



Ziel des Projekts

Die mit dem Temperatursensor DHT22 gemessenen Werte sollen mit einem Arduino WiFi im Browser angezeigt werden.

← → ↺ 🏠 🔒 192.168.1.160/WiFi.localIP() ☆

Temperatur und Luftfeuchtigkeit messen

Letzte Messung: Dienstag, 07.05.2024 13:28:33 Uhr

Temperatur:

22,10 °C

Luftfeuchtigkeit:

65,70 %

Gespeicherte Werte:

1: Dienstag, 07.05.2024 13:23:29 Uhr	Temperatur: ► 22,20 °C	Luftfeuchtigkeit: ► 65,70 %
2: Dienstag, 07.05.2024 13:24:30 Uhr	Temperatur: ► 22,20 °C	Luftfeuchtigkeit: ► 65,70 %
3: Dienstag, 07.05.2024 13:25:30 Uhr	Temperatur: ► 22,20 °C	Luftfeuchtigkeit: ► 65,70 %
4: Dienstag, 07.05.2024 13:26:30 Uhr	Temperatur: ► 22,10 °C	Luftfeuchtigkeit: ► 65,70 %
5: Dienstag, 07.05.2024 13:27:30 Uhr	Temperatur: ► 22,10 °C	Luftfeuchtigkeit: ► 65,70 %
6: Dienstag, 07.05.2024 13:27:58 Uhr	Temperatur: ► 22,10 °C	Luftfeuchtigkeit: ► 65,70 %
7: Dienstag, 07.05.2024 13:28:01 Uhr	Temperatur: ► 22,10 °C	Luftfeuchtigkeit: ► 65,70 %
8: Dienstag, 07.05.2024 13:28:03 Uhr	Temperatur: ► 22,10 °C	Luftfeuchtigkeit: ► 65,70 %
9: Dienstag, 07.05.2024 13:28:26 Uhr	Temperatur: ► 22,10 °C	Luftfeuchtigkeit: ► 65,70 %
10: Dienstag, 07.05.2024 13:28:33 Uhr	Temperatur: ► 22,10 °C	Luftfeuchtigkeit: ► 65,70 %

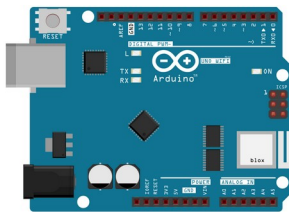
aktualisieren

Eigene IP: 192.168.1.12

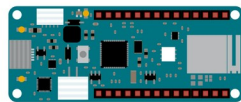
IP Arduino: 192.168.1.160

Die Hardware

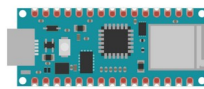
Für diese Anleitung benötigst du einen Arduino mit WiFi.



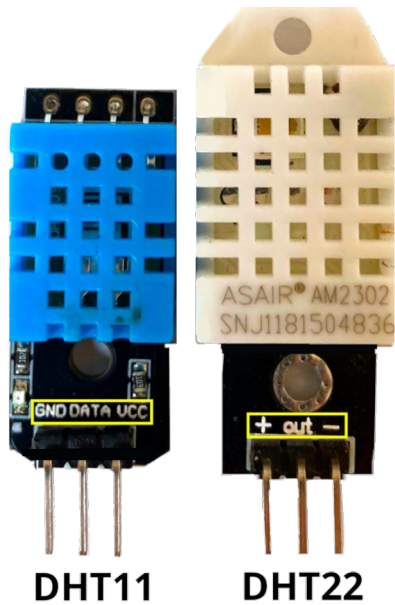
Arduino WiFi Rev 2



Arduino MKR WiFi 1010



Arduino Nano 33 IoT



DHT11

DHT22

Beispiele für DHT11/DHT22 Sensoren

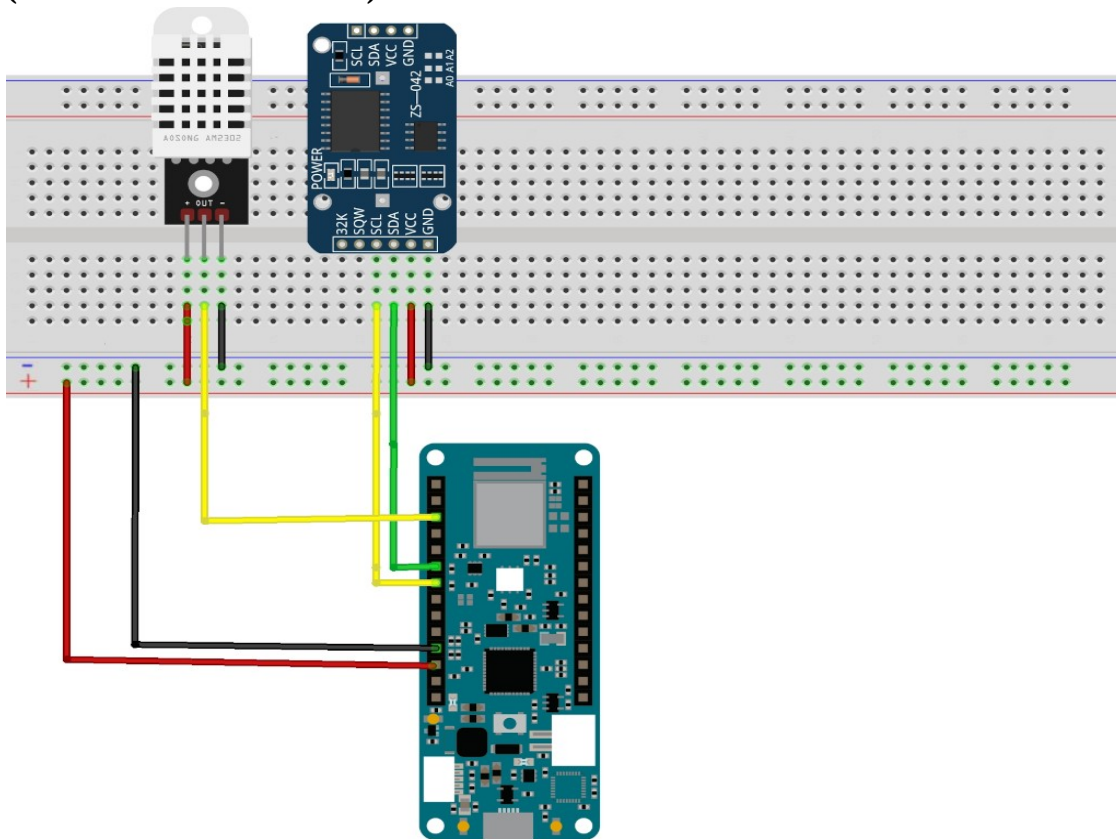
Benötigte Bauteile

- ➔ Temperatursensor DHT11 oder DHT22
- ➔ RTC-Modul DS3231
- ➔ Leitungsdrähte

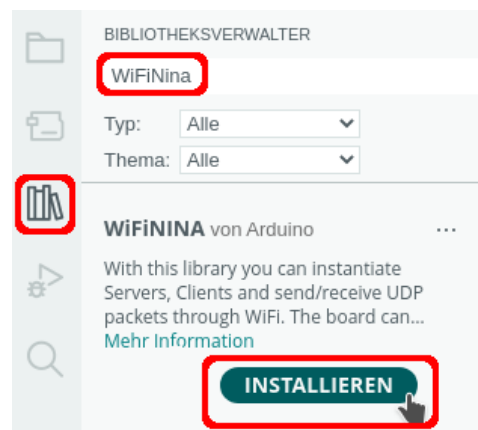
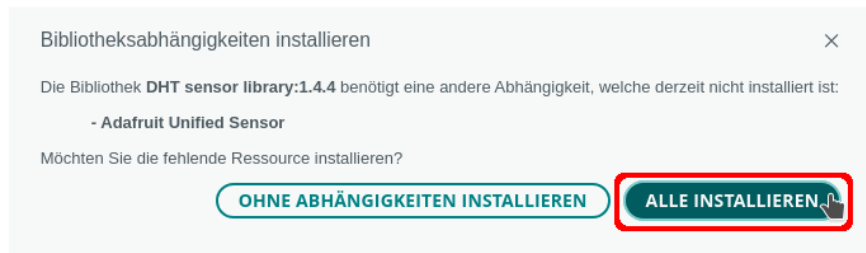
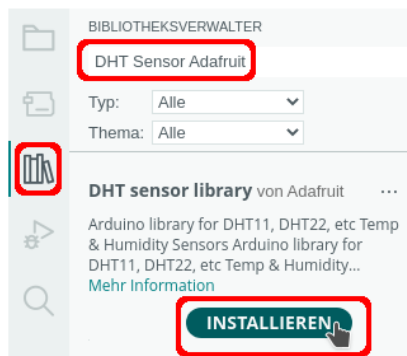
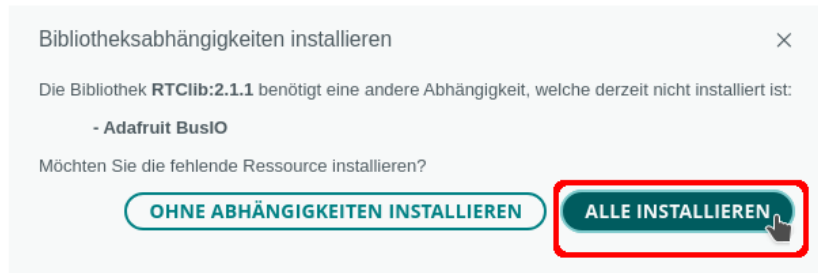
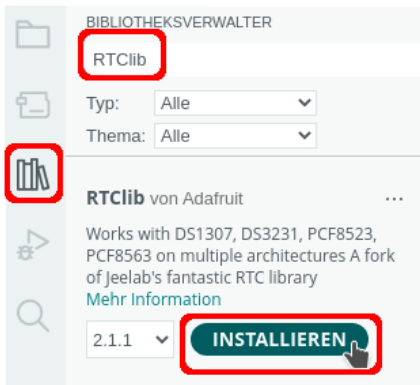
Der Schaltplan

Der Schaltplan

(Arduino MKR WiFi 1010)



Benötigte Bibliotheken



Das Programm

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. Im Programm wird eine über DHCP vergebene Adresse verwendet.



Binde die benötigte Bibliotheken ein und definiere die Variablen.

```
#include "DHT.h"
#include "RTClib.h"
#include "WiFiNINA.h"

// Name des RTC-Moduls
RTC_DS3231 rtc;

// DHT22
#define DHTTYPE DHT22
#define DHTPIN 8

// Sensor zuordnen
DHT dht(DHTPIN, DHTTYPE);

// Router: Name des Routers
// Passwort: WLAN-Passwort
char Router[] = "Router-SSID";
char Passwort[] = "xxxxxxx";

WiFiServer server(80);
WiFiClient client = server.available();

byte Zaehler = 0;

// maximale Anzahl gespeicherter Werte
const byte AnzahlDaten = 11;

// Arrays für die Speicherung
String gespeicherteTemperatur[AnzahlDaten];
String gespeichertesDatum[AnzahlDaten];
String gespeicherteLuftfeuchtigkeit[AnzahlDaten];
```

Der setup-Teil:

```
void setup()
{
    // RTC-Modul starten
    rtc.begin();

    /*
        wenn Datum und Zeit nicht korrekt -> Datum/Zeit setzen
        Jahr, Monat, Tag, Stunde, Minute, Sekunde
        rtc.adjust(DateTime(2025, 11, 13, 13, 30, 30));
    */

    // rtc.adjust(DateTime(2025, 11, 13, 13, 30, 30));

    // dht starten
    dht.begin();

    Serial.begin(9600);
    while (!Serial);
    delay(1000);
```



```
Serial.print("Verbindung aufbauen mit ");
Serial.println(Router);

// WiFi starten
WiFi.begin(Router, Passwort);
while (WiFi.status() != WL_CONNECTED)
{
    delay(200);
    Serial.print(".");
}
server.begin();
Serial.print("Router: ");
Serial.println(WiFi.SSID());

IPAddress ip = WiFi.localIP();
Serial.print("IP Adresse Arduino: ");
Serial.println(ip);

DateTime aktuell = rtc.now();

// Datum und Zeit holen
char Datum[] = "DD.MM.YYYY ";
char Zeit[] = "hh:mm:ss Uhr";

Serial.print(aktuell.toString(Datum));
Serial.println(aktuell.toString(Zeit));
}
```

Im Seriellen Monitor werden die IP des Arduinos und das aktuelle Datum und die aktuelle Zeit angezeigt. Diese Adresse wird in einem Browser eingegeben.



Da die IP-Adresse über DHCP übergeben wird, kann sie nach einem erneuten Hochladen anders sein.



Der loop-Teil. Beachte die Kommentare.

```
void loop()
{
    float Temperatur;
    float Luftfeuchtigkeit;
    String Nummer;
    String AnzeigeTemperatur;
    String AnzeigeLuftfeuchtigkeit;

    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 und Luftfeuchtigkeit 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("Sonntag");
    break;
  case 1:
    client.print("Montag");
    break;
  case 2:
    client.print("Dienstag");
    break;
  case 3:
    client.print("Mittwoch");
    break;
  case 4:
    client.print("Donnerstag");
    break;
  case 5:
    client.print("Freitag");
    break;
  case 6:
    client.print("Samstag");
    break;
}

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

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

AnzeigeLuftfeuchtigkeit = String(dht.readHumidity());
AnzeigeLuftfeuchtigkeit.replace(".", ",");

client.print("<b>Temperatur:</b><blockquote>");
client.println(AnzeigeTemperatur + " &deg;C</blockquote>");
client.println("<br>");

client.print("<b>Luftfeuchtigkeit:</b><blockquote>");
client.println(AnzeigeLuftfeuchtigkeit + " %</blockquote><br>");
client.println("<form>");

// Datum speichern
gespeichertesDatum[Zaehler] = aktuell.toString(Datum);
gespeichertesDatum[Zaehler] = gespeichertesDatum[Zaehler] + aktuell.toString(Zeit);

// Temperatur speichern
gespeicherteTemperatur[Zaehler] = AnzeigeTemperatur;

// Luftfeuchtigkeit speichern
gespeicherteLuftfeuchtigkeit[Zaehler] = AnzeigeLuftfeuchtigkeit;
```



```
client.println("<b>Gespeicherte Werte:<br></b>");

// gespeicherte Daten anzeigen
for (int i = 0; i <= Zaehler; i ++)
{
    client.println(String(i + 1) + ": ");

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

    client.println(gespeichertesDatum[i]);
    client.print(" | Temperatur: &#9654; ");
    client.println(gespeicherteTemperatur[i]);
    client.print(" &deg;C");
    client.print(" | Luftfeuchtigkeit: &#9654; ");
    client.println(gespeicherteLuftfeuchtigkeit[Zaehler]);
    client.println("%");
    client.println("<br>");
}
client.println("<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("<b>Eigene IP: ");
client.print(client.remoteIP());
client.print("</b>");
```



```
    client.print("<br><b>IP Arduino: ");
    client.print(WiFi.localIP());
    client.print("</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();

// Zaehler erhöhen und bei > AnzahlDaten zurücksetzen
if (Zaehler < AnzahlDaten) Zaehler ++;
if (Zaehler >= AnzahlDaten) Zaehler = 0;
}
}
```