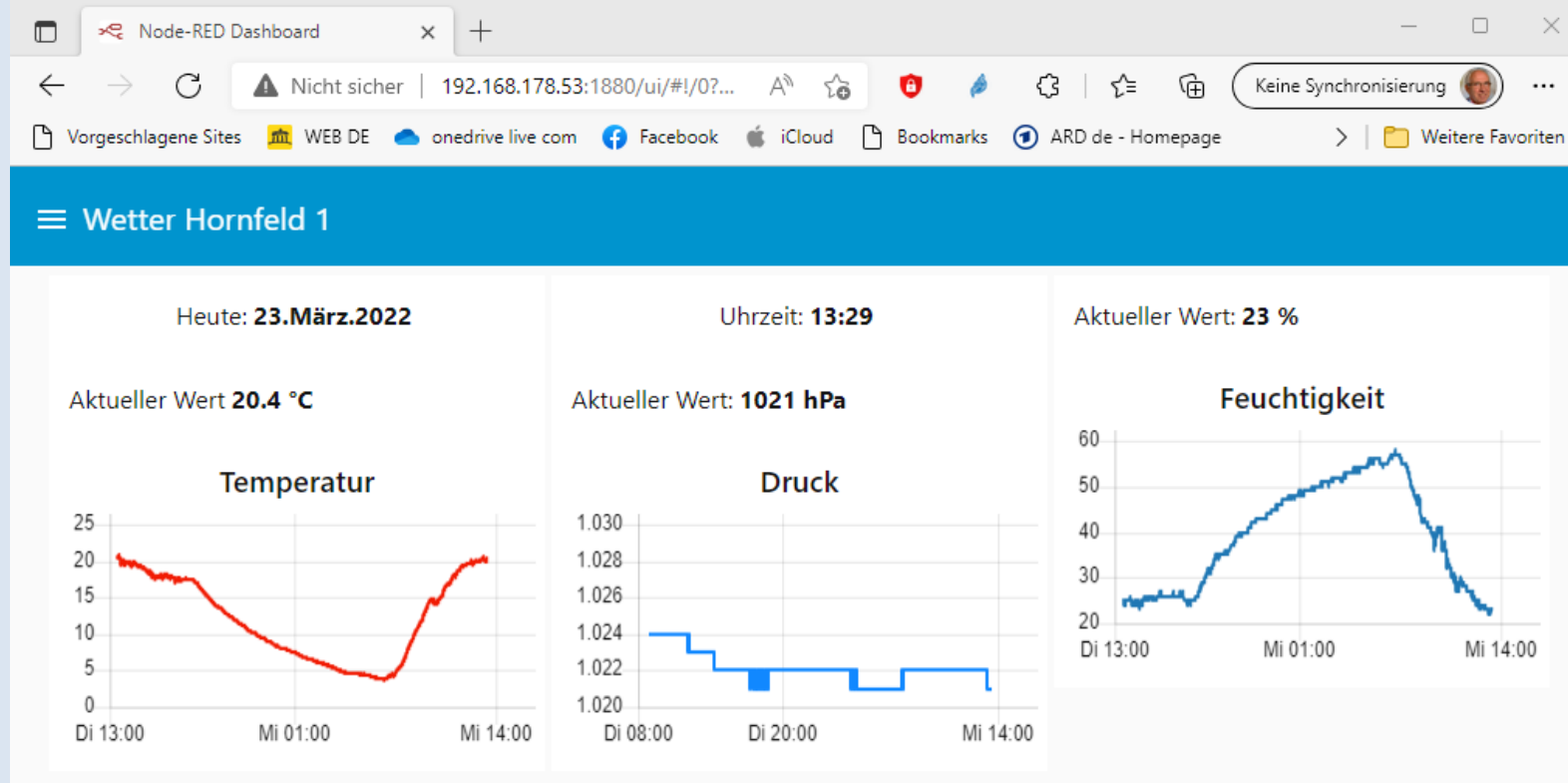


IoT & Node-RED & RPI

- IoT & Node-RED & RPI
- Wetterdaten im PC-Browser
- Raspberry Pi
- Wetterstation
- MQTT, Node-RED, ESP8266
- Projektbausteine
- Raspberry Pi 32-Bit (Bullseye)
- Node-RED
- Palette erweitern
- Node-RED
- MQTT Broker
- Mosquitto 2.x RPI-broker
- Mosquitto 2.x Enable Access
- Node-RED Flow
- ESP8266
- BME Libraries installieren (2022)
- ESP8266
- BME280-Breakout von watterott.com
- I2C-Betriebsart
- „ESP8266 ESP-01“; Schaltbild
- „ESP8266 ESP-01“ & Sketch-Beispiel
- „ESP-12E nodeMCU“; Schaltbild
- „ESP-12E nodeMCU“ & Sketch-Beispiel
- Weitere Infos

Wetterdaten im PC-Browser

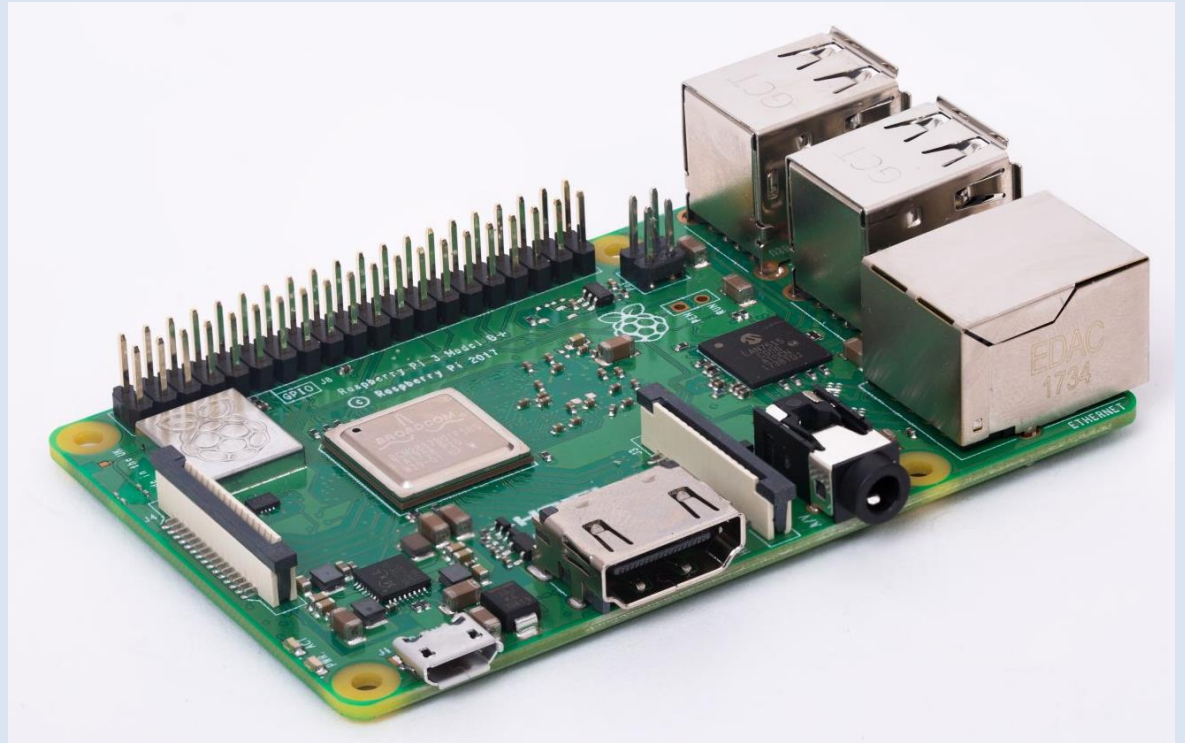


Raspberry Pi

Raspberry Pi 4 Model B

oder

Raspberry Pi 3 Model B oder B+



Quelle: <https://www.raspberrypi.org/products/>

Wetterstation

Mikrokontroller:

ESP8266 ESP-01 Serial Port WIFI Transceiver Wireless Modul

ESP8266 Breadboard-Adapter:

ESP-01 Adapter Breakout Board Platine ESP01

Spannungsregler Step Down Power Modul:

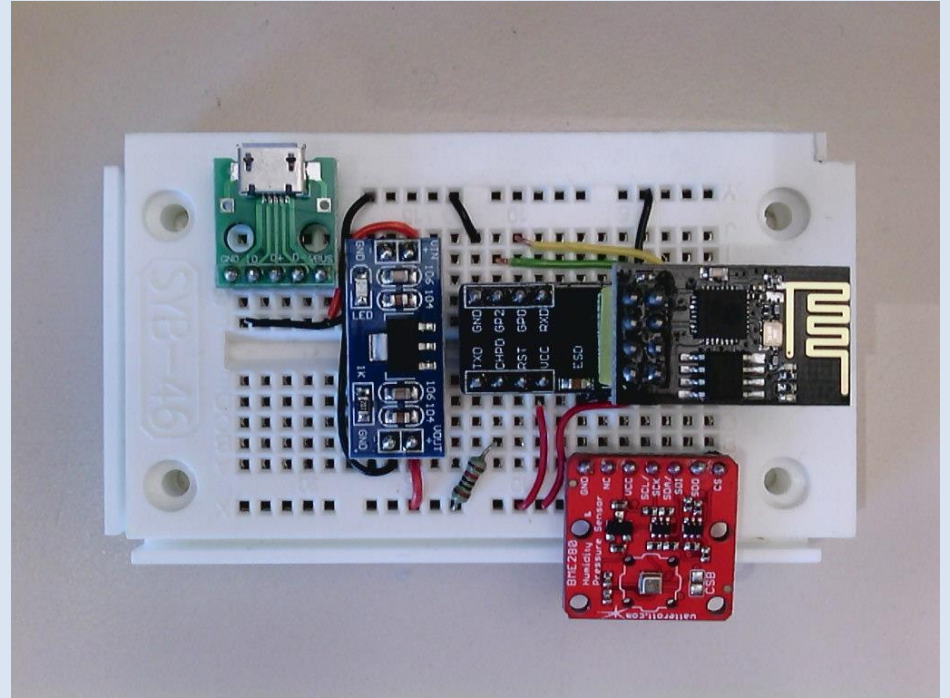
AMS1117 3,3 V

Mikro-USB Anschluss:

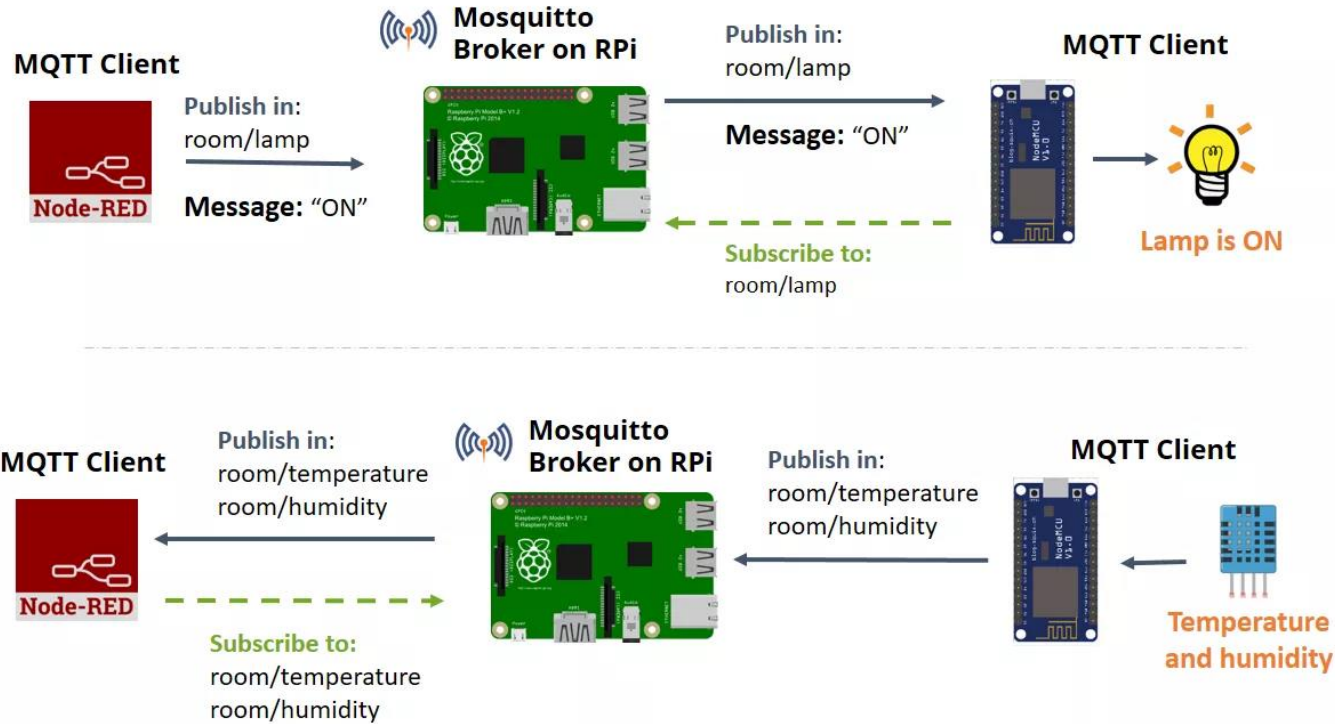
Micro USB Breakout Modul Board Platine

Wettersensor:

BME280-BREAKOUT (watterott.com)



MQTT, Node-RED, ESP8266



Quelle <https://i2.wp.com/randomnerdtutorials.com/wp-content/uploads/2017/08/MQTT-ESP8266-publish-and-subscribe-Node-RED.png?ssl=1>

Projektbausteine

Das Projekt hat als Basis die Informationen von:

<https://randomnerdtutorials.com/esp8266-and-node-red-with-mqtt/>

Raspberry Pi (RPI)

Raspberry Pi mit Betriebssystem „Debian Version: 11 “ einrichten.

Node-RED

Raspberry Pi mit Node-RED konfigurieren. Als Dienst einrichten.

Broker Mosquitto

Raspberry Pi mit Mosquitto konfigurieren. Als Dienst einrichten.

Betriebsart

Node-RED und Mosquitto laufen 24h auf dem RPI.

ESP8266 ESP-01

ESP8266 Sketch anpassen und flashen.

Wetterstation

BME280-Sensor & Zusatzkomponenten aufbauen.

Node-RED flow

Einen Flow, der die Messdaten erfasst, und auf einem Dashboard (Webinterface) anzeigt.

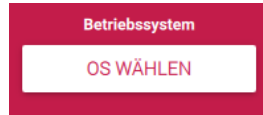
Raspberry Pi 32-Bit (Bullseye)

Raspberry Pi OS Raspbian Betriebssystem herunterladen.

Windows 10 SD-Karte in SD-Card-Slot bereitstellen

Imager installieren und ausführen https://downloads.raspberrypi.org/imager/imager_latest.exe

OS wählen



Raspberry Pi OS (32-bit)

A port of Debian Bullseye with the Raspberry Pi Desktop (Recommended)

Veröffentlicht: 2022-01-28

Online - 1.2 GB Download

SD-Karte wählen



Generic- Multi-Card USB Device - 31.7 GB

Als D:\ eingebunden

SSH aktivieren

Macht einen Zugriff vom PC über puTTY möglich.



Schreiben



Raspberry Pi 32-Bit (Bullseye)

Schreiben bestätigen

Alle vorhandenen Daten auf 'Generic- Multi-Card USB Device'
werden gelöscht.
Möchten Sie wirklich fortfahren?

NEIN

JA

RPI

SD-Karte entnehmen und Setup auf dem RPI weiter durchführen.

Deinstallation

Auf dem Windows-Rechner „imager_1.7.1.exe“ deinstallieren.

Node-RED

Anleitung

<https://nodered.org/docs/getting-started/raspberrypi>

Linux-Prompt

Bitte nicht das „\$“ Zeichen eingeben.

LXTerminal: \$ bash <(curl -sL <https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered>)

LXTerminal

Anzeige:

```
Running Node-RED install for user pi at /home/pi on debian

This can take 20-30 minutes on the slower Pi versions - please wait.

Stop Node-RED                      ✓
Remove old version of Node-RED     ✓
Remove old version of Node.js      ✓
Install Node.js 14 LTS              ✓   v14.19.0   Npm 6.14.16
Clean npm cache                    ✓
Install Node-RED core               ✓   2.2.2
Move global nodes to local         -
Npm rebuild existing nodes         ✓
Install extra Pi nodes             ✓
Add shortcut commands              ✓
Update systemd script              ✓

Any errors will be logged to /var/log/nodered-install.log
```

Node-RED

LXTerminal

Nach erfolgter erfolgreicher Installation:

```
All done.  
You can now start Node-RED with the command node-red-start  
or using the icon under Menu / Programming / Node-RED  
Then point your browser to localhost:1880 or http://{your_pi_ip-address}:1880  
  
Started : So 6. Mär 18:26:03 CET 2022  
Finished: So 6. Mär 18:34:38 CET 2022  
  
You may want to run node-red admin init  
to configure your initial options and settings.
```

Erster Start:

`$ node-red-start`

```
Start Node-RED  
  
Once Node-RED has started, point a browser at http://192.168.178.54:1880  
On Pi Node-RED works better with the Firefox or Chrome browser  
  
Use node-red-stop to stop Node-RED  
Use node-red-start to start Node-RED again  
Use node-red-log to view the recent log output  
Use sudo systemctl enable nodered.service to autostart Node-RED at every boot  
Use sudo systemctl disable nodered.service to disable autostart on boot  
  
To find more nodes and example flows - go to http://flows.nodered.org
```

Node-RED

LXTerminal

Weitere Anzeige:

```
Starting as a systemd service.
6 Mar 18:40:09 - [info]
Willkommen bei Node-RED!
=====
6 Mar 18:40:09 - [info] Node-RED Version: v2.2.2
6 Mar 18:40:09 - [info] Node.js Version: v14.19.0
6 Mar 18:40:09 - [info] Linux 5.10.92-v8+ arm64 LE
6 Mar 18:40:11 - [info] Paletten-Nodes werden geladen
6 Mar 18:40:15 - [info] Einstellungsdatei: /home/pi/.node-red/settings.js
6 Mar 18:40:15 - [info] Kontextspeicher: default [module=memory]
6 Mar 18:40:15 - [info] Benutzerverzeichnis: /home/pi/.node-red
6 Mar 18:40:15 - [warn] Projekte deaktiviert: editorTheme.projects.enabled=false
6 Mar 18:40:15 - [info] Flow-Datei: /home/pi/.node-red/flows.json
6 Mar 18:40:15 - [info] Neue 'flow'-Datei wird erstellt
6 Mar 18:40:15 - [warn]
-----
6 Mar 18:40:15 - [info] Server wird jetzt auf http://127.0.0.1:1880/ ausgeführt
6 Mar 18:40:15 - [info] Flows werden gestartet
6 Mar 18:40:15 - [info] Flows sind gestartet
```

Node-RED

Raspberry Pi „bullseye“

Automatischen Start einrichten	LXTerminal: <code>\$ sudo systemctl enable nodered.service</code>
Service beenden	LXTerminal: <code>\$ sudo systemctl disable nodered.service</code>
Node-RED starten	LXTerminal: <code>\$ node-red-start</code> (bei automatischem Start nicht erforderlich)
Node-RED beenden	LXTerminal: <code>\$ node-red-stop</code>
Node-RED auf RPI	Browser: <code>http://127.0.0.1:1880</code>
IP vom RPI?	LXTerminal: <code>\$ hostname -I</code> ergibt bei mir „192.168.178.xxx“
Node-RED von PC	Browser: <code>http://192.168.178.xxx:1880</code>

Palette erweitern

Serielle Schnittstelle (USB)
nachrüsten

> Hamburger-Menü



[node-red-node-serialport \(node\) - Node-RED \(nodered.org\)](#)

> Manage Palette

> Tab Install

> Search modules > „node-red-node-serialport“

> install

[dashboard-evi \(node\) - Node-RED \(nodered.org\)](#)

Dashboard
nachrüsten

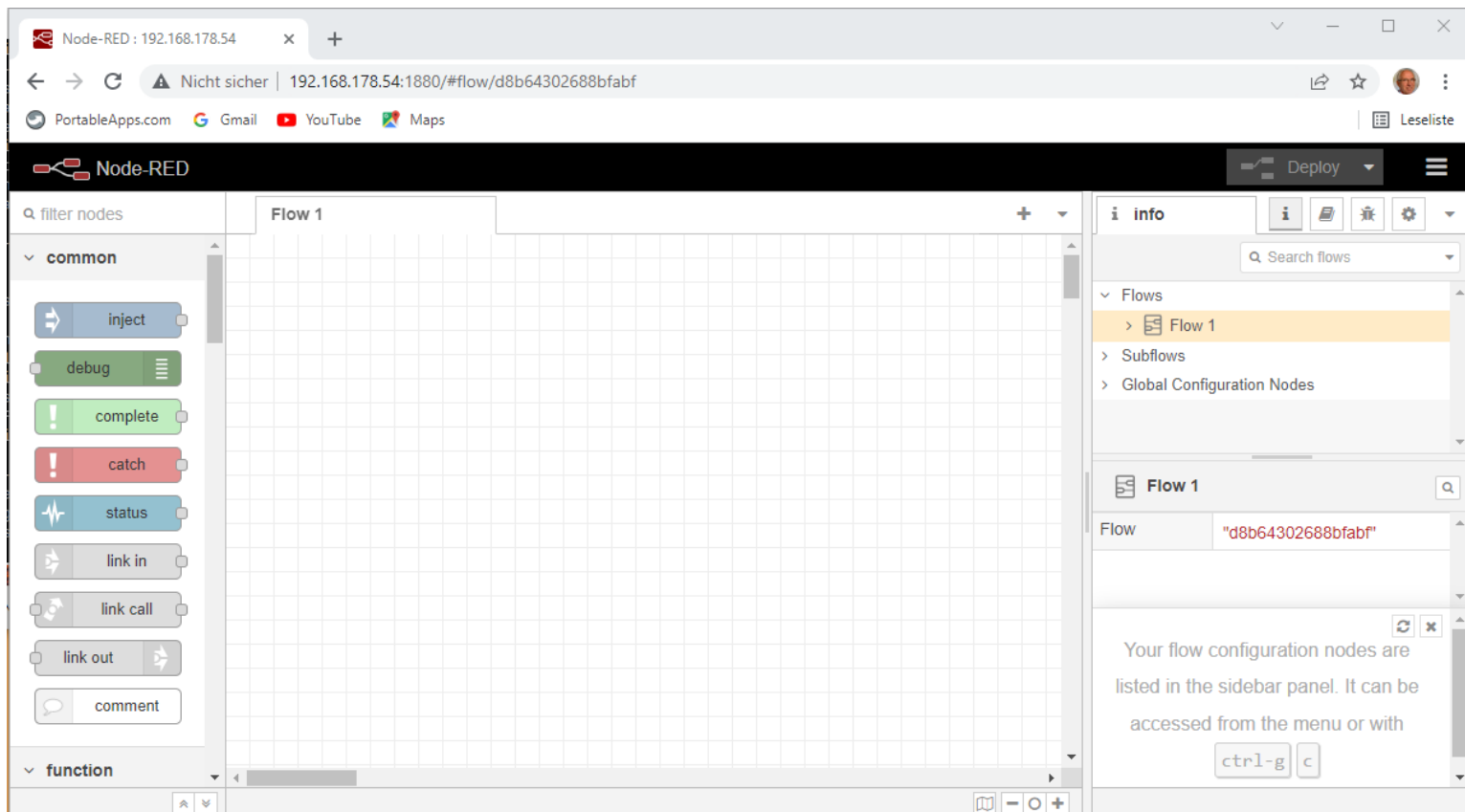
> Search modules > „dashboard-evi“

> install

Node-RED

Windows Browser

192.168.178.xx:1880 (deine IP:1880)



MQTT Broker

MQTT

Message Queue Telemetry Transport

Das Nachrichten-Protokoll MQTT wurde für die Kommunikation von Geräten zu Geräten entwickelt.

Grundlange für die Kommunikation bildet ein sogenannter Broker.

Ein Broker ist eine Server-Anwendung.

Ein Gerät kann sich nun mit dem Broker verbinden und Nachrichten Publischen (veröffentlichen), also an den MQTT-Broker, unter Nennung von Topics (Bezeichner), senden.

Ein anderes Gerät (hier: Node-RED) kann dann Topics Subscriben (abonnieren), und bekommt dann eine Nachricht vom MQTT-Broker zugestellt.

Information

<https://randomnerdtutorials.com/what-is-mqtt-and-how-it-works/>

Mosquitto 2.x RPI-broker

	Mosquitto ist ein MQTT Broker für den Raspberry Pi
Anleitungen von	https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/ https://randomnerdtutorials.com/testing-mosquitto-broker-and-client-on-raspbbery-pi/
Terminal	<pre>\$ sudo su # sudo apt update # sudo apt install -y mosquitto mosquitto-clients # sudo systemctl enable mosquitto.service # mosquitto -v</pre>
Nur „local mode“	<pre>root@raspberrypi:/home/pi# sudo systemctl enable mosquitto.service Synchronizing state of mosquitto.service with SysV service script with /lib/systemd/systemd-sysv-install. Executing: /lib/systemd/systemd-sysv-install enable mosquitto root@raspberrypi:/home/pi# mosquitto -v 1647348118: mosquitto version 2.0.11 starting 1647348118: Using default config. 1647348118: Starting in local only mode. Connections will only be possible from clients running on this machine. 1647348118: Create a configuration file which defines a listener to allow remote access. 1647348118: For more details see https://mosquitto.org/documentation/authentication-methods/ 1647348118: Opening ipv4 listen socket on port 1883. 1647348118: Error: Address already in use 1647348118: Opening ipv6 listen socket on port 1883. 1647348118: Error: Address already in use root@raspberrypi:/home/pi#</pre>
	Fehlermeldung bei der Mosquitto-Version 2.x: Nur lokaler Modus!

Mosquitto 2.x Enable Access

	Eine neue *.conf-Datei im Ordner “conf.d” erstellen.
Terminal	<i># sudo nano /etc/mosquitto/conf.d/RPI.conf</i>
	Hier folgende Zeilen hinzufügen: listener 1883 allow_anonymous true
	Nano-Editor: Datei speichern „Strg+O“ und schließen „Strg+X“.
Mosquitto Restart	<i>\$ sudo systemctl restart mosquitto</i>
Mosquitto Status	<i>\$ sudo systemctl status mosquitto</i>
IP?	Terminal: <i>\$ hostname -I</i> ergibt hier „192.168.178.xxx“. Wird für den ESP8266-Sketch gebraucht, daher notieren.

Node-RED Flow

Vorhanden

Raspberry Pi mit Node-RED und Mosquitto konfiguriert.

Aufgabe von Node-RED

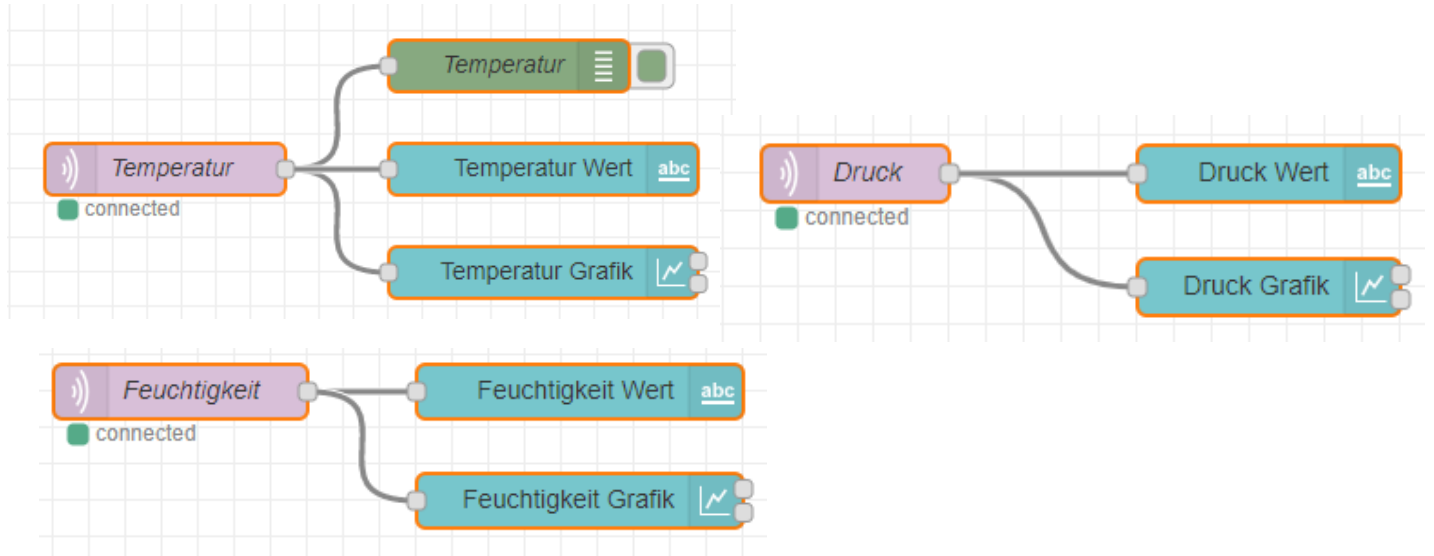
Node-RED & Mosquitto laufen 24h auf einem Raspberry Pi.

Node-RED öffnen

RPI-Browser: <http://127.0.0.1:1880>

PC-Browser: <http://192.168.178.xxx:1880>

Flow Wetterstation



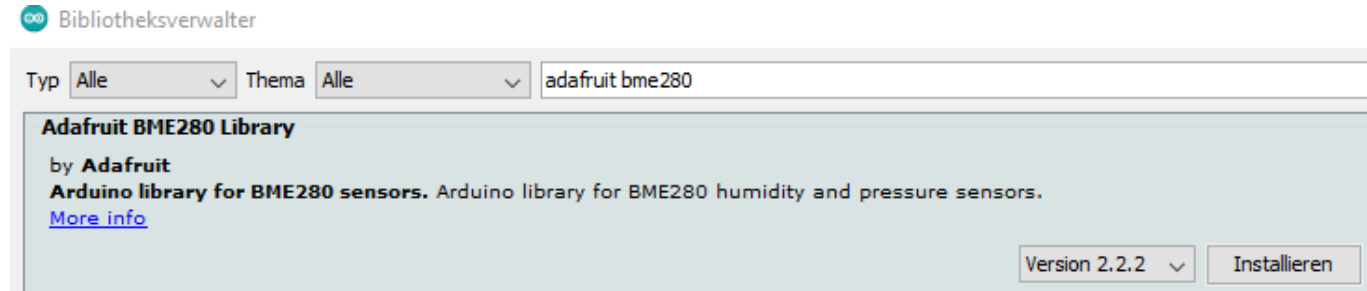
ESP8266

Sketch	Der ESP8266 muss mit einem Sketch programmiert werden, der WLAN , MQTT und die Sensordaten vom BME280 verarbeitet.
PubSubClient Library	Erzeugt einen Client auf dem ESP8266, um Publish/Subscribe Messages mit einem MQTT Server zu ermöglichen.
Download	<u>https://github.com/knolleary/</u> <u>https://github.com/knolleary/pubsubclient/archive/master.zip</u>
BME280 Sensor Libraries	Die Libraries dienen zum Einbinden des BME280-Sensors. <i>Siehe Installationsanweisungen weiter unten...</i>
Information	<u>https://randomnerdtutorials.com/esp8266-and-node-red-with-mqtt/</u>

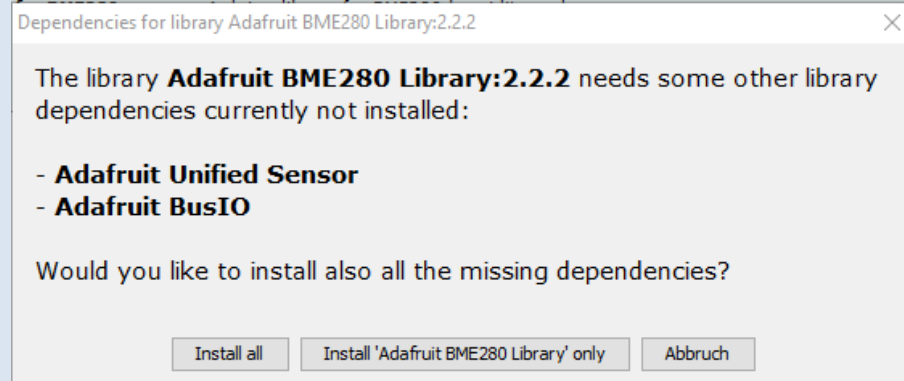
BME Libraries installieren (2022)

Bevor der BME280 mit der Arduino-IDE programmiert werden kann, müssen Libraries installiert werden.

- > Arduino-IDE starten
- > Werkzeuge
- > Bibliotheken verwalten
- > Suchen nach
- > „adafruit bme280“



- > Installieren



- > Install all




BME Libraries installieren (2022)

Hinweise Der Speicherort der Libraries richtet sich nach der Art der Arduino-IDE Installation.

Tipp Direkt nach der Installation der Arduino-IDE den Ordner „portable“ erstellen.
Z.B.: „...\\Arduino-1.8.19\\portable“

Ergebnis Alle weiteren Erweiterungen der IDE werden in „portable“ gespeichert und nicht unter „C:\\Users\\...\\AppData\\“.

Speicherort der hinzugefügten Libraries „...\\arduino-1.8.19\\portable\\staging\\libraries“

Name	Änderungsdatum
 Adafruit_BME280_Library-2.2.2.zip	23.03.2022 13:14
 Adafruit_BusIO-1.11.3.zip	23.03.2022 13:14
 Adafruit_Unified_Sensor-1.1.5.zip	23.03.2022 13:14

Quelle https://github.com/adafruit/Adafruit_BME280_Library

ESP8266

Sketch Ich habe einen Sketch von „randomnerdtutorials“ angepasst. Den angepassten Sketch hier runterladen:

Sketch herunterladen <https://c.web.de/@334322739298962515/QiUNyKKfSuK8YbeYhvO4aQ>

Sketch editieren „ssid“ und „password“ eintragen.
„IP“ vom MQTT-Server eintragen.

```
24 // Change the credentials below, so your ESP8266 connects to your router
25 const char* ssid = "Insert your WLAN ssid";
26 const char* password = "Insert your WLAN password";
27
28 // Change the variable to your Raspberry Pi IP address, so it connects to your MQTT broker
29 const char* mqtt_server = "Insert the IP of your MQTT-Server";
```

Sketch auf „ESP8266“ flashen.

Information <https://randomnerdtutorials.com/esp8266-and-node-red-with-mqtt/>

BME280-Breakout von watterott.com

www.watterott.com

BME280-Breakout (Luftfeuchtigkeits-, Druck & Temperatursensor)

Der BME280 ist einer der neuesten Luftfeuchtigkeits-, Druck- und Temperatursensoren von Bosch mit einem digitalen I2C und einem SPI Interface.

Auf dem Breakout befinden sich ein Spannungsregler und ein Pegelwandler für die I2C/SPI Schnittstelle, daher kann der Sensor von 3V - 5,5V betrieben werden.

Features



- Humidity sensor
- Pressure sensor
Pressure range 300 ... 1100 hPa
- Temperatur Sensor
Operating range Operational -40°C - +85°C

Weitere Infos

github.com/watterott/BME280-Breakout

Quelle:

<https://www.watterott.com/de/BME280-Breakout-Luftfeuchtigkeits-Druck-Tempertursensor>

I2C-Betriebsart

Wertvolle Tipps in <https://learn.sparkfun.com/tutorials/i2c>

<https://tronixstuff.com/2010/10/20/tutorial-arduino-and-the-i2c-bus/>

I2C Die Idee ist, dass über 2 Leitungen, SDA und SCL genannt, mehrere Geräte kommunizieren können. Daher auch I2C-Bus.

Dazu wird z.B. ein Temperatursensor in einem „Breakout Board“ integriert, das I2C bereit stellt, erkennbar an den Kontakten SDA und SCL.

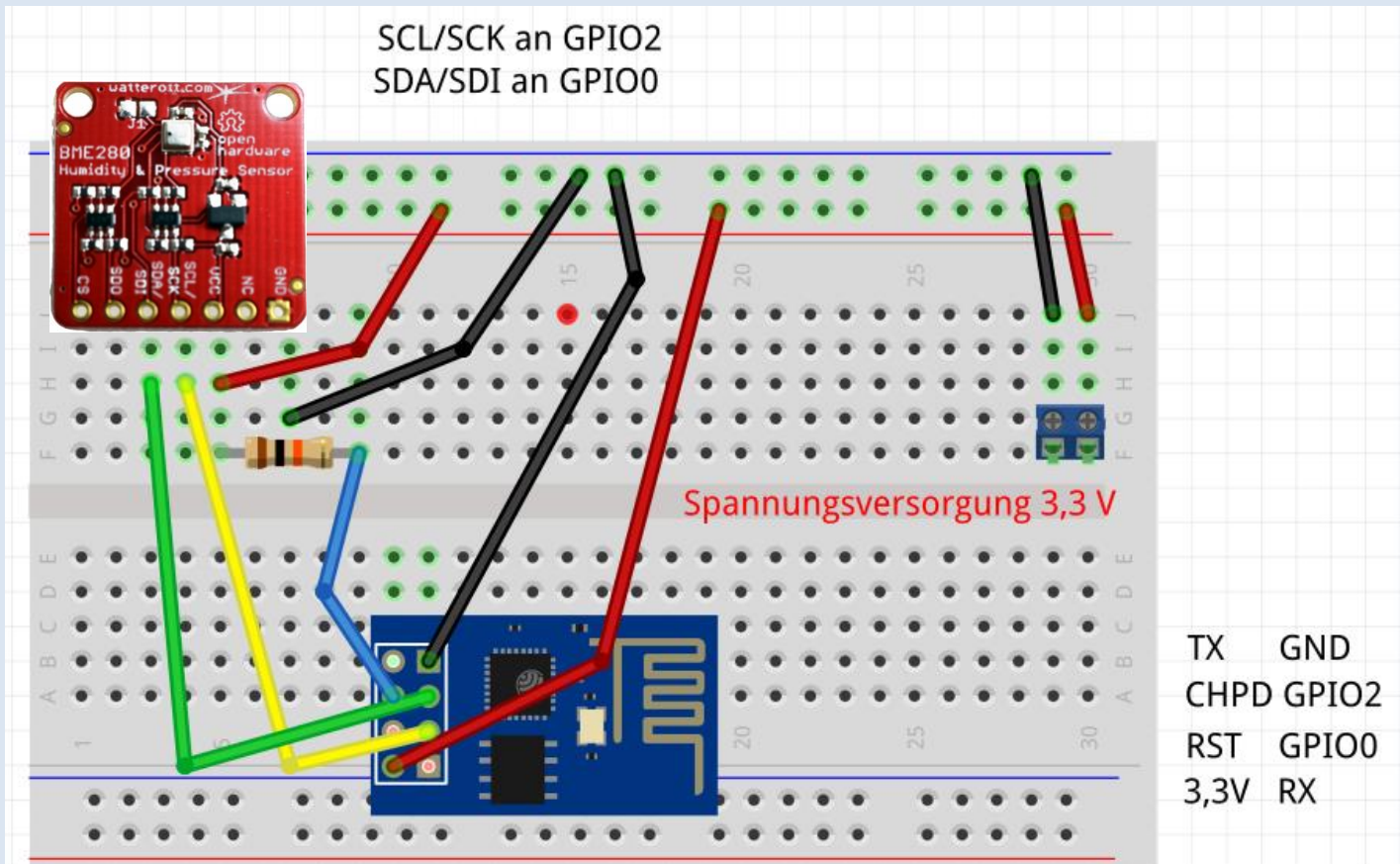
Für I2C sind festgelegt:

Board	SDA (data)	SCL (clock)
Arduino UNO	A4	A5
Arduino Mega	DPIN 20	DPIN 21 (nicht getestet)
ESP-12E nodeMCU	D2	D1
** ESP-01	GPIO0	GPIO2

** Anderer DPIN mit Wire.begin(SDA, SCL) im Setup() einfügen und definieren.
Wire.begin(0, 2);

Quelle <https://www.arduino.cc/en/Tutorial/MasterReader>

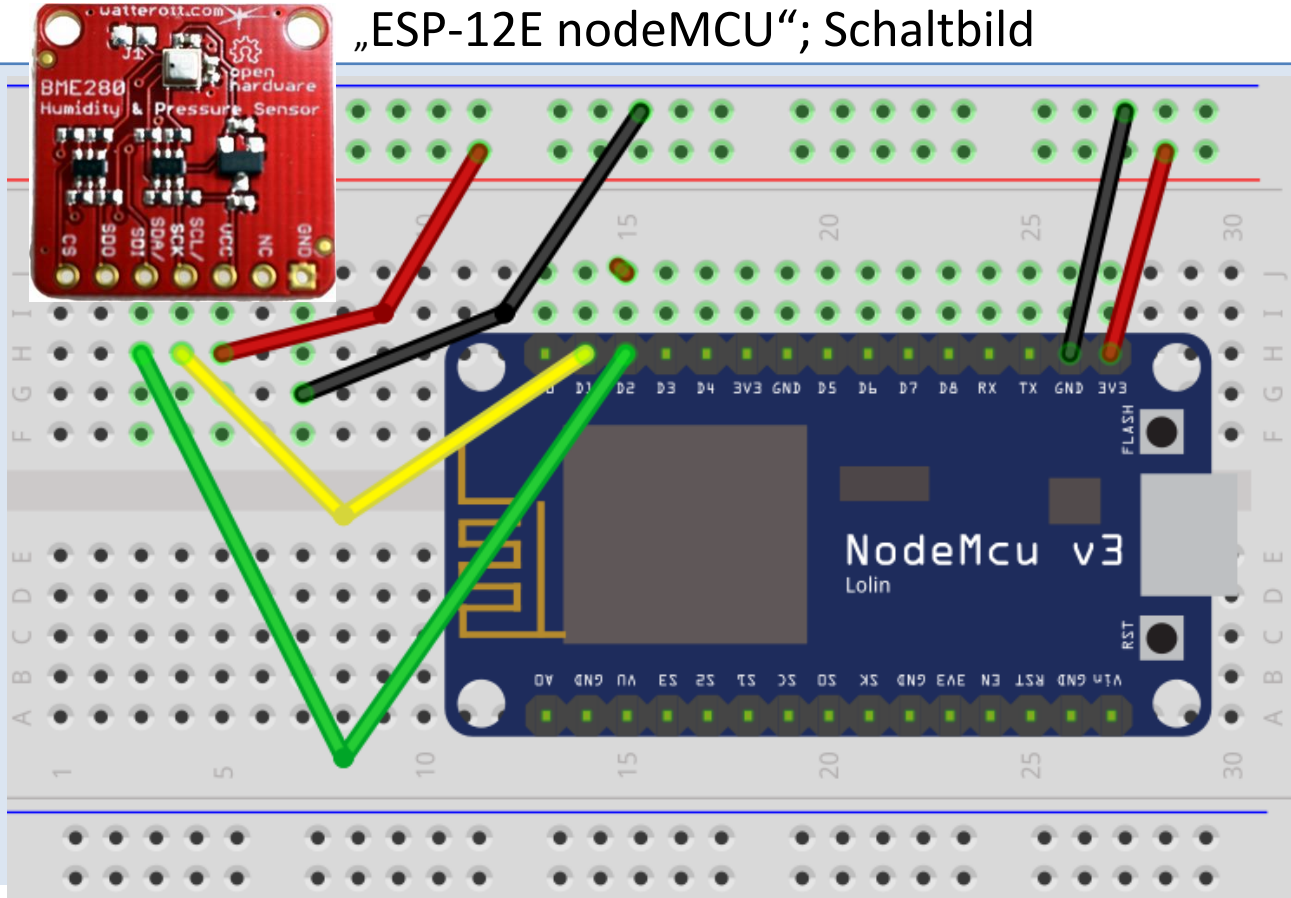
„ESP8266 ESP-01“; Schaltbild



„ESP8266 ESP-01“ & Sketch-Beispiel

Wertvolle Tipps in	http://raphuscucullatus.blogspot.com/2017/07/chinesischer-bme280-sensor-und-esp8266.html		
Die Zeilen 23 bis 26 mit Kommentaren?	Da im I2C-Modus die SPI-DPIN-Zuordnung nicht benötigt wird, kann sie auskommentiert werden.		
I2C in Zeile 30 aktivieren	30 <i>Adafruit_BME280 bme; // I2C</i>		
	Board	SDA (data)	SCL (clock)
I2C-Verkabelung „ESP8266 ESP-01 “	ESP-01	GPIO0	GPIO2
Im Codeabschnitt setup() noch hinzufügen:	Wire.begin(0, 2);		

„ESP-12E nodeMCU“; Schaltbild



DPIN2 an SDA/SDI und DPIN1 an SCL/SCK

„ESP-12E nodeMCU“ & Sketch-Beispiel

Wertvolle Tipps in	http://raphuscucullatus.blogspot.com/2017/07/chinesischer-bme280-sensor-und-esp8266.html
Sketch öffnen	Datei > Beispiele > „Adafruit BME280 Library“ > „ bme280test.ino “ öffnen
Die Zeilen 23 bis 26 mit Kommentaren?	Da im I2C-Modus die SPI-DPIN-Zuordnung nicht benötigt wird, kann sie auskommentiert werden.
I2C in Zeile 30 ohne Kommentare	30 <i>Adafruit_BME280 bme; // I2C</i>
I2C-Verkabelung „ESP-12E nodeMCU“	<pre>//I2C-wiring nodeMCU // BME280 nodeMCU ESP8266 12E // SCL/SCK (Clock) DPIN 1 GPIO5 Serial-Clock // SDA/SDI (Serial Data In) DPIN 2 GPIO4 Serial-Data (bi-directed)</pre>

Der Sketch „bme280test.ino“ sollte ohne Anpassungen funktionieren!

Weitere Infos

github.com/watterott/BME280-Breakout

Interessante Informationen, die für den Arduino-Programmierer zunächst nicht relevant sind!

Link:
Bosch BME280

Produktinformation:
https://www.bosch-sensortec.com/en/bst/products/all_products/bme280

Relevante Informationen finden sich hier:

Link:
learn.watterott.com

<http://learn.watterott.com/sensors/bme280/>

Für den „BME280 MOD-1022 Weather Multi Sensor“.

<https://github.com/embeddedadventures/BME280>

Für den Bosch BME280 Arduino/Teensy Library

https://github.com/Protoinfy/BME280_Library
