Lab1: Wifi-Controlled LED Stoplight

cj.cornel

September 2018

Contents

0.1 Online Link 2

  0.1.1 Blog 2

  0.1.2 Github 2

0.2 Objective 2

0.3 Materials 2

0.4 Procedure 2

0.5 Thought Questions 4

0.6 Sources 5

0.7 Appendix A 5

0.1 Online Link

0.1.1 Blog

`http://projectmyaccess.blogspot.com/2018/09/lab-1-led-stoplight.html`

0.1.2 Github

`https://github.com/Sh0rtyTheCircuit/TrafficLights`

0.2 Objective

1. Code and create a circuit to turn on and off traffic light colored LEDs (green, yellow, and red)
2. All LEDs must be able to turn on, off, and cycle based on corresponding buttons presented on a website

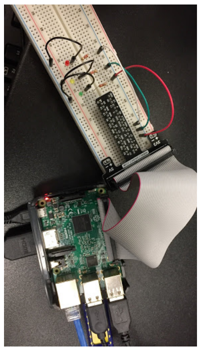
0.3 Materials

- 1 Raspberry Pi
- 3 LEDs (Green, Yellow, and Red)
- 3 220 Ω Resistors
- 6 male to male wires
- 1 Pi T-cobbler
- 1 ribbon cable
- 1 breadboard

0.4 Procedure

Please remember to constantly save and check your code!

- Set up the circuit according to the following image:

  I connected the GPIO pins to the resistors and then the resistors to the LEDS. The LEDs will then connect to Ground

- Open up the terminal and create a python file Don't forget to download any necessary libraries! Libraries I used can be found in my code in Appendix A

```
nano TrafficLight.py
```

- Create a function to turn off all LEDs - I called mine "OFF"

```
def Off():
    GPIO.output(4,GPIO.LOW)
    GPIO.output(17,GPIO.LOW)
    GPIO.output(22,GPIO.LOW)
```

- Create a function to turn on the green LED and turn off the rest of the LEDs (You can call the "OFF" function to do this for you)

```
 def GREEN():
    OFF()
    GPIO.output(4,GPIO.HIGH)
```

- Do the same for the rest of the LEDs
- Create a function that will cycle through the LEDs. One LED will turn on for 3 seconds and then the next will turn on. Red should turn on for 3 seconds and then turn off and Green will turn on for 3 seconds and then turn off with Yellow doing the same. This process should repeat until another button is pressed.

```
def CYCLE (client,stopper):
    while not stopper.is_set():
        OFF()
        GPIO.output(4,GPIO.HIGH)
        time.sleep(3)
        OFF()
        GPIO.output(17,GPIO.HIGH)
        time.sleep(3)
        OFF()
        GPIO.output(17,GPIO.HIGH)
        time.sleep(3)
```

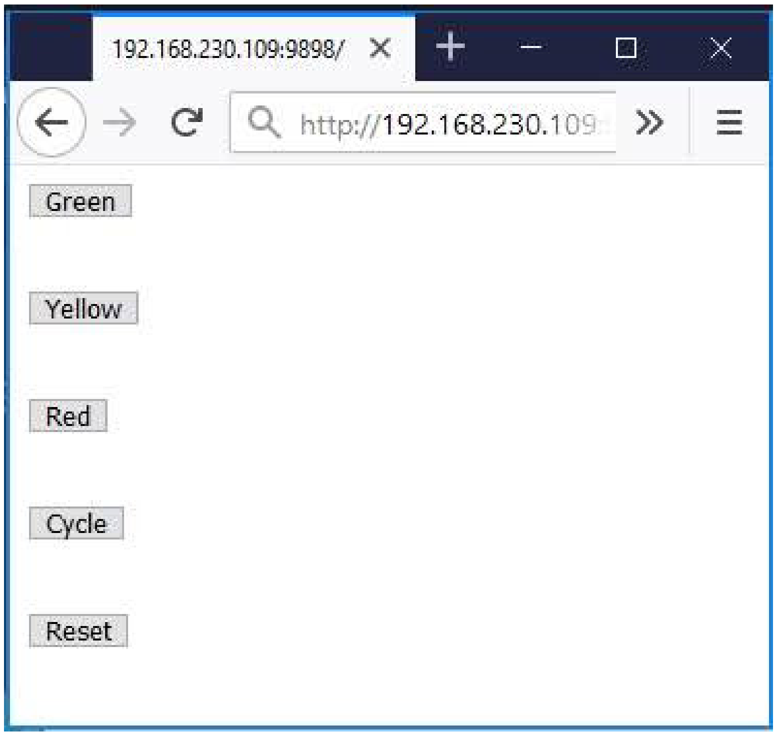
- Create a webserver that integrates the functions. Each function should correspond to a button on the webserver (see Appendix A for code)
- Exit and compile using the following commands

```
ctrl + x
python TrafficLight.py
```

- Now open up your browser and type in your IP address with the port number picked

```
192.168.230.109:9898
```

- You should now see your buttons and be able to use them!



0.5 Thought Questions

- What Language did you choose for implementing this project? Why?

  Python! It's the programming language I know and understand the most.

- What is the purpose of the resistor in this simple circuit? What would happen if you omitted it?

  The resistor essentially limits the amount of power flowing through the circuit. If the resistor was not there, there would be too much power going through the LEDs, creating a short circuit.

- What are practical applications of this device? What enhancements or modifications would you make?

  Applying the same or similar programming used can help with home light automation, club light automation, as well as applying to different objects/systems such as turning on and off a dryer, washer machine, or dishwasher. There are multiple possible applications. Enhancements or modifications I would make are for the lights to strobe, to increase and decrease in lumines, to make the lights flash when something is finished or make a sound when finished, and to add more lights.

- Please estimate the total time you spent on this lab and report.

  about 10.5 hours

0.6 Sources

- https://www.instructables.com/id/Raspberry-Pi-Home-Automation-Control-lights-comput/
- https://thepihut.com/blogs/raspberry-pi-tutorials/27968772-turning-on-an-led-with-your-raspberry-p
- https://ruslanspivak.com/lsbaws-part1/
- https://stackoverflow.com/questions/16036041/can-a-html-button-perform-a-post-request
- https://docs.python.org/2/library/socket.html
- overleaf.com
- https://www.sharelatex.com/learn/latex/

0.7 Appendix A

```
### WEBSERVER SETUP ###
import socket
import select
import sys
import threading

### LED SETUP ###
import RPi.GPIO as GPIO                 #RPi for Raspberry Pi
import time
GPIO.setmode(GPIO.BCM)


GPIO.setwarnings(False)                 #Turn off warnings
GPIO.setup(4,GPIO.OUT)                  #GREEN
GPIO.setup(17,GPIO.OUT)                 #YELLOW
GPIO.setup(22,GPIO.OUT)                 #RED


def OFF():                              #Turn all lights off

        GPIO.output(4,GPIO.LOW)
        GPIO.output(17,GPIO.LOW)
        GPIO.output(22,GPIO.LOW)

def GREEN():                            #Only Green is on
        OFF()
        GPIO.output(4,GPIO.HIGH)

def YELLOW():                           #Only Yellow is on
        OFF()
        GPIO.output(17,GPIO.HIGH)

def RED():                              #Only Red is on
        OFF()
        GPIO.output(22,GPIO.HIGH)

def CYCLE(client,stopper):              #Green to Yellow to Red Repeat
```

```
        while not stopper.is_set():     #Listen for stopper.
                                        #As long as stopper is not set,
                                        #will continue loop
                OFF()
                GPIO.output(4,GPIO.HIGH)
                time.sleep(3)
                OFF()
                GPIO.output(17,GPIO.HIGH)
                time.sleep(3)
                OFF()
                GPIO.output(22,GPIO.HIGH)
                time.sleep(3)
                #return 0

## CREATE WEBSERVER ##

HOST, PORT = '', 9898

listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)    #socket setup
listener.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
listener.bind((HOST,PORT))              #Where the server is now located
listener.listen(1)                      #Listen for this many clients

print 'Listener is set up'
stopper = threading.Event()             #Handler for starting and stopping the thread

while True:
        client_connection, client_address = listener.accept()
        #Accept connection from user
        '''TIMEOUT
        client_connection.setblocking(0)    #If does not receive anything from client
                                            #, will continue code
        request = "post /clear  "                               #default request
        ready = select.select([client_connection],[],[],0)
        #Looking every One second for a different client request
        if ready[0]:'''                     #If client sends new request
        request = client_connection.recv(1024)
        #Waiting for GET REQUEST(refresh/load website).
        #Request is new request now.
        print request
        #Request = whatever is Posted when btns are pressed
        check_status = request[0:13]
        print check_status
        if check_status.find("green")>0:
        #Look for "Green" in request.
        #If not found, will return -1 which breaks the code
                stopper.set()
                GREEN()
        if check_status.find("yellow")>0:
                stopper.set()
                YELLOW()
        if check_status.find("red")>0:
                stopper.set()
                RED()
        if check_status.find("cycle")>0:
                stopper.clear()
                cycle_handler =
                threading.Thread
                (target=CYCLE,args=(client_connection,stopper))
                cycle_handler.start()
        if check_status.find("reset")>0:
                stopper.set()
                OFF()
        disp_body = """\
<html>
        <title> Choose Wisely </title>
        <body>

                <form action="http://localhost:9898/green" method= "post">
                        <button>Green</button>
                </form>
        <br>
                <form action="http://localhost:9898/yellow" method= "post">
                        <button> Yellow </button>
                </form>
        <br>
                <form action="http://localhost:9898/red" method= "post">
                        <button> Red </button>
                </form>
        <br>
                <form action="http://localhost:9898/cycle" method= "post">
                        <button> Cycle </button>
                </form>
        <br>
                <form action="http://localhost:9898/reset" method= "post">
                        <button> Reset </button>
                </form>
        </body>

</html>
        """
        display = """\
HTTP/1.1 302 OK
Content-Type: text/html
Content-Length: %d
Connection: close
""" % len(disp_body)

        client_connection.sendall(display + disp_body)
        client_connection.close()
```