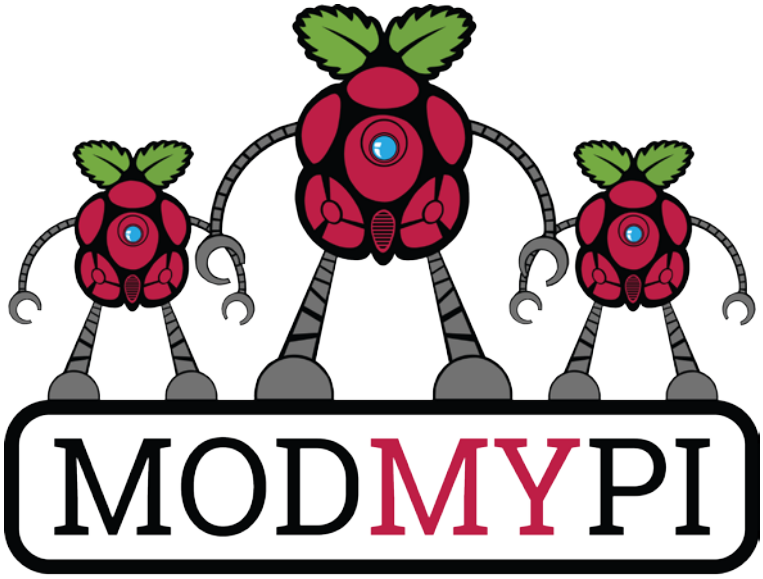


YouTube

Workshop Kit v2.0

Tutorial 8/9: LDR

Contents

Things you will need

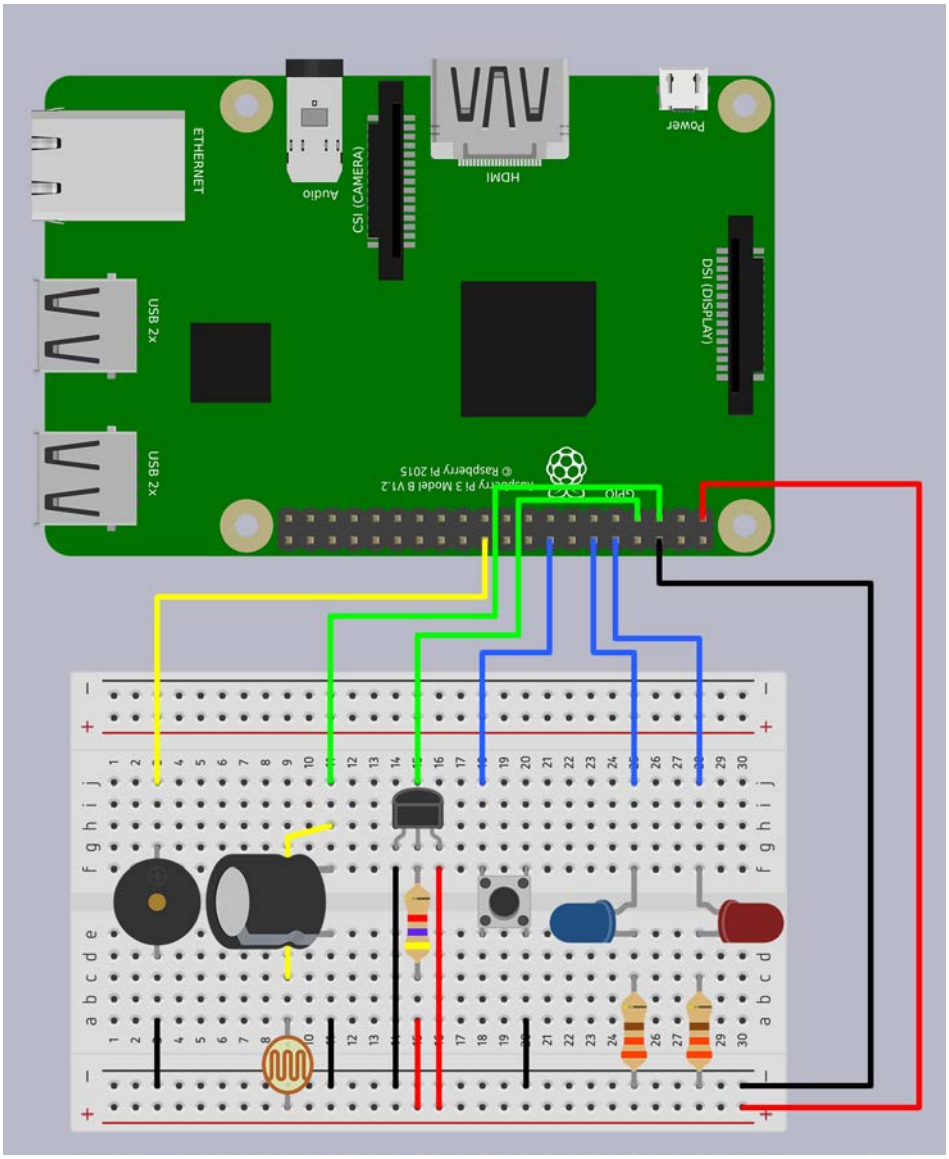
Raspberry Pi 3 Model B
Class 10 Micro SD Card
Keyboard + Mouse
Monitor + HDMI Cable
Power Supply (Recommended: 5V 2.5A)
Breadboard
1x Red LED
1x Blue LED
2x 330Ω Resistor
5x M/M Jumper Wire
8x M/F Jumper Wire
1x Button
1x Buzzer
1x DS18B20 Temperature Sensor
1x 4k7Ω Resistor
1x 1uF Capacitor
1x Light Dependent Resistor (LDR)

If you are connecting to your Raspberry Pi remotely using VNC or other means then Keyboard, mouse Monitor and HDMI cable are optional.

Prerequisites

You will need to install the latest version of Raspbian on to your Micro SD Card. Initial setup will require a keyboard, mouse, HDMI cable and Monitor/TV.

Wiring Diagram



Introduction

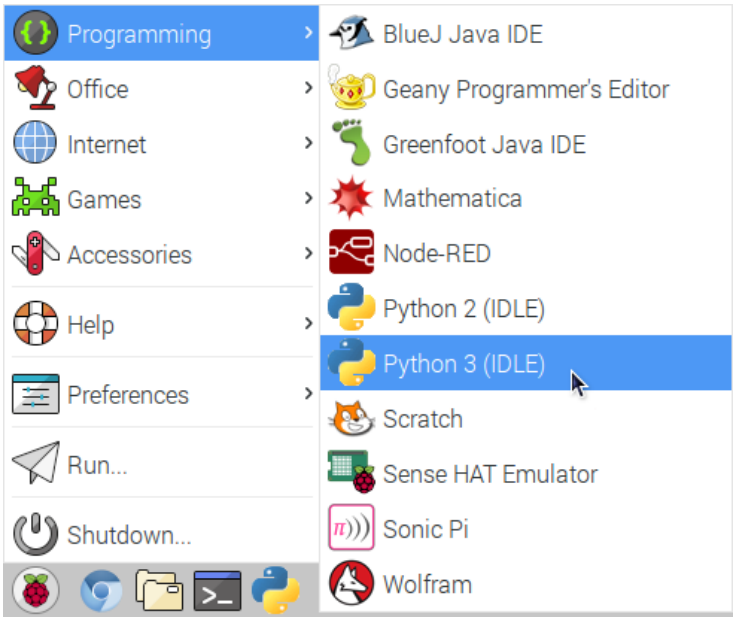
In this tutorial we will back to communicating with the GPIO directly again measuring the light levels from a Light Dependant Resistor (LDR).

These tutorials are written on a Raspberry Pi 3 Model B with a clean install of Rasbian 4.4.

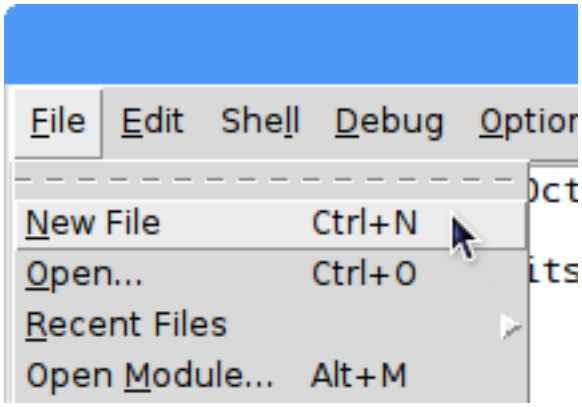
Some parts of the tutorials may look familiar as the code examples are written in a way that there is very little work needed and minor modifications to the previous code to get you up and running faster.

Getting Started

To get started, first you need to open Python 3 (IDLE). To do this click on the Raspberry Pi icon on the task bar, highlight “Programming” then click on “Python 3 (IDLE)”

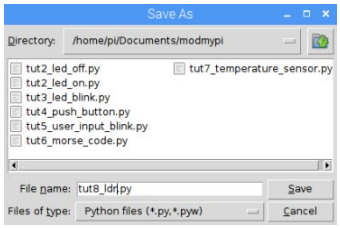


When IDLE has loaded, you will want to start working on a new file. You can do this by clicking on File and select “New File” or by pressing Ctrl+N



Save the Program

Now that we have IDLE running, first save a new file. First open up the File Manager by clicking on this icon on the taskbar and open the Documents folder.

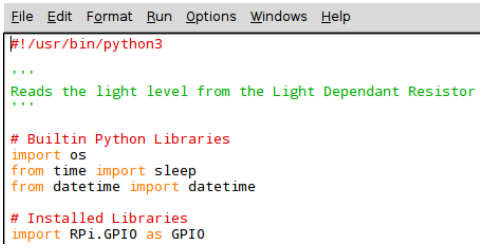


Go back to the IDLE and click on File and select “Save As”. Navigate to /home/pi/Documents/modmypi and enter tut8_ldr.py for the filename then click Save.

Writing the Code

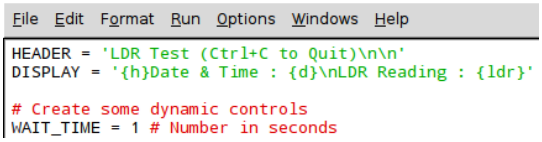
The first thing you should type is a shebang line, docstring and import your modules

```
#!/usr/bin/python3

'''
Reads the light level from the Light Dependant Resistor
'''

# Builtin Python Libraries
import os
from time import sleep
from datetime import datetime

# Installed Libraries
import RPi.GPIO as GPIO
```



Display Variables

Now you need to create some variables, 2 string variables and 1 integer variables. The string variables will be used to display text in the terminal to let us know what the program is doing. The integer variables will be used for a sleep timer.

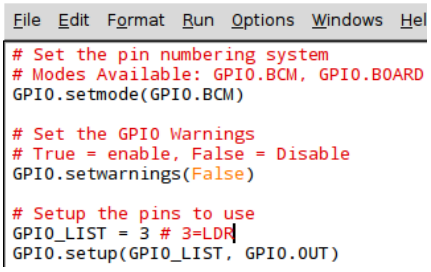



```
# Display Variables
HEADER = 'LDR Test (Ctrl+C to Quit)\n\n'
DISPLAY = '{h}Date & Time : {d}\nLDR Reading : {ldr}'

# Create some dynamic controls
WAIT_TIME = 1 # Number in seconds
```

Setting up the GPIO

Time to setup the GPIO. We will set mode and warnings and also which pins that you will be using.

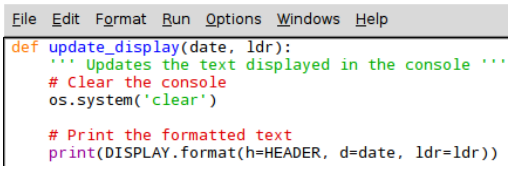
```
# Set the pin numbering system
# Modes Available: GPIO.BCM, GPIO.BOARD
GPIO.setmode(GPIO.BCM)

# Set the GPIO Warnings
# True = enable, False = Disable
GPIO.setwarnings(False)

# Setup the pins to use
GPIO_LIST = 3 # 3=LDR
GPIO.setup(GPIO_LIST, GPIO.OUT)
```



Update Display Method

Next, it is time to create a Method that will update the text that will be displayed in the console. This will be called when you want to update the information in the console.

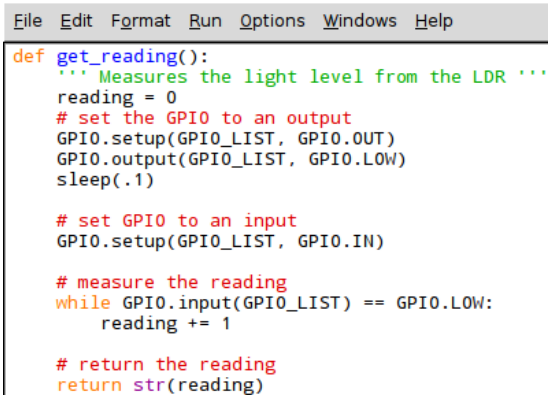
```
def update_display(date, ldr):
    ''' Updates the text displayed in the console
'''
    # Clear the console
    os.system('clear')
    # Print the formatted text to the console
    print(DISPLAY.format(h=HEADER, d=date, ldr=ldr))
```



Get Reading Method

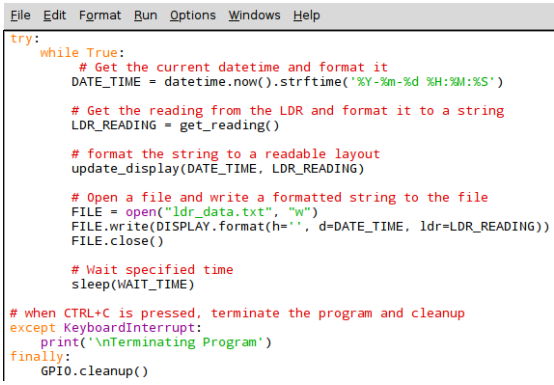
This method changes the GPIO pin 3 between input and output to charge a capacitor then take a measurement of the light level. This is done by using a loop to increase a counter till the capacitor has fully discharged. When the capacitor has discharged it will return the value.

```
def get_reading():
    ''' Measures the light level from the LDR '''
    reading = 0
    # set the GPIO to an output
    GPIO.setup(GPIO_LIST, GPIO.OUT)
    GPIO.output(GPIO_LIST, GPIO.LOW)
    sleep(.1)

    # set GPIO to an input
    GPIO.setup(GPIO_LIST, GPIO.IN)

    # measure the reading
    while GPIO.input(GPIO_LIST) == GPIO.LOW:
        reading += 1

    # return the reading
    return str(reading)
```



The Action Code – Get Reading and Save to File

This code will get the reading from the get_reading method and update the text in the console. While it is not needed, we will also save the results to a text file.

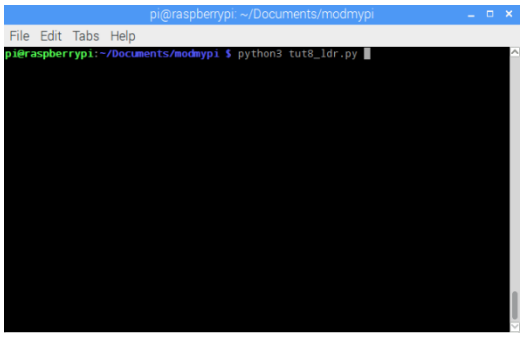



```
try:
    while True:
        # Get the current datetime and format it
        DATE_TIME = datetime.now().strftime('%Y-%m-%d %H:%M:%S')

        # Get the reading from the LDR and format it to a string
        LDR_READING = get_reading()

        # format the string to a readable layout
        update_display(DATE_TIME, LDR_READING)

        # Open a file and write a formatted string to the file
        FILE = open('ldr_data.txt', 'w')
        FILE.write(DISPLAY.format(h='', d=DATE_TIME, ldr=LDR_READING))
        FILE.close()

        # Wait specified time
        sleep(WAIT_TIME)

# when CTRL+C is pressed, terminate the program and cleanup
except KeyboardInterrupt:
    print('\nTerminating Program')
finally:
    GPIO.cleanup()
```

Make sure to save your work by clicking File and select Save, or press Ctrl+S

Running the Program

Save your work and it is time to run it so that you can make sure that it works as it should. Go back to the File Manager and open the modmypi folder you created. Next click on tools and select “Open Current Folder in Terminal” or press F4.



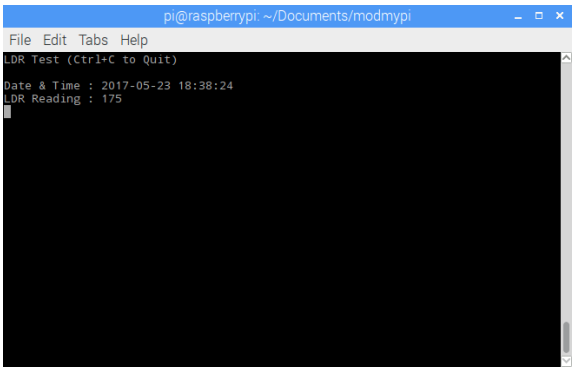
In the terminal, type
python3 tut8_ldr.py and press enter

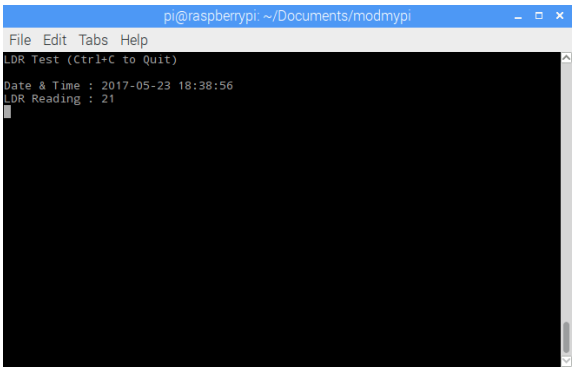
Results

If everything is working correctly you should see the console showing the light level reading. The reading will vary depending on the level of light, you can change this by covering the LDR or shining a torch directly on the LDR.

The brighter the light level the lower of the measured value, the darker the light level will increase the measured value.

In the console you should see something similar to this when running:





Code on GitHub

If you would like to download a copy of the code, you can download it from along with all the other tutorials, code and wiring diagrams from GitHub here

Thanks

Thank you for taking the time to follow this tutorial and hope that you have found this useful. Please feel free to follow the other tutorials that have been created for the ModMyPi YouTube Workshop Kit.