# 6. Network operation

Difficult Level: 

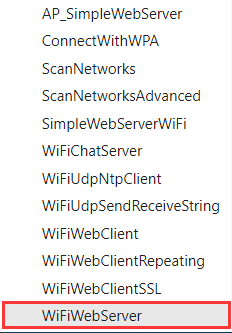
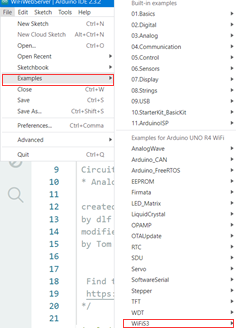
## A. Connect the network

The Arduino UNO R4 Wi-Fi has a built in ESP32-S3 module that enables you to connect to Wi-Fi® networks, and perform network operations. Protocols including HTTPS, MQTT, and UDP are tested and supported, and in this article, you will get an example that will get you started.

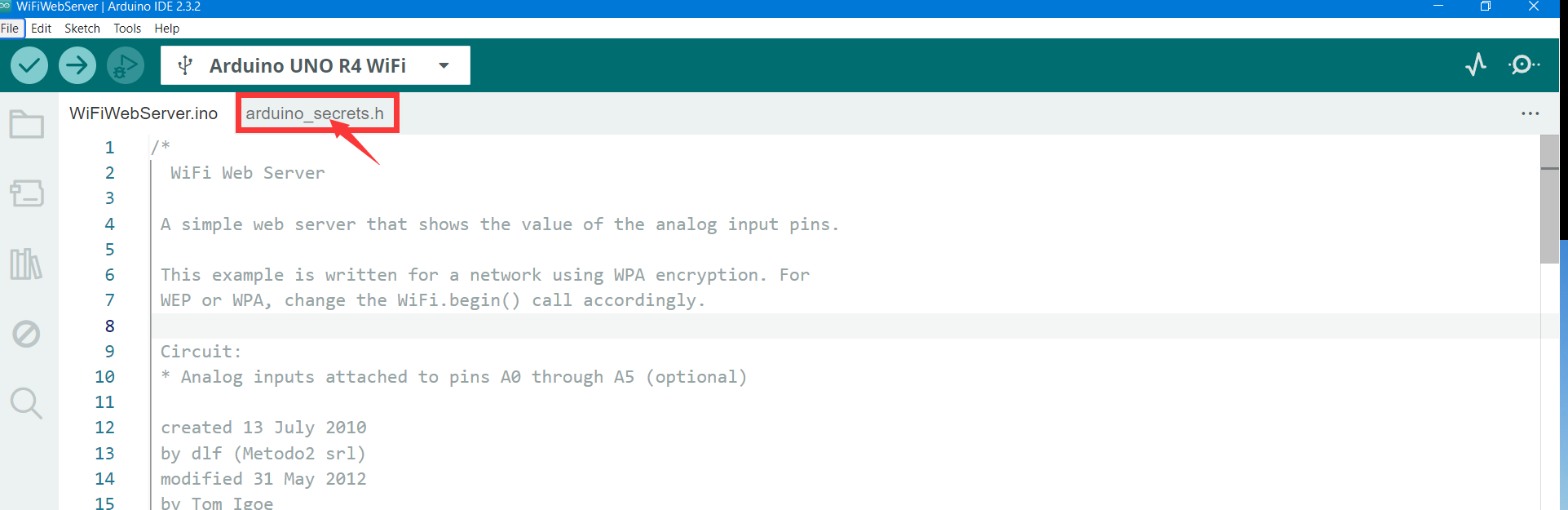
Wi-Fi® support is enabled via the built-in WiFiS3 library that is shipped with the Arduino UNO R4 Board Package. Installing the Board Package automatically installs the WiFiS3 library.

Step 1: Open the Arduino IDE and modifying the code. Find the examples called: WiFiWeb

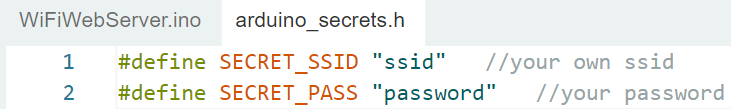
Server



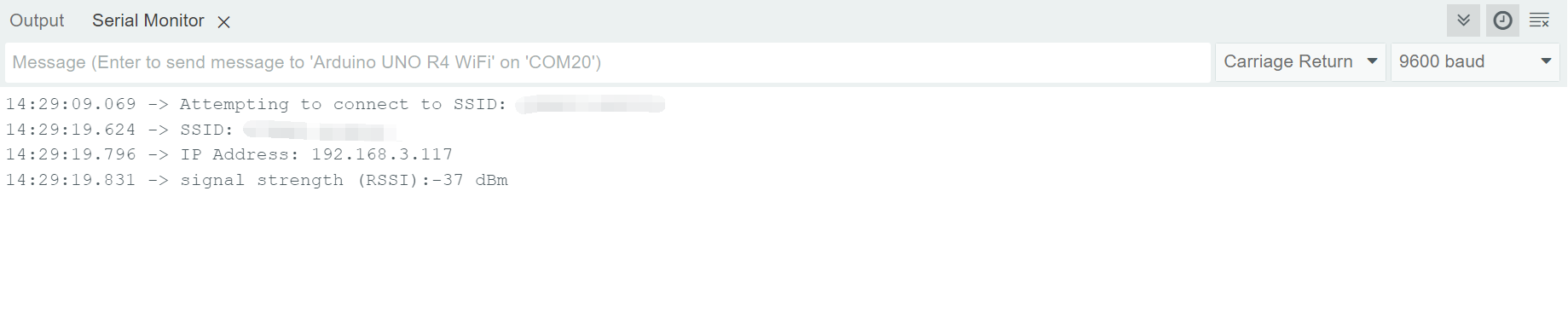
Step 2: Open the arduino\_secrets.h and write down your own SSID and PASS.



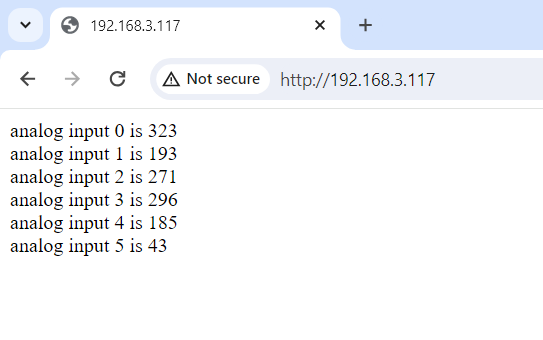
Like this:



Upload it to your Arduino UNO R4 WIFI. Open the serial monitor.



Then open your local browser and type in the address. If there is data on the web page, it means that the connection was successful, and the data was uploaded to the web server successfully.



## B. Visit a specific URL

Visit a specific URL, if successful, light green, otherwise light red (via external connection LED).

**Note:** If you have been unable to connect to the Internet during this project, you can power off the Arduino Uno R4 Wi-Fi for a while and then reconnect.

### What you need

* 2 x LED light (one red, another green)
* 1 x Arduino UNO R4 Wi-Fi
* 1 x USB 2.0 cable Type C
* 4 x male to male dumper wires

### How to wiring circuit diagram?

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | LED(Red) |
| Pin 4 | VCC(the longer pin) |
| GND | GND |

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | LED(Green) |
| 2 | VCC(the longer pin) |
| GND | GND |

Then copy the code and upload it.

**#include "WiFiS3.h"**

**#include "arduino\_secrets.h"**

**#define LED\_PIN 2**

**#define LED2\_PIN 4**

**/////////please enter your sensitive data in the Secret tab/arduino\_secrets.h**

**char ssid[] = SECRET\_SSID;        // your network SSID (name)**

**char pass[] = SECRET\_PASS;    // your network password (use for WPA, or use as key for WEP)**

**int keyIndex = 0;            // your network key index number (needed only for WEP)**

**int status = WL\_IDLE\_STATUS;**

**// if you don't want to use DNS (and reduce your sketch size)**

**// use the numeric IP instead of the name for the server:**

**char server[] = "www.github.com";    // name address for github (using DNS)**

**// Initialize the Ethernet client library**

**// with the IP address and port of the server**

**// that you want to connect to (port 80 is default for HTTP):**

**WiFiClient client;**

**/\* ------------------------------------------------- \*/**

**void setup() {**

**/\* -------------------------------------------- \*/**

**//Initialize serial and wait for port to open:**

**Serial.begin(9600);**

**while (!Serial) {**

**; // wait for serial port to connect. Needed for native USB port only**

**}**

**pinMode(LED\_PIN, OUTPUT);      // set the LED pin mode**

**pinMode(LED2\_PIN, OUTPUT);      // set the LED2 pin mode**

**// check for the WiFi module:**

**if (WiFi.status() == WL\_NO\_MODULE) {**

**Serial.println("Communication with WiFi module failed!");**

**// don't continue**

**while (true);**

**}**

**String fv = WiFi.firmwareVersion();**

**if (fv < WIFI\_FIRMWARE\_LATEST\_VERSION) {**

**Serial.println("Please upgrade the firmware");**

**}**

**// attempt to connect to WiFi network:**

**while (status != WL\_CONNECTED) {**

**Serial.print("Attempting to connect to SSID: ");**

**Serial.println(ssid);**

**// Connect to WPA/WPA2 network. Change this line if using open or WEP network:**

**status = WiFi.begin(ssid, pass);**

**// wait 10 seconds for connection:**

**delay(10000);**

**}**

**printWifiStatus();**

**Serial.println("\nStarting connection to server...");**

**// if you get a connection, report back via serial:**

**if (client.connect(server, 80)) {**

**Serial.println("connected to server");**

**// Make a HTTP request:**

**client.println("GET /search?q=arduino HTTP/1.1");**

**client.println("Host: www.github.com");**

**client.println("Connection: close");**

**client.println();**

**digitalWrite(LED\_PIN, HIGH);**

**}else {**

**digitalWrite(LED2\_PIN,HIGH);**

**}**

**}**

**/\* just wrap the received data up to 80 columns in the serial print\*/**

**/\* ----------------------------------------------- \*/**

**void read\_response() {**

**/\* ----------------------------------------- \*/**

**uint32\_t received\_data\_num = 0;**

**while (client.available()) {**

**/\* actual data reception \*/**

**char c = client.read();**

**/\* print data to serial port \*/**

**Serial.print(c);**

**/\* wrap data to 80 columns\*/**

**received\_data\_num++;**

**if(received\_data\_num % 80 == 0) {**

**Serial.println();**

**}**

**}**

**}**

**/\* ------------------------------------------------------ \*/**

**void loop() {**

**/\* ------------------------------------------------- \*/**

**read\_response();**

**// if the server's disconnected, stop the client:**

**if (!client.connected()) {**

**Serial.println();**

**Serial.println("disconnecting from server.");**

**client.stop();**

**// do nothing forevermore:**

**while (true);**

**}**

**}**

**/\* ---------------------------------------------------- \*/**

**void printWifiStatus() {**

**/\* ------------------------------------------------------ \*/**

**// print the SSID of the network you're attached to:**

**Serial.print("SSID: ");**

**Serial.println(WiFi.SSID());**

**// print your board's IP address:**

**IPAddress ip = WiFi.localIP();**

**Serial.print("IP Address: ");**

**Serial.println(ip);**

**// print the received signal strength:**

**long rssi = WiFi.RSSI();**

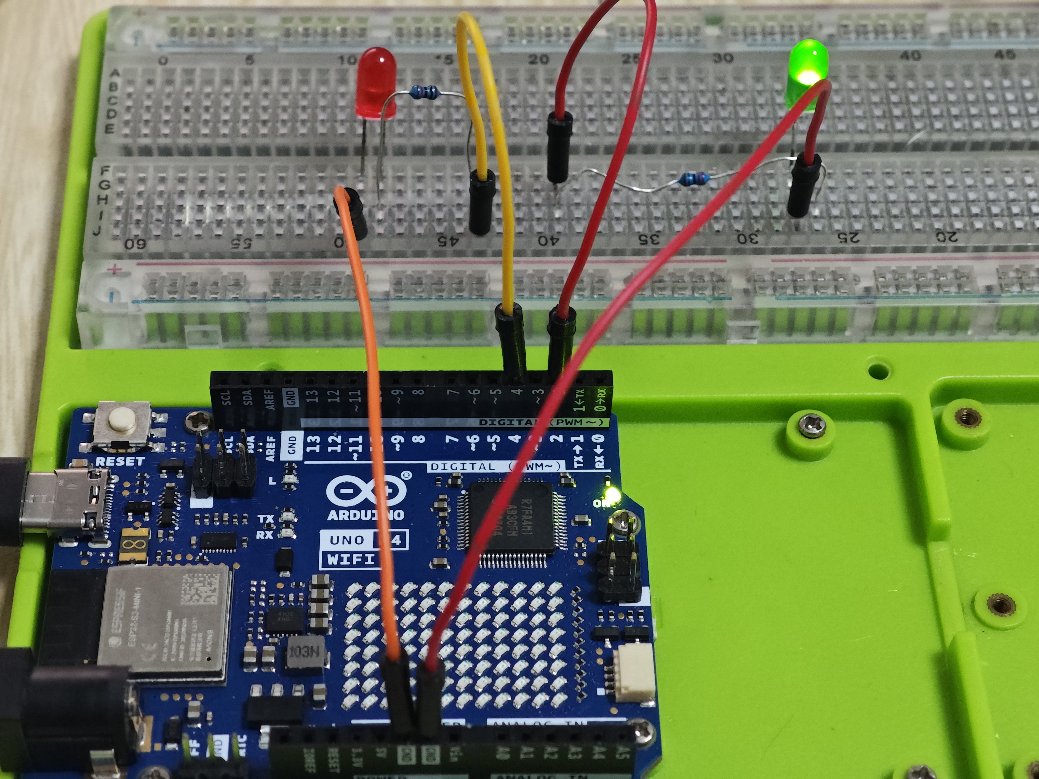
**Serial.print("signal strength (RSSI):");**

**Serial.print(rssi);**

**Serial.println(" dBm");**

**}**

What I did was just turn on the LED when I was checking whether I was connected to the website, green if it was connected, and red if it was not.



## C. Extension

Statistical delay. When the delay is greater than 7000, the yellow light is on, the green light of the network process is on, and the network delay is too long, the red light is on.

### What you need

* 3 x LED light (one red, one yellow, and one green)
* 1 x Arduino UNO R4 Wi-Fi
* 1 x USB 2.0 cable Type C
* 6 x male to male dumper wires

### How to wiring circuit diagram?

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | LED(Red) |
| 4 | VCC(the longer pin) |
| GND | GND |

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | LED(Yellow) |
| 3 | VCC(the longer pin) |
| GND | GND |

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | LED(Green) |
| 2 | VCC(the longer pin) |
| GND | GND |

Then copy the code and upload it.

**#include "WiFiS3.h"**

**#include "arduino\_secrets.h"**

**#define LED\_PIN 2**

**#define LED2\_PIN 3**

**#define LED3\_PIN 4**

**///////please enter your sensitive data in the Secret tab/arduino\_secrets.h**

**char ssid[] = SECRET\_SSID;        // your network SSID (name)**

**char pass[] = SECRET\_PASS;    // your network password (use for WPA, or use as key for WEP)**

**int keyIndex = 0;            // your network key index number (needed only for WEP)**

**int status = WL\_IDLE\_STATUS;**

**// if you don't want to use DNS (and reduce your sketch size)**

**// use the numeric IP instead of the name for the server:**

**char server[] = "www.github.com";    // name address for github (using DNS)**

**// Initialize the Ethernet client library**

**// with the IP address and port of the server**

**// that you want to connect to (port 80 is default for HTTP):**

**WiFiClient client;**

**unsigned long myTime1;**

**unsigned long myTime2;**

**unsigned long myTime3;**

**/\* ------------------------------------------------------ \*/**

**void setup() {**

**/\* ------------------------------------------------------ \*/**

**//Initialize serial and wait for port to open:**

**Serial.begin(9600);**

**while (!Serial) {**

**; // wait for serial port to connect. Needed for native USB port only**

**}**

**pinMode(LED\_PIN, OUTPUT);      // set the LED pin mode**

**pinMode(LED2\_PIN, OUTPUT);      // set the LED2 pin mode**

**pinMode(LED3\_PIN, OUTPUT);      // set the LED3 pin mode**

**// check for the WiFi module:**

**if (WiFi.status() == WL\_NO\_MODULE) {**

**Serial.println("Communication with WiFi module failed!");**

**// don't continue**

**while (true);**

**}**

**String fv = WiFi.firmwareVersion();**

**if (fv < WIFI\_FIRMWARE\_LATEST\_VERSION) {**

**Serial.println("Please upgrade the firmware");**

**}**

**// attempt to connect to WiFi network:**

**while (status != WL\_CONNECTED) {**

**Serial.print("Attempting to connect to SSID: ");**

**Serial.println(ssid);**

**// Connect to WPA/WPA2 network. Change this line if using open or WEP network:**

**status = WiFi.begin(ssid, pass);**

**// wait 10 seconds for connection:**

**delay(10000);**

**}**

**printWifiStatus();**

**Serial.print("Time1: ");**

**myTime1 = millis();**

**Serial.println("\nStarting connection to server...");**

**// if you get a connection, report back via serial:**

**if (client.connect(server, 80)) {**

**Serial.println("connected to server");**

**// Make a HTTP request:**

**client.println("GET /search?q=arduino HTTP/1.1");**

**client.println("Host: www.github.com");**

**client.println("Connection: close");**

**client.println();**

**}**

**}**

**/\* just wrap the received data up to 80 columns in the serial print\*/**

**/\* ------------------------------------------------------ \*/**

**void read\_response() {**

**/\* ------------------------------------------------------ \*/**

**uint32\_t received\_data\_num = 0;**

**while (client.available()) {**

**/\* actual data reception \*/**

**char c = client.read();**

**/\* print data to serial port \*/**

**Serial.print(c);**

**/\* wrap data to 80 columns\*/**

**received\_data\_num++;**

**if(received\_data\_num % 80 == 0) {**

**Serial.println();**

**}**

**}**

**}**

**/\* -------------------------------------------------------- \*/**

**void loop() {**

**/\* -------------------------------------------------------- \*/**

**read\_response();**

**// if the server's disconnected, stop the client:**

**if (!client.connected()) {**

**Serial.println();**

**Serial.println("disconnecting from server.");**

**client.stop();**

**Serial.print("Time2: ");**

**myTime2 = millis();**

**myTime3 = myTime2 - myTime1;**

**if(myTime3 < 3000)digitalWrite(LED\_PIN,HIGH);**

**else if(myTime3 < 7000)digitalWrite(LED2\_PIN,HIGH);**

**else digitalWrite(LED3\_PIN,HIGH);**

**// do nothing forevermore:**

**while (true);**

**}**

**}**

**/\* ---------------------------------------------------- \*/**

**void printWifiStatus() {**

**/\* ---------------------------------------------------- \*/**

**// print the SSID of the network you're attached to:**

**Serial.print("SSID: ");**

**Serial.println(WiFi.SSID());**

**// print your board's IP address:**

**IPAddress ip = WiFi.localIP();**

**Serial.print("IP Address: ");**

**Serial.println(ip);**

**// print the received signal strength:**

**long rssi = WiFi.RSSI();**

**Serial.print("signal strength (RSSI):");**

**Serial.print(rssi);**

**Serial.println(" dBm");**

**}**

Then after you run it, when you connect to the GitHub site, light up different LED lights according to the time difference before and after the connection.

**Congratulations！**



You have already got this new skills!!! Once the Arduino with sensors is connected to the network, its potential is unlimited, and we are about to start a smart home chapter! come on! Keep going!

Let’s get into next chapter!