

ทดลองปฏิบัติ

Outline

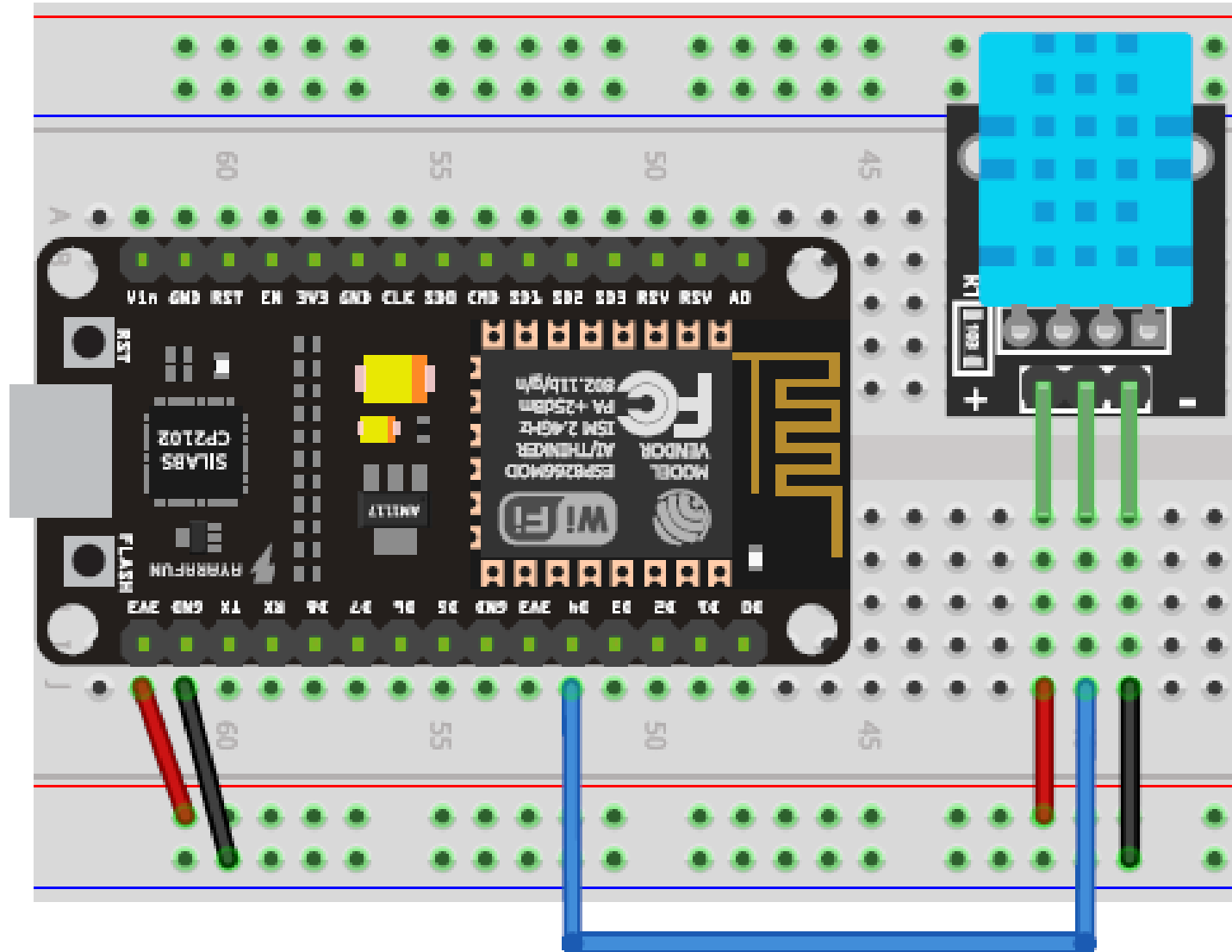
- ทดลองใช้งาน DHT 11 Module
- ทดลองใช้งาน BH1750 Module
- ทดลองใช้งาน Ultra Sonic Module HC-SR04+
- ทดลองใช้งาน LCD 16x2 i2c Display Module
- ประยุกต์การใช้งานลักษณะ IoT

DHT 11 Module

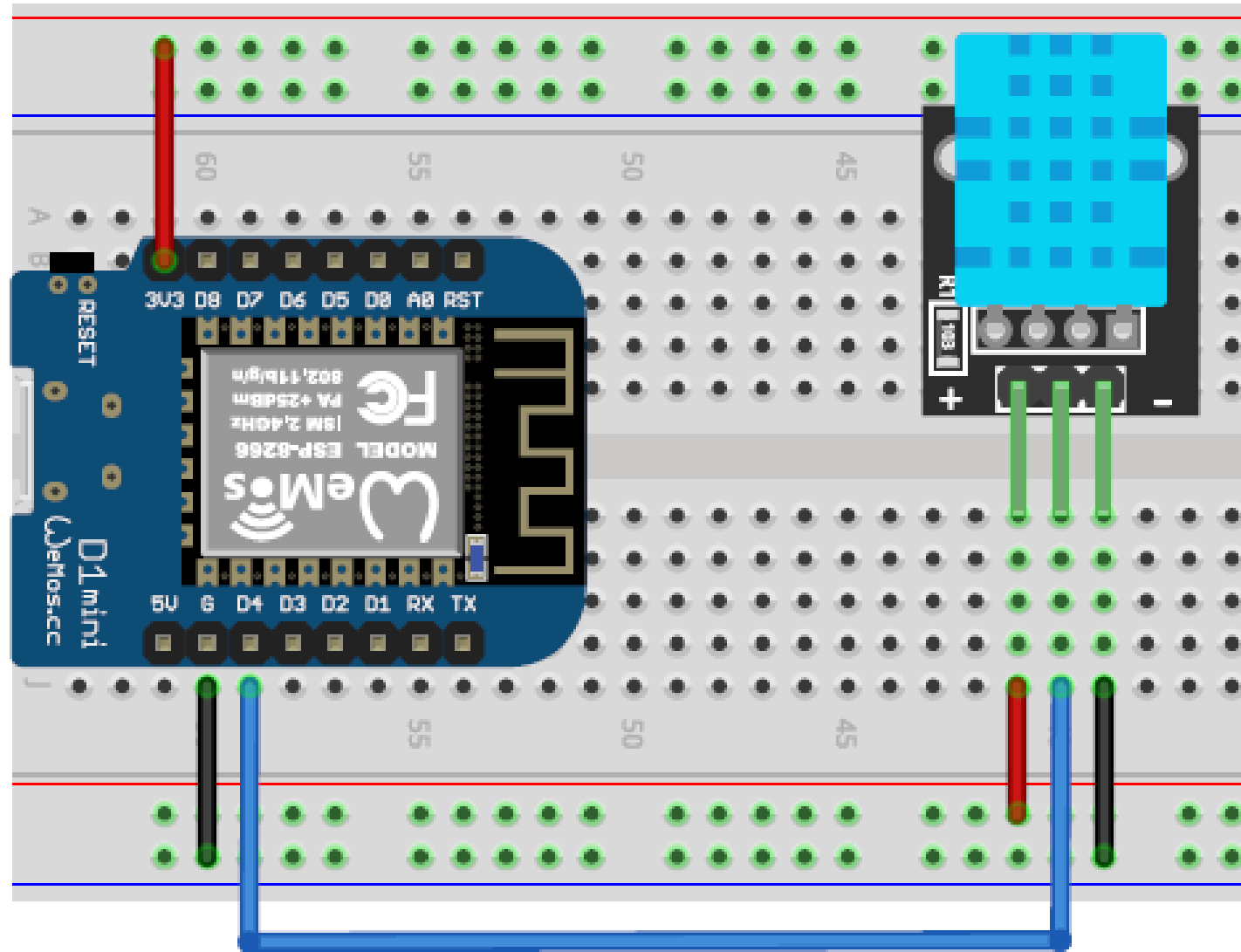
- โมดูลวัดอุณหภูมิ และ ความชื้น
- 3 to 5V power and I/O
- วัดความชื้นระดับ 20-80% โดยมีความผิดพลาดในการวัดไม่เกิน 5%
- วัดอุณหภูมิ 0-50°C โดยมีความผิดพลาดในการวัดไม่เกิน $\pm 2^{\circ}\text{C}$
- ความถี่ในการวัด 1 Hz (อ่านค่าได้วินาทีละครั้ง)



NodeMCU DHT11 Wiring Diagram



WeMos DHT11 Wiring Diagram



esp8266_dht11_lib_demo-1

```
#include "DHT.h" // -> https://github.com/adafruit/DHT-sensor-library

// define the DHT sensor type to be used (tested: DHT11 and DHT22)
#define DHT_TYPE  DHT11
// #define DHT_TYPE  DHT22

#define DHT_PIN  (2)    // use the D4 pin (GPIO-2)

// note: use Vcc=3.3V for the DHTxx sensor

DHT dht( DHT_PIN, DHT_TYPE );

// global variable used for sprintf()
char sbuf[32];
```

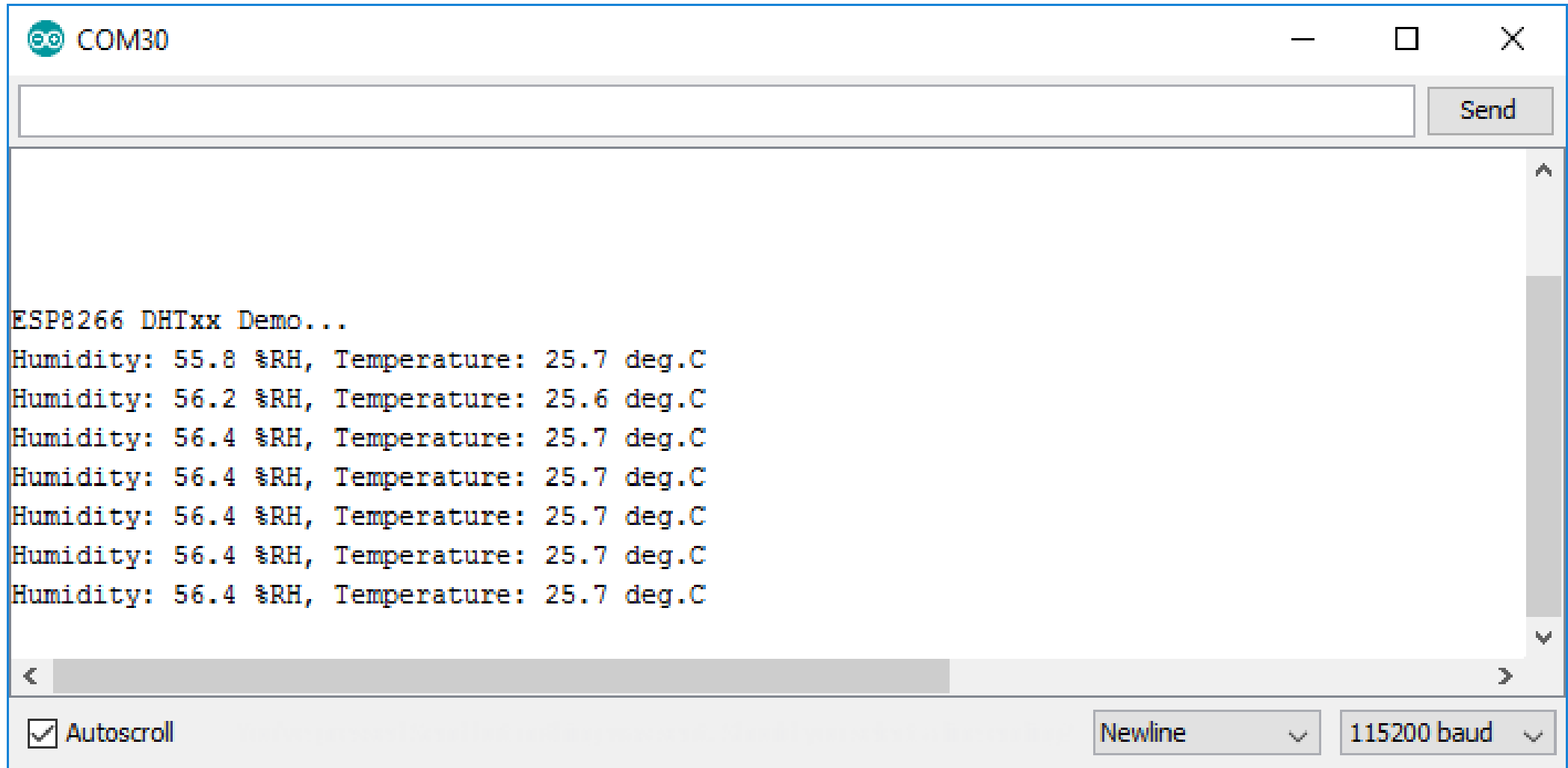
esp8266_dht11_lib_demo-1

```
void setup() {  
  Serial.begin( 115200 );  
  Serial.println( F("\n\n\n\n") );  
  delay(1000);  
  Serial.println( F("ESP8266 DHTxx Demo...") );  
  dht.begin();  
}
```

esp8266_dht11_lib_demo-1

```
void loop() {  
  float humid = dht.readHumidity(); // read the humidity  
  float temp = dht.readTemperature(); // read temperature as Celsius  
  
  // Check if any reads failed and exit early (to try again).  
  if ( isnan(humid) || isnan(temp) ) {  
    Serial.println( F("Failed to read from DHT sensor!") );  
    delay(1000);  
    return;  
  }  
  Serial.print( F("Humidity: ") );  
  dtostrf( humid, 3, 1, sbuf );  
  Serial.print( sbuf );  
  Serial.print( F(" %RH, ") );  
  Serial.print( F("Temperature: ") );  
  dtostrf( temp, 3, 1, sbuf );  
  Serial.print( sbuf );  
  Serial.println( F(" deg.C") );  
  // wait a few seconds between measurements.  
  delay(2000);  
}
```


esp8266_dht11_lib_demo-1



COM30

Send

```
ESP8266 DHTxx Demo...  
Humidity: 55.8 %RH, Temperature: 25.7 deg.C  
Humidity: 56.2 %RH, Temperature: 25.6 deg.C  
Humidity: 56.4 %RH, Temperature: 25.7 deg.C  
Humidity: 56.4 %RH, Temperature: 25.7 deg.C  
Humidity: 56.4 %RH, Temperature: 25.7 deg.C  
Humidity: 56.4 %RH, Temperature: 25.7 deg.C  
Humidity: 56.4 %RH, Temperature: 25.7 deg.C
```

☒ Autoscroll Newline 115200 baud

esp8266_dht11_lib_demo-2

```
#include <ESP8266WiFi.h>
#include "PubSubClient.h"
#include "DHT.h"

#define DHTPIN D4    // what pin we're connected to

// Uncomment whatever type you're using!
#define DHTTYPE DHT11  // DHT 11
// #define DHTTYPE DHT22  // DHT 22 (AM2302)
// #define DHTTYPE DHT21  // DHT 21 (AM2301)

DHT dht(DHTPIN, DHTTYPE);
```

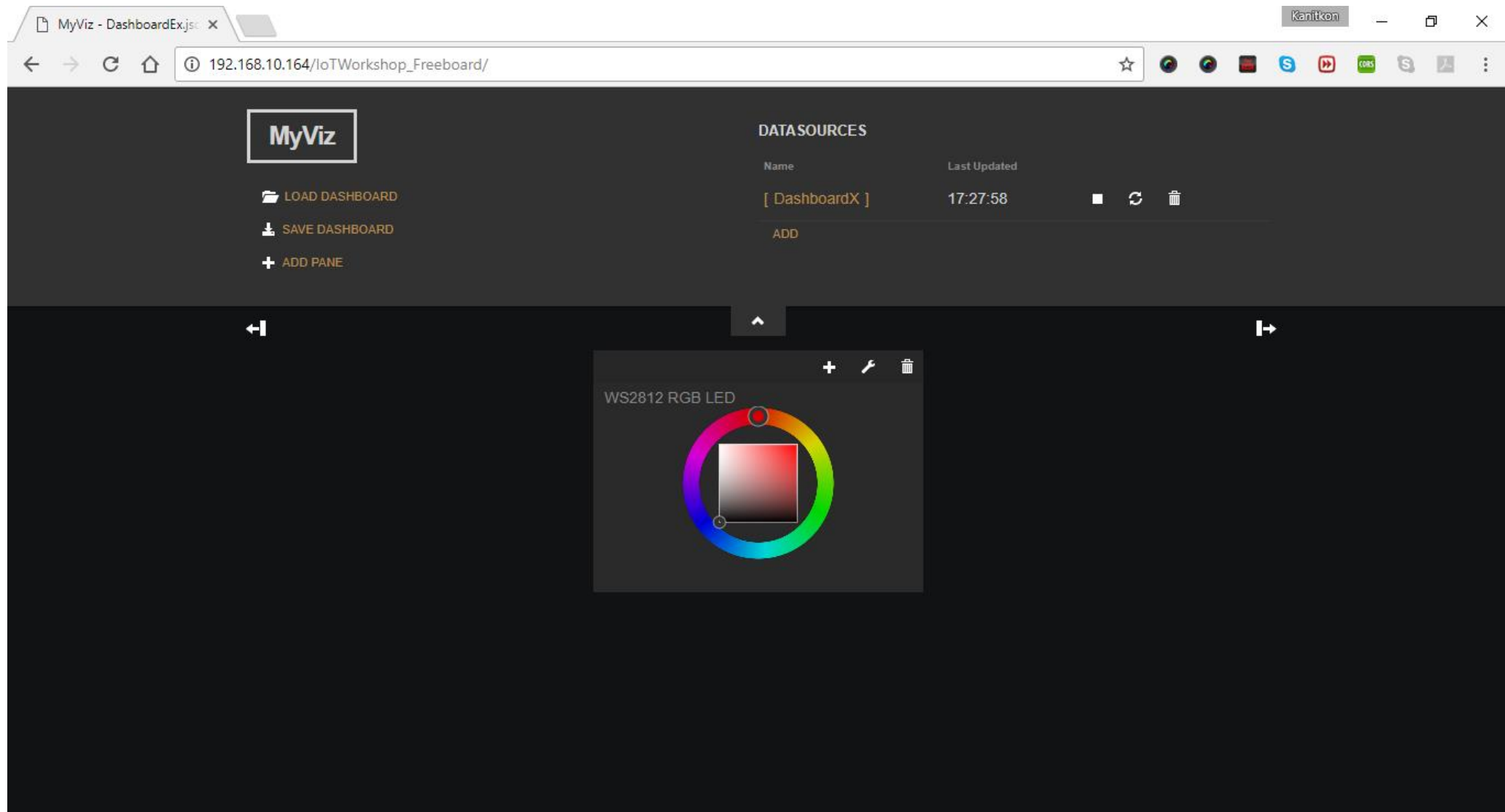
esp8266_dht11_lib_demo-2

```
void setup() {  
  Serial.begin(115200);  
  Serial.println("DHTxx test!");  
  setup_wifi();  
  client.setServer(mqtt_server, mqtt_port);  
  dht.begin();  
  sprintf(temperature_topic, "/esp8266/%d/temperature", esp_id);  
  sprintf(humidity_topic, "/esp8266/%d/humidity", esp_id);  
}
```

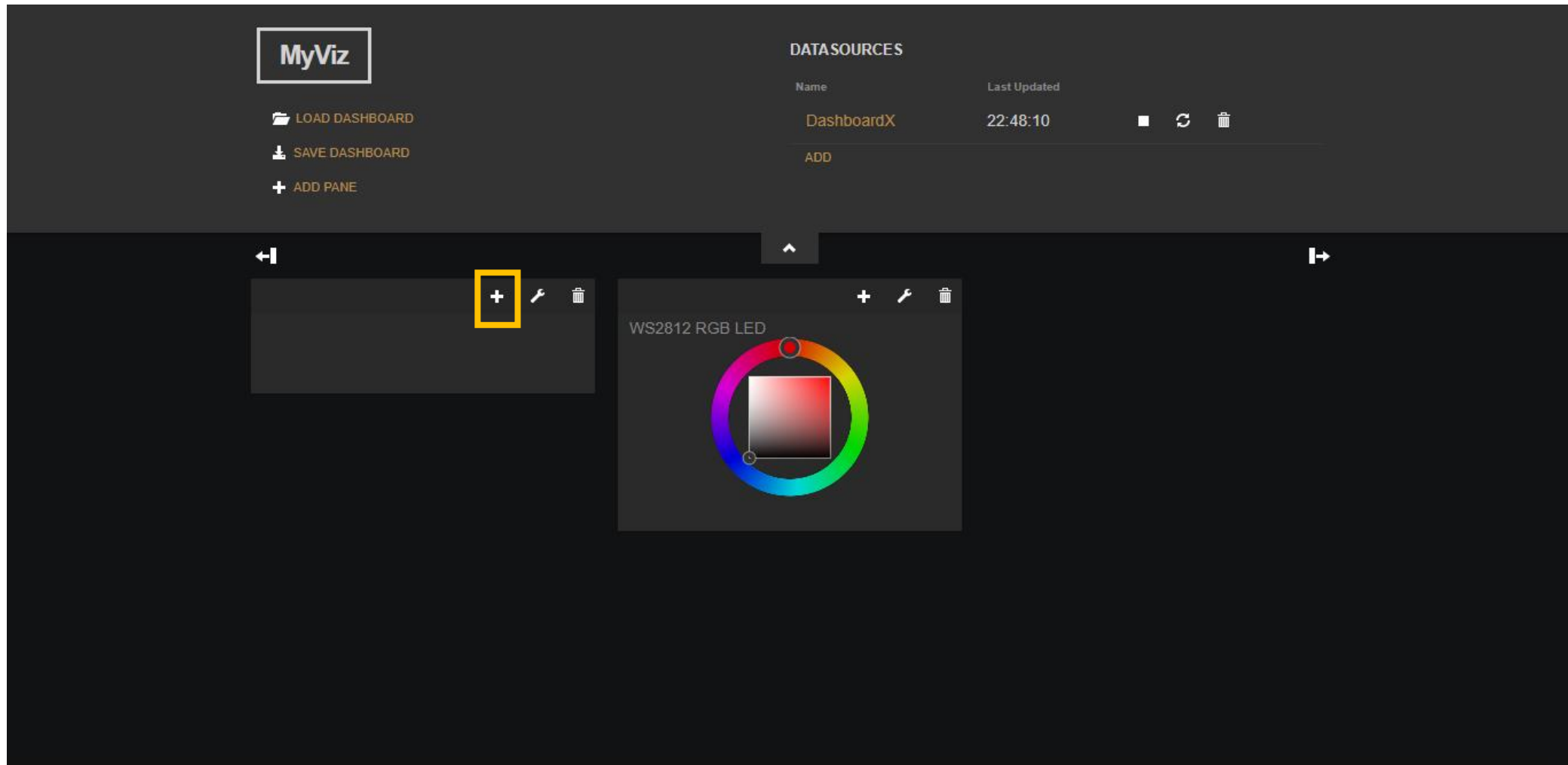
esp8266_dht11_lib_demo-2

```
void loop() {  
  if (!client.connected()) {  
    reconnect();  
  }  
  client.loop();  
  long now = millis();  
  if (now - lastMsg > 5000) {  
    lastMsg = now;  
    // Reading temperature or humidity takes about 250 milliseconds!  
    // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)  
    float newHum = dht.readHumidity();  
    float newTemp = dht.readTemperature();  
    Serial.print("New temperature:");  Serial.println(String(newTemp).c_str());  
    client.publish(temperature_topic, String(newTemp).c_str(), true);  
    Serial.print("New humidity:");  Serial.println(String(newHum).c_str());  
    client.publish(humidity_topic, String(newHum).c_str(), true);  
  }  
}
```

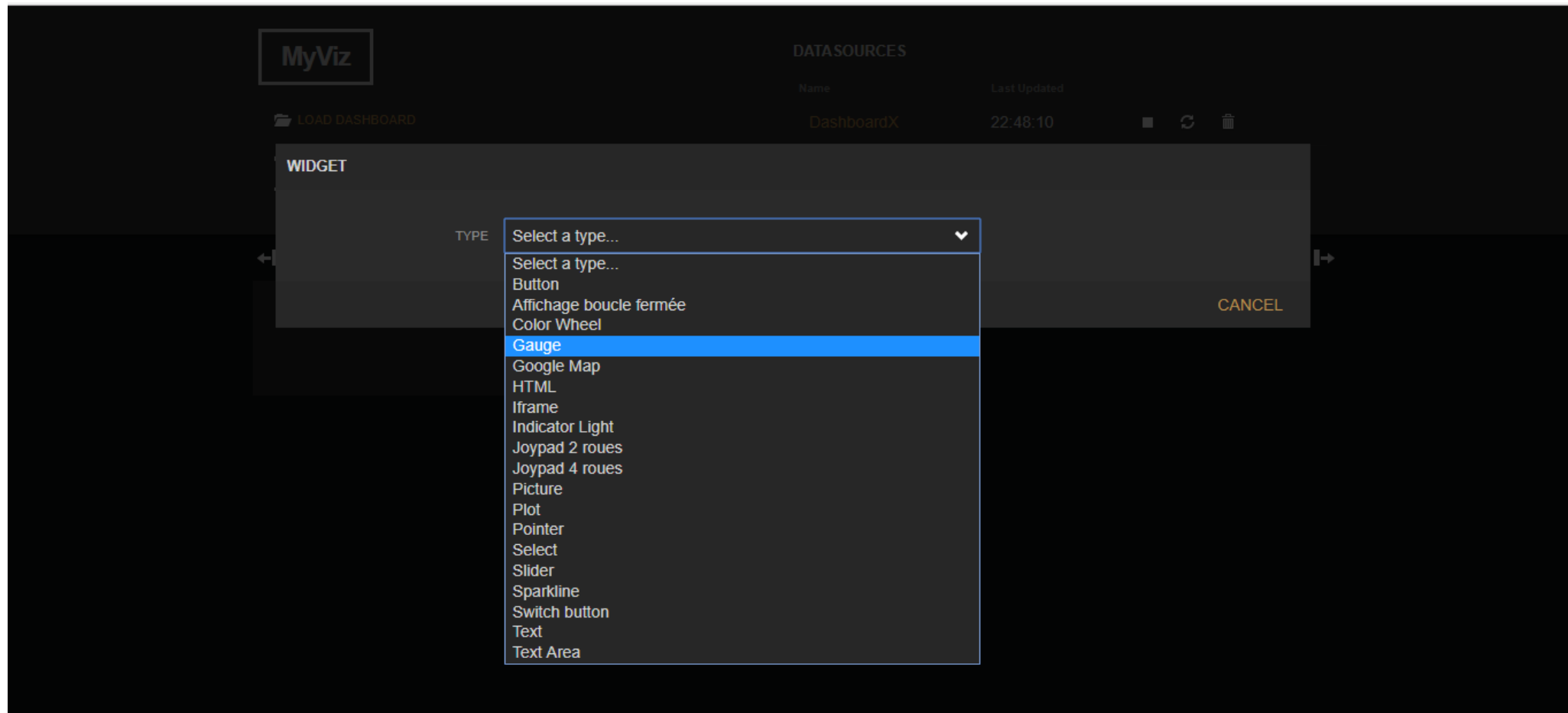
ใช้งาน Freeboard เรียกดูค่า DHT11



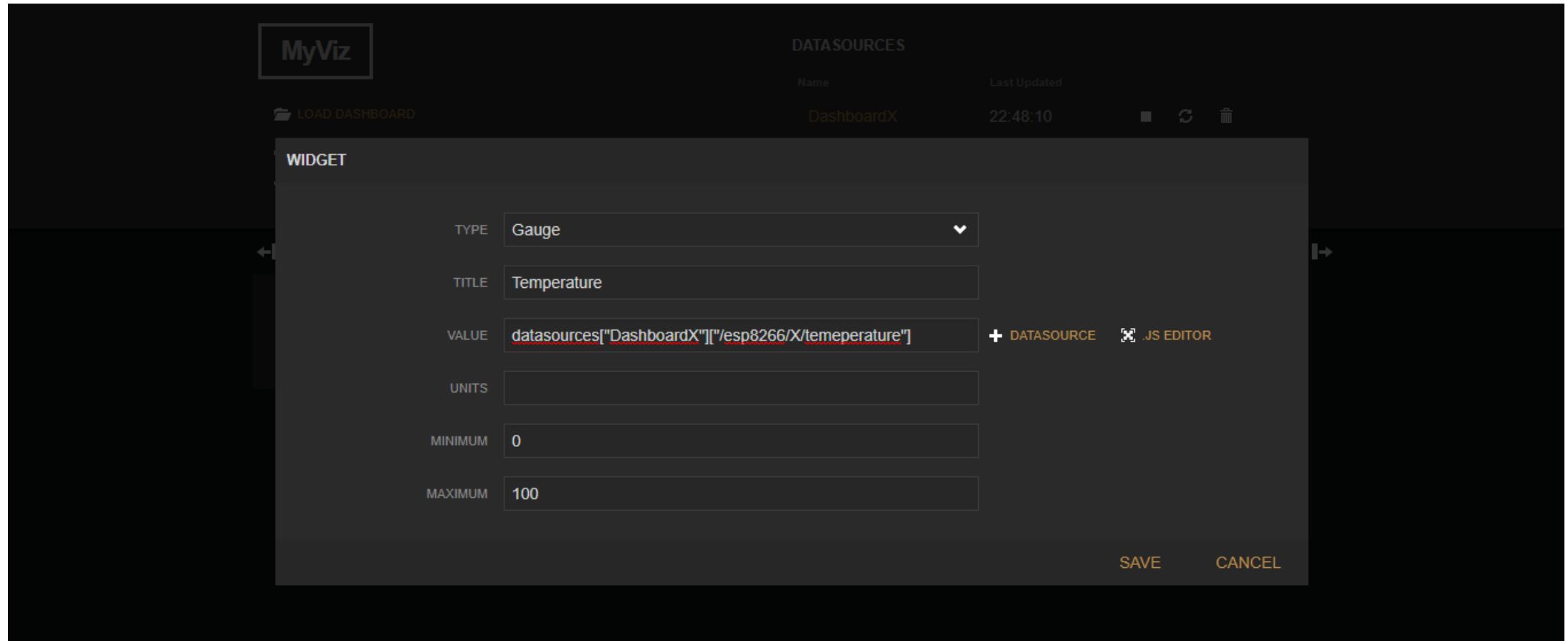
ใช้งาน Freeboard เรียกดูค่า DHT11



ใช้งาน Freeboard เรียกดูค่า DHT11



ใช้งาน Freeboard เรียกดูค่า DHT11



ใช้งาน Freeboard เรียกดูค่า DHT11

The screenshot displays the MyViz dashboard interface. At the top left, the 'MyViz' logo is visible. Below it are three buttons: 'LOAD DASHBOARD', 'SAVE DASHBOARD', and '+ ADD PANE'. To the right, a 'DATASOURCES' table lists the data sources. The table has columns for 'Name' and 'Last Updated'. The first entry is 'DashboardX' with a last update time of '22:48:10'. Below the table is an 'ADD' button. The main dashboard area contains two panes. The left pane, titled 'Temperature', features a semi-circular gauge with a needle pointing to '0' on a scale from 0 to 100. The right pane, titled 'WS2812 RGB LED', shows a color wheel with a red-to-white gradient square in the center. Navigation icons for zooming and panning are located at the top of the dashboard area.

Name	Last Updated
DashboardX	22:48:10

ADD

Temperature

0 100

WS2812 RGB LED

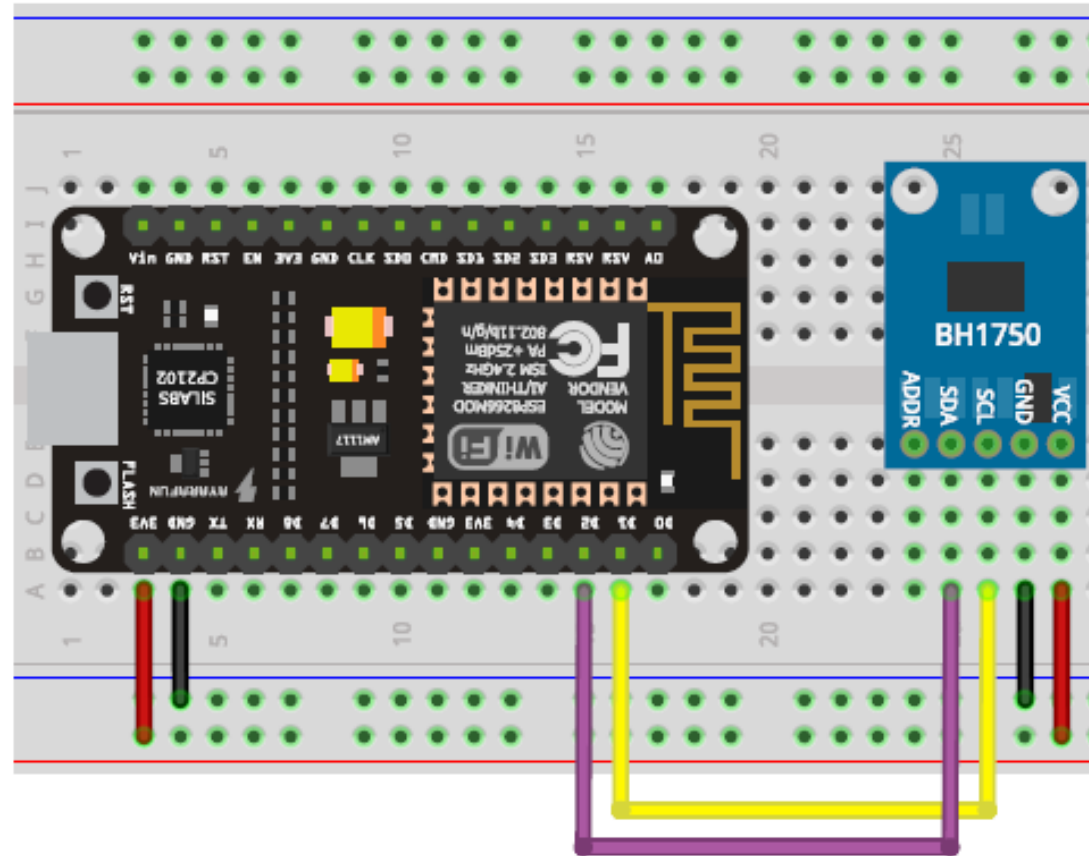
BH1750 Module

- โมดูลวัดความเข้มแสง
- เชื่อมต่อแบบ I2C
- แรงดันไฟเลี้ยงในช่วง 2.4V - 3.6V
- ความละเอียด: 16 บิต ได้ค่า 1-65536 หน่วยเป็น Lux (step: 0.5 Lux, 1 Lux, หรือ 4 Lux ขึ้นอยู่กับโหมดการวัดที่เลือก)
- ระยะเวลาในการวัดแต่ละครั้ง: ประมาณ 120 msec (สำหรับ 0.5 Lux หรือ 1 Lux), 16 msec (สำหรับ 4 Lux) ขึ้นอยู่กับโหมดการวัดที่เลือก

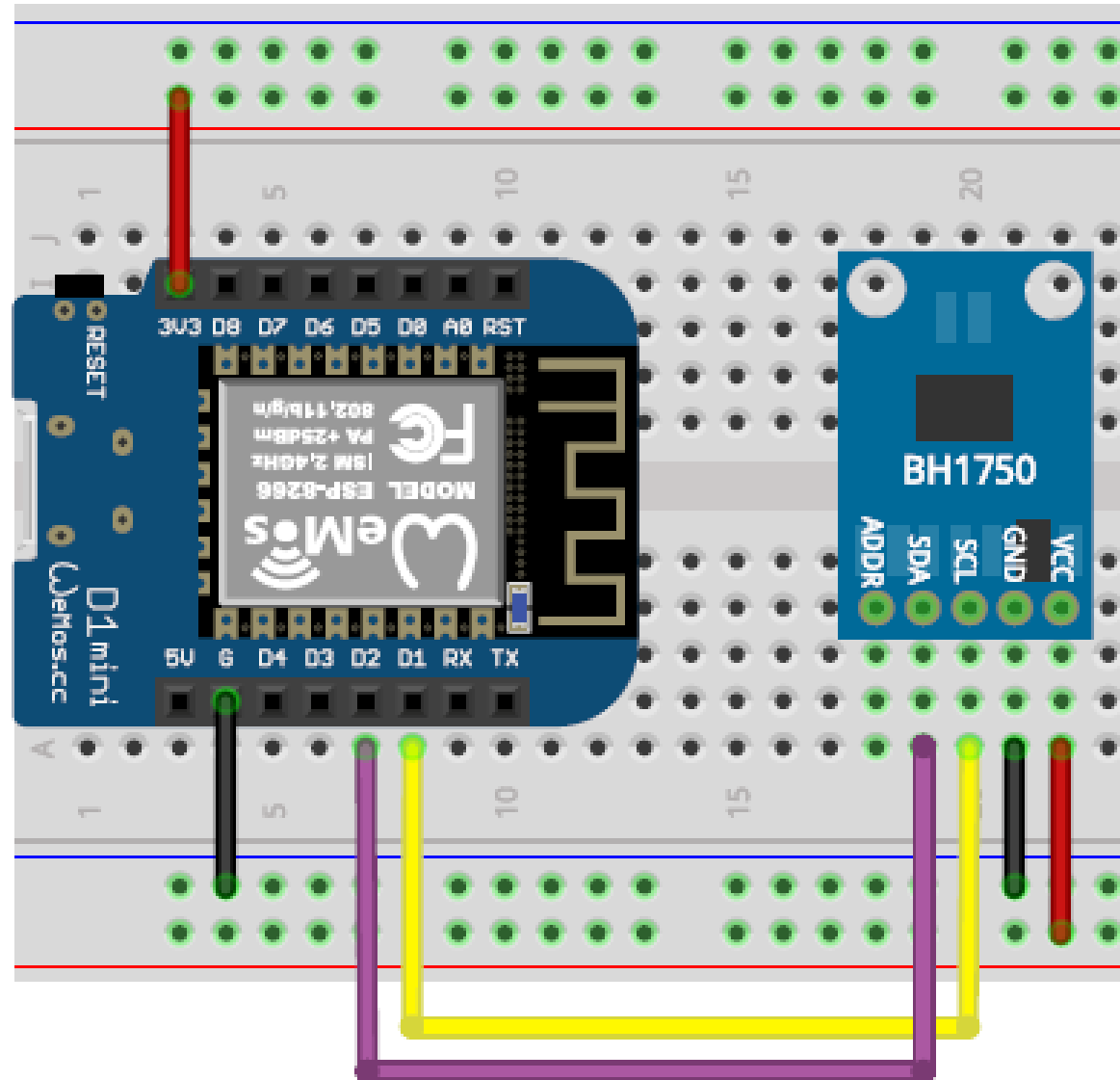


ที่มา : <https://goo.gl/G74yWP>

NodeMCU BH1750 Wiring Diagram



WeMos BH1750 Wiring Diagram



esp8266_bh1750_lib_demo-1

```
#include <Wire.h>
#include "BH1750.h"          // -> https://github.com/claws/BH1750

#define LED_PIN      (D4)    // D4 pin (GPIO-2)
#define SDA_PIN      (D2)    // D2 pin (GPIO-4)
#define SCL_PIN      (D1)    // D1 pin (GPIO-5)

#define DEV_ADDR      (0x23) // set the I2C device address (BH1750)

#define INTERVAL_MSEC (1000) // update interval: 1 sec

BH1750 bh( DEV_ADDR );      // create BH1750 object

// global variables
char sbuf[64];              // char buffer for sprintf()
uint32_t ts;                // used to save timestamp
```

esp8266_bh1750_lib_demo-1

```
void setup() {  
  pinMode( LED_PIN, OUTPUT );  
  digitalWrite( LED_PIN, LOW );  
  Serial.begin( 115200 );  
  
  for ( int i=0; i < 10; i++ ) {  
    delay(100);  
    Serial.println();  
  }  
  Serial.flush();  
  
  Serial.println( F("BH1750 Sensor Reading...") );  
  bh.begin( BH1750_CONTINUOUS_HIGH_RES_MODE,  
            SDA_PIN, SCL_PIN, 400000 /* set I2C frequency */ );  
  ts = millis();  
}
```

esp8266_bh1750_lib_demo-1

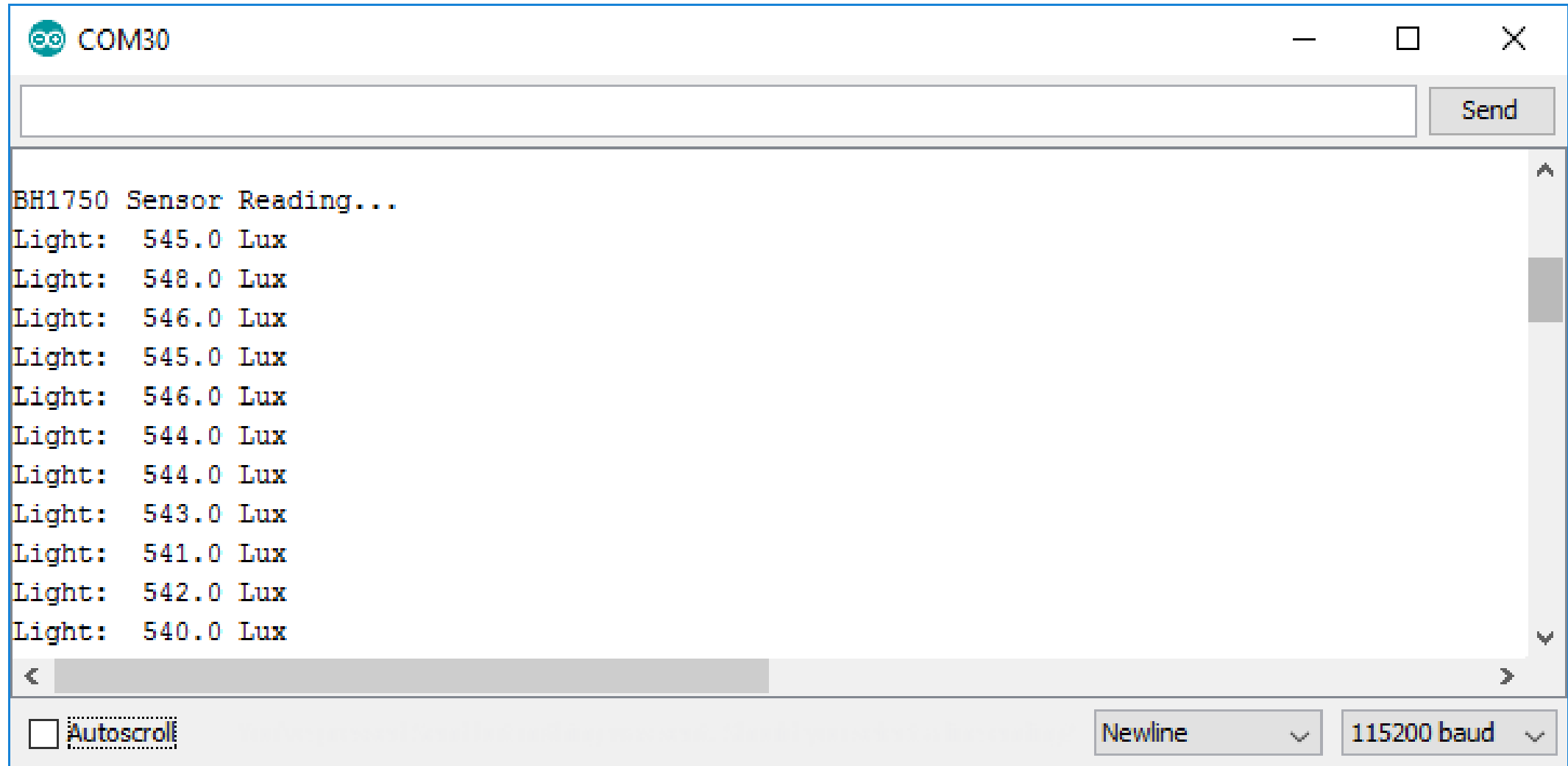
```
void process() {
    uint16_t lux = bh.readLightLevel(); // read the sensor value

    String str;
    str += "Light: ";
    dtostrf( lux, 6, 1, sbuf );
    str += sbuf;
    str += " Lux";

    Serial.println( str.c_str() );
}

void loop() {
    if ( millis() - ts >= INTERVAL_MSEC ) {
        ts += INTERVAL_MSEC;
        digitalWrite( LED_PIN, HIGH );
        process();
        digitalWrite( LED_PIN, LOW );
    }
    delay(1); // delay for 1 msec
}
```

esp8266_bh1750_lib_demo-1



The screenshot shows a serial monitor window with the title 'COM30'. At the top, there is a text input field and a 'Send' button. The main area displays the following text:

```
BH1750 Sensor Reading...  
Light: 545.0 Lux  
Light: 548.0 Lux  
Light: 546.0 Lux  
Light: 545.0 Lux  
Light: 546.0 Lux  
Light: 544.0 Lux  
Light: 544.0 Lux  
Light: 543.0 Lux  
Light: 541.0 Lux  
Light: 542.0 Lux  
Light: 540.0 Lux
```

At the bottom, there is a horizontal scrollbar. Below the scrollbar, there is a checkbox labeled 'Autoscroll' which is currently unchecked. To the right of the checkbox are two dropdown menus: the first is labeled 'Newline' and the second is labeled '115200 baud'.

esp8266_bh1750_lib_demo-2

```
#include <Wire.h> // use the Wire library
#include "BH1750.h" // https://github.com/claws/BH1750
#include <ESP8266WiFi.h>
#include "PubSubClient.h"

#define I2C_SCL_PIN (D1) // D1 pin (GPIO-5)
#define I2C_SDA_PIN (D2) // D2 pin (GPIO-4)
#define I2C_BH1750_ADDR (0x23)

BH1750 bh( I2C_BH1750_ADDR );

char sbuf[64];
uint32_t ts;
```

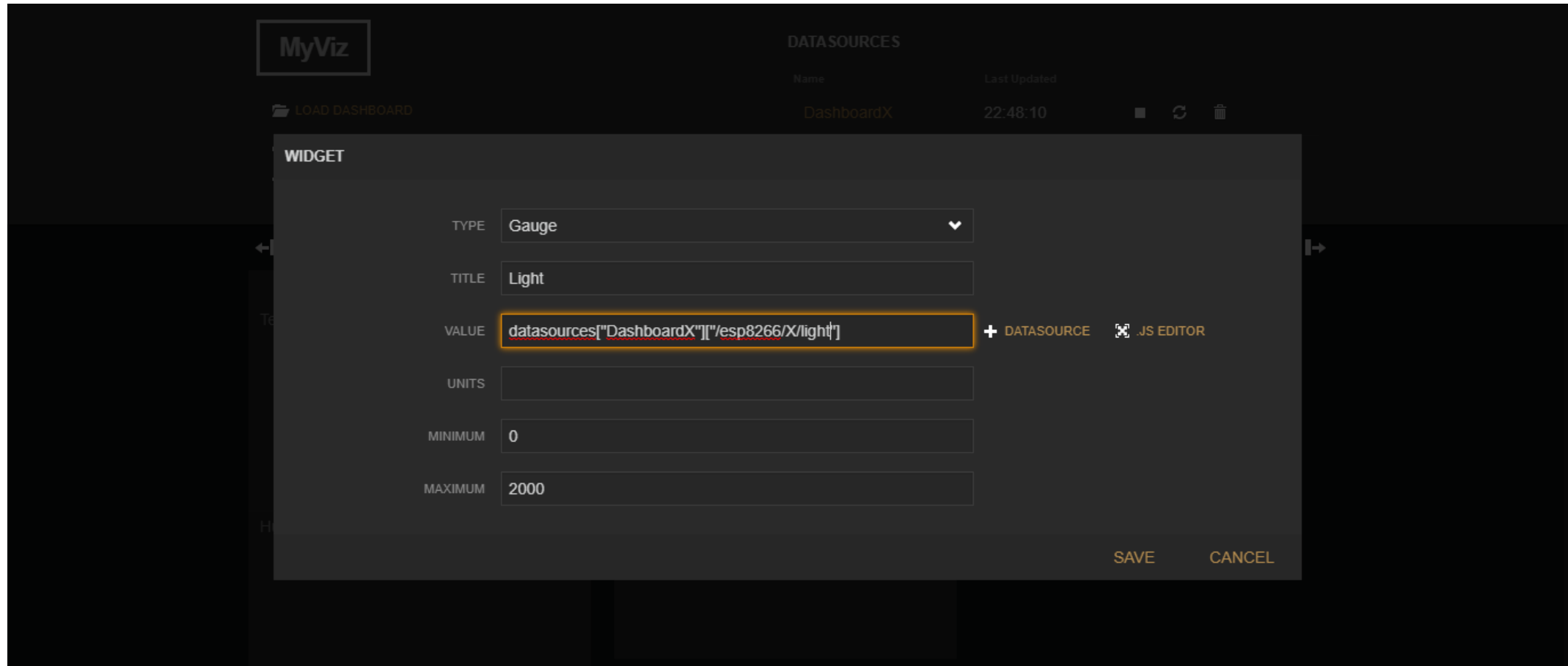
esp8266_bh1750_lib_demo-2

```
void setup() {  
  // put your setup code here, to run once:  
  Serial.begin( 115200 );  
  Serial.println( "\n\n\n" );  
  
  setup_wifi();  
  client.setServer(mqtt_server, mqtt_port);  
  
  Wire.begin( I2C_SDA_PIN, I2C_SCL_PIN );  
  delay(1000);  
  i2c_scan();  
  delay(1000);  
  
  bh.begin( BH1750_CONTINUOUS_HIGH_RES_MODE, I2C_SDA_PIN, I2C_SCL_PIN, 400000 );  
  
  Serial.print( "Light Intensity" );  
  ts = millis();  
}
```

esp8266_bh1750_lib_demo-2

```
void loop() {  
  // put your main code here, to run repeatedly:  
  if (!client.connected()) {  
    reconnect();  
  }  
  client.loop();  
  
  if ( millis() - ts >= INTERVAL_MSEC ) {  
    char val_str[8];  
    ts += INTERVAL_MSEC;  
  
    uint16_t lux = bh.readLightLevel();  
    dtostrf( lux, 6, 1, val_str );  
  
    char light_topic[64];  
    sprintf(light_topic, "/esp8266/%d/light", esp_id);  
    sprintf( sbuf, "  %s Lux", val_str );    Serial.println( sbuf );  
    client.publish(light_topic, val_str, true);  
  }  
  delay(1);  
}
```

ใช้งาน Freeboard เรียกดูค่า BH1750



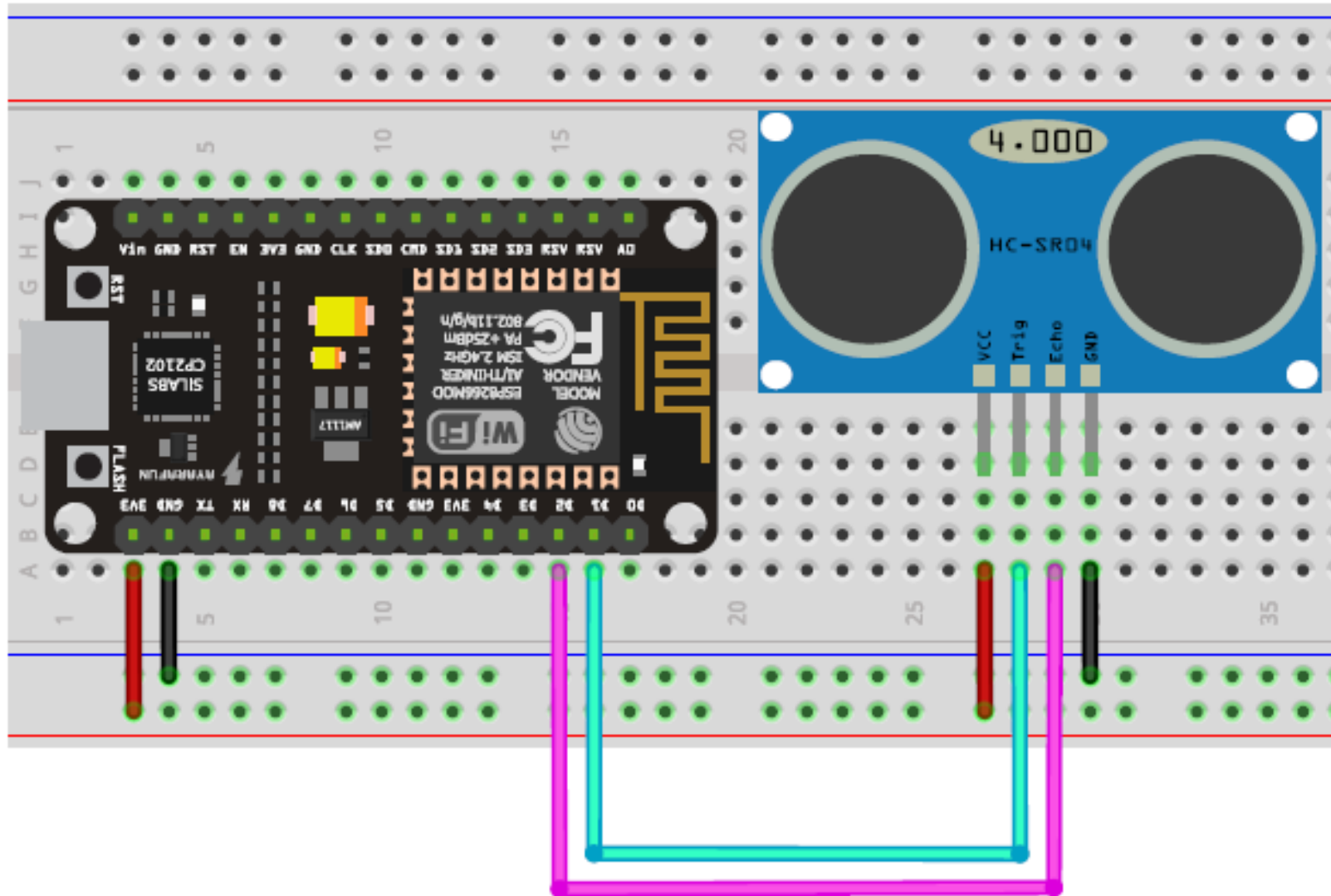
Ultrasonic Module HC-SR04+

- โมดูลวัดระยะห่างโดยใช้คลื่นเสียง
- 3.3V power and I/O
- ช่วงในการวัดที่ 2-400 cm

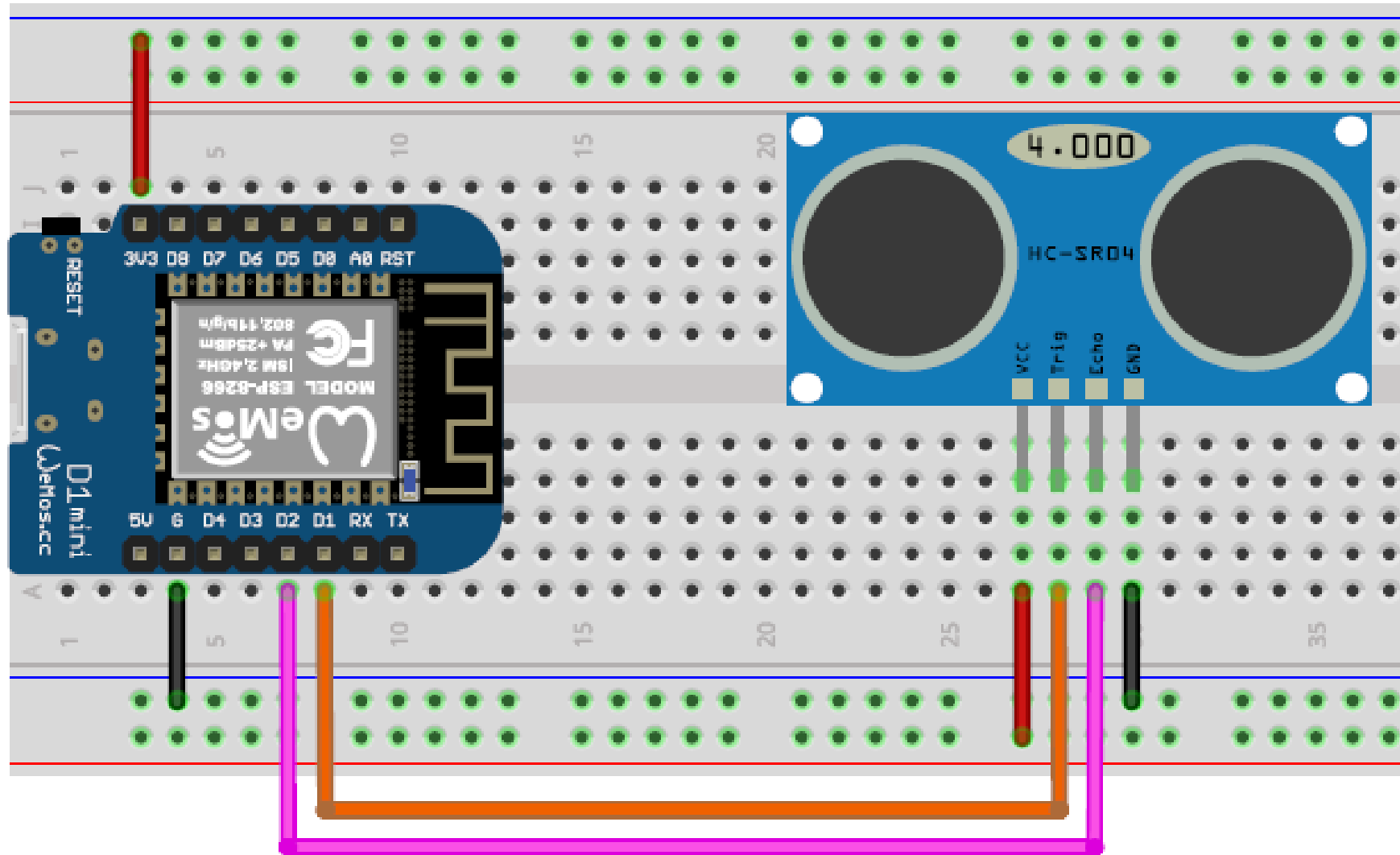


ที่มา : <https://goo.gl/5tUDuq>

NodeMCU Ultrasonic Wiring Diagram



WeMos Ultrasonic Wiring Diagram



esp8266_ultrasonic_sr04p_demo-1

```
#define TRIG_PIN 5 // GPIO-5 / D1 pin
#define ECHO_PIN 4 // GPIO-4 / D2 pin

const uint32_t timeout_usec = 40000; // timeout in microseconds
const uint32_t sound_speed = 34300; // in centimeters/second

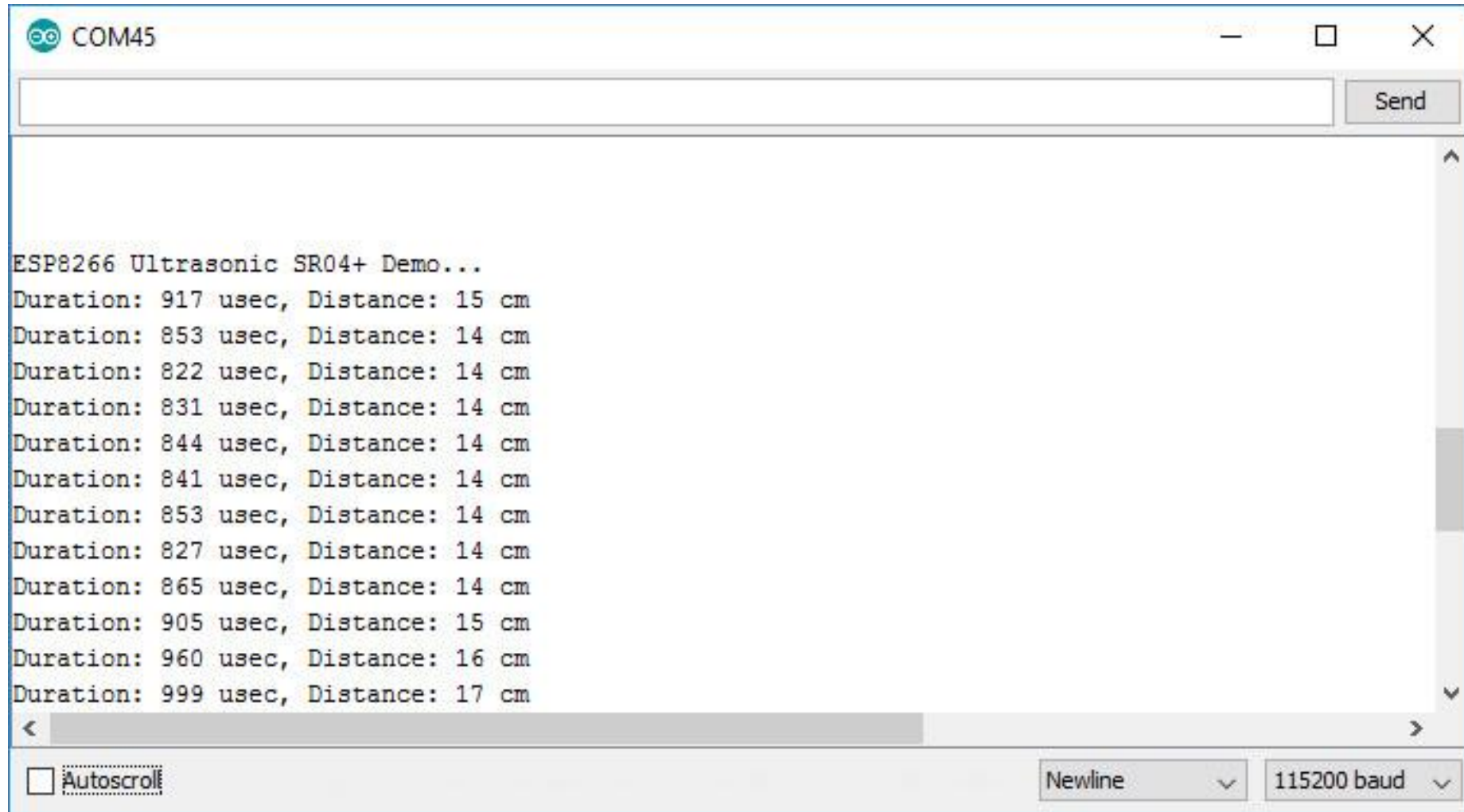
String str;

void setup() {
  Serial.begin( 115200 );
  Serial.println( F("\n\n\n") );
  delay(100);
  Serial.println( F("ESP8266 Ultrasonic SR04+ Demo...") );
  pinMode( ECHO_PIN, INPUT );
  pinMode( TRIG_PIN, OUTPUT );
}
```


esp8266_ultrasonic_sr04p_demo-1

```
void loop() {  
    // send a PING signal (a short-pulse signal on TRIG pin)  
    digitalWrite( TRIG_PIN, HIGH );  
    delayMicroseconds( 20 );  
    digitalWrite( TRIG_PIN, LOW );  
  
    // see: https://www.arduino.cc/en/Reference/pulseIn  
    uint32_t duration = pulseIn( ECHO_PIN, HIGH, timeout_usec );  
    str = F("Duration: ");  
    str += duration;  
    str += F(" usec, ");  
    str += F("Distance: ");  
    str += (sound_speed * duration / 1000000) / 2;  
    str += F(" cm");  
    Serial.println( str );  
  
    delay(500);  
}
```

esp8266_ultrasonic_sr04p_demo-1



```
ESP8266 Ultrasonic SR04+ Demo...
Duration: 917 usec, Distance: 15 cm
Duration: 853 usec, Distance: 14 cm
Duration: 822 usec, Distance: 14 cm
Duration: 831 usec, Distance: 14 cm
Duration: 844 usec, Distance: 14 cm
Duration: 841 usec, Distance: 14 cm
Duration: 853 usec, Distance: 14 cm
Duration: 827 usec, Distance: 14 cm
Duration: 865 usec, Distance: 14 cm
Duration: 905 usec, Distance: 15 cm
Duration: 960 usec, Distance: 16 cm
Duration: 999 usec, Distance: 17 cm
```

☐ Autoscroll Newline 115200 baud

esp8266_ultrasonic_sr04p_demo-2

```
void setup() {  
  Serial.begin(115200);  
  pinMode( ECHO_PIN, INPUT );  
  pinMode( TRIG_PIN, OUTPUT );  
  setup_wifi();  
  client.setServer(mqtt_server, mqtt_port);  
}  
  
uint32_t read_distance() {  
  // send a PING signal (a short-pulse signal on TRIG pin)  
  digitalWrite( TRIG_PIN, HIGH );  
  delayMicroseconds( 20 );  
  digitalWrite( TRIG_PIN, LOW );  
  // see: https://www.arduino.cc/en/Reference/pulseIn  
  // measure pulse width of the ECHO signal  
  uint32_t duration = pulseIn( ECHO_PIN, HIGH, timeout_usec );  
  uint32_t distance_cm = (sound_speed * duration / 1000000) / 2;  
  return distance_cm;  
}
```

esp8266_ultrasonic_sr04p_demo-2

```
void loop() {  
  if (!client.connected()) {  
    reconnect();  
  }  
  client.loop();  
  
  long now = millis();  
  if (now - lastMsg > 500) {  
    lastMsg = now;  
  
    char distance[64];  
    sprintf(distance, "/esp8266/%d/distance", esp_id);  
  
    uint32_t newDistance = read_distance();  
    Serial.print("New distance:");   Serial.println(String(newDistance).c_str());  
    client.publish(distance, String(newDistance).c_str(), true);  
  }  
}
```

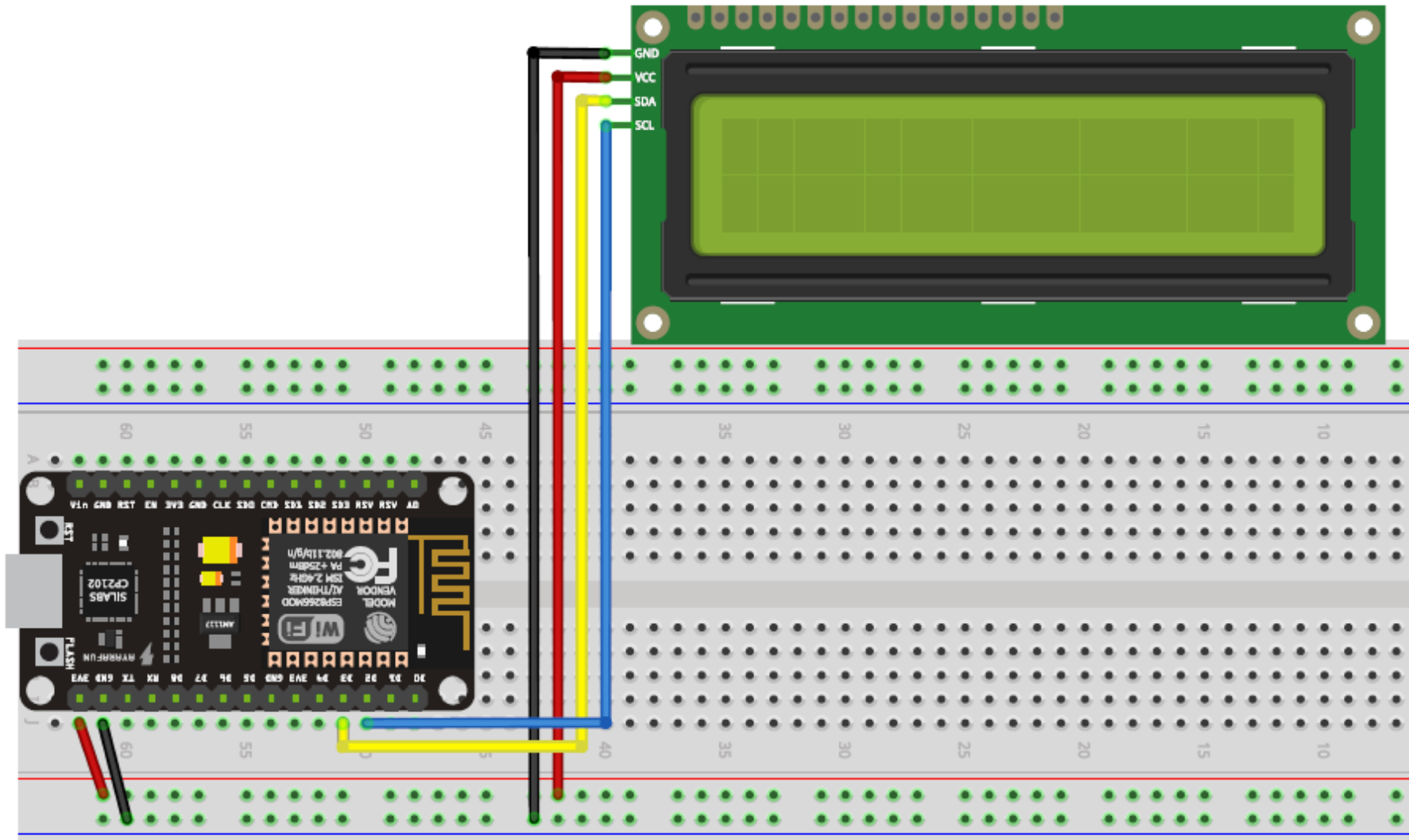
LCD 16x2 i2c Display Module

- โมดูลแสดงข้อความขนาด 16 ตัวอักษร 2 แถว
- มีการเชื่อมต่อแบบ i2c โดยใช้โมดูลที่ใช้ไอซี PCF8574

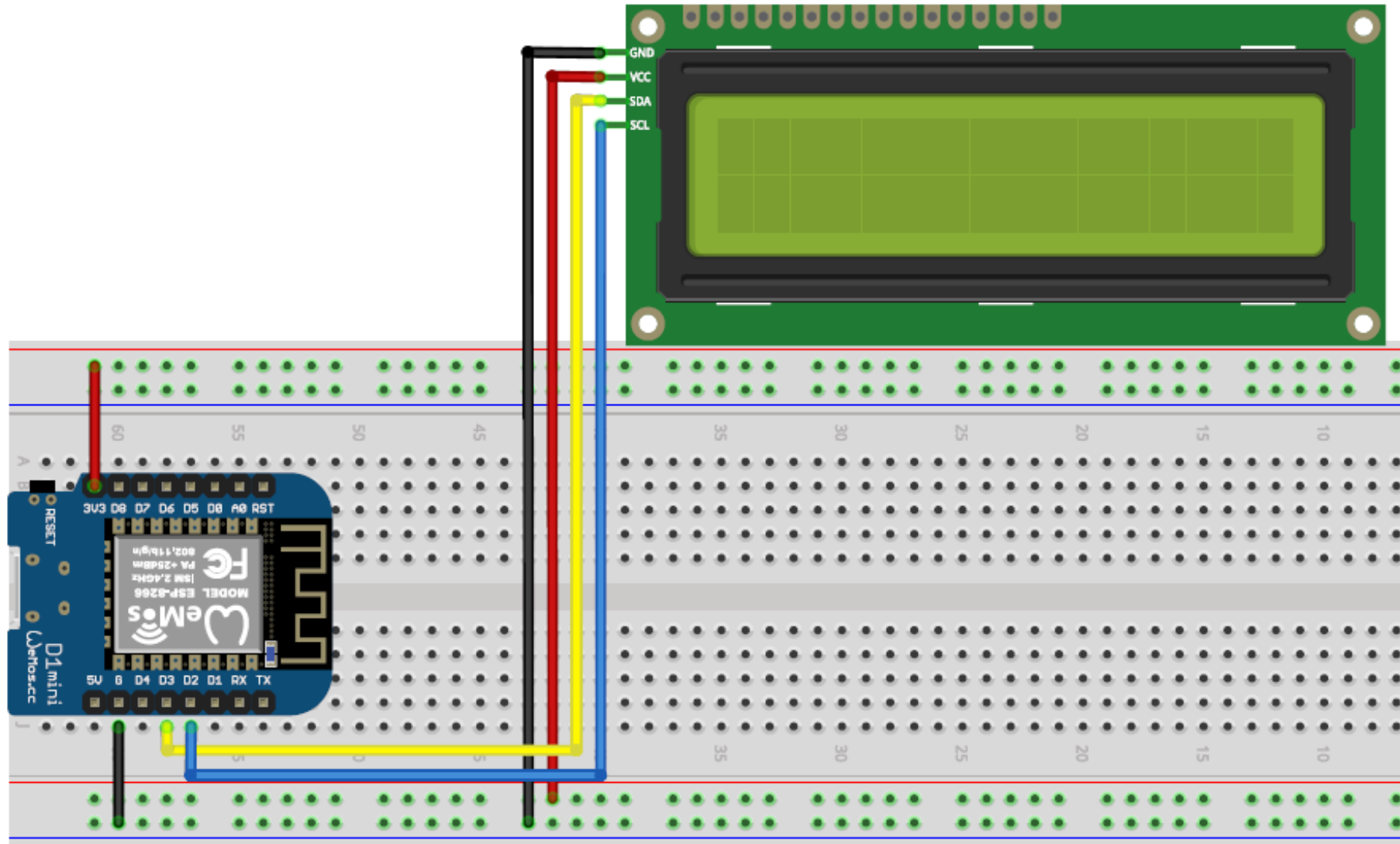


ที่มา : <https://goo.gl/EcD6p2>

NodeMCU LCD Wiring Diagram



WeMos LCD Wiring Diagram



esp8266_pcf8574a_lcd_lib_demo-1

```
#include <Wire.h> // use the Wire library
#include "LiquidCrystal_I2C.h" // -> https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library

#define I2C_SCL_PIN    (4)    // D2 pin (SCL / GPIO-4)
#define I2C_SDA_PIN    (0)    // D3 pin (SDA / GPIO-0)
#define I2C_ADDR        (0x3F) // set the I2C address for PCF8574 LCD adapter (0x27 or 0x3F)

LiquidCrystal_I2C lcd( I2C_ADDR, 16, 2 ); // 16x2 LCD display, set I2C address

// global variables
char sbuf[64]; // used for sprintf()
uint32_t ts;   // used to save timestamp value

#define INTERVAL_MSEC (1000)
```


esp8266_pcf8574a_lcd_lib_demo-1

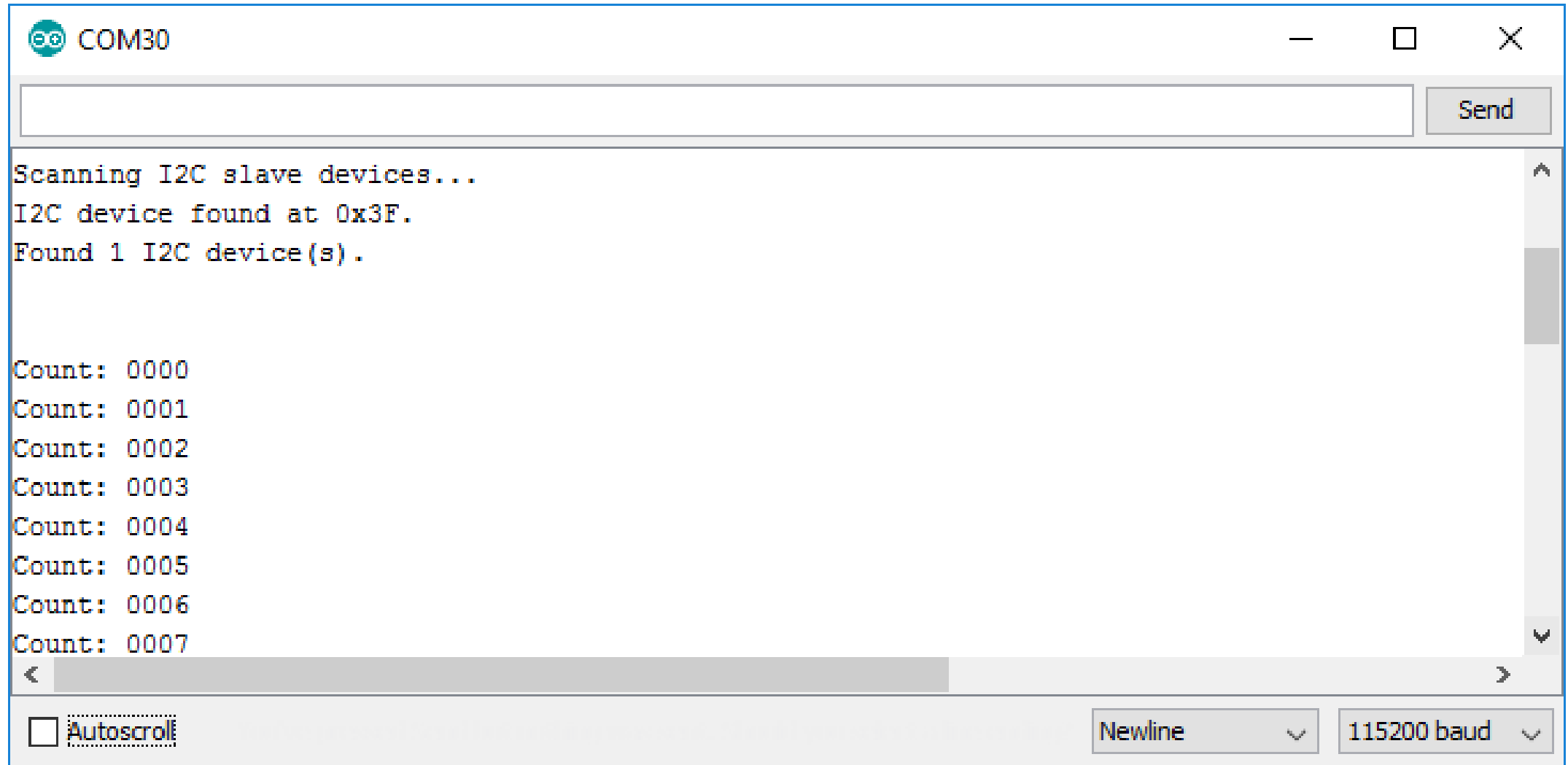
```
void setup() {  
    Serial.begin( 115200 );  
  
    Wire.begin( I2C_SDA_PIN, I2C_SCL_PIN );  
    delay(1000);  
    i2c_scan();  
    delay(1000);  
  
    lcd.begin( I2C_SDA_PIN, I2C_SCL_PIN, 400000 );  
    lcd.backlight();  
    lcd.clear();  
    lcd.setCursor(0,0); // set cursor at top-level position on the first row  
    lcd.print( F("ESP8266 Demo") );  
    lcd.setCursor(0,1); // set cursor at the start position on the second row  
    lcd.print( F("ESL KMUTNB") );  
    lcd.clear();  
    lcd.setCursor(0,0); // set cursor at top-level position on the first row  
    lcd.print( F("16x2 LCD Adapter") );  
    delay(1000);  
    ts = millis();  
}
```

esp8266_pcf8574a_lcd_lib_demo-1

```
uint16_t count = 0;

void loop() {
  if ( millis() - ts >= INTERVAL_MSEC ) {
    ts += INTERVAL_MSEC;
    sprintf_P( sbuf, PSTR("Count: %04u"), count );
    count = (count+1) % 10000;
    lcd.setCursor(0 /*col*/, 1 /*row*/);
    lcd.print( sbuf );
    Serial.println( sbuf );
  }
  delay(1);
}
```

esp8266_pcf8574a_lcd_lib_demo-1



A screenshot of a serial terminal window titled "COM30". The window has a title bar with standard minimize, maximize, and close buttons. Below the title bar is a text input field and a "Send" button. The main area of the window displays the following text:

```
Scanning I2C slave devices...  
I2C device found at 0x3F.  
Found 1 I2C device(s).  
  
Count: 0000  
Count: 0001  
Count: 0002  
Count: 0003  
Count: 0004  
Count: 0005  
Count: 0006  
Count: 0007
```

At the bottom of the window, there is a status bar containing three elements: an unchecked checkbox labeled "Autoscroll", a dropdown menu currently showing "Newline", and another dropdown menu currently showing "115200 baud".

esp8266_pcf8574a_lcd_lib_demo-2

```
#include <ESP8266WiFi.h>
#include "PubSubClient.h"
#include <Wire.h> // use the Wire library
#include "LiquidCrystal_I2C.h" // -> https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library

#define I2C_SCL_PIN    (4)    // D2 pin (SCL / GPIO-4)
#define I2C_SDA_PIN    (0)    // D3 pin (SDA / GPIO-0)
#define I2C_ADDR        (0x3F) // set the I2C address for PCF8574 LCD adapter (0x27 or 0x3F)

LiquidCrystal_I2C lcd( I2C_ADDR, 16, 2 ); // 16x2 LCD display, set I2C address

// global variables
char sbuf[64]; // used for sprintf()
uint32_t ts;   // used to save timestamp value

#define INTERVAL_MSEC (1000)
```

esp8266_pcf8574a_lcd_lib_demo-2

```
void setup() {  
    Serial.begin( 115200 );  
  
    setup_wifi();  
    client.setServer(mqtt_server, mqtt_port);  
    client.setCallback(callback);  
  
    Wire.begin( I2C_SDA_PIN, I2C_SCL_PIN );  
    delay(1000);  
    i2c_scan();  
    delay(1000);  
  
    lcd.begin( I2C_SDA_PIN, I2C_SCL_PIN, 400000 );  
    lcd.backlight();  
    lcd.clear();  
    lcd.setCursor(0,0); // set cursor at top-level position on the first row  
    lcd.print( F("IoT Workshop") );  
    delay(1000);  
    ts = millis();  
}
```

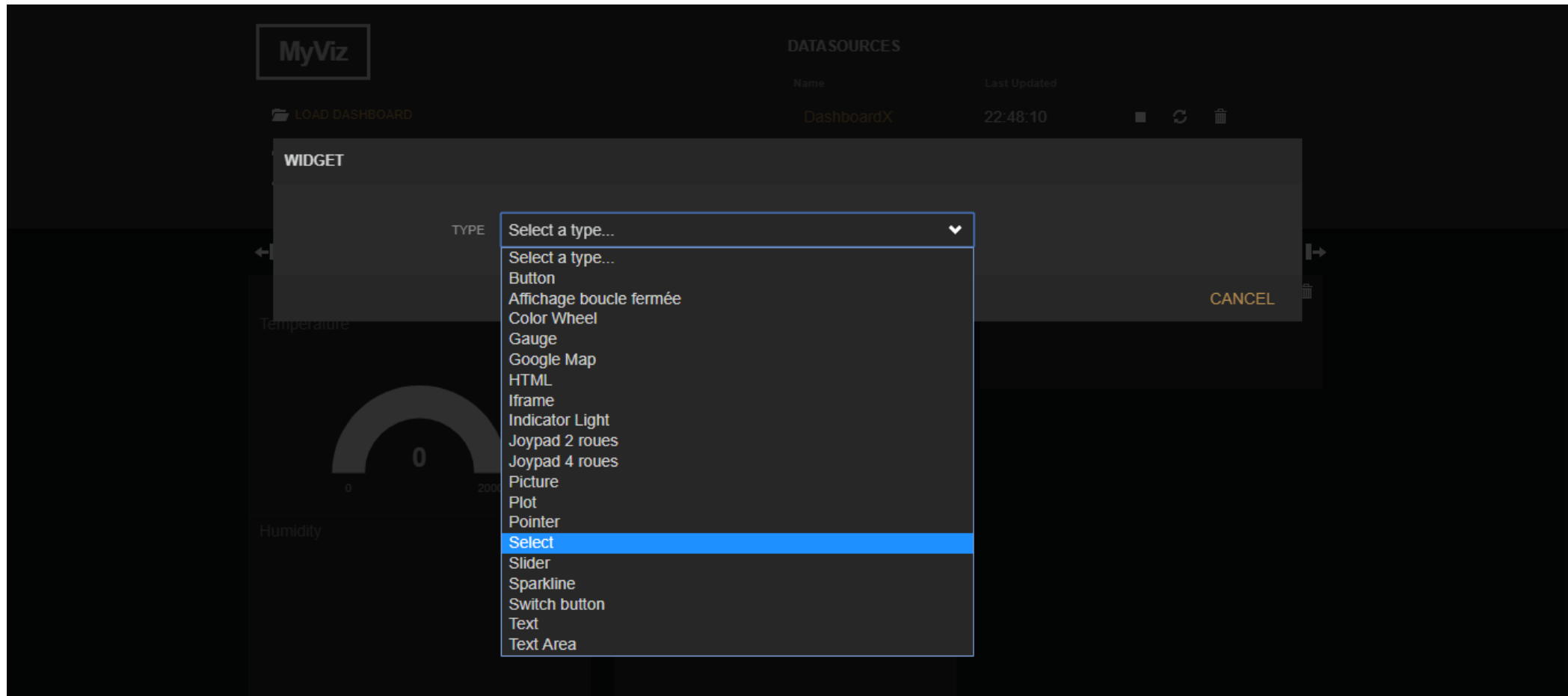
esp8266_pcf8574a_lcd_lib_demo-2

```
void loop() {  
  connectCloudMQTT();  
  if( millis() - ts >= INTERVAL_MSEC){  
    ts += INTERVAL_MSEC;  
    if (count_scroll%(text.length()+16) == 0){  
      lcd.clear();  
      lcd.setCursor(16,0); // set cursor at top-level position on the first row  
      lcd.print(text);  
    }else{  
      lcd.scrollDisplayLeft();  
    }  
    count_scroll++;  
  }  
}
```

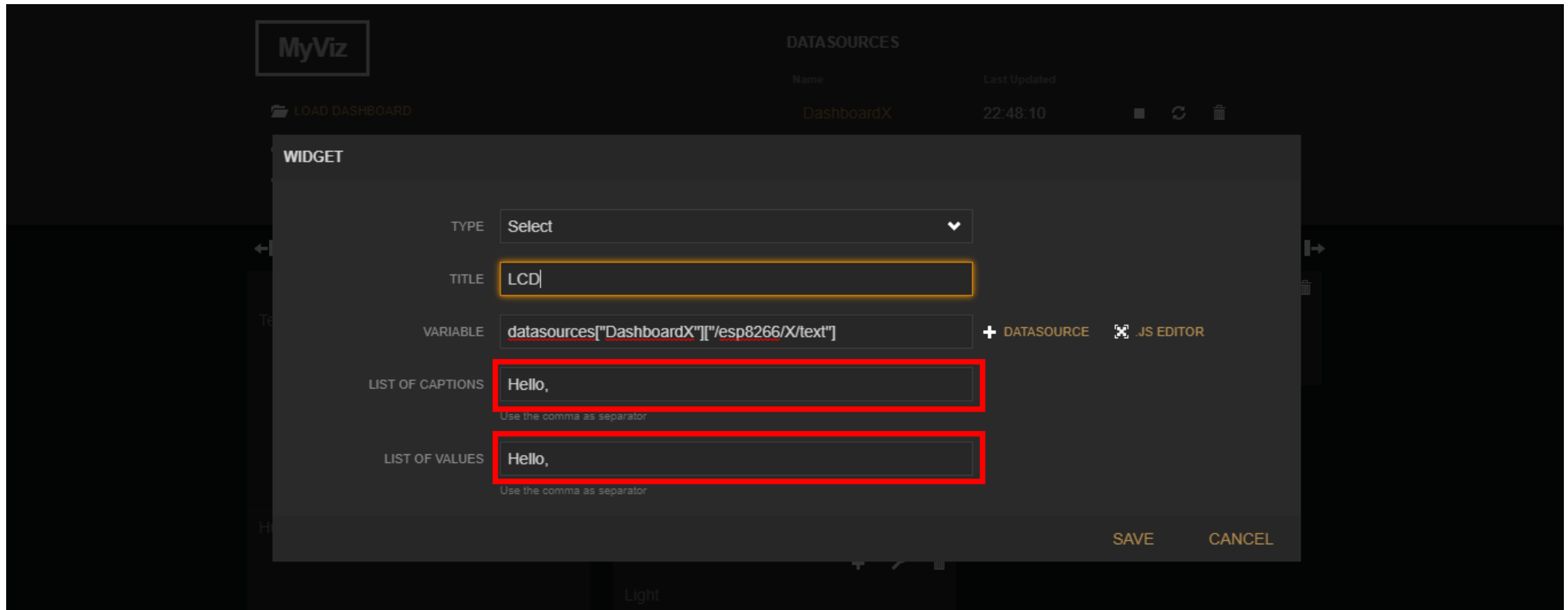
esp8266_pcf8574a_lcd_lib_demo-2

```
void callback(char* topic, byte* payload, unsigned int length) {  
    const char s[2] = "/";  
    char *token;  
    token = strtok(topic, s);  
    token = strtok(NULL, s);  
    token = strtok(NULL, s);  
    if(!strcmp(token, "text")){  
        payload[length] = '\0';  
        text = String((char*)payload);  
        Serial.println(text);  
    }  
}
```

ใช้งาน Freeboard ส่งข้อความแสดงที่ LCD



ใช้งาน Freeboard ส่งข้อความแสดงที่ LCD



ใช้งาน Freeboard ส่งข้อความแสดงที่ LCD

The screenshot displays the MyViz dashboard interface. At the top left, the 'MyViz' logo is visible. Below it, there are three buttons: 'LOAD DASHBOARD', 'SAVE DASHBOARD', and 'ADD PANE'. To the right, the 'DATASOURCES' section shows a table with columns 'Name' and 'Last Updated'. The table contains one entry, 'DashboardX', with a timestamp of '22:48:10'. Below the table is an 'ADD' button. The main dashboard area is divided into three panes. The left pane is titled 'Temperature' and shows a semi-circular gauge with a needle pointing to '0' on a scale from 0 to 2000. The right pane is titled 'WS2812 RGB LED' and shows a color wheel with a red square in the center. The bottom pane is titled 'LCD' and shows a dropdown menu with three options: 'Hello', 'Hello', and 'test123'. The 'test123' option is currently selected and highlighted in blue.

MyViz

LOAD DASHBOARD

SAVE DASHBOARD

ADD PANE

DATASOURCES

Name	Last Updated
DashboardX	22:48:10

ADD

Temperature

0 2000

Humidity

WS2812 RGB LED

LCD

Hello

Hello

test123