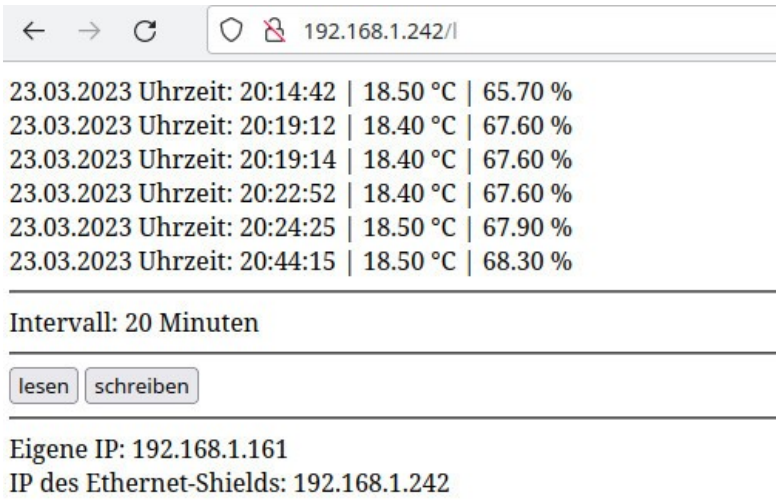
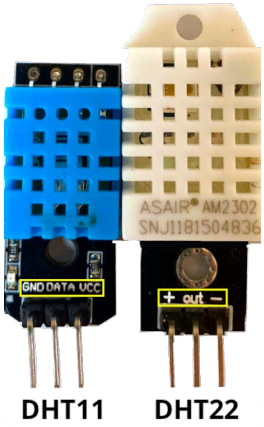
Der Temperatursensor DHT22 (oder DHT11) misst Temperatur und Luftfeuchtigkeit, die Daten werden auf der SD-Karte des Ethernet-Shields gespeichert und können im Browser verarbeitet werden.

So sieht es aus:

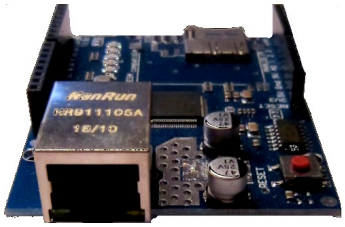




Pinbelegung:

DHT11
GND Daten 5V

DHT22
5V Daten GND

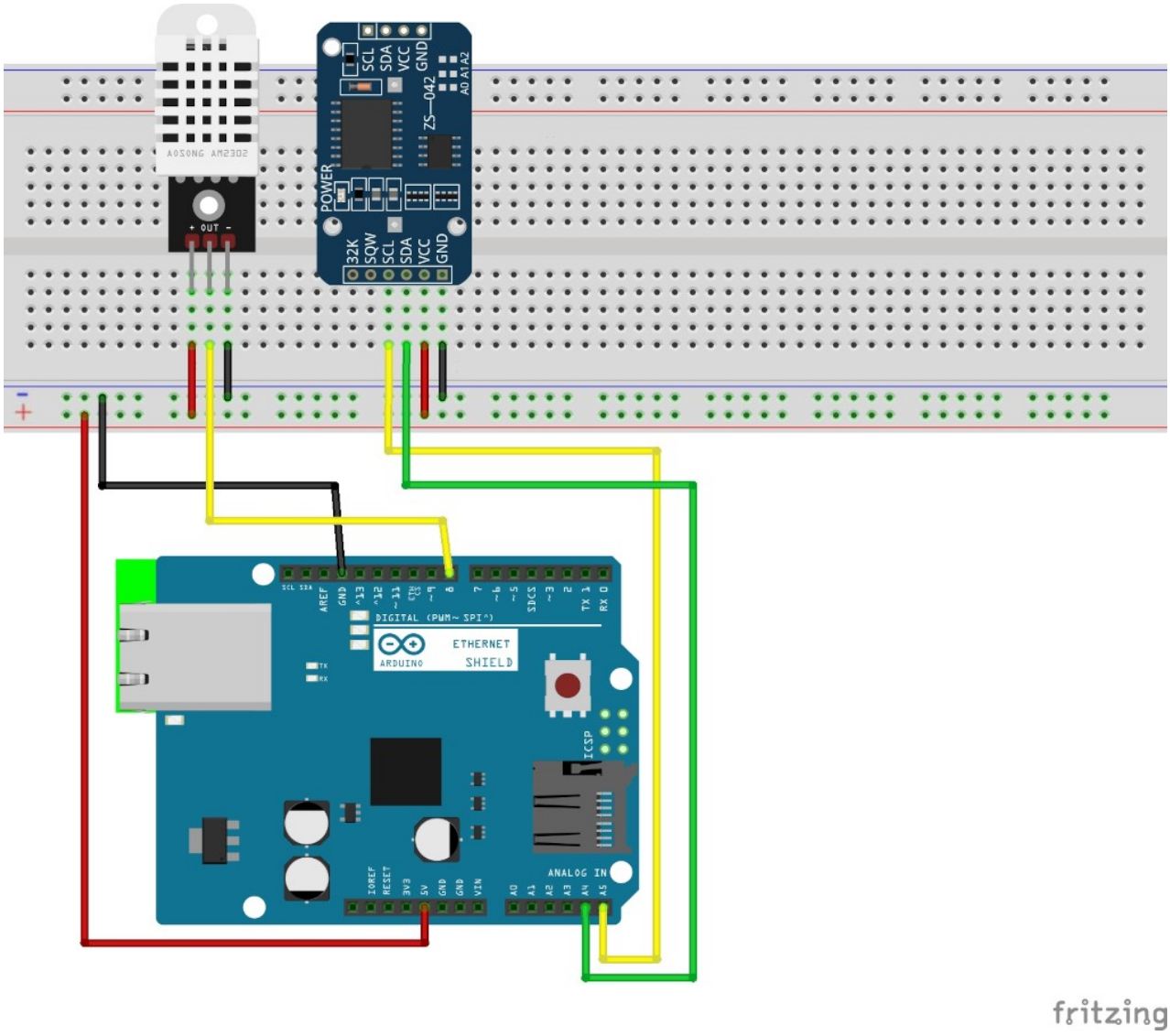


Für diese Anleitung benötigst du ein sogenanntes „Shield“, eine Platine, die einfach auf den Arduino aufgesteckt wird. Auf ihr befindet sich ein LAN-Anschluss (RJ45). Alle digitalen und analogen Anschlüsse stehen auch weiterhin zur Verfügung.

Benötigte Bauteile:

- Temperatursensor DHT22/DHT11
- Ethernet-Shield
- RTC-Modul
- Leitungsdrähte

Baue die Schaltung auf



Für das Programm brauchst du eine freie IP-Adresse und eine freie MAC-Adresse in deinem lokalen Netzwerk.

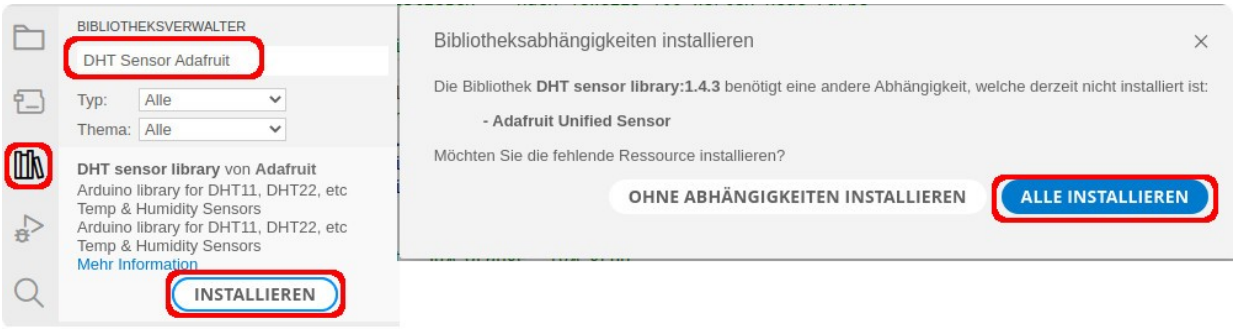
Im Regelfall befindet sich in einem lokalen Netzwerk ein DHCP-Server, der jedem Gerät im Netzwerk automatisch eine IP-Adresse zuteilt. Wenn du dich für diese Alternative entscheidest, musst die im Seriellen Monitor anzeigte Adresse verwenden.
Die MAC-Adresse ist die Hardware-Adresse jeder einzelnen Netzwerkschnittstelle (LAN oder WLAN), mit der jedes Gerät im Netzwerk eindeutig identifiziert werden kann.
Sie besteht aus sechs Bytes in hexadezimaler Schreibweise, die durch „:“ oder „-“ getrennt werden.

Du kannst die im Programm verwendete „erfundene“ MAC-Adresse übernehmen: Die Gefahr, dass sich ein Gerät mit der gleichen MAC-Adresse im Netzwerk befindet, ist äußerst gering.

Im Seriellen Monitor siehst du die IP-Adresse des Ethernet-Shields:

Benötigte Bibliotheken:

Sketch → Bibliothek einbinden → Bibliotheken verwalten



Binde die benötigten Bibliotheken ein und definiere die Variablen:

```
# include <Ethernet.h>
# include <RTClib.h>
# include <DHT.h>
# include <SD.h>

// Bezeichnung der Textdatei
File Messung;

// Datenpin für das SD-Kartenmodul
int DatenPin = 4;

// Name des RTC-Moduls
RTC_DS3231 rtc;
```

```
// Sensorpin
int SENSOR_DHT = 8;

// Sensortyp festlegen
// DHT22
# define SensorTyp DHT22

// DHT11
// # define SensorTyp DHT11

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

// MAC-Adresse und IP definieren
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};

// festeIP = false -> IP-Adresse über DHCP vergeben
bool festeIP = false;

// feste IP
IPAddress ip(192, 168, 1, 200);

// Name des Servers vergeben
EthernetServer server(80);

// Intervall in Millisekunden
// 600000 = 10 Minuten
// 3600000 = 60 Minuten
const long Intervall = 1200000;
const unsigned int Minuten = Intervall / 60000;

// Start der Zeitmessung
unsigned long StartZeit = 0;
```

Der setup-Teil. Beachte die Kommentare.

```
void setup()
{
  // RTC-Modul starten
  rtc.begin();
  /*
    wenn Datum und Zeit nicht korrekt -> Datum/Zeit setzen
    Jahr, Monat, Tag, Stunde, Minute, Sekunde
    keine führende 0 setzen
    Beispiel:
    rtc.adjust(DateTime(2023, 3, 10, 7, 2, 30));
  */

  Serial.begin(9600);
  delay(500);

  // Ethernet starten feste IP
  if (festeIP) Ethernet.begin(mac, ip);
```

```
  // Ethernet starten DHCP
  else Ethernet.begin(mac);

  // Server starten
  server.begin();

  // zur Kontrolle IP-Adresse im Seriellen Monitor anzeigen
  // localIP -> Adresse, die im Browser eingegeben wird
  Serial.print(F("IP des Ethernet-Shields: "));
  Serial.println(Ethernet.localIP());

  // SD-Karte starten
  SD.begin(DatenPin);

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

Der loop-Teil. Beachte die Kommentare.

```
void loop()
{
  String Auswertung = "";

  // Variablen für Temperatur und Luftfeuchtigkeit
  float Temperatur;
  float Luftfeuchtigkeit;

  // auf Clienten warten ...
  EthernetClient Client = server.available();

  // abgelaufene Zeit mit millis() ermitteln
  unsigned long VerstricheneZeit = millis();

  // Zeit messen und Daten speichern, wenn das Zeitintervall erreicht ist
  if (VerstricheneZeit - StartZeit >= Intervall)
  {
    // StartZeit zurücksetzen
    StartZeit = VerstricheneZeit;

    // Temperatur, Luftfeuchtigkeit messen
    Temperatur = dht.readTemperature();
    Luftfeuchtigkeit = dht.readHumidity();

    // aktuelle Zeit ermitteln
    DateTime aktuell = rtc.now();

    // in Datei speichern
    Messung = SD.open("Messung.txt", FILE_WRITE);
```

```
  if (Messung)
  {
    // Datum schreiben
    char Datum[] = "DD.MM.YYYY";
    Messung.print(aktuell.toString(Datum));

    // Zeit schreiben
    char Zeit[] = " Uhrzeit: hh:mm:ss";
    Messung.println(aktuell.toString(Zeit));

    // Messdaten schreiben
    Messung.print(F(" | "));
    Messung.println(Temperatur);
    Messung.print(F(" &deg;C | "));
    Messung.println(Luftfeuchtigkeit);
    Messung.print(F("%"));
    Messung.println(F("<br>"));

    // Datei schließen
    Messung.close();
  }
}

// neue Anfrage
if (Client)
{
  // solange der Client verbunden ist ...
  while (Client.connected())
  {
    if (Client.available())
    {
      char Zeichen = Client.read();

      /*
        einegegebene Zeichen zum String zusammensetzen
        der String enthält eines der Zeichen (l, s, e)
        Beispiel lesen:
        GET /l HTTP/1.1
      */
      Auswertung += Zeichen;

      // \n =return: Seite vom Clienten vollständig geladen
      if (Zeichen == '\n')
      {
        // HTTP-Anforderung senden
        Client.println(F("HTTP/1.1 200 OK"));
        Client.println(F("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
          (jede Anweisung in einer eigenen Zeile)
        */
        Client.println(F("<!doctype html>"));
        Client.println(F("<html>"));
        Client.println(F("<body>"));

        // String Auswertung lesen
        // indexOf -> Vorhandensein eines Zeichens prüfen
        // Daten speichern
        if (Auswertung.indexOf("s") > 0)
        {
          Temperatur = dht.readTemperature();
          Luftfeuchtigkeit = dht.readHumidity();
          DateTime aktuell = rtc.now();

          Messung = SD.open("Messung.txt", FILE_WRITE);

          // wenn die Datei vorhanden ist
          if (Messung)
          {
            char Datum[] = "DD.MM.YYYY";
            Messung.print(aktuell.toString(Datum));

            // Zeit schreiben
            char Zeit[] = " Uhrzeit: hh:mm:ss";
            Messung.println(aktuell.toString(Zeit));
            Messung.print(F(" | "));

            // Temperatur und Luftfeuchtigkeit schreiben
            Messung.println(Temperatur);
            Messung.print(F(" &deg;C | "));
            Messung.println(Luftfeuchtigkeit);
            Messung.print(F("%"));
            Messung.println(F("<br>"));

            // Datei schließen
            Messung.close();
          }
        }

        // Datei lesen
        if (Auswertung.indexOf("l") > 0)
        {
          Messung = SD.open("Messung.txt");
```

```
            if (Messung)
            {
              // solange sich Zeilen in der Datei befinden ...
              while (Messung.available())
              {
                // ... werden sie gelesen und auf der Webseite ausgegeben
                Client.write(Messung.read());
              }

              // Datei schließen
              Messung.close();
            }
          }

          // Zeitintervall der Messungen anzeigen
          Client.println(F("<hr />"));
          Client.println(F("Intervall: "));
          Client.println(Minuten);
          Client.println(F("Minuten<br> "));
          Client.println(F("<hr />"));

          // Buttons anzeigen
          Client.print(F("<input type='button' "));
          Client.println(F(" onClick=\"location.href='l'\""));
          Client.println(F(" value='lesen'>"));
          Client.print(F("<input type='button' "));
          Client.println(F(" onClick=\"location.href='s'\""));
          Client.println(F(" value='schreiben'>"));

          Client.println(F("<hr />"));

          // IPs anzeigen
          Client.print(F("Eigene IP: "));
          Client.print(Client.remoteIP());
          Client.print(F("<br>IP des Ethernet-Shields: "));
          Client.print(Ethernet.localIP());
          Client.println(F("</body>"));
          Client.println(F("</html>"));

          // Zeit, um die Antwort zu übertragen
          delay(1);

          // Verbindung beenden
          Client.stop();
        }
      }
    }
  }
}
```