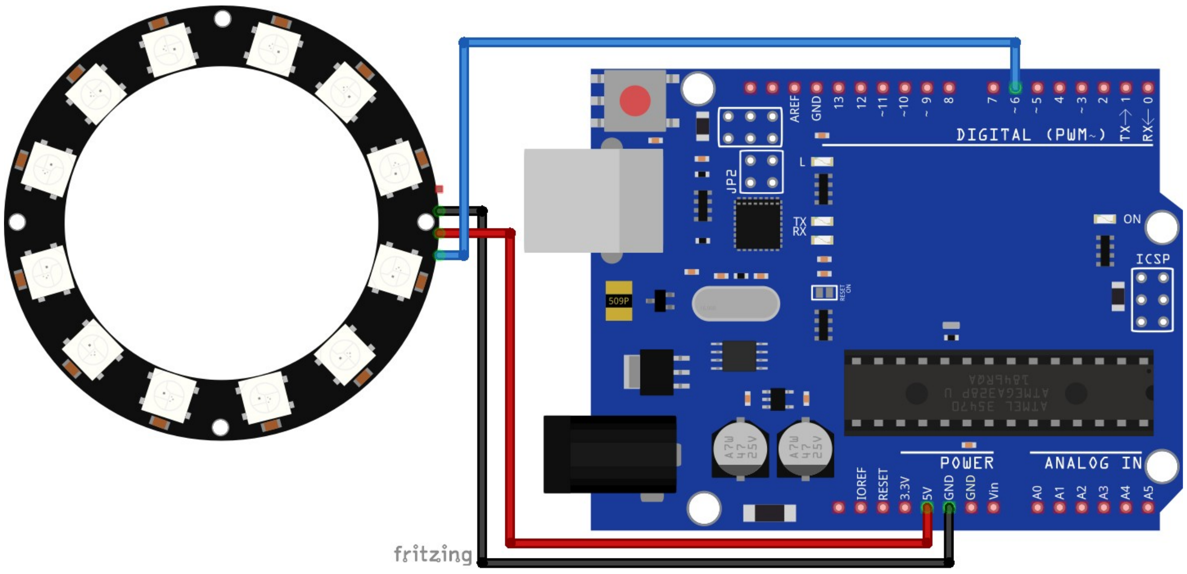
How to set-up the Raspberry Pi and Python

For the Raspberry Pi, first the operating system has to be installed, then everything has to be set-up so that it can be used in the `Headless` mode. The `Headless` mode enables remote connection to the Raspberry Pi, without the need for a *PC* screen Monitor, mouse or keyboard. The only things that are used in this mode are the Raspberry Pi itself, power supply and internet connection. All of this is explained minutely in the free eBook:
Raspberry Pi Quick Startup Guide

The *Raspberry Pi OS (*operating system), previously known as Raspbian, comes with `Python` preinstalled.

Connecting the module with Uno

Connect the module with the Uno as shown on the following connection diagram:



Module pin	Uno pin	Wire color
GND	GND	**Black wire**
5V	5V	**Red wire**
DI	~ 6	**Blue wire**

Sketch example

```
#include <Adafruit_NeoPixel.h>

#define LED_PIN    6
#define LED_COUNT 12

Adafruit_NeoPixel ring(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);

void setup() {
  ring.begin();
  ring.show();
  ring.setBrightness(50);
}

void loop() {

  colorWipe(ring.Color(255,   0,   0), 50); // Red
  colorWipe(ring.Color(  0, 255,   0), 50); // Green
  colorWipe(ring.Color(  0,   0, 255), 50); // Blue


  theaterChase(ring.Color(127, 127, 127), 50); // White, half brightness
  theaterChase(ring.Color(127,   0,   0), 50); // Red, half brightness
  theaterChase(ring.Color(  0,   0, 127), 50); // Blue, half brightness

  rainbow(10);             // Flowing rainbow cycle along the whole ring
  theaterChaseRainbow(50); // Rainbow-enhanced theaterChase variant
}
```

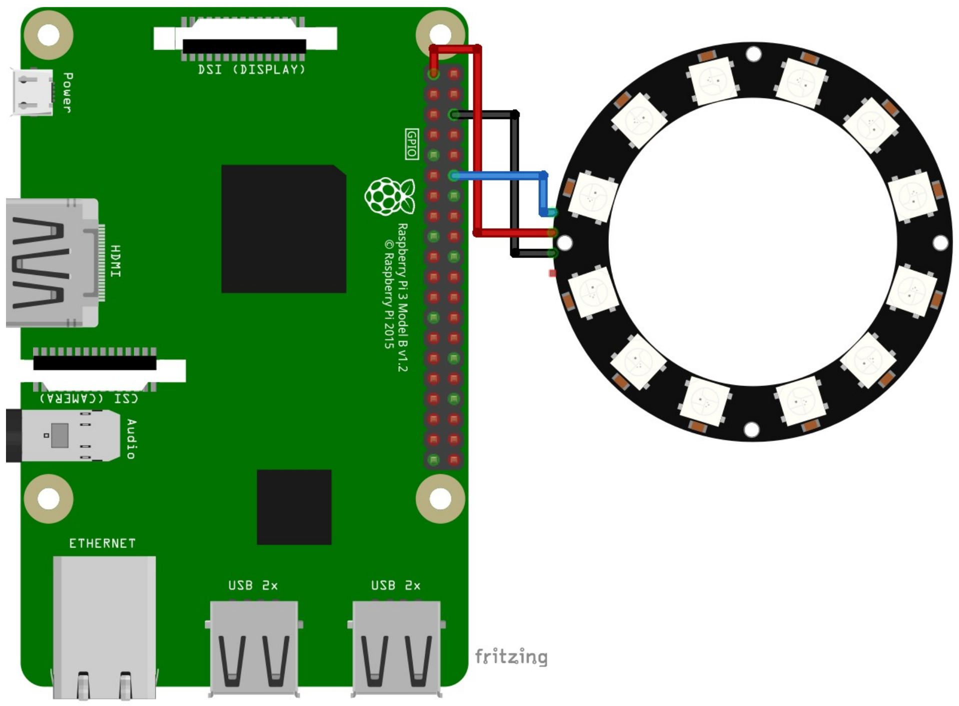
```
void colorWipe(uint32_t color, int wait) {
  for(int i=0; i<ring.numPixels(); i++) {
    ring.setPixelColor(i, color);
    ring.show();
    delay(wait);
  }
}

void theaterChase(uint32_t color, int wait) {
  for(int a=0; a<10; a++) {
    for(int b=0; b<3; b++) {
      ring.clear();
      for(int c=b; c<ring.numPixels(); c += 3) {
        ring.setPixelColor(c, color); // Set pixel 'c' to value 'color'
      }
      ring.show();
      delay(wait);
    }
  }
}

void rainbow(int wait) {

  for(long firstPixelHue = 0; firstPixelHue < 5*65536; firstPixelHue += 256) {
    for(int i=0; i<ring.numPixels(); i++) {

      int pixelHue = firstPixelHue + (i * 65536L / ring.numPixels());

      ring.setPixelColor(i, ring.gamma32(ring.ColorHSV(pixelHue)));
    }
    ring.show();
    delay(wait);
  }
}
```

```
void theaterChaseRainbow(int wait) {
  int firstPixelHue = 0;     // First pixel starts at red (hue 0)
  for(int a=0; a<30; a++) {
    for(int b=0; b<3; b++) {
      ring.clear();
      for(int c=b; c<ring.numPixels(); c += 3) {

        int      hue   = firstPixelHue + c * 65536L / ring.numPixels();
        uint32_t color = ring.gamma32(ring.ColorHSV(hue));
        ring.setPixelColor(c, color);
      }
      ring.show();
      delay(wait);
      firstPixelHue += 65536 / 90;
    }
  }
}
```

NOTE: For "daisy-chaining" two LED Rings, simply increase the number LED_COUNT to 24.

Connecting the module with Raspberry Pi

Connect the module with the Raspberry Pi as shown on the following connection diagram:



Module pin	Raspberry Pi pin	Physical pin	Wire color
VCC	3V3	1	**Red wire**
GND	GND	6	**Black wire**
DI	GPIO18	12	**Blue wire**

Libraries and tools for Python

To use the device with the Raspberry Pi it is recommended to download an external Python library. The library that is used in this eBook is called the *Adafruit_CircuitPython_NeoPixel*.

Before the library can be used, run the following commands:

```
sudo apt-get update
sudo pip3 install rpi_ws281x adafruit-circuitpython-neopixel
sudo python3 -m pip install --force-reinstall adafruit-blinka
```

Next, to download an external library, run the following command:

```
git clone https://github.com/adafruit/Adafruit_CircuitPython_NeoPixel.git
```

To install it, first change to the new directory *Adafruit_CircuitPython_NeoPixel*, by running the following command:

```
cd  Adafruit_CircuitPython_NeoPixel
```

and install the library with the following command:

```
sudo python3 setup.py install
```

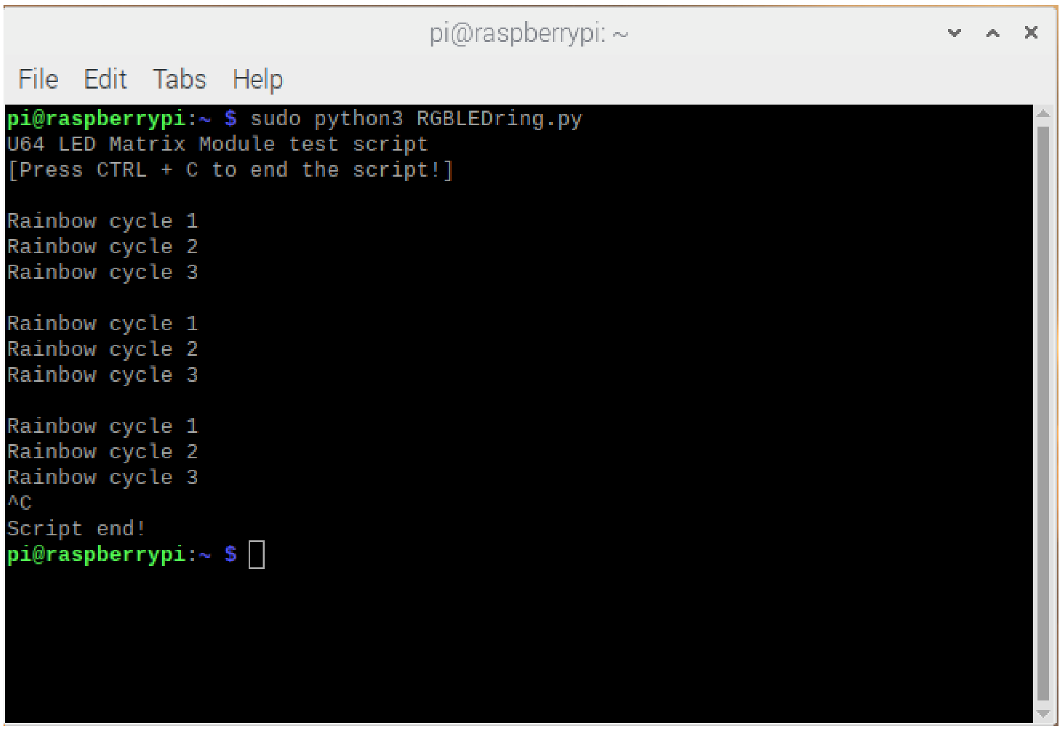
Python script

```
import time
import board
import neopixel

pixel_pin = board.D18
num_pixels = 12
ORDER = neopixel.GRB

pixels = neopixel.NeoPixel(
    pixel_pin, num_pixels, brightness=0.2, auto_write=False, pixel_order=ORDER
)

def wheel(pos):
    if pos < 0 or pos > 255:
        r = g = b = 0
    elif pos < 85:
        r = int(pos * 3)
        g = int(255 - pos * 3)
        b = 0
    elif pos < 170:
        pos -= 85
        r = int(255 - pos * 3)
        g = 0
        b = int(pos * 3)
    else:
        pos -= 170
        r = 0
        g = int(pos * 3)
        b = int(255 - pos * 3)
    return (r, g, b) if ORDER in (neopixel.RGB, neopixel.GRB) else (r, g, b,
0)
```

```
def rainbow_cycle(wait):
    for j in range(255):
        for i in range(num_pixels):
            pixel_index = (i * 256 // num_pixels) + j
            pixels[i] = wheel(pixel_index & 255)
        pixels.show()
        time.sleep(wait)

print('U64 LED Matrix Module test script')
print('[Press CTRL + C to end the script!]')

try:
    while True:
        print('\nRainbow cycle 1')
        pixels.fill((255, 0, 0))
        pixels.show()
        time.sleep(1)
        print('Rainbow cycle 2')
        pixels.fill((0, 255, 0))
        pixels.show()
        time.sleep(1)
        print('Rainbow cycle 3')
        pixels.fill((0, 0, 255))
        pixels.show()
        time.sleep(1)

        rainbow_cycle(0.001)

except KeyboardInterrupt:
    print('\nScript end!')
```

Save the script under the name *`RGBLEDring.py`*. To run the script open the terminal in the directory where the script is saved and run the following command: **`sudo python3 RGBLEDring.py`**

The result should look like as on the following image:



```
pi@raspberrypi:~ $ sudo python3 RGBLEDring.py
U64 LED Matrix Module test script
[Press CTRL + C to end the script!]

Rainbow cycle 1
Rainbow cycle 2
Rainbow cycle 3

Rainbow cycle 1
Rainbow cycle 2
Rainbow cycle 3

Rainbow cycle 1
Rainbow cycle 2
Rainbow cycle 3
^C
Script end!
pi@raspberrypi:~ $
```

To end the script press 'CTRL + C' on the keyboard.

You've done it!

Now you can use your module for various projects.

Now it is the time to learn and make your own projects. You can do that with the help of many example scripts and other tutorials, which can be found on the Internet.

If you are looking for the high quality products for Arduino and Raspberry Pi, AZ-Delivery Vertriebs GmbH is the right company to get them from. You will be provided with numerous application examples, full installation guides, eBooks, libraries and assistance from our technical experts.

https://az-delivery.de

Have Fun!

Impressum

https://az-delivery.de/pages/about-us