C. RODRIGUES	TP Internet of Things

TP 1 : Tutorial

Goals:

1. **Learn to manipulate a Raspberry Pi**
2. **Learn to assemble and use different devices with a Raspberry Pi**
3. **Implement simple applications in Python, using sensors and actuators connected to the Raspberry**

I – Tutorial: [for more details check the annex]

Before proceeding to the exercises, you MUST complete the following examples to understand how to assemble and manipulate the devices we are going to use during this course.

The Raspberry PI normally updates the date and time automatically. If not, use the following command to update manually:

```
sudo date -s "Mon Nov 29 14:00:00 2021"
```

1 – Example - LED:

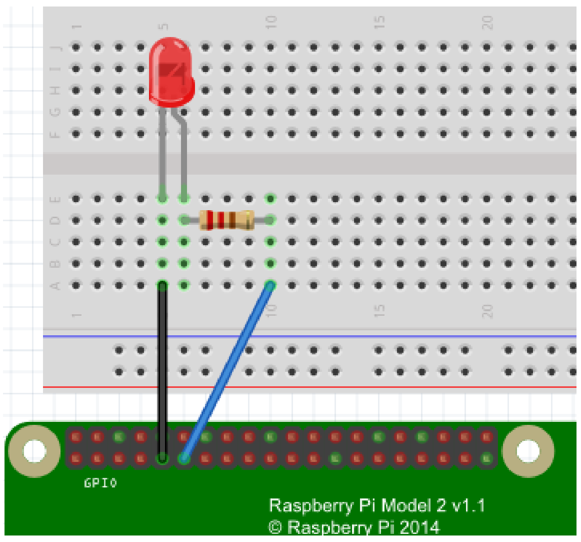
The example below must be followed as a model. It is a code sequence to turn on a LED, wait for 1 second and turn it off. Every line has comments to clarify the subsequent commands. **Attention**: the number at the beginning of each line is there only to guide you.

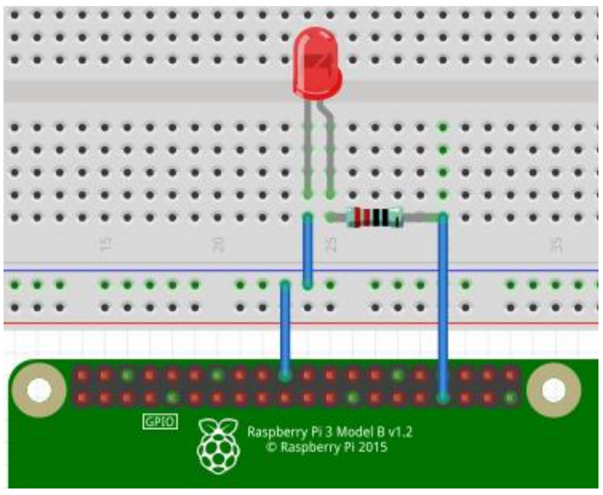
Code 01: LED

```
#! /usr/bin/env python
# enable functions and methods to manage the Pi
import RPi.GPIO as GPIO
# enable use of functions of time
import time

# GPIO used to connect an anode leg LED
# GPIO 17 is equal PIN 11 follow the physical enumeration
LED=22

#time to wait
FRESH=1

# setup the enumeration based on GPIO references 1
GPIO.setmode(GPIO.BCM)

# setup the port 17 as output, in this PIN the energy will go out
```

```
GPIO.setup(LED, GPIO.OUT)

# setup to have not energy by the port 17
GPIO.output(LED, GPIO.LOW)

# code of protection; if has no problem the block try will keep always on
try:
# infinite loop
    while True:
    # Active the transmission of energy by port 17
        GPIO.output(LED, GPIO.HIGH)
    # Wait 1 second
        time.sleep(FRESH)
    # Cut off the transmission of energy
        GPIO.output(LED, GPIO.LOW)
    # again, wait 1 second
        time.sleep(FRESH)

# exceptions are anything that interrupts the try block.
# if CTRL_C is pressed
except KeyboardInterrupt:
# setup the GPIO to default values; finish any transmission of energy
    GPIO.cleanup()
```

Two possible sketches for this code:





Attention: A red LED requires a resistor of 180Ω, but we don't have it, so we used a resistor of 220Ω or 330Ω! Since the resistor used is greater than 180Ω, there is no problem. The higher resistance will only reduce the voltage and the luminosity of the LED.

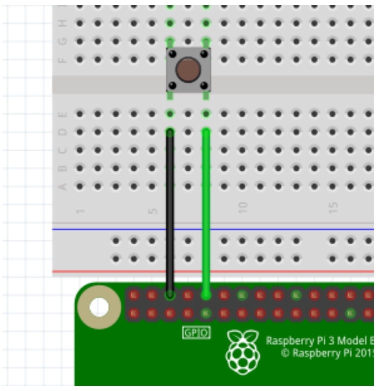
2 – Example - Button:

The next example is a model about how to use a button. In this example, the program prints the number 0 if the button is not pressed, otherwise, the value 1 is written on screen [when the button is pressed].

Code 02: Button

```
#! /usr/bin/env python

import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(27, GPIO.IN, pull_up_down=GPIO.PUD_UP)

try:
    while True:
        button_state=GPIO.input(27)
        if button_state == True:
            print("Button not pressed")
        else:
            print("Button pressed")
except KeyboardInterrupt:
    GPIO.cleanup()
```

Sketch of prototype of Code 02:

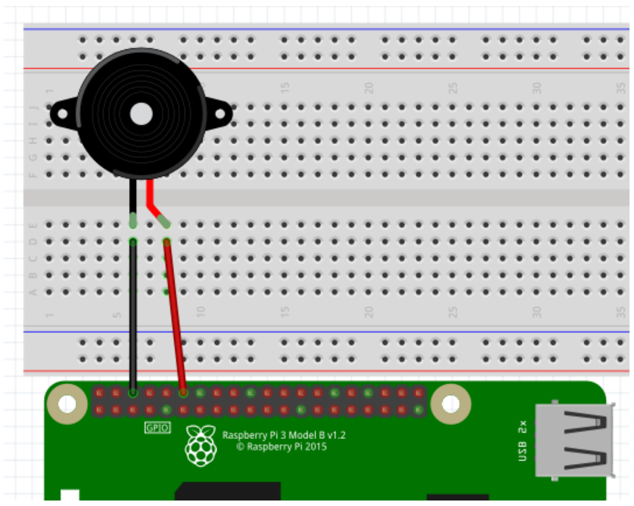


3 – Example - PassiveBuzzer:

In this example, we will use an active buzzer. We will control a GPIO to make the buzzer sound and to make it stop.

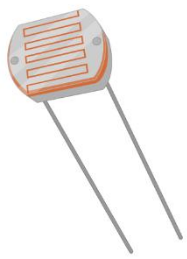
Code 3 - Buzzer

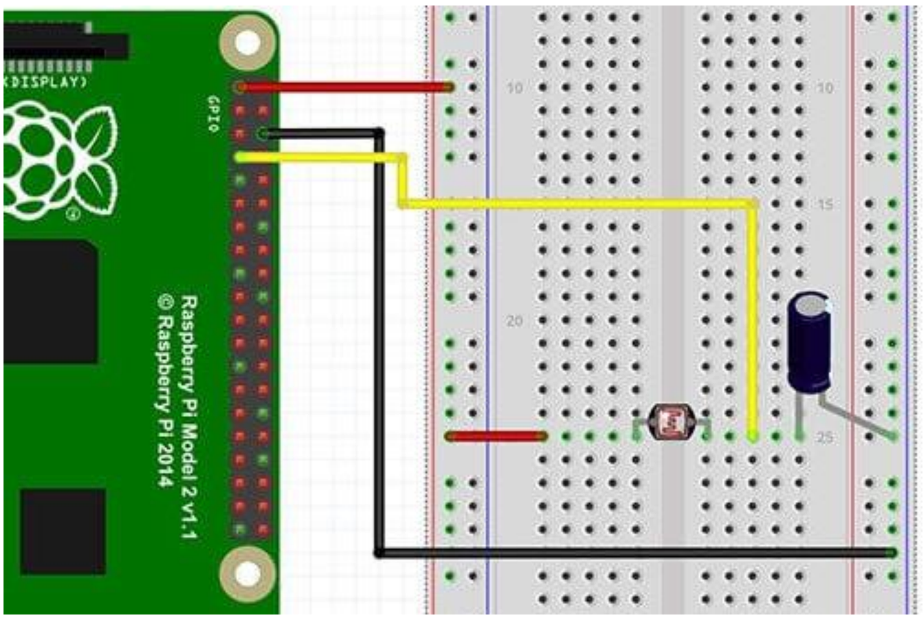
```python
#! /usr/bin/env python
import RPi.GPIO as GPIO
import time
LED=18
FRESH=2
GPIO.setmode(GPIO.BCM)
GPIO.setup(LED, GPIO.OUT)
GPIO.output(LED, GPIO.HIGH)
try:
    while True:
        GPIO.output(LED, GPIO.HIGH)
        time.sleep(FRESH)
        GPIO.output(LED, GPIO.LOW)
        time.sleep(FRESH)
except KeyboardInterrupt:
    GPIO.cleanup()
    
```



4 - Example Light-dependent Resistors (LDR):

An LDR (sometimes called a photocell) is a special type of resistor. LDR's resistance is very low when it's light, but very high when it's dark.



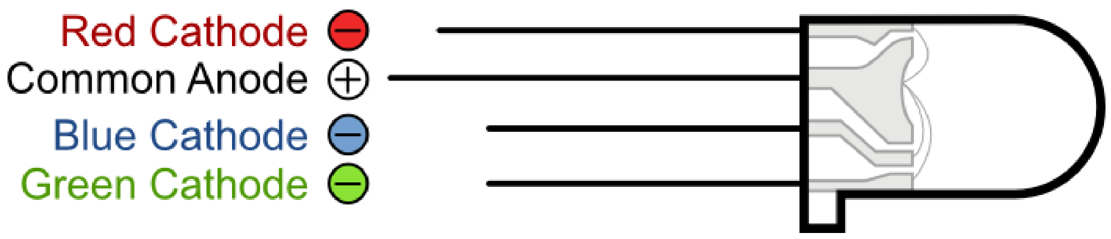


Code 04: LDR

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
turn_off_GPIO_time = 0.1
value = 0 # this variable will be used to store the ldr value
ldr = 4

def ldr_read (ldr):
    count = 0

    #Output on the pin to turn it off before reading it again
    GPIO.setup(ldr, GPIO.OUT)
    GPIO.output(ldr, False)
    time.sleep(turn_off_GPIO_time)

    #Change the pin back to input
    GPIO.setup(ldr, GPIO.IN)

    #Count until the pin goes high
    while (GPIO.input(ldr) == 0):
        count += 1

    return count


try:
    while True:
        print("LDR Value:")
        value = ldr_read(ldr)
        print(value)
        if ( value <= 4000 ):
                print("Lights are ON")
        else:
                print("Lights are OFF")
except KeyboardInterrupt:
    GPIO.cleanup()
```

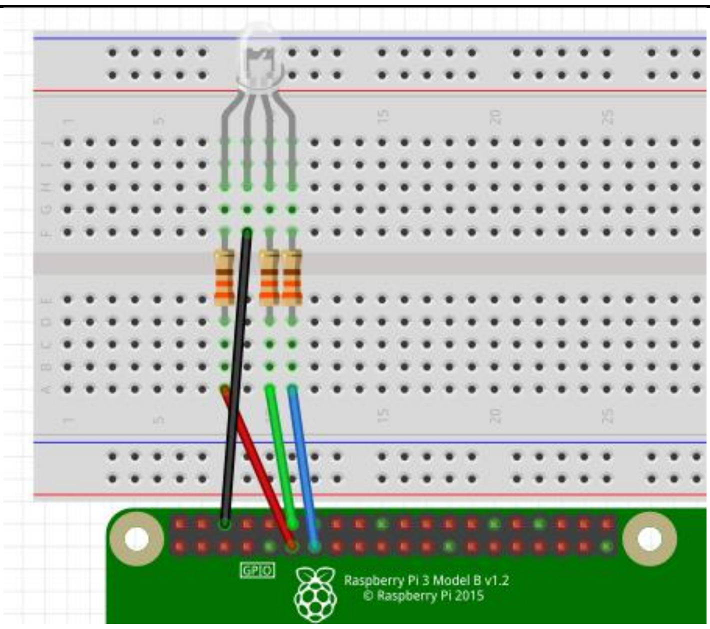
5 - Example - LED RGB:

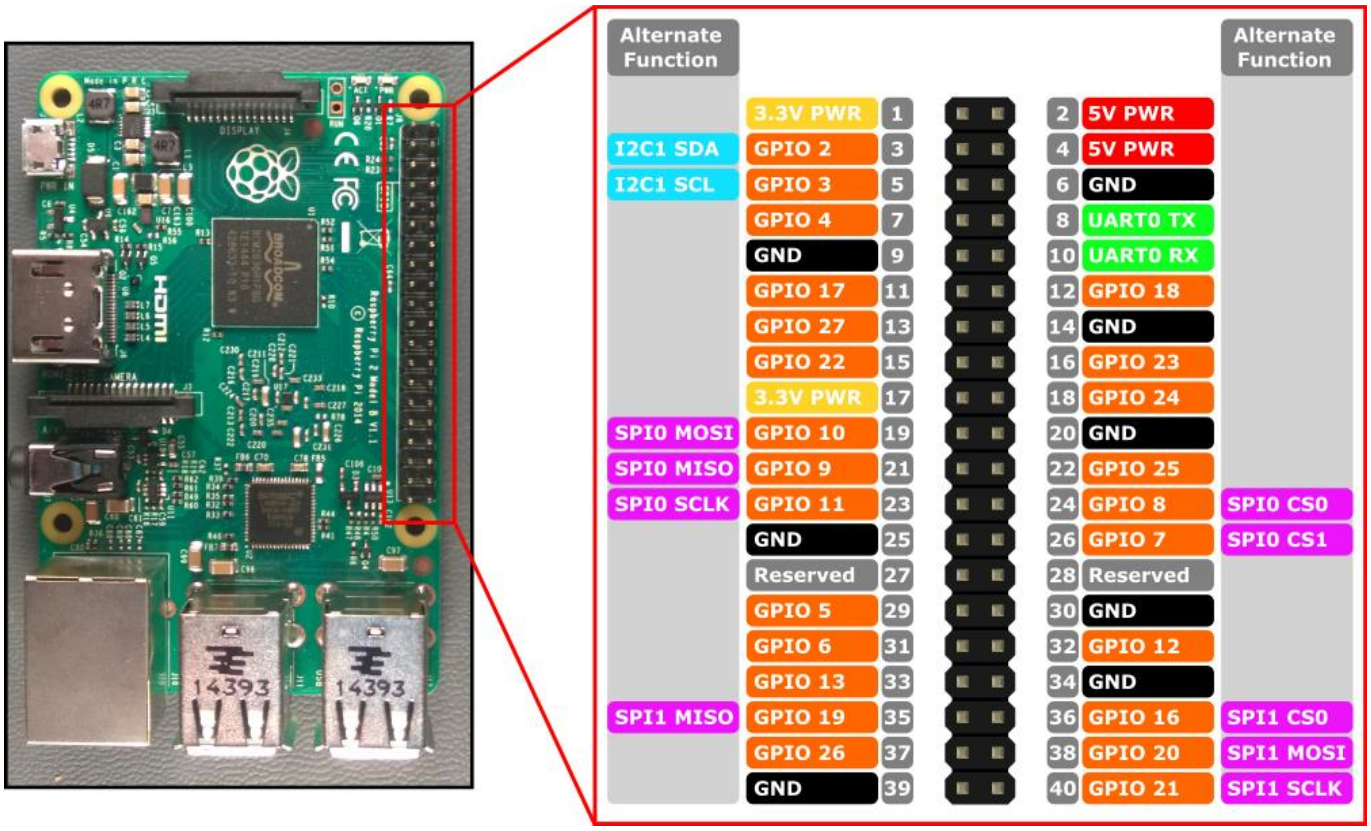
RGB stands for the red, green, and blue color channels and it is an industry color standard. RGB displays colors by changing the three channels and superimposing them. Each of the three-color channels of red, green, and blue has 255 stages of brightness. When the three primary colors are set to 0, light will be off.



Code 05: LED RGB

```python
#! /usr/bin/env python
import RPi.GPIO as gpio
import time
import random
RED=17
GREEN=187
BLUE=27
legs = (RED, GREEN, BLUE)
gpio.setmode(gpio.BCM)
gpio.setup(RED,gpio.OUT)
gpio.setup(GREEN,gpio.OUT)
gpio.setup(BLUE, gpio.OUT)
gpio.output(RED, gpio.LOW)
gpio.output(GREEN, gpio.LOW)
gpio.output(BLUE, gpio.LOW)
try:
    while True:
        x = int(random.random()*2)
        gpio.output(legs[x], gpio.HIGH)
        time.sleep(0.05)
        x = int(random.random()*2)
        gpio.output(legs[x], gpio.LOW)
except KeyboardInterrupt:
    gpio.cleanup()
```





Alternate Function					Alternate Function
	3.3V PWR	1	2	5V PWR	
I2C1 SDA	GPIO 2	3	4	5V PWR	
I2C1 SCL	GPIO 3	5	6	GND	
	GPIO 4	7	8	UART0 TX	
	GND	9	10	UART0 RX	
	GPIO 17	11	12	GPIO 18	
	GPIO 27	13	14	GND	
	GPIO 22	15	16	GPIO 23	
	3.3V PWR	17	18	GPIO 24	
SPI0 MOSI	GPIO 10	19	20	GND	
SPI0 MISO	GPIO 9	21	22	GPIO 25	
SPI0 SCLK	GPIO 11	23	24	GPIO 8	SPI0 CS0
	GND	25	26	GPIO 7	SPI0 CS1
	Reserved	27	28	Reserved	
	GPIO 5	29	30	GND	
	GPIO 6	31	32	GPIO 12	
	GPIO 13	33	34	GND	
SPI1 MISO	GPIO 19	35	36	GPIO 16	SPI1 CS0
	GPIO 26	37	38	GPIO 20	SPI1 MOSI
	GND	39	40	GPIO 21	SPI1 SCLK

For more information about Raspberry Pi GPIO ports, check:

https://pinout.xyz