

# Wetterstation Montage

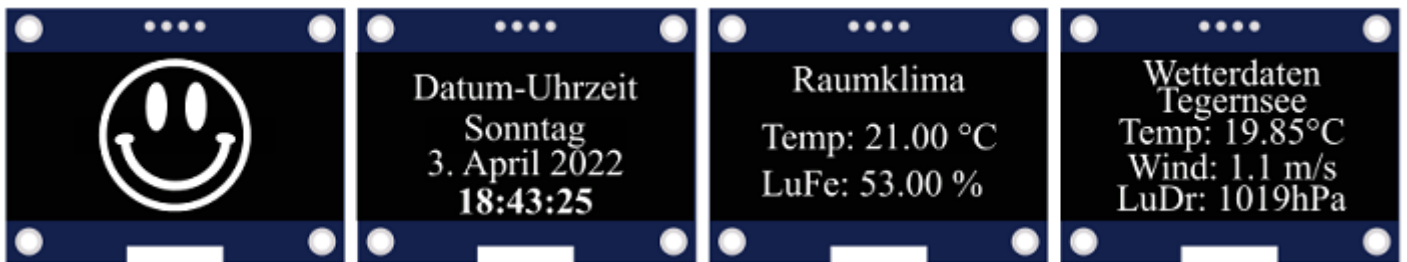
## Anleitung

### Einleitung

Diese Anleitung beschreibt den Aufbau einer Wetterstation in einem Holzgehäuse mit einem ESP32. Beim ESP32 ist ein DHT11-Sensor und am I<sup>2</sup>C-Bus ein OLED 1,3"-Display angeschlossen. Die Wetterstation beinhaltet einen WiFi-Manager für die Einstellung der WLAN-Zugangsdaten (SSID und Passwort). Nach dem ersten Start der Wetterstation, muss in den WLAN-Einstellungen des Handy oder Tablet das Netzwerk "Wetterstation" ausgewählt und über einen Browser die IP-Adresse 192.168.4.1 aufgerufen werden. Danach startet die "Wetterstation WLAN-Einstellung" und die Zugangsdaten für das WLAN können eingetragen werden. Sind die Zugangsdaten korrekt, startet die Wetterstation.

Die Wetterstation zeigt beim Start für 5 Sekunden einen Smiley. In dieser Zeit werden die aktuelle Uhrzeit, das Datum und die aktuellen Wetterdaten aus der Region Tegernsee von [de.pool.ntp.org](http://de.pool.ntp.org) und von [openweathermap.org](http://openweathermap.org) geladen.

Sind die Daten geladen, verschwindet der Smiley und es werden in regelmäßigen Abständen drei Seiten mit Datum und Uhrzeit, danach das Raumklima mit Temperatur und Luftfeuchte, zuletzt die Wetterdaten Außen-Temperatur, Windgeschwindigkeit und Luftdruck angezeigt.





# Hardware

## Das benötigen wir für die Wetterstation

- 1 Sperrholzplatte 4mm x 600x300
- 1 ESP32
- 1 ESP32-Shield
- 3D-Druck Teile für den ESP32, das OLED-Display und den DHT11
- 1 DHT11-Sensor
- 1 Gas-Sensor (optional)
- 1 OLED Display 1,3" I<sup>2</sup>C
- 1 Sekundenkleber und Holzkleber
- 1 9V Batterie oder ein 8V Netzteil
- 1 Batterieclip
- Schalt draht

## Der Aufbau erfolgt in 5 Schritten

### 1. 3D-Druckteile zeichnen und drucken

ESP32-Pins d=5mm h=4mm

OLED-Display Pins 10x4x1,6mm

## 2. Gehäuse lasern

Wir erstellen mit [BOXES.PY](https://box.es.py) eine Lasercutter Vorlage

Wähle die ClosedBox - Fully closed box mit den Maßen  $x=100$ ,  $y=80$ ,  $h=80$  und  $thickness=4$

Zusätzlich wird eine Aussparung für den DHT11-Sensor in der Rückwand und eine Aussparung für den Spannungsanschluss des ESP32 in der Seitenwand eingefügt

Wir lasern mit den Einstellungen Geschw./Leistung = 12.0/100 in einem Durchgang



## 3. Verdrahten

Wir löten auf das ESP32-Shield die Buchsenleisten

Wir schließen mit Schottdraht den DHT11-Sensor am ESP32-Shield GPIO27 an

Am ESP32-Shield GPIO27 wird noch ein 10kΩ Pullupwiderstand gegen +5V angelötet

Wir verdrahten den DHT11 mit +5V und GND

Wir verbinden mit Schottdraht den I<sup>2</sup>C-Bus mit dem Display

Dazu werden die Lötunkte SLA, SLC, 3.3V und GND mit dem Display verbunden

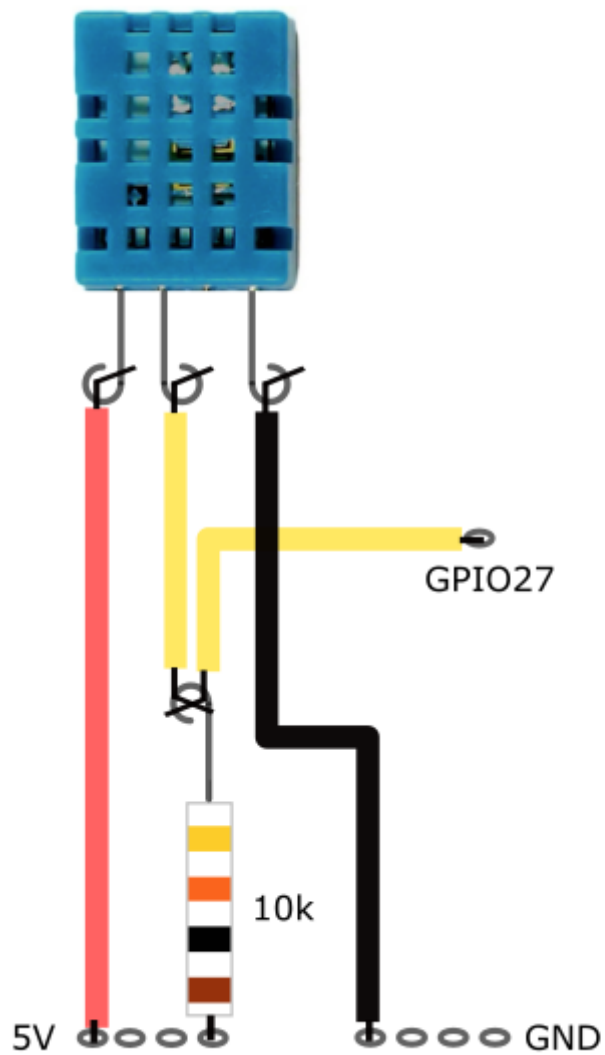
Ist der ESP32 auf dem Shield, der Sensor und das Display angelötet, erfolgt der erste Test

Wir laden einen I<sup>2</sup>C-Scanner in den ESP32 und kontrollieren ob der ESP32 die Busteilnehmer erkennt

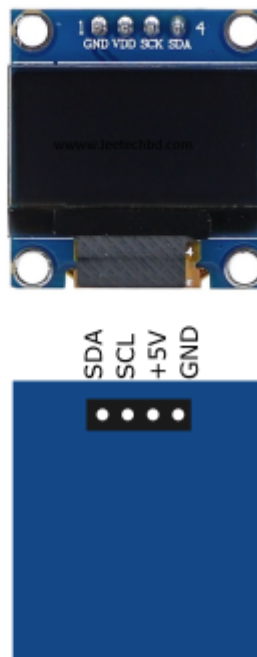
Wird das Display erkannt, laden wir das Haupt-Programm in den ESP32

Verdrahtung des DHT11:

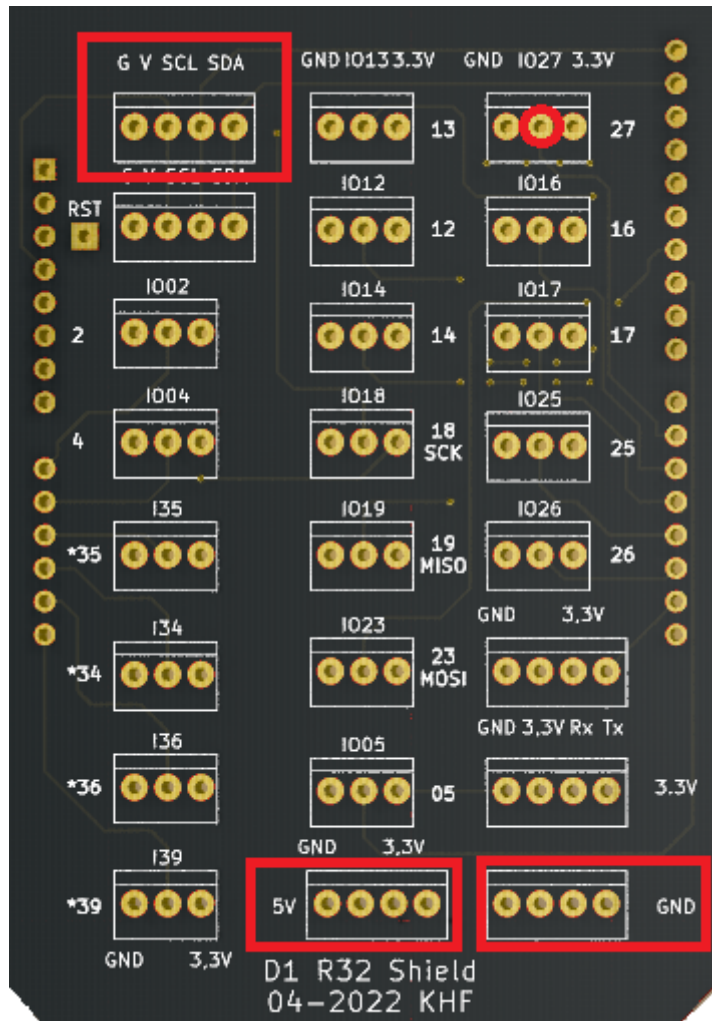
## DHT11 Verdrahtung



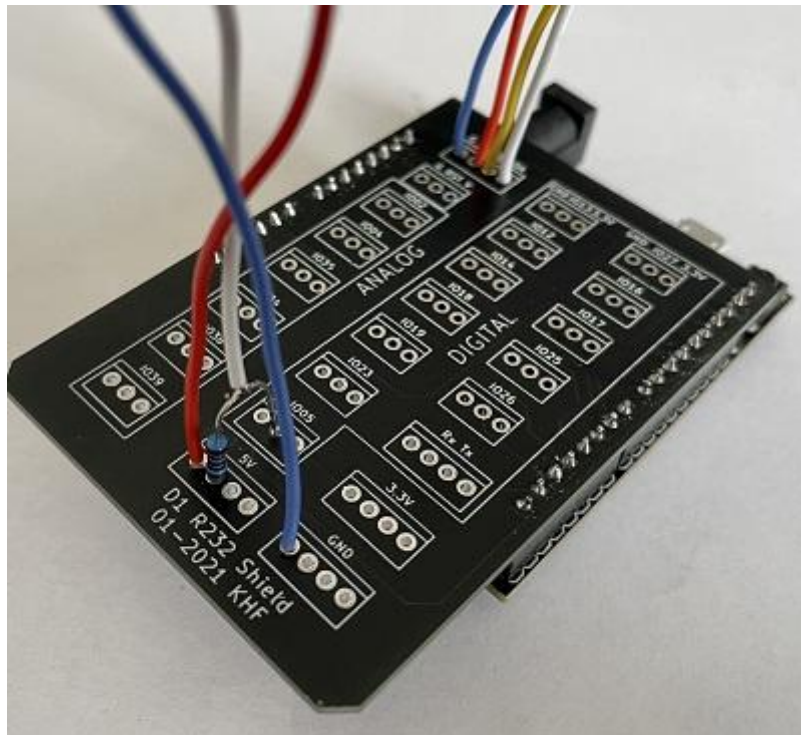
## OLED Verdrahtung



ESP32 Shield:



Verdrahtetes ESP32 Shield:



## 4. Die Montage

Mit Sekundenkleber die 3D-Druck Bauteilfüße auf den ESP32 kleben

Die 3D-Druck Leisten auf das OLED-Display kleben

Den ESP32 auf die Grundplatte und die Sensorhalterung  
an die Gehäuse-Rückwand kleben

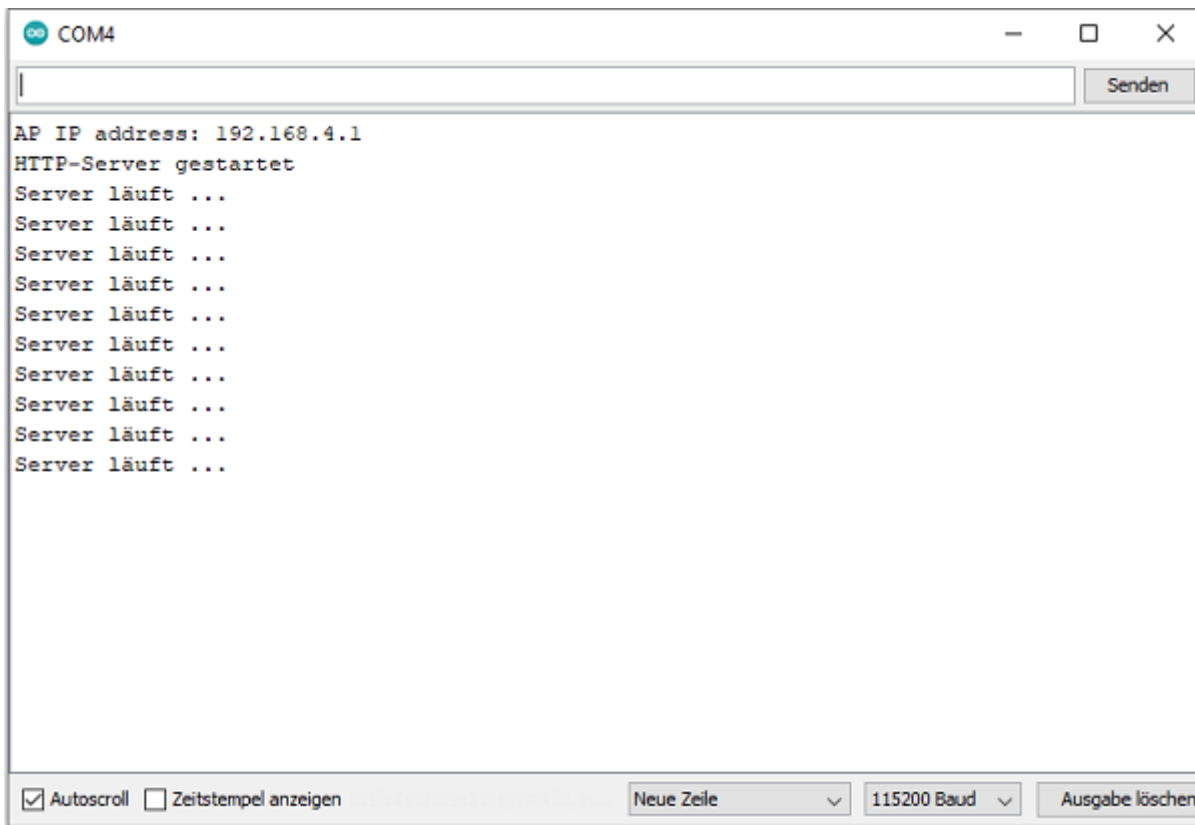
VORSICHTIG das Display an die Frontseite (Durchbruch) kleben

Vor dem finalen verkleben des Gehäuses erfolgt ein Funktionstest

War der Funktionstest erfolgreich, kann das Gehäuse vollständig verklebt werden



Erster Funktionstest am Seriellen Monitor:



## 5. Inbetriebnahme

Im Handy oder Tablet die WLAN-Einstellungen öffnen

Das Netzwerk "Wetterstation" aufrufen

Im Browser die IP-Adresse 192.168.4.1 aufrufen

Die Seite "Wetterstation WLAN Einstellung" öffnet sich

Die WLAN-Zugangsdaten SSID und Passwort eingeben

Der Zeitserver ist voreingestellt und muss nicht verändert werden

Ist die WLAN-Verbindung korrekt, startet die Wetterstation mit einem Smiley am Display

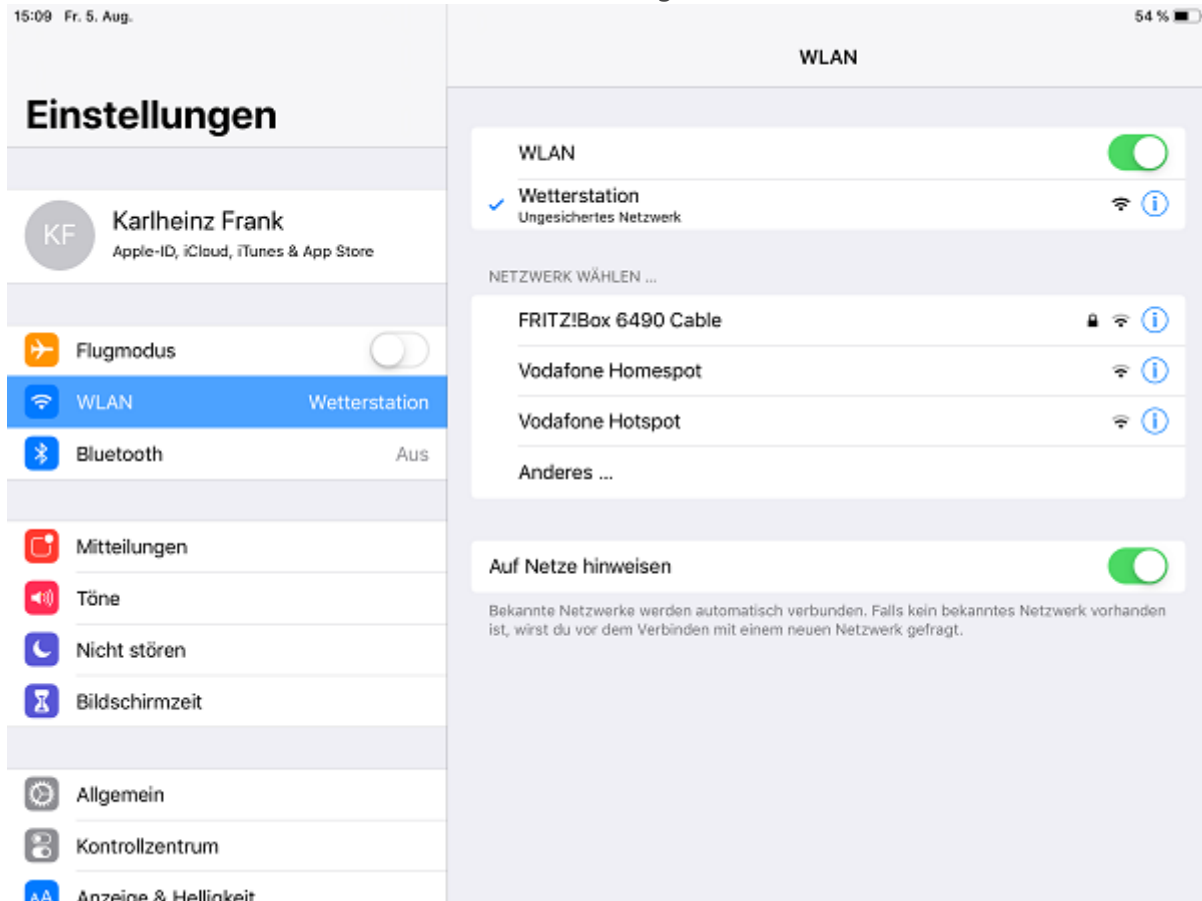
Danach werden abwechselnd das aktuelle Datum mit Uhrzeit, danach das Raumklima mit

Temperatur und Luftfeuchte und die aktuellen Wetterdaten aus der Region Tegernsee angezeigt

Die angezeigten Daten wechseln in regelmäßigen Abständen



## Aufruf "Wetterstation" in den WLAN-Einstellungen



Im Browser die IP-Adresse 192.168.4.1 öffnen, die WLAN Zugangsdaten eingeben und speichern



## Wetterstation WLAN-Einstellung

### WLAN-Einstellungen

WLAN-Name (SSID):

WLAN-Passwort:

Zeitserver (NTP):

Falls nicht klar ist, was das ist,  
dann sollte es **de.pool.ntp.org** bleiben.

speichern



Sind die WLAN-Zugangsdaten korrekt, startet die Wetterstation



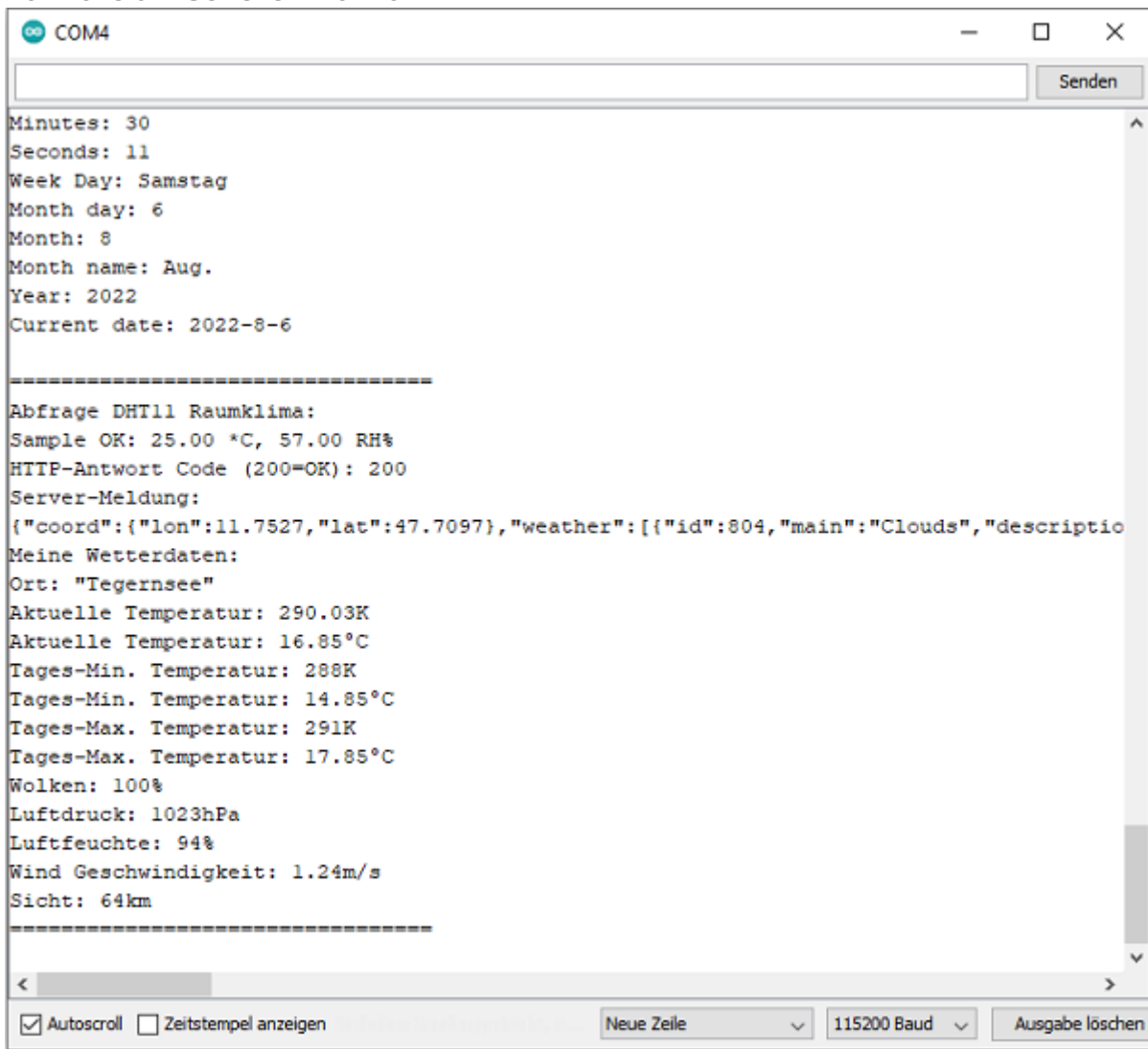
## Wetterstation WLAN-Einstellung

**Verbindung wird gesucht...**

**Sollte die Wetterstation nicht starten, bitte in 20 Sek. wieder mit dem WLAN "Wetterstation" verbinden und die Seite erneut aufrufen.**

Startet die Wetterstation nicht, muss der Vorgang wiederholt werden

Kontrolle am Seriellen Monitor:





**Fertig, Glückwunsch!**

# Troubleshooting, mögliche Fehlerursachen

- Sichtkontrolle - sind alle Lötstellen ok?
- Sind Lötbrücken zusehen oder sind Bauteilfüße fälschlicherweise mit einander verbunden?
- Sitzt der ESP32 korrekt auf dem Shield? Auf die Polung achten und und Bauteilfüße kontrollieren.
- Sind die Anschlussdrähte zum Sensor und Display ok?
- Ist die Spannungsversorgung für den ESP32 ok?
- Wurde das Programm korrekt geladen? Meldung: Hochladen abgeschlossen.
- Wird das Display vom ESP32 erkannt? Mit dem I<sup>2</sup>C-Scanner prüfen.
- Werden die Sensordaten und die Uhrzeit/Datum ausgelesen? Mit dem Seriellen Montitor der Arduino IDE prüfen.
- Wurden die WLAN-Zugangsdaten korrekt eingegeben?

# Abkürzungen

- OLED - Organic Light Emitting Diode
- u8g2 Library - Universal 8Bit Graphics Library

- I<sup>2</sup>C - Inter-Integrated Circuit (Serieller Datenbus)
- SCL - Serial Clock
- SDA - Serial Data

# C++ Programm I2C Scanner

```
// ESP32 I2C Scanner
// ESP32 DevKit / ESP32vn IoT UNO - Arduino IDE 1.8.

#include <Wire.h>

void setup()
{
  Serial.begin (115200);
  Wire.begin (21, 22); // sda= GPIO_21 /scl= GPIO_22
}

void Scanner ()
{
  Serial.println ();
  Serial.println ("I2C scanner. Scanning ...");
  byte count = 0;

  Wire.begin();
  for (byte i = 8; i < 120; i++)
  {
    Wire.beginTransmission (i);          // Begin I2C transmission Address (i)
    if (Wire.endTransmission () == 0) // Receive 0 = success (ACK response)
    {
      Serial.print ("Found address: ");
      Serial.print (i, DEC);
      Serial.print (" (0x");
      Serial.print (i, HEX); // PCF8574 7 bit address
      Serial.println (");");
      count++;
    }
  }
}
```

```
Serial.print ("Found ");  
Serial.print (count, DEC);    // numbers of devices  
Serial.println (" device(s).");  
}  
  
void loop()  
{  
  Scanner ();  
  delay (100);  
}
```

---

Version #1

Erstellt: 2 April 2025 21:57:52 von Joel Hatsch

Zuletzt aktualisiert: 2 April 2025 22:12:55 von Joel Hatsch