Raspberry Pi LEGO® GPIOZero Traffic Lights

Tutorial by Andrew Oakley

www.cotswoldjam.org

Components

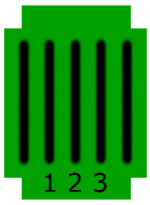
An LED has a short leg and a long leg. If you feel around the rim, you'll also find a flat edge. The short leg and flat edge always connect to negative (ground).

LEDs always need to be connected with a resistor. If you connect them without a resistor, they will burn out and probably won't work again.

We use Broadcom 5mm LEDS which fit into LEGO® Technic holes. Some other brands fit, some don't. We ordered part numbers HLMP-3301 (Red), HLMP-3401 (Yellow) and HLMP-3507 (Green) from cpc.farnell.com . Order a small quantity and check the fit before ordering in bulk.

The resistor can be connected any way around. Lower value (lower ohm) resistors will allow more current through and will make the LED brighter; higher values will let less current through and make it dimmer. We're using a 270 ohm resistor but anything between 220-470 ohms will work fine. We will use one resistor connected to ground to cover all 3 LEDs.

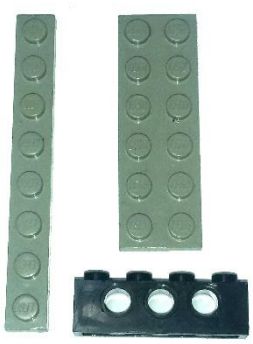
This micro breadboard has 25 connection points. The columns are numbered from 0 to 4 (some have the middle columns 1, 2 and 3 marked). All points on a column are connected together, but the different columns are not connected. You can use a larger breadboard if you wish.



You will also need:
• Four 10cm male-to-female jumper cables
• Three 20cm female-to-female jumper cables
Male ends have sticky-out wire, female ends have sockets.

You will also need an up-to-date version of Raspbian on your Raspberry Pi. It must be no older than 24 November 2015, so that GPIOZero is included.

These three LEGO® bricks can be made into a traffic light, and the LEDs fit inside the holes of the Technic brick.



The part numbers for these bricks – which can be purchased second-hand on sites such Bricklink.com – are:

3701 – Technic, Brick, 1x4 with holes, black
3460 – Plate, 1x8, grey
3795 – Plate, 2x6, grey

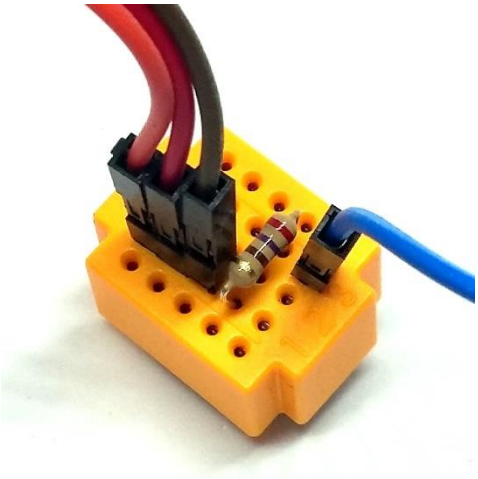
Push the LEDs into the Technic brick so that the short legs of each LED are at the top of the brick.

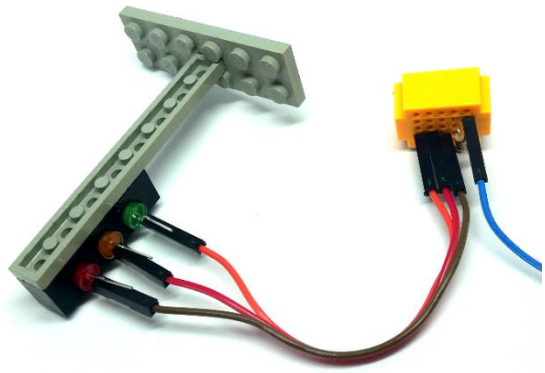
Now take the four 10cm male-to-female wires.

Plug three of the 10cm wires into column 1.

Plug another 10cm wire into column 3.

It does not matter what colour the wires are – the author of this tutorial is colour-blind.

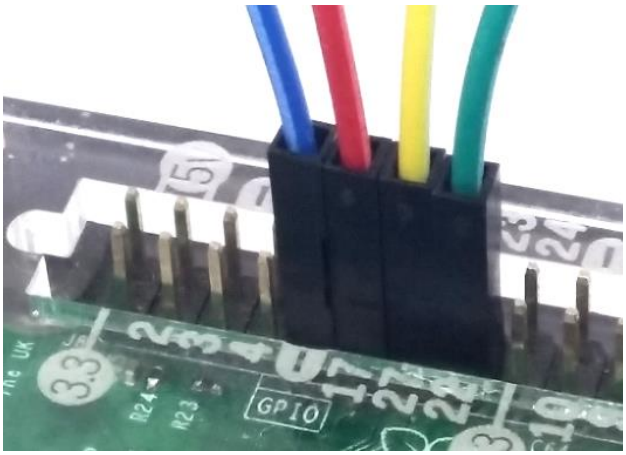




Next plug the resistor so that it connects column 1 and column 3.

Now plug in the 3 wires from column 1, into the negative (ground) legs of the LEDs. The ground legs are the short legs.

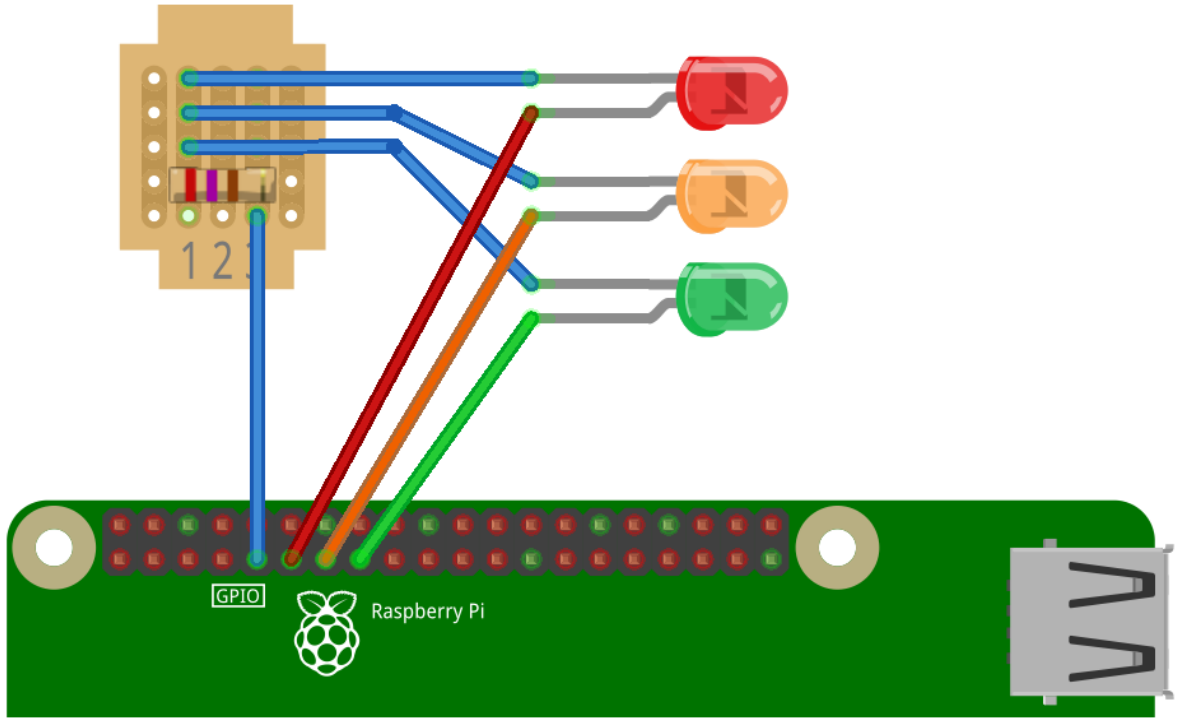
Next, take the three 20cm female-to-female wires and plug them into the positive (long) legs of the LEDs.



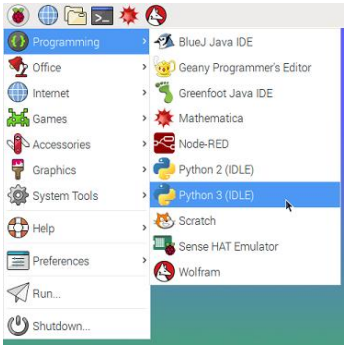
Connect the other ends of the female-female wires as follows. All the pins are on the bottom row, as you look at your Raspberry Pi with the writing the correct way up.

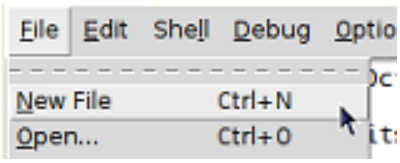
Red – GPIO17 – 6th from the left
Yellow – GPIO27 – 7th from the left
Green – GPIO22 – 8th from the left

Finally connect the remaining 10cm male-female wire from column 3 of the breadboard, to any Ground pin (negative, minus) such as the 5th from the left.



Turn on your Raspberry Pi. From the desktop, select the Raspberry Pi menu – Programming – Python 3 (IDLE) .





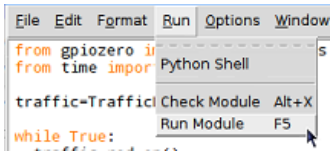
Create a new program with the File menu – New File .

Type in the following program or load it with File menu – Open and pick the Python directory – traffic subdirectory – traffic.py

```
from gpiozero import TrafficLights
from time import sleep

traffic=TrafficLights(17,27,22)

while True:
  traffic.red.on()
  sleep(2)
  traffic.yellow.on()
  sleep(1)
  traffic.red.off()
  traffic.yellow.off()
  traffic.green.on()
  sleep(2)
  traffic.green.off()
  traffic.yellow.on()
  sleep(1)
  traffic.yellow.off()
```



Run the program with Run menu – Run Module.

You may be asked to save the program. Type a name for the program (such as mytraffic) and click Save.

Your program should run, and you should see the lights move through the British traffic light sequence; red; red and yellow; green; yellow; then back to red.

To stop the program, hold down the Ctrl key and press C, or select File menu – Close from the Shell window (you'll be asked "Do you want to kill it?" – click OK).

Didn't work? Check your wiring and check the spelling and spacing of your program. If you are using an original first edition Raspberry Pi and the yellow LED doesn't work, you'll need to change TrafficLights(17,27,22) to TrafficLights(17,21,22) – that's 21 not 27.

What is this program doing?

```
from gpiozero import TrafficLights
from time import sleep
```

The Python programming language doesn't know about GPIO pins, nor how to wait for a few seconds. We teach it how to do this using a "library" – a small program that someone else has written, from which we select new commands.

```
traffic=TrafficLights(17,27,22)
```

Here we're creating a set of traffic lights connected to GPIO pins 17, 27 and 22. We call our set "traffic". The technical term for a set of things like this, is called an Object.

```
while True:
```

This means that we repeat the following commands forever. Note how all the commands which we want to repeat, are indented by a couple of spaces. Commands that are grouped together like this are called a "block".

```
  traffic.green.on()
  sleep(2)
  traffic.green.off()
```

These commands turn a light on, wait for 2 seconds, and then turn a light off. We can change the colour of the light (red, yellow, green) or the length of the wait.

What next?

Try to get two sets of traffic lights working – maybe you could have some Lego roadworks! Make sure that one set is green when the other set is red – you don't want a car crash!

You could use GPIO 18, 23 and 24 which are the 6th, 8th and 9th pins from the left on the top row. There's an extra ground on the 7th pin from the left on the top row.