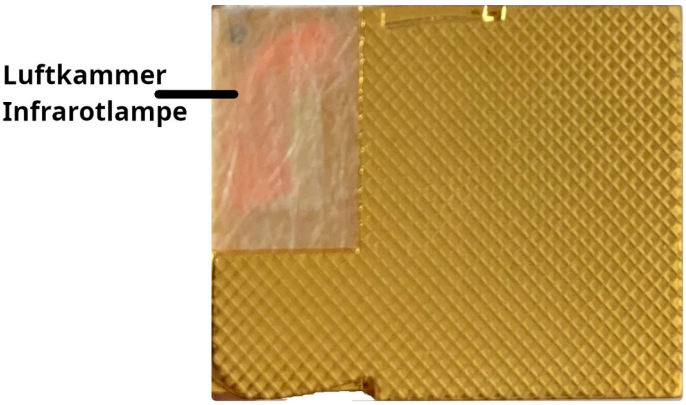
Inhaltsverzeichnis

Messprinzip von CO2-Sensoren....1
Arbeitsweise des Programms....1
Ansicht im Browser....1
Anschluss am Wemos D1....2
Kalibrierung des Sensors....2
Händische Kalibrierung:....2
Automatische Kalibrierung:....3
Mit einer Bibliothek....3
Benötigte Bauteile:....4
Der Schaltplan....4
Vorbereitungen:....5
Benötigte Bibliotheken:....5
Installation des Boards....5
Das Programm....7
Bibliotheken und Variable....7
Der setup-Teil:....8
Der loop-Teil....9
Quellen:....12

Messprinzip von CO_2-Sensoren

Der Sensor MH-Z19C misst nach dem NDIR-Prinzip (Nichtdispersiver Infrarotsensor). Er besteht aus einer Infrarot-Lampe und einem Detektor. Dazwischen befindet sich die zu messende Luft. Je mehr CO_2 sie enthält, desto mehr Infrarot Strahlung wird absorbiert und entsprechend weniger kommt am Detektor an.



Arbeitsweise des Programms

Das Programm misst mit dem Sensor DHT22 Temperatur und Luftfeuchtigkeit und mit dem MH-Z19C den CO_2-Gehalt der Luft.
Die Messdaten werden im Browser angezeigt.

Ansicht im Browser

Anschluss am Wemos D1

Kalibrierung des Sensors

Das Sensor MH-Z19C muss auf einen „Nullpunkt“ eingestellt werden. Es wird angenommen, dass draußen oder in einem gut gelüftetem Raum die CO^2-Konzetration 400 ppm (parts per million) beträgt. Dieser Wert wird als „Nullwert“ festgelegt. Meine Messungen haben gezeigt, dass dieser Wert in der Regel nur leicht überschritten wird.

Letzte Messung: Dienstag, 25.04.2023 08:16:10 Uhr

Temperatur:

12,70 °C

Luftfeuchtigkeit:

66,80 %

CO_2-Gehalt:

405 ppm

Für die Kalibrierung gibt es drei Möglichkeiten:

Händische Kalibrierung:

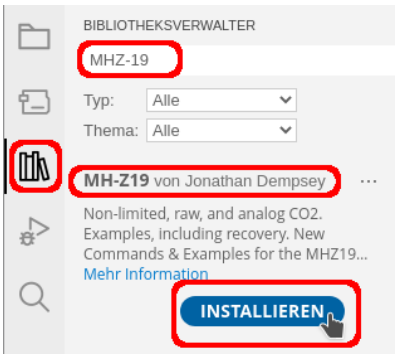
Der Sensor muss mindestens 20 Minuten in einer gut gelüfteten Umgebung (am besten draußen) Messungen durchführen, dann musst du den HD Pin wird für mehr als 7 Sekunden mit GND verbinden.

Automatische Kalibrierung:

Die automatische Kalibrierung kann nur dann funktionieren, wenn der Sensor über längere Zeit im Einsatz ist und sich zwischendurch immer wieder in einem gut durchlüftetem Raum befindet. Der „Nullpunkt“ von 400 ppm muss dann näherungsweise erreicht werden.

Mit einer Bibliothek

Installiere zunächst die Bibliothek MH-Z19.



In den Beispielen zur Bibliothek befindet sich das Programm Calibration, ich habe es ein wenig angepasst.

```
# include <Arduino.h>
# include <MHZ19.h>
# include <SoftwareSerial.h>

// RX/TX Pins zuordnen
// TX MH-Z19C auf D6, RX-Pin MH-Z19C auf D7 (überkreuz)
# define RX D6
# define TX D7

// Name des Moduls MH-Z19
MHZ19 MHZCO2;

// SoftwareSerial -> Name zuordnen
SoftwareSerial MHZSerial(RX, TX);

unsigned long Wartezeit = 0;

void setup()
{
  Serial.begin(9600);
  delay(500);

  MHZSerial.begin(9600);
  MHZCO2.begin(MHZSerial);

  // automatische Kalibrierung ausschalten
  MHZCO2.autoCalibration(false);

  Serial.println("20 Minuten warten um die Messwerte zu stabilisieren...");
  Wartezeit = 12e5;
  delay(Wartezeit);
```

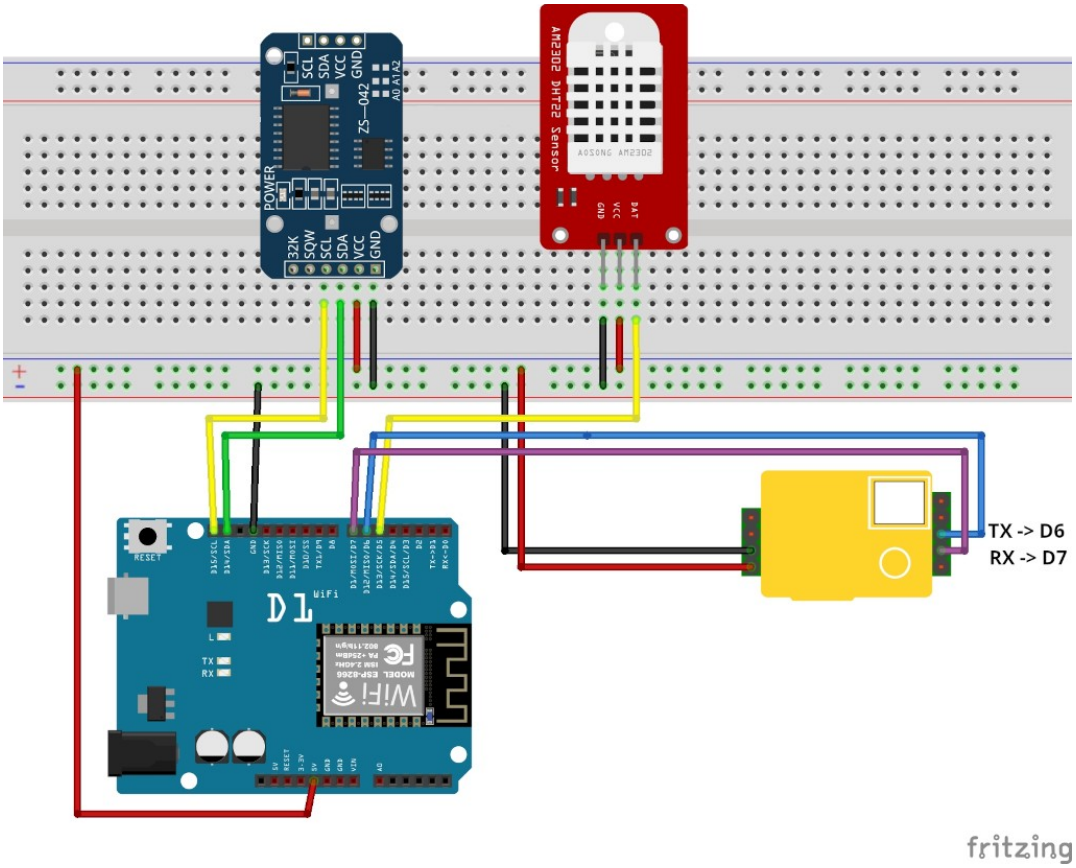
```
  Serial.println("Kalibriere..");

  // Sensor kalibrieren
  MHZCO2.calibrate();
}

void loop()
{
  if (millis() - Wartezeit >= 2000)
  {
    int CO2;
    CO2 = MHZCO2.getCO2();

    Serial.print("CO2 (ppm): ");
    Serial.println(CO2);

    Wartezeit = millis();
  }
}
```

Nach der Kalibrierung darfst du das Programm nicht erneut hochladen!

Benötigte Bauteile:

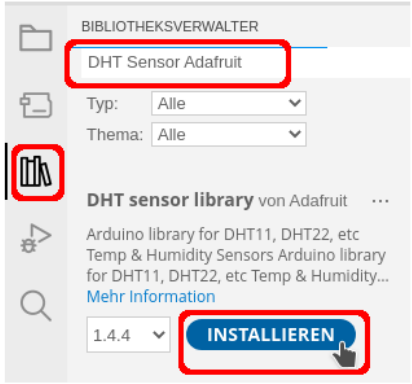
- CO_2-Sensor MH-Z19C
- RTC-Modul DS3231
- Temperatur-/Feuchtigkeitssensor DHT22
- Leitungsdrähte

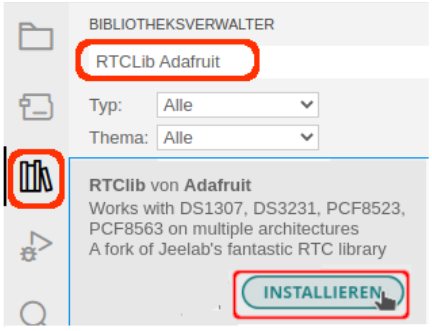
Der Schaltplan

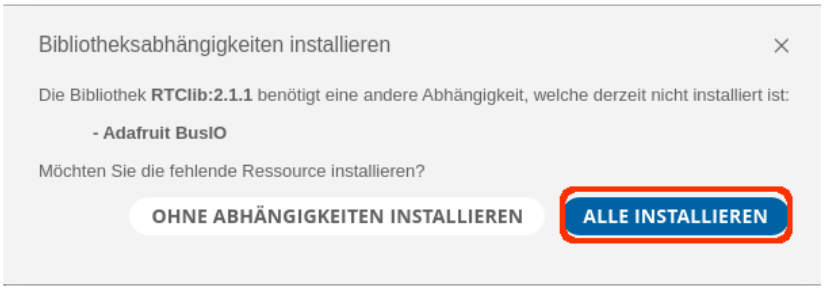


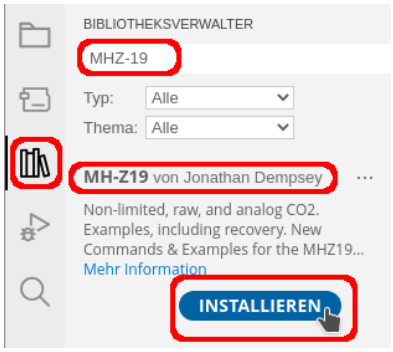
Vorbereitungen:

Benötigte Bibliotheken installieren:







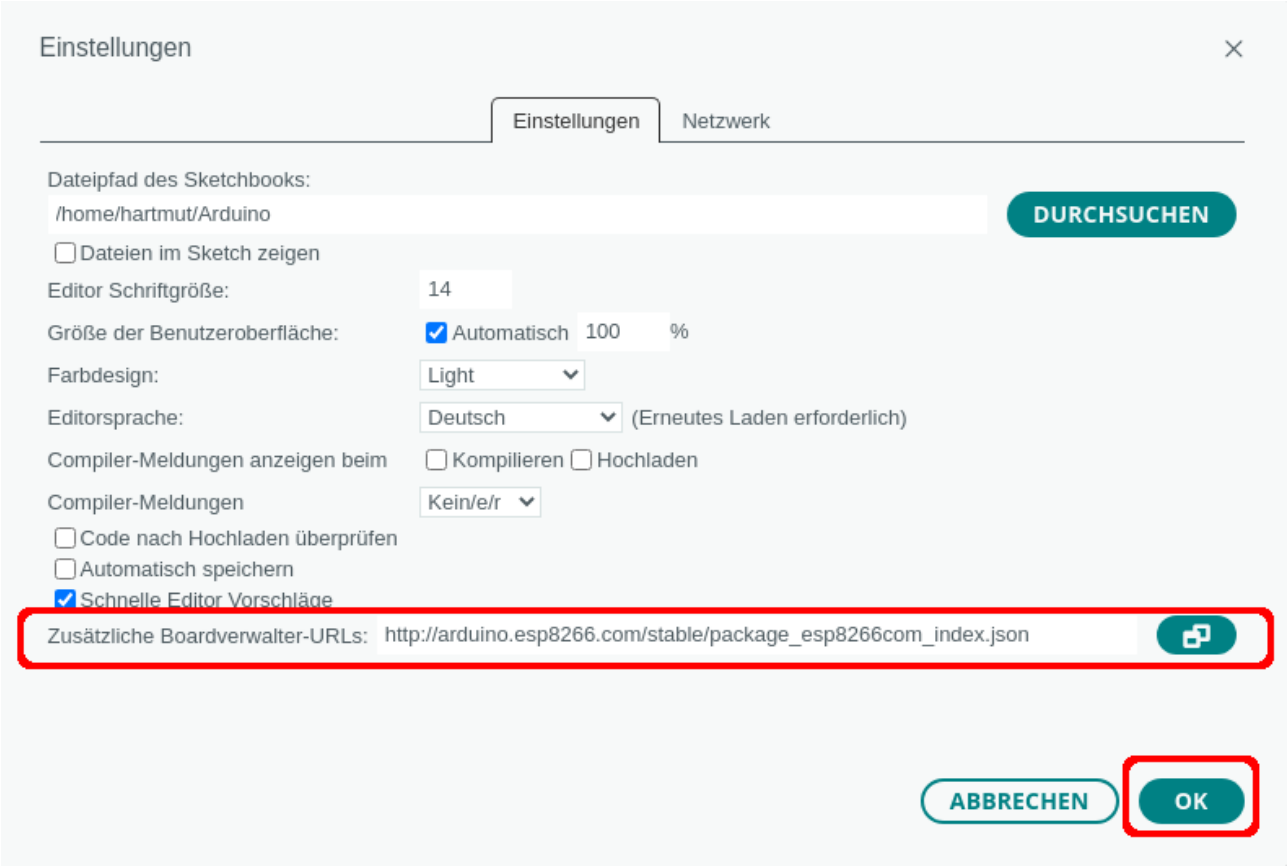


Installation des Boards

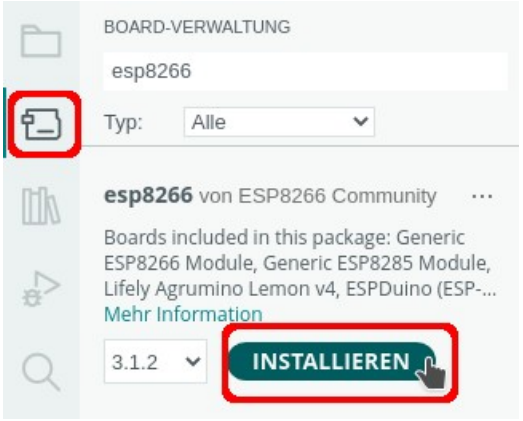
Unter Datei → Einstellungen muss du ein zusätzliches Board installieren.

Trage unter Zusätzliche Boardverwalter-URLs das esp8266 ein:

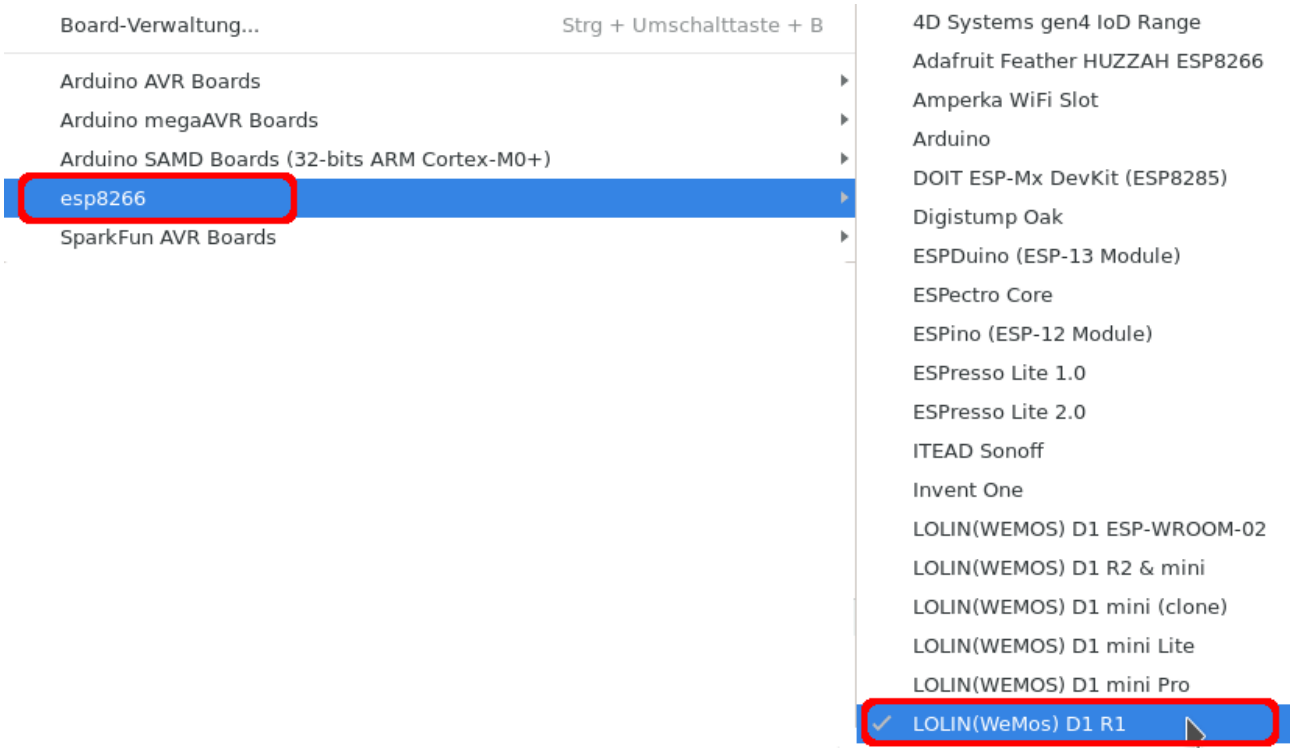
http://arduino.esp8266.com/stable/package_esp8266com_index.json



Anschließend kann das Board esp8266 ausgewählt werden:



Zum Schluss musst du noch unter der Boardverwaltung unter esp8266 das Board LOLIN (Wemos) D1 R1 auswählen.



Das Programm

Bibliotheken und Variable

```
# include <ESP8266WebServer.h>
# include <Arduino.h>
# include <MHZ19.h>
# include <SoftwareSerial.h>
# include <RTClib.h>
# include <DHT.h>

// Router: Name des Routers
// Passwort: WLAN-Passwort
char Router[] = "FRITZ!Box 7590 LB";
char Passwort[] = "xxxxxxx";

// festeIP = false -> IP-Adresse über DHCP vergeben
// festeIP = true -> IP Gateway und Subnetz festlegen
bool festeIP = false;

// IP, Gateway und Subnetz festlegen
IPAddress ip(192, 168, 1, 200);
IPAddress gateway(192, 168, 1, 1);
IPAddress subnet(255, 255, 255, 0);

WiFiServer server(80);
WiFiClient client;
```

```
// Pin des DHT22
int SENSOR_DHT = D5;

// Sensortyp festlegen
// DHT22
#define SensorTyp DHT22

// Sensor DHT einen Namen zuweisen
DHT dht(SENSOR_DHT, SensorTyp);

// Name des RTC-Moduls (rtc)
RTC_DS3231 rtc;

// RX/TX Pins zuordnen
// TX MH-Z19C auf D6, RX-Pin MH_Z19C auf D7 (überkreuz)
# define RX D6
# define TX D7

// Name des Moduls MH-Z19
MHZ19 MHZCO2;

// SoftwareSerial -> Name zuordnen
SoftwareSerial MHZSerial(RX, TX);

unsigned long VerstricheneZeit = 0;
```

Der setup-Teil:

```
void setup()
{
  Serial.begin(9600);
  delay(500);

  // WiFi starten
  WiFi.begin(Router, Passwort);

  if (festeIP) WiFi.config(ip, gateway, subnet);

  // verbinden
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  server.begin();

  // IP des Servers/des verbunden Computers anzeigen
  Serial.print("Server: ");
  Serial.println(WiFi.SSID());
```

```
  // IP des Wemos D1 anzeigen
  if (festeIP) Serial.print("Statische IP Adresse: ");
  else Serial.print("IP Adresse DHCP: ");
  Serial.println(WiFi.localIP());

  MHZSerial.begin(9600);
  MHZCO2.begin(MHZSerial);

  // automatische Kalibrierung bei Dauerbetrieb einschalten
  MHZCO2.autoCalibration(true);

  // RTC-Modul starten
  rtc.begin();

  // Datum/Zeit einmalig setzen, beim nächsten Starten auskommentieren
  // rtc.adjust(DateTime(2023, 4, 23, 8, 50, 30));

  // Sensor DHT starten
  dht.begin();
}
```

Der loop-Teil

```
void loop()
{
  VerstricheneZeit = millis();
  client = server.available();
  if (client)
  {
    String SchaltungLesen = "";
    while (client.connected())
    {
      if (client.available())
      {
        char Zeichen = client.read();

        if (Zeichen == '\n')
        {
          if (SchaltungLesen.length() == 0) {

            // HTTP-Anforderung senden
            client.println("HTTP/1.1 200 OK");
            client.println("Content-Type: text/html");

            // Leerzeile zwingend erforderlich
            client.println();

            /*
              HTML-Seite aufbauen
              die folgenden Anweisungen müssen mit print oder println gesendet werden
              println "verschönert" den Quelltext
              " muss mit \" maskiert werden
            */
            client.println("<!doctype html>");
```

```
client.println("<html>");
client.println("<body>");

// alle 60 Sekunden aktualisieren mit meta-Tag
client.println("<meta http-equiv=\"refresh\" content=\"60\">");

client.println("<h1> Temperatur, Luftfeuchtigkeit und CO&sup2;-Gehalt in der Luft messen</h1>");
client.println("<hr />");

client.print("<h2>Letzte Messung: ");
DateTime aktuell = rtc.now();
char Datum[] = "DD.MM.YYYY ";
char Zeit[] = "hh:mm:ss Uhr";

switch (aktuell.dayOfTheWeek())
{
  case 0:
    client.print(F("Sonntag"));
    break;
  case 1:
    client.print(F("Montag"));
    break;
  case 2:
    client.print(F("Dienstag"));
    break;
  case 3:
    client.print(F("Mittwoch"));
    break;
  case 4:
    client.print(F("Donnerstag"));
    break;
  case 5:
    client.print(F("Freitag"));
    break;
  case 6:
    client.print(F("Samstag"));
    break;
}

client.print(", ");
client.print(aktuell.toString(Datum));
client.println(aktuell.toString(Zeit));
client.println("</h2>");
client.println("<hr />");

// Daten lesen
String AnzeigeTemperaturDHT = String(dht.readTemperature());
AnzeigeTemperaturDHT.replace(".", ",");

String AnzeigeLuftfeuchtigkeit = String(dht.readHumidity());
AnzeigeLuftfeuchtigkeit.replace(".", ",");
client.print("<b>Temperatur:</b><blockquote>");
client.println(AnzeigeTemperaturDHT + " &deg;C</blockquote>");
client.println("<br>");
client.print("<b>Luftfeuchtigkeit:</b><blockquote>");
client.println(AnzeigeLuftfeuchtigkeit + " %</blockquote>");
```

```
            // kurze Pause
            delay(500);

            // CO2-Wert ermitteln
            int CO2 = MHZCO2.getCO2();

            // kurze Pause
            delay(500);

            client.print("<b>CO&sup2;-Gehalt:</b><blockquote>");
            client.println(String(CO2) + " ppm</blockquote>");
            client.println("<form>");
            client.println(F("<hr />"));

            // Button formatieren
            client.print("<input style=\"font-size:16pt; font-weight:bold;");
            client.print("background-color:#55A96B;");
            client.print("display:block; cursor:pointer;\"type=\"button\"");
            client.println(" onClick=\"location.href='WiFi.localIP()'\" value=\"aktualisieren\">");
            client.println("</form>");
            client.println("<hr />");

            // IPs anzeigen
            client.print(F("<b>Eigene IP: "));
            client.print(client.remoteIP());
            client.print(F("</b>"));
            client.print(F("<br><b>IP Klient: "));
            client.print(WiFi.localIP());
            client.print(F("</b>"));
            client.println("</b>");
            client.println("</body>");
            client.print("</html>");

            // HTTP-Antwort endet mit neuer Zeile
            client.println();

            // Seite vollständig geladen -> loop verlassen
            break;
          }
          else
          {
            SchaltungLesen = "";
          }
        }

        // bei einem anderen Zeichen als return (\r)
        // -> Zeichen dem String SchaltungLesen hinzufügen
        else if (Zeichen != '\r')
        {
          SchaltungLesen += Zeichen;
        }
      }
    }
    client.stop();
  }
}
```

Quellen:

MH-Z19 Sensoren bei Wolles Elektronikkiste

MHZ-19B bei Unsinnsbasis

Datenblatt MH-Z19C bei Winsen Eletronics

Beitrag bei Wikipedia zum NDIR-Prinzip

Hartmut Waller letzte Änderung: 14.05.23