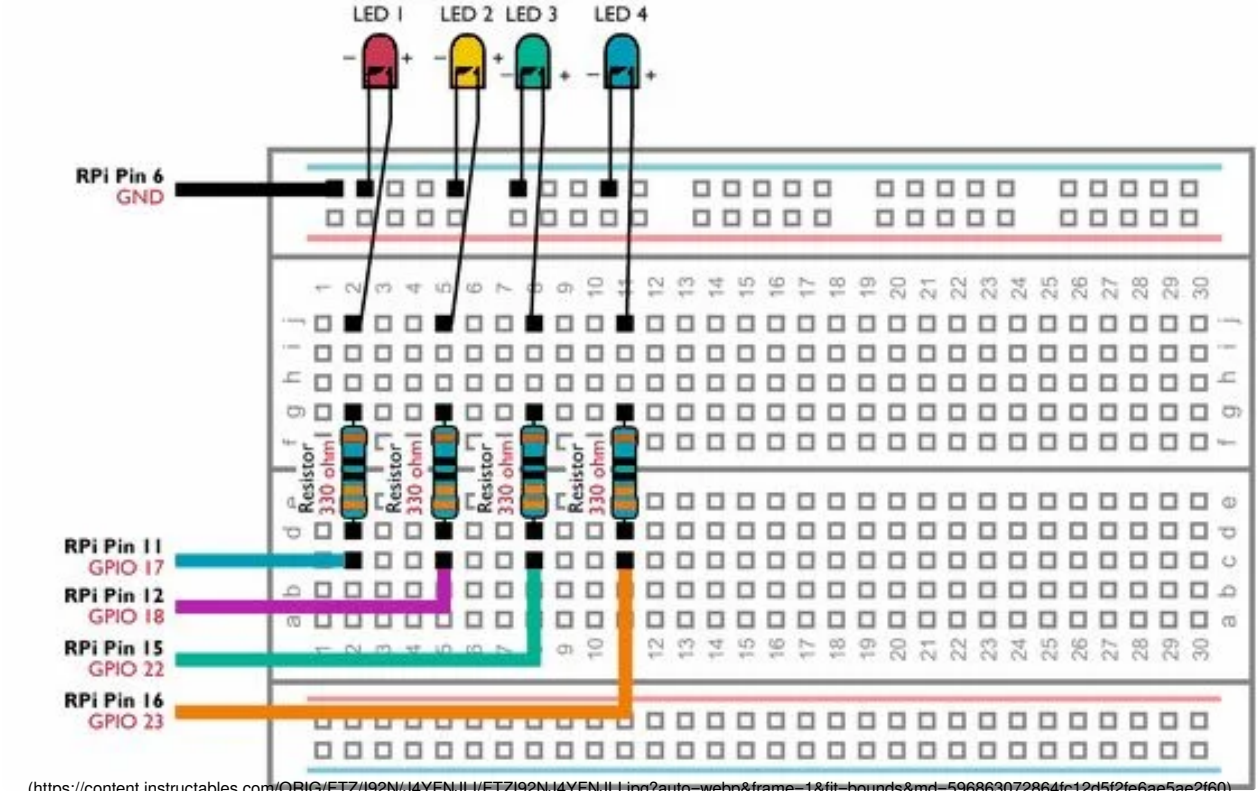
Controlling Multiple LEDs With Python and Your Raspberry Pi's GPIO Pins

By ComponentsPlus (/member/ComponentsPlus/) in Circuits (/circuits/) > Raspberry Pi (/circuits/raspberry-pi/projects/) 28,603 40 7 Featured

 Download Favorite



(/member/ComponentsPlus/)
By **ComponentsPlus (/member/ComponentsPlus/)**
Click here to visit my site (http://componentsplus.co.uk/search-electrical-components) Follow

More by the author:

About: Components Plus specialise in electronic components for hobbyists and makers alike. We are passionate about anything Arduino, RaspberryPi and robotics. We provide a wide range of cheap components aimed at insp… More About ComponentsPlus » (/member/ComponentsPlus/)

This Instructable demonstrates how to control multiple GPIO pins on your RaspberryPi to power 4 LEDs. It will also introduce you to **parameters** and **conditional statements** in Python.

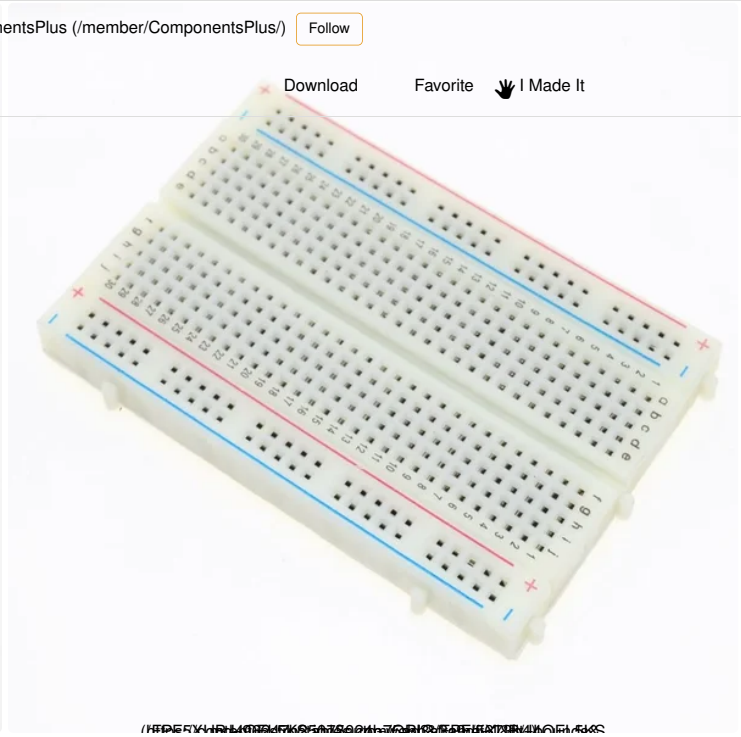
Our previous Instructable Using Your Raspberry Pi's GPIO Pins to Control an LED (https://www.instructables.com/id/Using-Your-Raspberry-Pis-GPIO-Pins-to-Control-an-L/) demonstrates how to switch a single LED on and off by using the **GPIO.output** command. This Instructable builds on that knowledge to teach you how to obtain more control over your circuit.

Add Tip Ask Question Comment Download

Step 1: What You Will Need

Controlling Multiple LEDs With Python and Your Raspberry Pi's GPIO Pins by ComponentsPlus (/member/ComponentsPlus/) Follow

Download Favorite I Made It



- A RaspberryPi with Raspbian already installed. You will also need to be able to access the Pi using a Monitor, Mouse and Keyboard or via Remote Desktop. You can use any model of Raspberry Pi. If you have one of the Pi Zero models, you may want to solder some header pins to the GPIO port.

- **Red, Blue, Yellow and Green LEDs** (http://componentsplus.co.uk/opto-electronics-and-displays/)

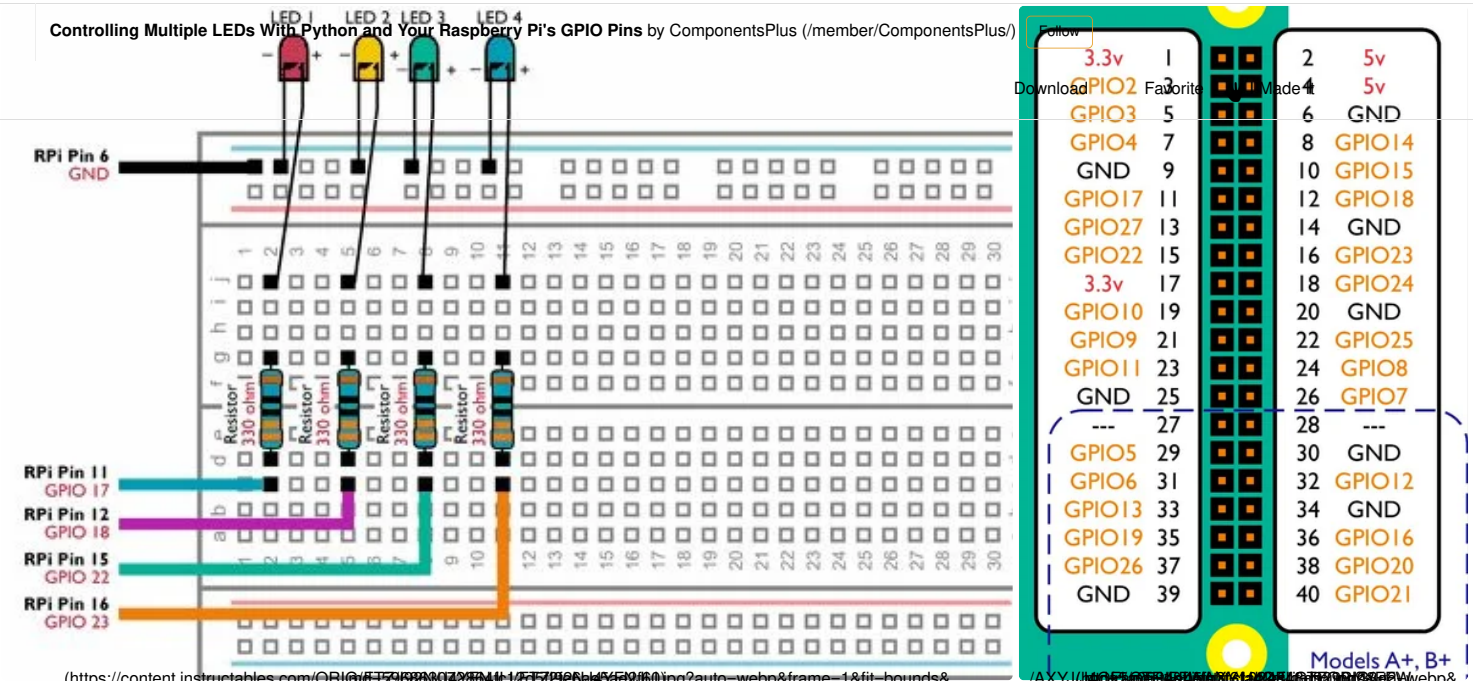
- A Solderless **Prototyping Breadboard** (http://componentsplus.co.uk/product/solderless-breadboard-for-prototyping-400-holes)

- 4 x **330 ohm Resistors** (http://componentsplus.co.uk/product/330-ohm-0-25-watt-metal-film-resistors-10-pack)

- Some **Male to Female jumper wires** (http://componentsplus.co.uk/product/dupont-jumper-cables-jumper-leads-male-to-female)

Add Tip Ask Question Comment Download

Step 2: Build Your Circuit



Build the above circuit on your breadboard making sure that none of the components leads are touching and that the LEDs are connected the correct way round.

How do you identify the positive and negative leads (the polarity) on your LEDs? If you look at an LED closely, you will see that it has two small pieces of metal inside the coloured casing. These are called the Anode and Cathode. The Cathode is the largest of the two and is also connected to the LEDs negative lead.

Once you have checked your circuit, connect the jumper cables your Raspberry Pi's GPIO pins by following the above diagram.

Step 3: Create a Script to Control and Test the LEDs

On your Raspberry Pi, open IDLE (Menu > Programming > Python 2 (IDLE)).

Open a new project go to File > New File. Then type (or copy and paste) the following code:

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(17, GPIO.OUT)
GPIO.setup(18, GPIO.OUT)
GPIO.setup(22, GPIO.OUT)
GPIO.setup(23, GPIO.OUT)
GPIO.output(17, True)
time.sleep(3)
GPIO.output(17, False)
time.sleep(1)
GPIO.output(18, True)
time.sleep(3)
GPIO.output(18, False)
time.sleep(1)
GPIO.output(22, True)
time.sleep(3)
GPIO.output(22, False)
time.sleep(1)
GPIO.output(23, True)
time.sleep(3)
GPIO.output(23, False)
```

Save your project as multilights.py (File > Save As) in your Raspberry Pis Documents folder.

On your Raspberry Pi open Terminal (Menu > Accessories > Terminal) and navigate to your Documents folder by typing the following:

```
cd /home/pi/Documents
```

You can now run your new script by typing the following:

```
python multilights.py
```

The lights will take it in turn to switch on and off. The above script uses the **time.sleep** command to create a pause between each step, making each light stay on for 3 seconds and to wait for 1 second before turning the next light on.

Step 4: Adding Flexibility by Using Parameters and Conditional Statements

By using **Parameters** and **Conditional Statements** we can make the above script much more flexible.

A **Parameter** allows you to store a value which you can use later in the script. The most common types of values are strings (text), integers (whole numbers) or floats (decimal numbers).

A Conditional Statement will determine whether or not a segment of code should be executed by checking whether a certain condition is met. The condition can also involve parameters.

Open IDLE on your Raspberry Pi and open a new project (File > New File). Then type the following. **Be careful to ensure that all of the indents (tabs) are included** by using the tab key:

```
import RPi.GPIO as GPIO
import time
from sys import argv
whichled=argv[1]
ledaction = argv[2]
LEDa=17
LEDb=18
LEDc=22
LEDd=23
GPIO.setmode(GPIO.BCM)
GPIO.setup(LEDa, GPIO.OUT)
GPIO.setmode(GPIO.BCM)
GPIO.setup(LEDb, GPIO.OUT)
GPIO.setmode(GPIO.BCM)
GPIO.setup(LEDc, GPIO.OUT)
GPIO.setmode(GPIO.BCM)
GPIO.setup(LEDd, GPIO.OUT)
if ledaction=="off":
    if whichled=="a":
        GPIO.output(LEDa, False)
    if whichled=="b":
        GPIO.output(LEDb, False)
    if whichled=="c":
        GPIO.output(LEDc, False)
    if whichled=="d":
```

Save your project as **controllight.py** (File > Save As) in your Documents folder.
Now open Terminal (Menu > Accessories > Terminal) and type the following command:

```
python controllight.py b on
```

The second LED should switch on. Now type the following:

python controllight.py b off

The second LED should switch off.

In lines 5, 6, 7 & 8, we create the parameters LEDa, LEDb, LEDc and LEDd to store which GPIO pin we have connected to which LED. This enables us to use alternative GPIO pins without having to make substantial changes to the script.

For example, if we were to connect the first LEDs lead to Pin 3 (GPIO 2) instead, we would just need to change line 5 to the following:

```
LEDa=2
```

Line 4 stores the values you typed after controllight.py into the parameters **whichled** (c) and **ledaction** (on). The script then uses these parameters, alongside a number of Conditional Statements to decide which LED to control and whether to switch it on or off.

Line 16 (if ledaction=="on":) is a **conditional statement**. The indented lines that follow this statement will only run if the statement's condition is met. In this scenario, the condition is that **ledaction** contains the text **on**.

By reading through the script's other Conditional Statements, can you predict what will happen when you type the following command in Terminal?

```
python controllight.py all on
```

Why not give it a go and post your answer in the comments section below.