# Lab 2 - Stoplight on D1 mini with a Webserver in Arduino

#### Online Link

The details and code to this lab can be found at: <https://github.com/Rhemsley/Arduino-Stoplight>

#### Objective

The purpose of this lab is to learn how to use the D1 mini, its WiFi chip, and Arduino to create a web server controlled Stoplight. This teaches principles like the following:

* Create circuits using a resistor, LED, and D1 mini
* Connect the D1 mini to WiFi with the ESP8266WiFi.h library
* Create a basic Webpage using Arduino and the ESP8266WebServer.h library
* Accessing the Web Server on a browser to control each of the circuits and create a Looping effect like a Stoplight.

#### Materials

I used the following materials to complete this lab:

D1 mini

4 x Male to Male Jumper Cables

1 x Red LED

1 x Yellow LED

1 x Green LED

3 x 100 Ω 5% Resistor (BrBlBrGd)

1 x Breadboard

#### References

I used the following resources to complete this lab:

<https://www.wemos.cc/en/latest/d1/d1_mini.html> D1 mini documentation and pin out

<https://microdigisoft.com/how-to-create-html-web-page-with-wemos-d1-mini/> Majority of my learning about D1 mini Webserver hosting and WiFi connection

<https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266WebServer> Learning how to add additional handlers

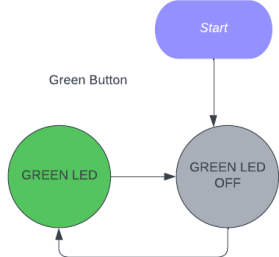
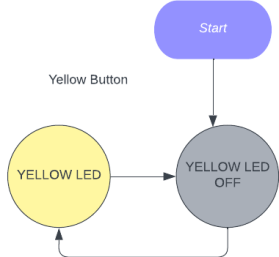
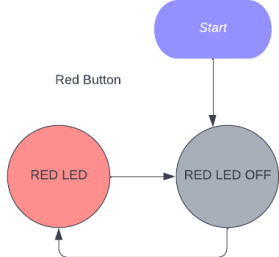
<https://www.instructables.com/Control-Led-Via-Browser-Nodemcu-Mini-D1-R1-HTML-CS/> Reference on LED control ideas and methods

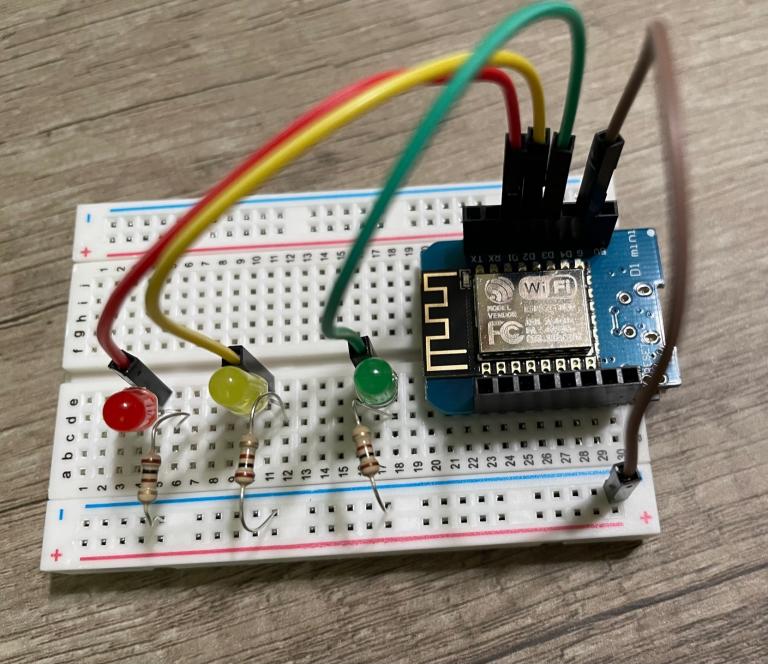
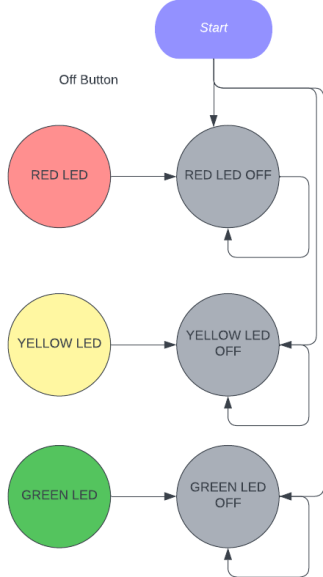
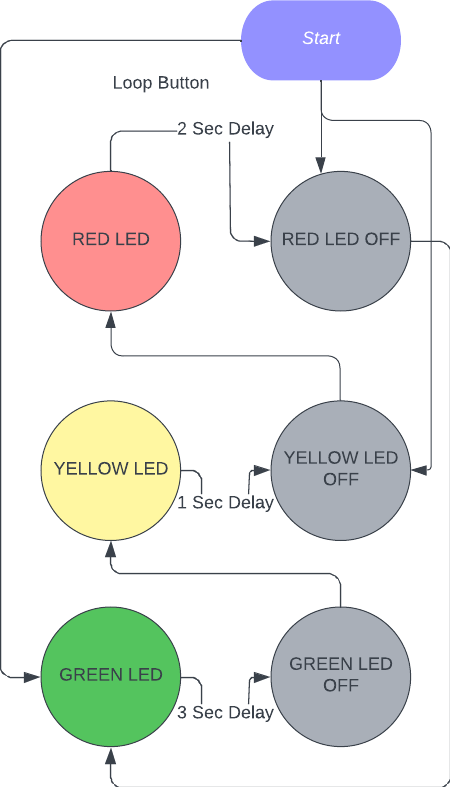
<https://www.programmingelectronics.com/arduino-sketch-with-millis-instead-of-delay/> How to use millis()

<https://linuxhint.com/make-button-link-another-page-html/#:~:text=In%20HTML%2C%20a%20button%20link,a%20link%20on%20the%20button.> HTML Button adjustment from previous build

#### Baseline Information

Below are the Flow charts for the five different buttons that control the LEDs. As seen each of the individual LED buttons cycle the LED between on and off. The Off button turns off all three of the LEDs. Then the Loop button loops the LEDs from Green to Yellow to Red with identified delays between each one.

To the right we have the view of the build with each of the LEDs connected in a circuit with a resistor. As seen, the Red LED goes to pin D1, the Yellow LED to pin D2, the Green LED to ping D3, and the Ground wire going to the Ground pin.

#### Procedures

1. Plug D1 mini chip into your computer with micro usb.
   1. Doing so will allow for upload of code and monitors. Once uploaded, the D1 mini could be powered by another mean and still work so long as it is on the same WiFi and uses the same IP address.
   2. Additional note: a D1 mini does not need to be used to recreate this, but using a different board will require some different code to connect to wifi and get the server up and running. Only slight adjustments would need to be made though.
2. Install and setup Arduino to work with D1 mini.
   1. Install the Arduino IDE as explained here. <https://www.arduino.cc/en/software>
   2. Once installed, go to Tools > Board > Board Manager and make sure the latest version of esp8266 is installed.
   3. Now go to Tools > Board > ESP8266 Boards > LOLIN(WEMOS) D1 R2 & mini to set the Arduino IDE to use the right board.
   4. Next go to Tools > Port > and then select the Port your D1 mini is plugged into. If nothing shows up, the Port may need new drivers which can be done in the Device Manager with finding the Unknown Device or USB Serial Port in use and updating its drivers. For me, my device was using Port COM3 or COM 4.
3. Modify code to match your Pin and WiFi setup.
   1. Now that the Arduino IDE is all setup, we can add the code into the IDE. Write or copy the code in.
   2. First at the top under the “// Needed variables for LED Pin association” section, adjust the Pins being used to match your setup if not using the same pins.
   3. Then go down to the “// wifi and server config” section and set the ssid variable to the name of your WiFi network. Then do the same for the password variable for your WiFi network password.
4. Verify and then Upload code.
   1. Now that the code is all setup to match your setup, we can hit the Checkmark at the top left to “Verify” the code. If any issues arise, make sure the above steps were followed, the #include librarys are still there, and that the above changes didn’t mess up any quotation marks or semi colons. Otherwise google the issue.
   2. Once verified, we can upload the code to the D1 mini board. This is done with the arrow button right to the right of the checkmark to “Upload” the code. This will then send that code through the USB cord to the D1 mini and then reboot the device.
5. Watch Serial Monitor, make sure it connects to WiFi, and get IP address
   1. Once the Upload button has been pressed it will be worth while to open the Serial Monitor and watch what is taking place on the device. This can be done by clicking the Magnifying glass button at the top right which opens the “Serial Monitor.”
   2. Once in the Serial Monitor, first make sure that the device connects to your WiFi. You know it has connected when the periods stop coming in and it says something like: Connected to [WIFI NAME] IP address: [IP ADDRESS] HTTP server started. If you do not see this, try rebooting the device with the Reset button or making sure that the WiFi Name and Password are in there correctly. Also, it may have issues connecting to networks that do not support 2.4Ghz devices.
6. Go to the IP address in browser.
   1. Now that the device is on the WiFi network, on a device that is also on that same network, open an internet browser and go to the IP address it gave you in the Serial Monitor. This should bring you to the basic HTML webpage the device is hosting.
7. Control LEDs as desired.
   1. The LEDs can now be controlled by one of two means.
   2. First they can simply be controlled by hitting the Buttons on the HTML page. These buttons will either alternate the color clicked, start the stoplight loop effect, or turn off all of the lights.
   3. The other way to control these LEDS is to simply send Get/Post requests to the webserver IP as outlined below. The [IP] indicates your devices IP address.
      1. [IP]/Red - Alternates the RED LED between Off and On
      2. [IP]/Yellow - Alternates the Yellow LED between Off and On
      3. [IP]/Green - Alternates the Green LED between Off and On
      4. [IP]/Off - Turns off all of the LEDs and stops the loop if looping
      5. [IP]/Loop - Turns off all the LEDs and then starts the loop cycle

#### Observations

I loved the thought adjustment that went into getting the Stoplight to work within Arduino and on a little D1 mini chip. To have first done this on the Raspberry Pi allowed for a strong foundation of what is going on in the circuitry and how to work with the GPIO pins.

Some of the difficulties of using the Raspberry Pi were avoided in this lab while some other difficulties were added. For example, the main difficulty for me of using the Raspberry Pi was getting the Apache Webserver to run the python code I had built to operate the LEDs as desired. I ended up having to rewrite that python code into PHP code and then I finally got it working. For this lab I was able to stay within the same code the entire time and didn’t have to deal with any issues of trying to get the Webserver to run needed code. The main two issues that replaced that Raspberry Pi issues were as follows. First, I had to learn how in the word to get the D1 mini to run a Webserver and connect to the WiFi. This took quite a bit of exploring but eventually I got it rolling, especially thanks to some of the references above. Second, I had to figure out how to get the LEDs, and in particular the looping aspect, to work with the structure of the Aduino since it does that continuous loop command. Luckily I was pretty quickly able to grasp the millis() function and how to utilize it to control the desired LEDs to loop. Combining everything added a tiny bit more over all but it really wasn’t too tough.

Last little note: I love that these can be controlled so easily remotely and can be powered so simply. For example, once I had everything working, I plugged my board/setup into a battery pack and then brought the thing around the house and to my WiFi and then controlled the lights from my phone like it was magic. It made it all the more real and applicable to me to be able to do this.

#### Thought Questions

1. What are some key differences between developing this lab on a Raspberry Pi, and developing on Arduino?

The biggest to me was absolutely first the coding languages used. This one was all in C and that was actually a good thing in lots of ways since we have taken several classes on coding in C. The other thing like mentioned earlier was figuring out how to get the stoplight loop to go in a contagiously running function.

1. What are the strengths and trade-offs of each of these platforms?

I think a big strength of the Raspberry Pi was its additional processing power and all of the potential avenues of running a server and messing with it. In some ways this was daunting but overall meant quite a bit of flexibility. The Arduino ended up being a bit easy though because of the lack of potential solutions even with its lesser capabilities. It also did potentially a better job at just handling the one task it is give and made for some real easy mobility with its simplicity and just need for power and not needing to control it like the Raspberry Pi desires more of.

3. How familiar were you with the Arduino platform prior to this lab?

Virtually none at all. I had heard the name and I knew a decent amount about coding in c but that is the extent. I had never used the Arduino platform.

4. What was the biggest challenge you overcame in this lab?

I would say the hardest part to me was figuring out how to get the webserver up and running and then how to get it to work correctly with my stoplight loop method I had came up with. Enough exploring, testing, time and then code modification lead to a decent solution though.

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

I would estimate I spent about 4 hours coding/building the project and then 3 hours creating the lab writeup.

#### Certification of Work

I certify that the results and solution to this lab were my own work. For the resources of information I found through exploring the internet, I referenced the website and what I pulled from it. All code written was of my own writing.

-Rylan Hemsley

#### Appendix

Code for reference:

#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

// Needed variables for LED Pin association

unsigned char greenLed = D3; // Create greenLed Label for pin 3

unsigned char yellowLed = D2; // Create yellowLed Label for pin 2

unsigned char redLed = D1; // Create redLed Label for pin 1

// Needed variables for looping

unsigned long previousTime = 0; // Create previousTime int for looping

unsigned long startTime = 0; // Create startTime int for looping

bool runStoplight = false; // Create runStoplight bool for looping

bool redStatus = false; // Create Red bool for cycling red

bool yellowStatus = false; // Create Yellow bool for cycling yellow

bool greenStatus = false; // Create Green bool for cycling green

// wifi and server config

const char\* ssid = "RYLANSLAPTOP"; // Set Wifi Name

const char\* password = "Onyourleft100"; // Set Password

ESP8266WebServer server(80); // Set server to port 80

// Simple HTML saved in a string with the different page references in buttons

const char MAIN\_page[] PROGMEM = R"=====(

<html>

<head>

<title>Home Webpage</title>

<h1>Home Webpage</h1>

<h2>Stoplight</h2>

</head>

<body>

<a href="/Off"><button style="background-color:grey">Off</button></a>

<a href="/Red"><button style="background-color:red">Red</button></a>

<a href="/Yellow"><button style="background-color:yellow">Yellow</button></a>

<a href="/Green"><button style="background-color:lightGreen">Green</button></a>

<a href="/Loop"><button style="background-color:lightGrey">Loop</button></a>

</body>

</html>

)=====";

// Handle default page call

void handleRoot() {

String page = MAIN\_page; // Store HTML in a string

server.send(200, "text/html", page); // Send HTML to user

}

// Handle Off call and reset needed variables and LEDs

void handleOff() {

String page = MAIN\_page; // Store HTML in a string

server.send(200, "text/html", page); // Send HTML to user

digitalWrite(greenLed, LOW); // Turn Green LED off

digitalWrite(yellowLed, LOW); // Turn Yellow LED off

digitalWrite(redLed, LOW); // Turn Red LED off

greenStatus = false; // Indicate green is off

yellowStatus = false; // Indicate yellow is off

redStatus = false; // Indicate red is off

runStoplight = false; // Stop stoplight looping

previousTime = 0; // Reset previousTime

startTime = 0; // Reset startTime

Serial.println("Turning Off LEDs");

}

// Handle Red call and cycle red LED depending on status

void handleRed() {

String page = MAIN\_page; // Store HTML in a string

server.send(200, "text/html", page); // Send HTML to user

// If off turn on and if on turn off

if (redStatus == false) {

digitalWrite(redLed, HIGH); // Turn Red LED on

redStatus = true; // Flip status

}

else {

digitalWrite(redLed, LOW); // Turn Red LED off

redStatus = false; // Flip status

}

Serial.println("Cycling Red");

}

// Handle Yellow call and cycle yellow LED depending on status

void handleYellow() {

String page = MAIN\_page; // Store HTML in a string

server.send(200, "text/html", page); // Send HTML to user

// If off turn on and if on turn off

if (yellowStatus == false) {

digitalWrite(yellowLed, HIGH); // Turn Yellow LED on

yellowStatus = true; // Flip status

}

else {

digitalWrite(yellowLed, LOW); // Turn Yellow LED off

yellowStatus = false; // Flip status

}

Serial.println("Cycling Yellow");

}

// Handle Green call and cycle green LED depending on status

void handleGreen() {

String page = MAIN\_page; // Store HTML in a string

server.send(200, "text/html", page); // Send HTML to user

// If off turn on and if on turn off

if (greenStatus == false) {

digitalWrite(greenLed, HIGH); // Turn Green LED on

greenStatus = true; // Flip status

}

else {

digitalWrite(greenLed, LOW); // Turn Green LED off

greenStatus = false; // Flip status

}

Serial.println("Cycling Green");

}

// Handle Loop call and setup needed variables

void handleLoop() {

String page = MAIN\_page; // Store HTML in a string

server.send(200, "text/html", page); // Send HTML to user

digitalWrite(greenLed, LOW); // Turn Green LED off

digitalWrite(yellowLed, LOW); // Turn Yellow LED off

digitalWrite(redLed, LOW); // Turn Red LED off

greenStatus = false; // Indicate green is off

yellowStatus = false; // Indicate yellow is off

redStatus = false; // Indicate red is off

runStoplight = true; // Start stoplight looping

previousTime = millis(); // Set previousTime to current runtime

startTime = millis(); // Then set startTime to current runtime

Serial.println("Starting Stoplight Loop");

}

void setup() {

// Setup Serial Monitor with 9600

Serial.begin(9600);

// Create Wifi connection and delay until connected

WiFi.begin(ssid, password);

Serial.println("Connecting to WiFi");

while(WiFi.status() != WL\_CONNECTED) { // Loop until connected to WiFi

delay(500);

Serial.print(".");

}

// If connection successful show IP address in serial monitor

Serial.println("");

Serial.print("Connected to ");

Serial.println(ssid);

Serial.print("IP address: ");

Serial.println(WiFi.localIP()); // Show your assigned IP address

// Setup all 3 LED pins and make sure they are off

pinMode(greenLed, OUTPUT); // Set Green LED to Output pin

pinMode(yellowLed, OUTPUT); // Set Yellow LED to Output pin

pinMode(redLed, OUTPUT); // Set Red LED to Output pin

digitalWrite(greenLed, LOW); // Turn Green LED off

digitalWrite(yellowLed, LOW); // Turn Yellow LED off

digitalWrite(redLed, LOW); // Turn Red LED off

// Setup the needed Handles for the client operations

server.on("/", handleRoot);

server.on("/Off", handleOff);

server.on("/Red", handleRed);

server.on("/Yellow", handleYellow);

server.on("/Green", handleGreen);

server.on("/Loop", handleLoop);

// Start server

server.begin();

Serial.println("HTTP server started");

}

void loop() {

// Run server client requests

server.handleClient();

// Stoplight Controlling only runs when set to true

if (runStoplight == true) {

// compare last run to starting time, turn to seconds, then

// check if within first second of three second loop

if ((((previousTime - startTime) / 1000)%3) == 0) {

digitalWrite(greenLed, HIGH); // Turn Green LED on

digitalWrite(yellowLed, LOW); // Turn Yellow LED off

digitalWrite(redLed, LOW); // Turn Red LED off

Serial.println("Loop Green ON");

previousTime = millis(); // Set previous time to current time

}

// compare last run to starting time, turn to seconds, then

// check if within second second of three second loop

else if ((((previousTime - startTime) / 1000)%3) == 1) {

digitalWrite(greenLed, LOW); // Turn Green LED off

digitalWrite(yellowLed, HIGH); // Turn Yellow LED on

digitalWrite(redLed, LOW); // Turn Red LED off

Serial.println("Loop Yellow ON");

previousTime = millis(); // Set previous time to current time

}

// compare last run to starting time, turn to seconds, then

// check if within third second of three second loop

else if ((((previousTime - startTime) / 1000)%3) == 2) {

digitalWrite(greenLed, LOW); // Turn Green LED off

digitalWrite(yellowLed, LOW); // Turn Yellow LED off

digitalWrite(redLed, HIGH); // Turn Red LED on

Serial.println("Loop Red ON");

previousTime = millis(); // Set previous time to current time

}

}

}