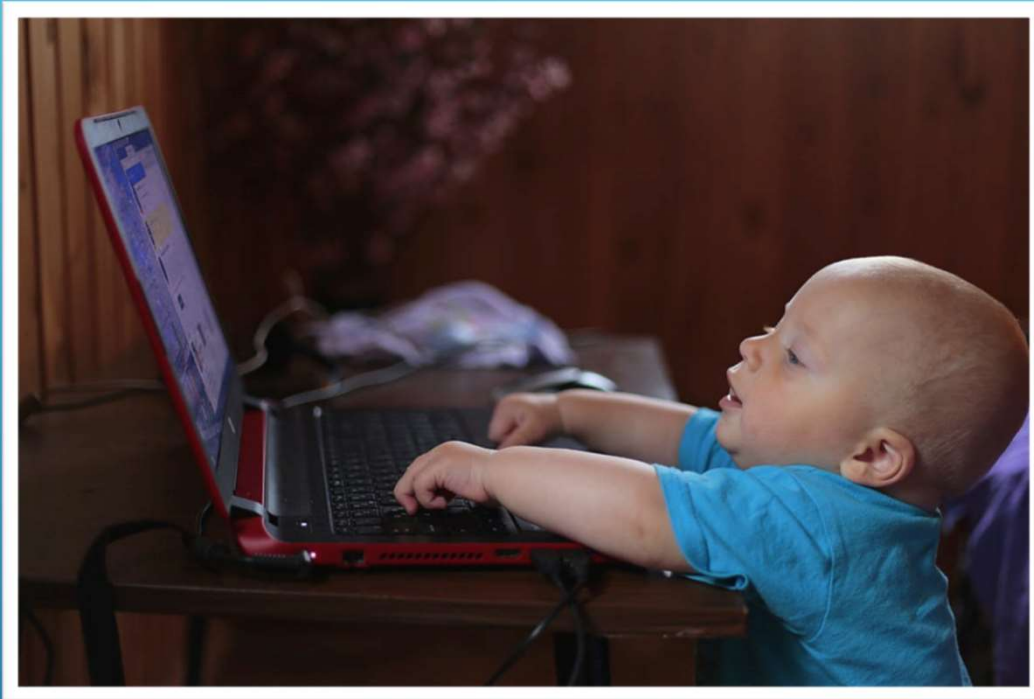




DIY WETTERSTATION V2

mit ESP8266 Mikrocontroller

DIY Wetterstation
DHT II | BME280 | UV-Sensor
Stefan Draeger Software
<https://draeger-it.blog.mh.hk.rs>



AGENDA

- »»» Vorstellungsrunde
- »»» Was wollen wir machen?
- »»» Einrichten der Entwicklungsumgebung
- »»» Aufbau der Platine
- »»» Programmieren der Sensoren & Aktoren
- »»» Anbindung an ThingSpeak
- »»» Programmieren einer Webseite mit Sensordaten

- ▶ Name: Stefan Draeger
- ▶ Alter: 42
- ▶ Familienstand: verheiratet, 2 Kinder
- ▶ Beruf: Softwareentwickler
- ▶ Hobbies: Mikrocontroller, Garten, Haus



VORSTELLUNGSRUNDE

Einen Mikrocontroller programmieren

- Arduino IDE einrichten & benutzen
 - C/C++ kennenlernen
- Mikrocontroller Wemos D1 Mini
- Programmieren von Sensoren & Aktoren
- Aufbau einer WiFi Verbindung
 - Einrichten des Mikrocontrollers als AccessPoint

Einen IoT Dienst ansprechen

- Einrichten
 - eines Accounts
 - eines Dashboards
- Aufbau der Verbindung zu ThingSpeak

Eine Webseite ausliefern

- programmieren einer einfachen HTML Seite
- Einrichten des ESP8266 als Webserver

WAS WOLLEN WIR MACHEN?

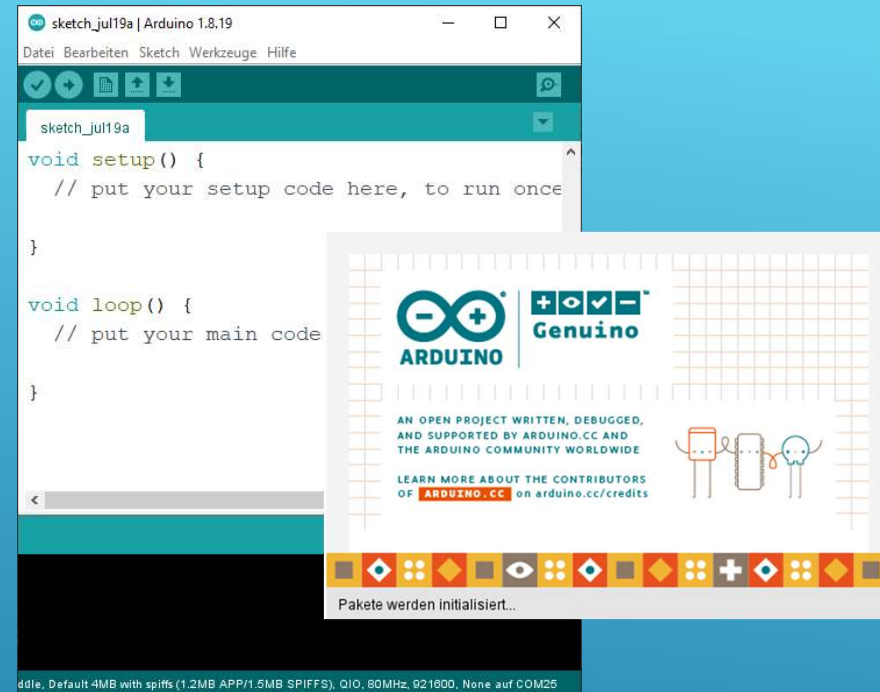


▶ Downloads & andere Ressourcen auf USB Stick bei mir 😊

RESSOURCEN

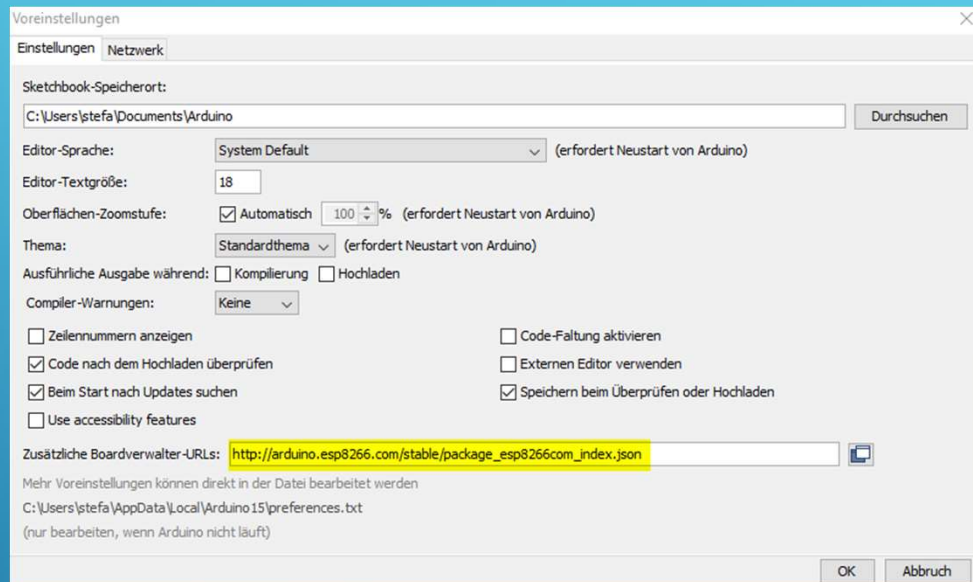
A decorative graphic consisting of several parallel white lines of varying lengths, slanted upwards from left to right, located in the bottom right corner of the slide.

- ▶ kostenloser Download unter <https://arduino.cc>
- ▶ verfügbar für Windows, macOS & Linux
- ▶ als EXE-Datei oder ZIP-Datei



EINRICHTEN DER ARDUINO IDE

Download & Installieren

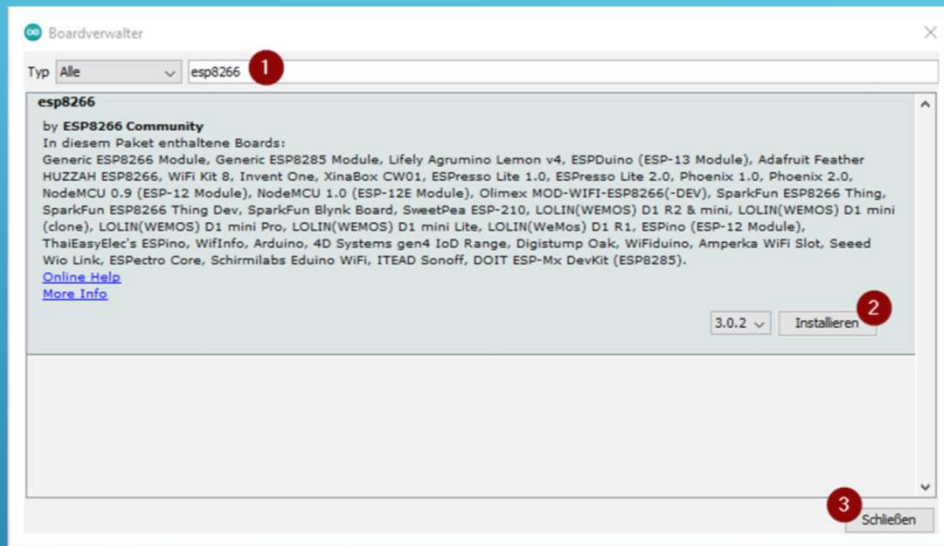


► zusätzliche URL für den Boardverwalter

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

EINRICHTEN DER ARDUINO IDE

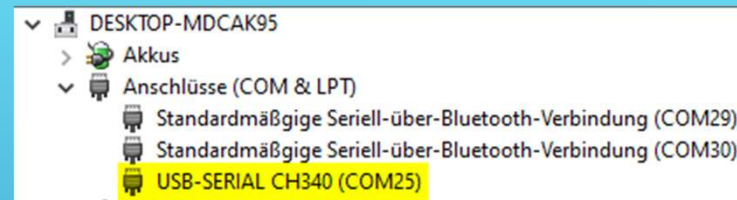
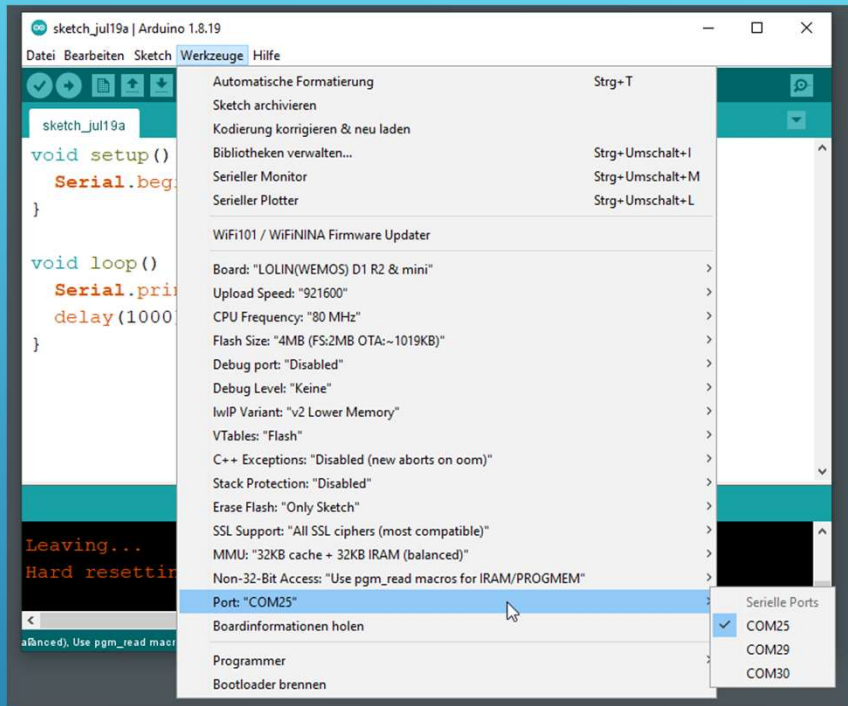
Einrichten des ESP8266



- ▶ Schritt 1 – suchen nach „esp8266“
- ▶ Schritt 2 – Schaltfläche „Installieren“ betätigen
- ▶ Schritt 3 – Schaltfläche „Schließen“ betätigen

EINRICHTEN DER ARDUINO IDE

Installieren des Boardtreibers



- ▶ Hauptmenü „Werkzeuge“ > „Board: xyz“
 - > „ESP8266 Boards (3.0.2)“
 - > „LOLIN(WEMOS) D1 R2 & mini“
- ▶ Port ggf. aus dem Geräte-Manager ablesen

EINRICHTEN DER ARDUINO IDE

Board wählen

Dateiname

Version der IDE



zusätzliche Datei
anlegen

unten rechts gewählter
Mikrocontroller

Überprüfen

Hochladen

Neu

Öffnen

Speichern

Serieller Monitor

LOLIN(WEMOS) D1 R2 & mini, 80 MHz, Flash, Disabled (new aborts on oom), Disabled, All SSL ciphers (most compatible), 32KB cache + 32KB IRAM (balanced), Use pgm_read macros for IRAM/PROGMEM, 4MB (FS:2MB OTA~1019KB), v2 Lower Memory, Disabled, None, Only Sketch, 921600 auf COM25

ARDUINO IDE

Toolbar & Funktionen

```
void setup() {  
  // put your setup code here, to run once:  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
}
```

- ▶ Funktion „setup“ wird einmalig ausgeführt wenn der Mikrocontroller gestartet wird.
- ▶ Funktion „loop“ wird dauerhaft ausgeführt.

ARDUINO IDE

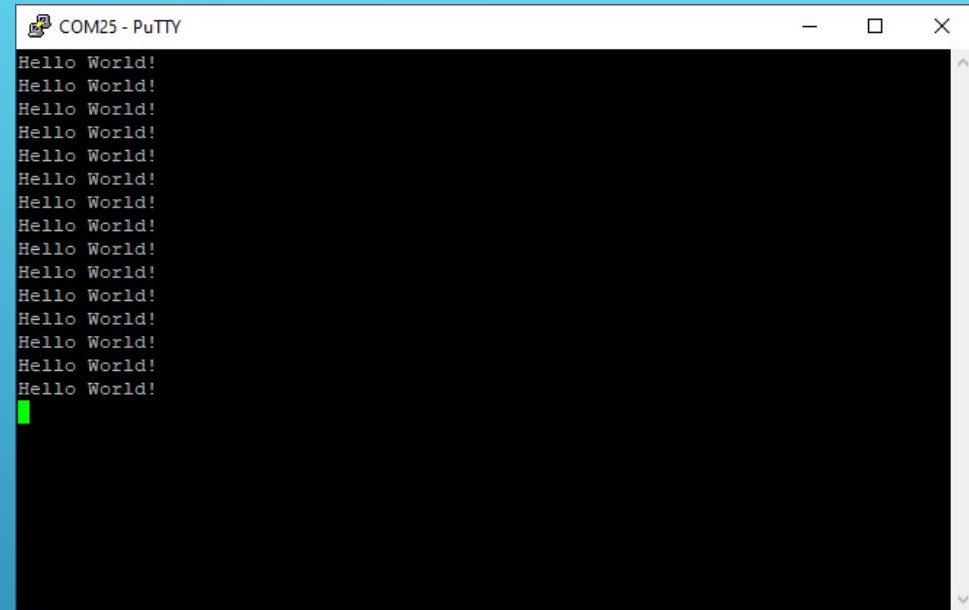
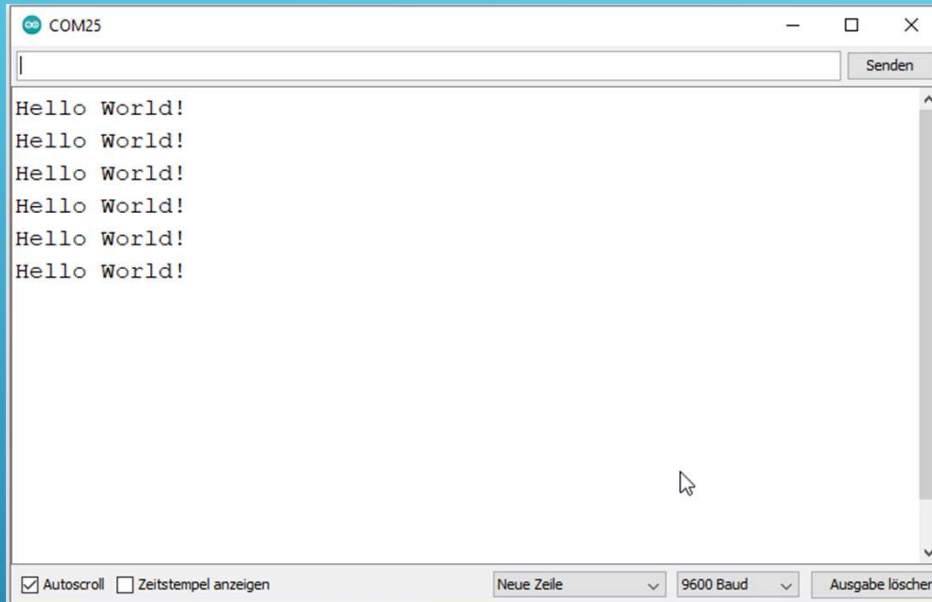
Funktionen setup & loop

```
void setup() {  
  //beginn der seriellen Kommunikation mit  
  //9600 baud  
  Serial.begin(9600);  
}  
  
void loop() {  
  //Ausgabe des Textes "Hello World!" auf  
  //der seriellen Schnittstelle  
  Serial.println("Hello World!");  
  //eine kleine Pause von 1 Sekunde (1000 ms.)  
  delay(1000);  
}
```

- ▶ es wird auf der seriellen Schnittstelle der Text "Hello World!" ausgeführt
- ▶ lesbar zbsp. im seriellen Monitor der Arduino IDE oder auch über Putty

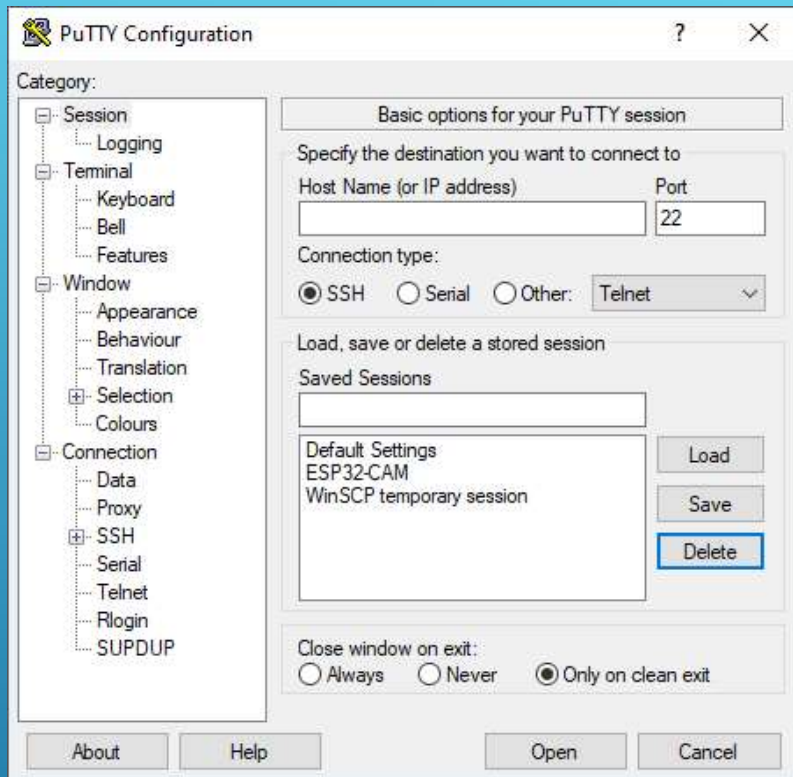
ARDUINO IDE

Ein kleines Beispiel



ARDUINO IDE

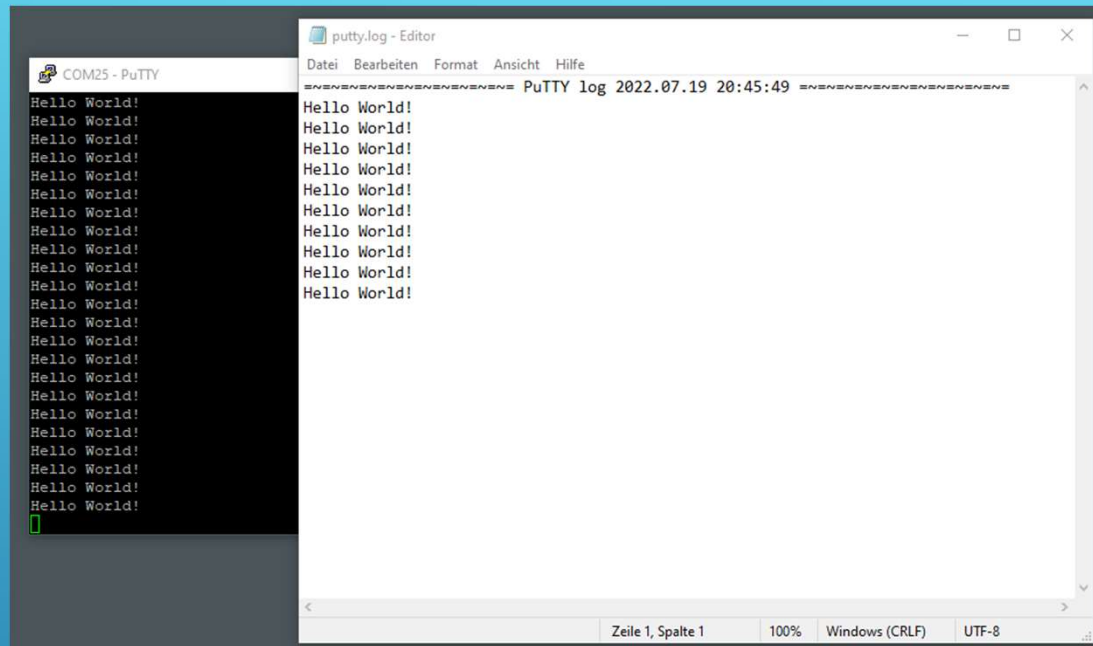
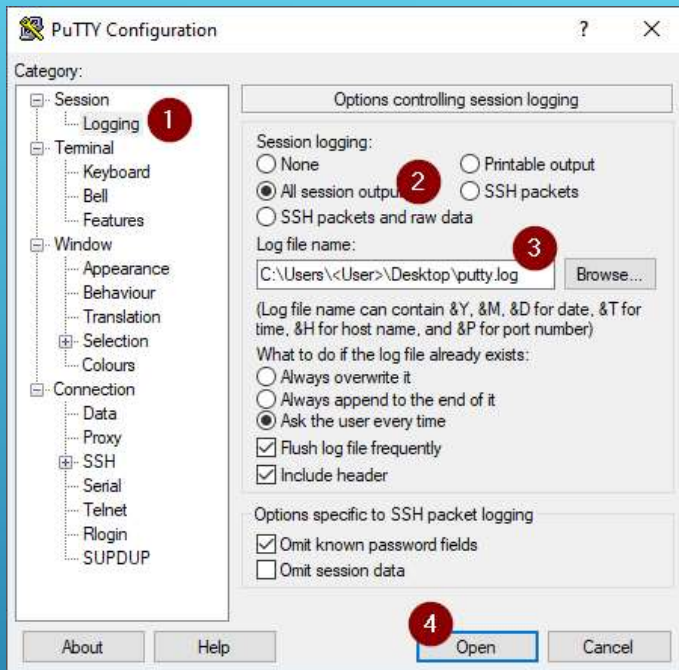
Ausgabe im seriellen Monitor & putty



- ▶ Download unter putty.org
- ▶ Programm zum Aufbau einer Verbindung zu einem entfernten Computer / Gerät per SSH, Serial oder anderen Schnittstellen
- ▶ Logging der Ausgaben in Datei möglich
- ▶ viele nützliche Features

EXKURS PUTTY

Was ist putty?



EXKURS PUTTY

Loggen der Daten in eine Datei

```
//ein Kommentar

/*
  ein mehrzeiliger Kommentar
*/

//eine Ganze Zahl
int zahl0 = 1;
long zahl1 = 1L;

//eine Gleitkomma Zahl
float zahl2 = 1.2f;
double zahl3 = 1.2d;

//ein Array
int liste[3] = {1,2,3};
```

- ▶ Kommentare am Code helfen diesen zu verstehen
- ▶ speichern von Werte in Feldern, Variablen
- ▶ Ganze Zahlen als Integer, Long
- ▶ Gleitkommazahlen als Float oder Double
- ▶ eine Liste von Werten als Array

C/C++ PROGRAMMIEREN

kurze Einführung


```
void setup() {  
  Serial.begin(9600);  
}  
  
void eineFunktion() {  
  //do something  
}  
  
String eineWeitereFunktion() {  
  return "Text";  
}  
  
String eineFunktionMitParameter(String name, String vorname) {  
  return name + " " + vorname;  
}
```

- ▶ Funktionen dienen zum strukturieren von Code
- ▶ redundanter Code entfällt

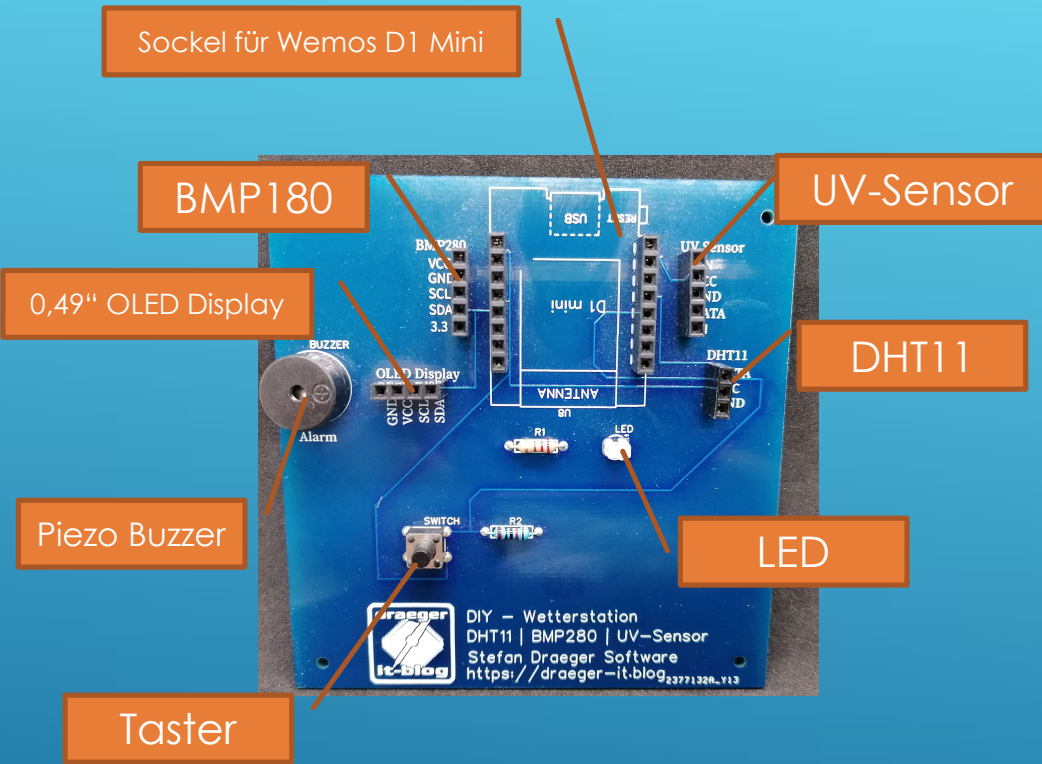
C/C++ PROGRAMMIEREN

Funktionen

- ▶ Funktionen für
 - ▶ addition,
 - ▶ subtraktion,
 - ▶ multiplikation,
 - ▶ division
- zweier Zahlen

C/C++ PROGRAMMIEREN

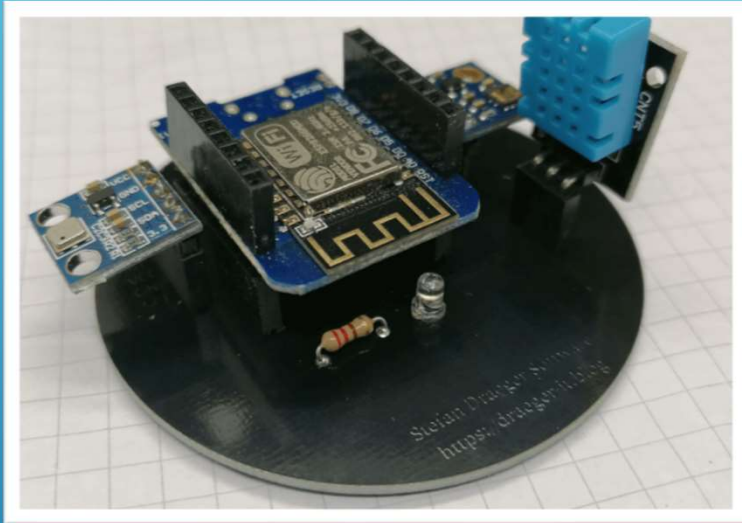
Einen kleinen Taschenrechner programmieren



- ▶ Platine verfügt über Buchsenleisten für die Sensoren
 - ▶ BMP180
 - ▶ UV-Sensor
 - ▶ DHT11
 - ▶ 0,42" OLED Display
- ▶ Piezo Buzzer für Soundausgabe
- ▶ LED, und
- ▶ Taster

PLATINE „DIY WETTERSTATION V2“

Aufbau & Überblick



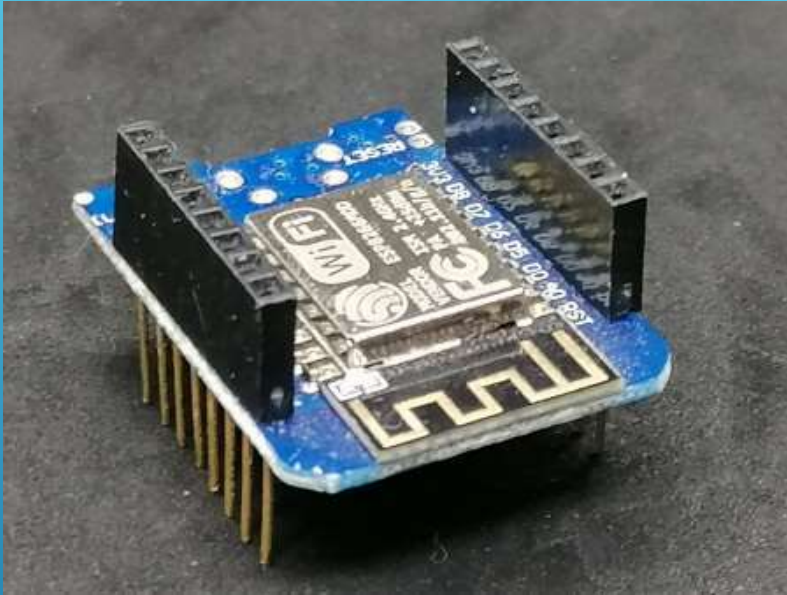
Version 1 der „DIY Wetterstation“

- ▶ Der WiFi Chip wird bei betrieb warm und erzeugt somit im Gehäuse in Klima welches die Werte der Sensoren verfälscht.
- ▶ Sensoren können besser platziert werden (zbsp. UV-Sensor an die Oberfläche)

PLATINE „DIY WETTERSTATION V2“

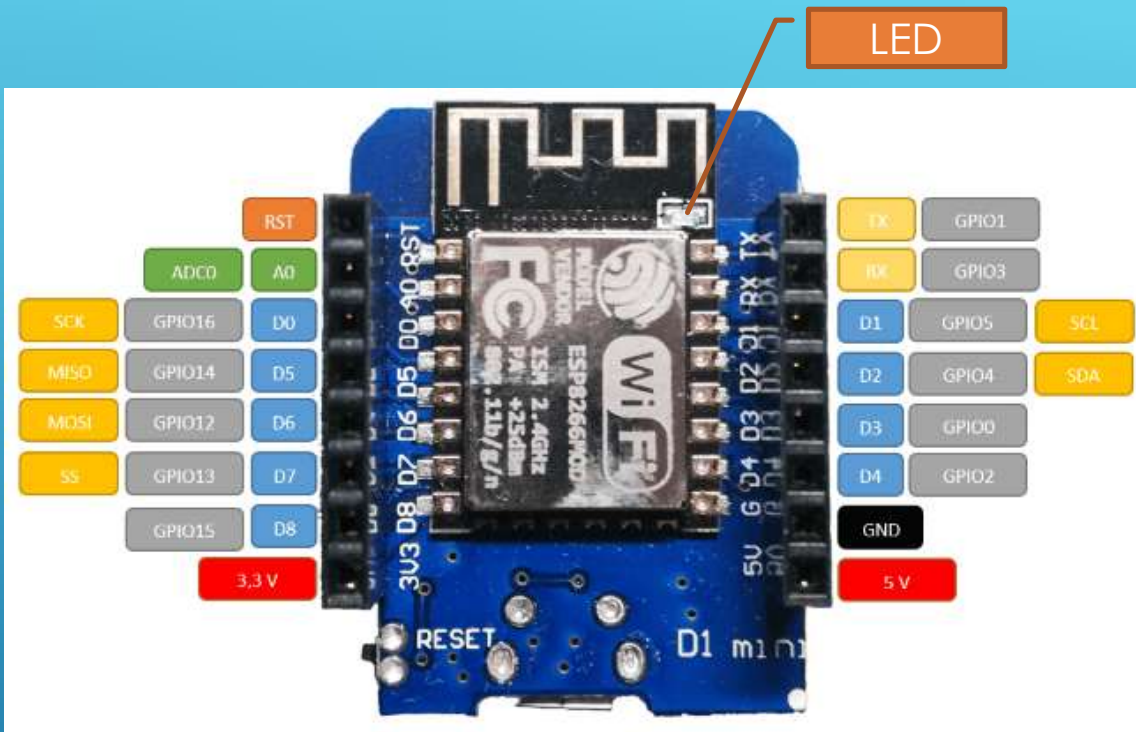
Warum die Sensoren auf Buchsenleisten?

Spezifikation



Länge	34,2 mm
Breite	25,6 mm
Gewicht	10 g
Flash Speicher	4 Mbyte
CPU Taktgeschwindigkeit	80 MHz / 160 MHz
Betriebsspannung	3.3 V
max. Stromstärke pro I/O Pin	20 mA
max. Stromstärke für die 3.3 V Versorgung	50 mA
digitale Eingänge / Ausgänge	11
digitale PWM Ausgänge	11
analoge Eingänge / Ausgänge	1

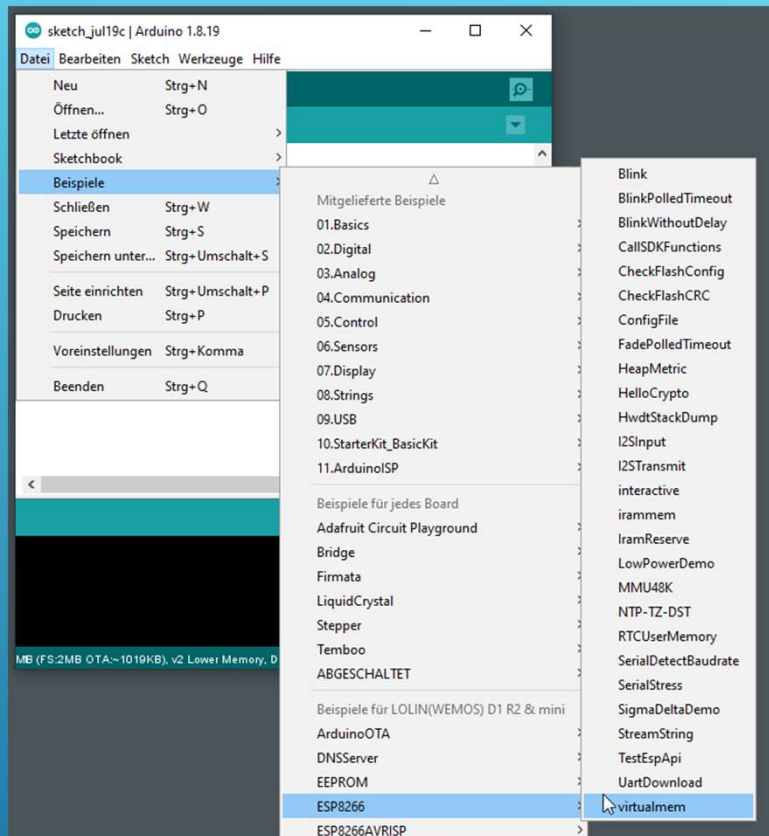
WEMOS D1 MINI (ESP8266)



- ▶ Pins für 3.3V & 5V
- ▶ Schnittstellen I²C, SPI, UART
- ▶ ein analoger Eingang
- ▶ 11 digitale Ein/Ausgänge
 - ▶ davon 10 als PWM

WEMOS D1 MINI (ESP8266)

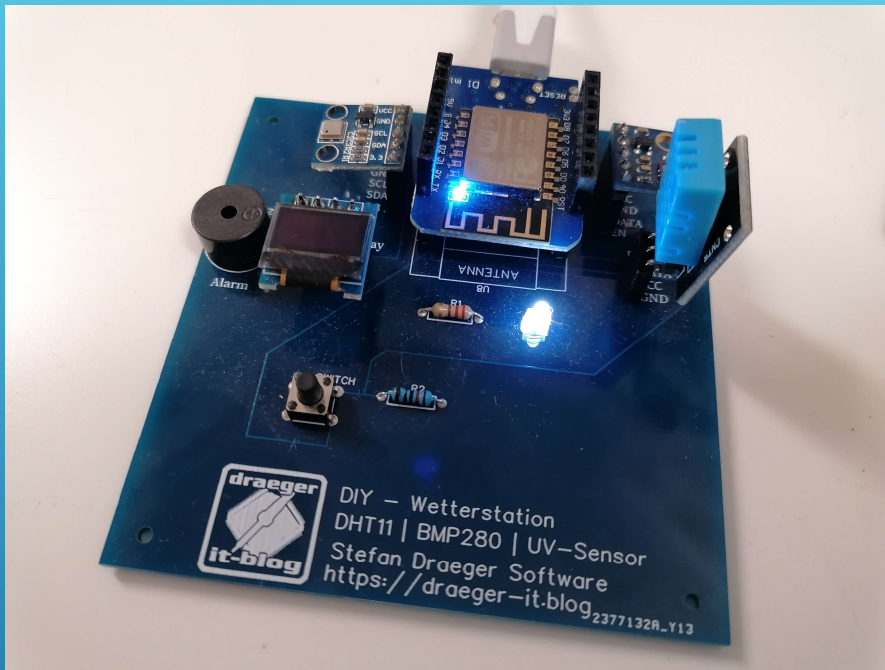
Pinout



► Beispiele unter „Datei“ > „Beispiele“ > „ESP8266“

WEMOS D1 MINI (ESP8266)

Beispiele in der Arduino IDE



LED blinken

LED Fade Effekt

WEMOS D1 MINI (ESP8266)

Beispiele


```
#define button D5

void setup() {
  Serial.begin(9600);
  pinMode(button, INPUT);
}

void loop() {
  int state = digitalRead(button);
  if(state == HIGH){
    Serial.println("Taster ist gedrückt!");
  }
}
```

- ▶ digitaler Pin muss als Eingang definiert werden
- ▶ Status des Tasters auslesen mit `digitalRead(<Pin>)`
 - ▶ HIGH = Taster gedrückt
 - ▶ LOW = Taster ist nicht gedrückt

ARDUINO IDE

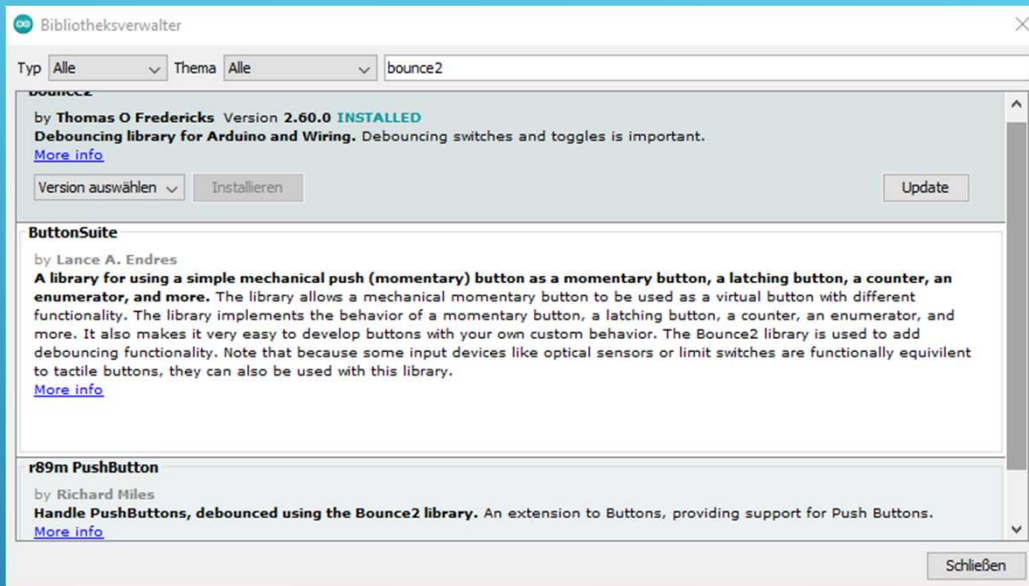
Einen Taster programmieren

```
COM25
12:02:26.048 -> Taster ist gedrückt!
12:02:26.048 -> Taster ist gedrückt!
12:02:26.048 -> Taster ist gedrückt!
12:02:26.094 -> Taster ist gedrückt!
12:02:26.094 -> Taster ist gedrückt!
12:02:26.142 -> Taster ist gedrückt!
12:02:26.142 -> Taster ist gedrückt!
12:02:26.188 -> Taster ist gedrückt!
12:02:26.188 -> Taster ist gedrückt!
12:02:26.235 -> Taster ist gedrückt!
12:02:26.281 -> Taster ist gedrückt!
12:02:26.281 -> Taster ist gedrückt!
```

- ▶ Mikrocontroller arbeitet mit max. 160 MHz und kann die Operation viele Millionen mal pro Sekunde ausführen
- ▶ Status wird somit mehrfach gelesen obwohl der Taster nur kurz betätigt wurde
- ▶ Wie entprellt man einen Taster?

ARDUINO IDE

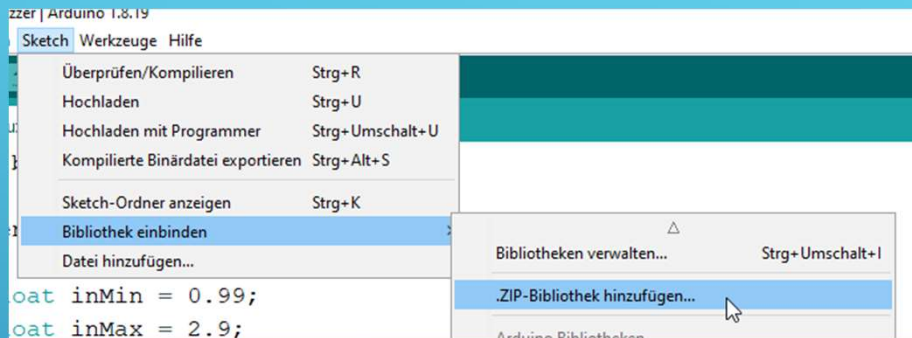
Taster prellt



- ▶ Bibliotheken ermöglichen einfachen Zugriff auf komplexe Programmierung
- ▶ einfacher Zugriff auf Sensoren & Aktoren
- ▶ „Sketch“ > „Bibliothek einbinden“ > „Bibliotheken verwalten...“

ARDUINO IDE

Taster entprellen & Installieren einer Bibliothek



► Öffnen der ZIP-Datei

- „Sketch“ > „Bibliothek einbinden“ > „.ZIP-Bibliothek hinzufügen...“

- Datei muss einem definierten Format entsprechen

Bibliothek wurde zu Ihren Bibliotheken hinzugefügt. Bitte im Menü "Bibliothek einbinden" nachprüfen

ARDUINO IDE

alternative Installation einer ZIP-Datei

The screenshot shows the Arduino IDE interface. The main editor window displays the following code:

```
#include <Bounce2.h>

#define button D5

Bounce btn = Bounce(button, 50);

void setup() {
  Serial.begin(9600);
  pinMode(button, INPUT);
}

void loop() {
  btn.update();

  if(btn.rose()){
    Serial.println("Taster gedrückt");
  }

  if(btn.fell()){
    Serial.println("Taster losgelassen");
  }
}
```

The serial monitor window (COM25) shows the following output:

```
12:28:18.788 -> Taster gedrückt
12:28:18.927 -> Taster losgelassen
12:28:19.767 -> Taster gedrückt
12:28:20.848 -> Taster losgelassen
12:28:21.689 -> Taster gedrückt
12:28:22.485 -> Taster losgelassen
12:28:23.283 -> Taster gedrückt
12:28:23.936 -> Taster losgelassen
```

At the bottom of the serial monitor, there are checkboxes for "Autoscroll" and "Zeitstempel anzeigen", both of which are checked.

- ▶ Schritt 1 - einbinden der Bibliothek
- ▶ Schritt 2 – initialisieren eines Objektes vom Typ Bounce mit Parameter button und Millisekunden für die Pause,
- ▶ Schritt 3 – in der Loop, aktualisieren des Tasterobjektes,
- ▶ Schritt 4 – auslesen des Tasters mit
 - ▶ rose() – Taster gedrückt
 - ▶ fell() – Taster losgelassen

ARDUINO IDE

Taster entprellen mit bounce2



LED aktivieren
mit dem Taster

LED bei
Tastendruck
aktivieren /
deaktivieren

WEMOS D1 MINI (ESP8266)

Beispiele



- ▶ Ausgabe einfacher Töne & kleine Lieder
- ▶ kann als Alarm für einen Schwellwert eingesetzt werden

ARDUINO IDE

Soundausgabe


```
//Piezo Buzzer am
//digitalen Pin D8 angeschlossen
#define buzzer D8

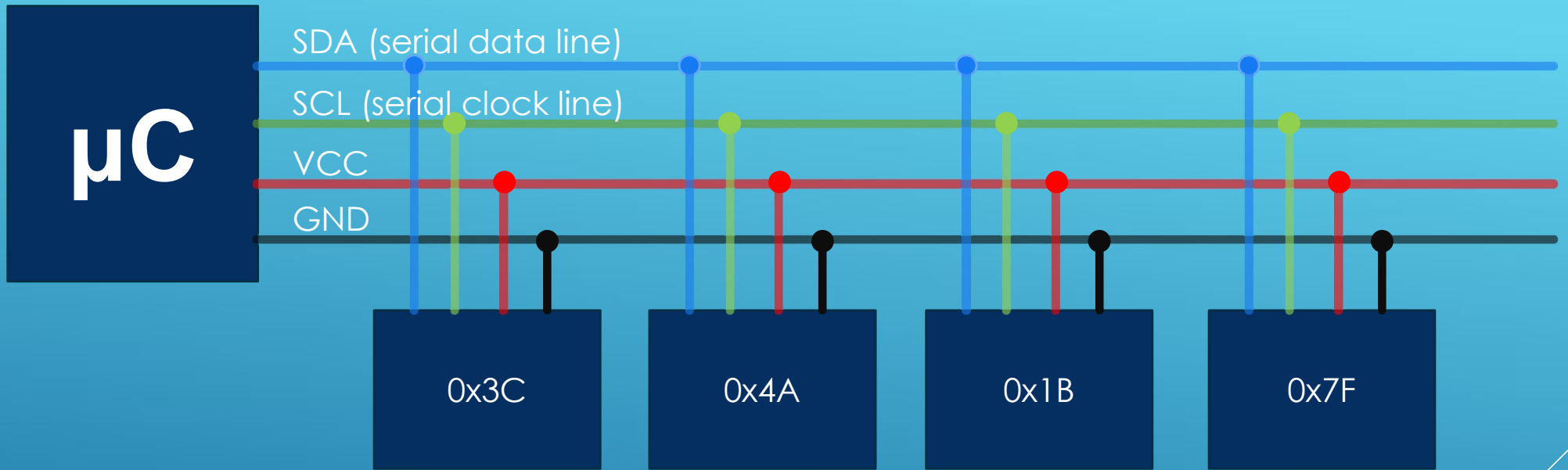
void setup() {
  //Pin des Buzzers als Ausgang definieren
  pinMode(buzzer, OUTPUT);
}

void loop() {
  //Ausgabe von einem Ton mit 750 Hertz
  tone(buzzer, 750);
  //eine kleine Pause von 500ms.
  delay(500);
  //den Ton deaktivieren
  noTone(buzzer);
}
```

ARDUINO IDE

Ausgabe von Tönen

- ▶ Erzeugen eines Tones mit der Funktion
 - ▶ `tone(<Pin>, <Frequenz>)`
 - ▶ `Tone(<Pin>, <Frequenz>, <Dauer>)`



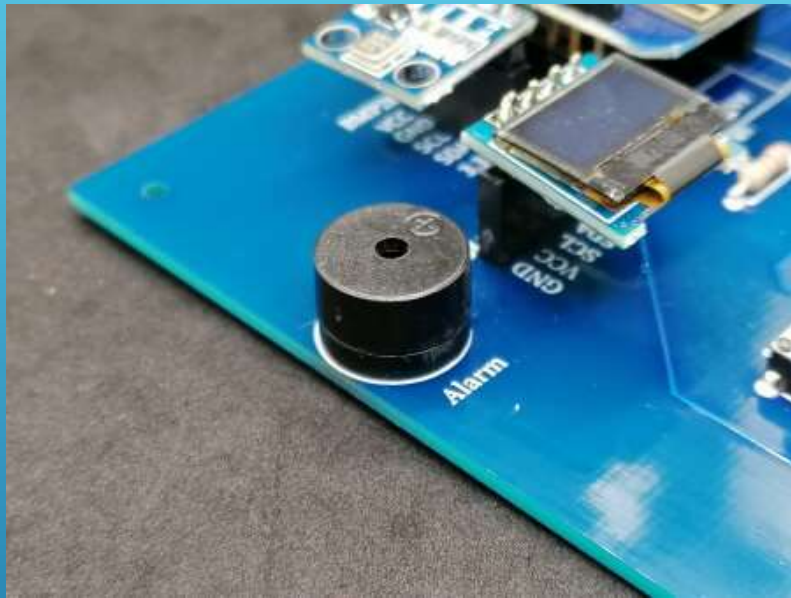
EXKURS I²C KOMMUNIKATION

```
COM25
18:02:23.775 -> ??
18:02:23.883 -> I2C Scanner
18:02:23.883 -> Scanning...
18:02:23.883 -> I2C device found at address 0x3C !
18:02:23.931 -> done
18:02:23.931 ->
```

- ▶ URL: <https://playground.arduino.cc/Main/I2cScanner/>
- ▶ ermöglicht die einfache suche nach Sensoren welche per I²C angebunden sind

EXKURS I²C KOMMUNIKATION

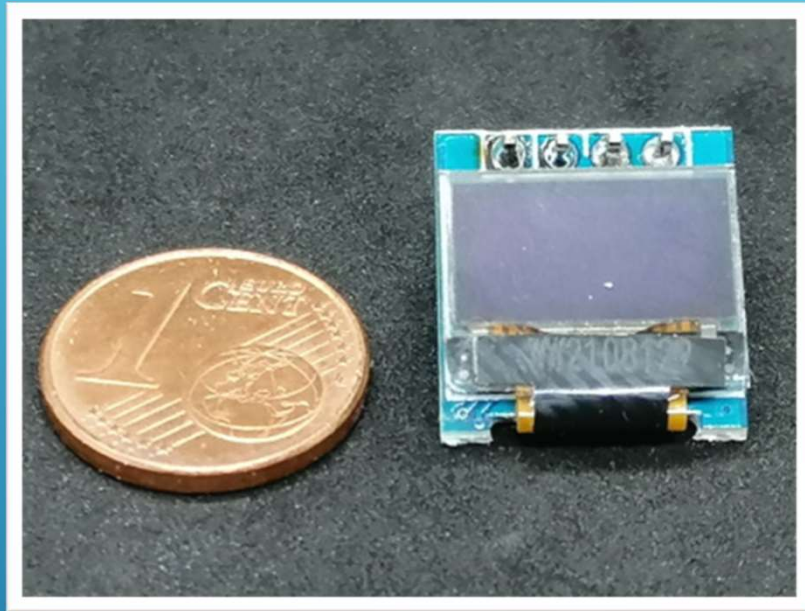
I²C Scanner



Abspielen von
Tönen per
Tastendruck

WEMOS D1 MINI (ESP8266)

Beispiele



- ▶ Auflösung 64 x 32 Punkte
- ▶ Schnittstelle I²C

ARDUINO IDE

0,49" OLED Display programmieren

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define OLED_RESET 0
Adafruit_SSD1306 display(OLED_RESET);

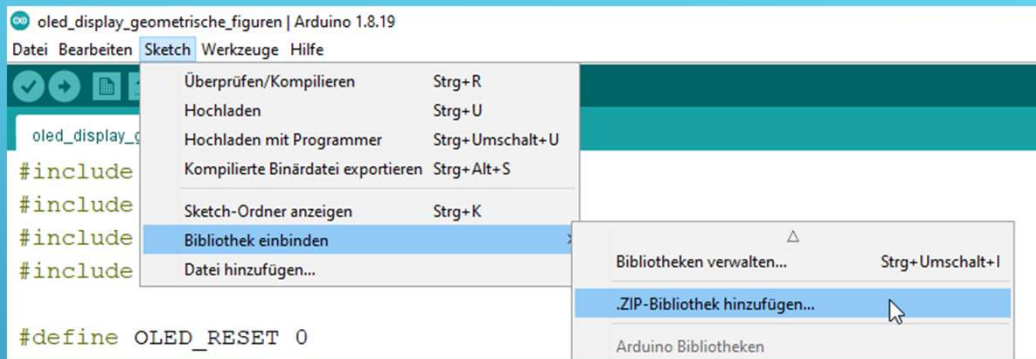
void setup() {
  Serial.begin(9600);
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.display();
  delay(2000);
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(60, 15);
  display.println("Hello");
  display.display();
}

void loop() {}
```

- ▶ I2C Adresse des Displays 0x3C
- ▶ Textgröße 1
- ▶ Textfarbe – weiß
- ▶ Cursor Position
 - ▶ Spalte 60
 - ▶ Zeile 15

ARDUINO IDE

Anzeigen des Textes „Hello“

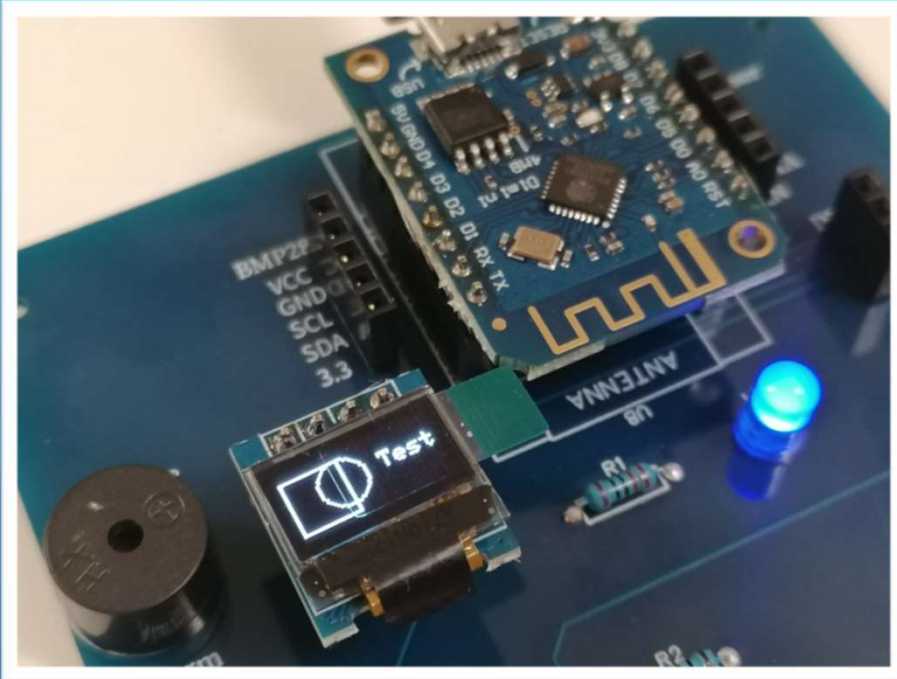


► Benötigte Bibliotheken:

- Adafruit_BusIO
- Adafruit_SSD1306_Wemos_OLED
- Adafruit-GFX-Library

ARDUINO IDE

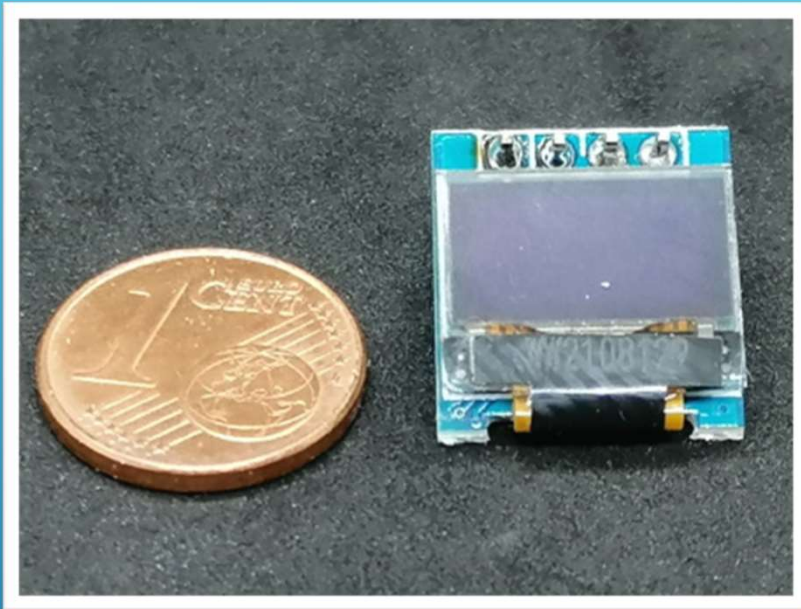
Installieren der benötigten Bibliotheken



- ▶ gedacht für große OLED Displays
- ▶ viele nützliche Funktionen
 - ▶ Schriftart, Schriftgröße, Farbe
 - ▶ Zeichnen von einfachen geometrischen Figuren
 - ▶ Zeichnen von XBMP Bilder
- ▶ weitere Informationen unter https://adafruit.github.io/Adafruit_SSD1306/html/class_adafruit___s_s_d1306.html

ARDUINO IDE

Besonderheiten bei der Adafruit Bibliothek

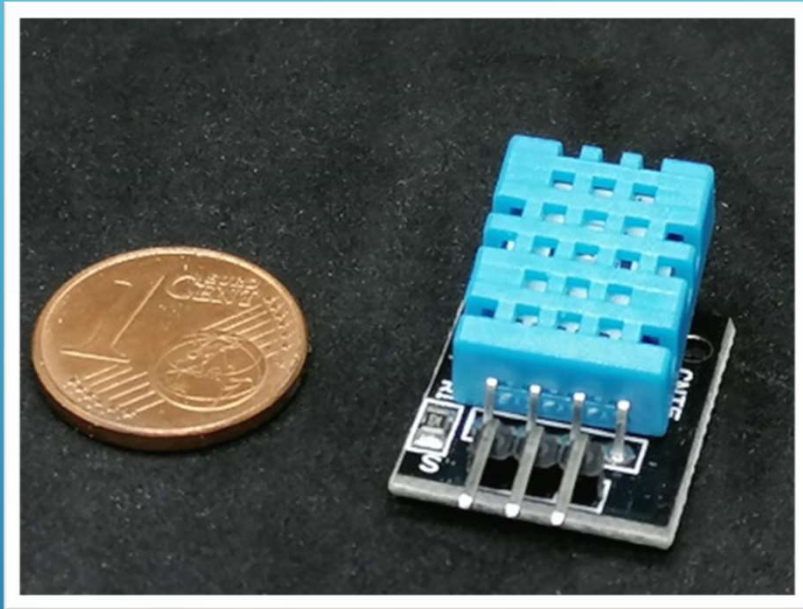


Anzeigen von
Text

Anzeigen von
geometrischen
Figuren

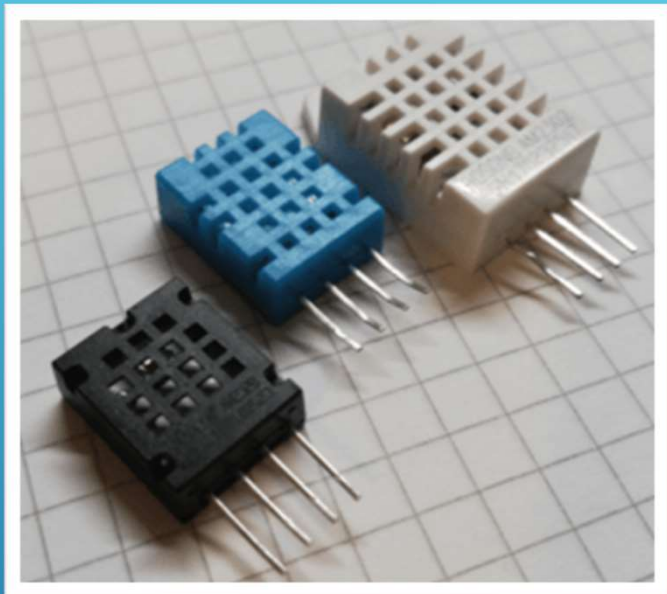
WEMOS D1 MINI (ESP8266)

Beispiele



- Technische Daten:
 - rel. Luftfeuchtigkeit
 - Messbereich 20 % bis 90 %
 - Toleranz ± 5 %
 - Temperatur
 - Messbereich 0 bis 60 °C
 - Toleranz ± 2 °C
 - Betriebsspannung 5 V

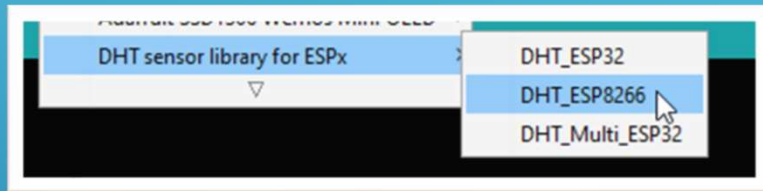
ARDUINO IDE
Sensor DHT11



- ▶ DHT11 – blau
- ▶ DHT22 – weiß
 - ▶ deutlich genauer,
 - ▶ teurer
- ▶ AM2320 – schwarz
 - ▶ deutlich genauer,
 - ▶ anderes Pinout,
 - ▶ teurer

ARDUINO IDE

Alternativen zum DHT11



- ▶ Bibliothek DHTesp
- ▶ Beispiele für ESP32, ESP8266

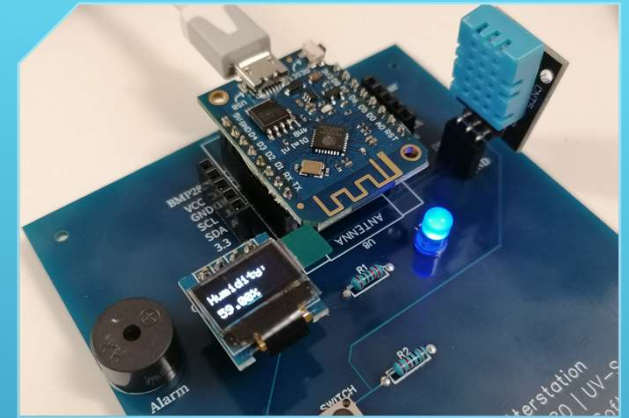
ARDUINO IDE
Installieren der benötigten bibliothek

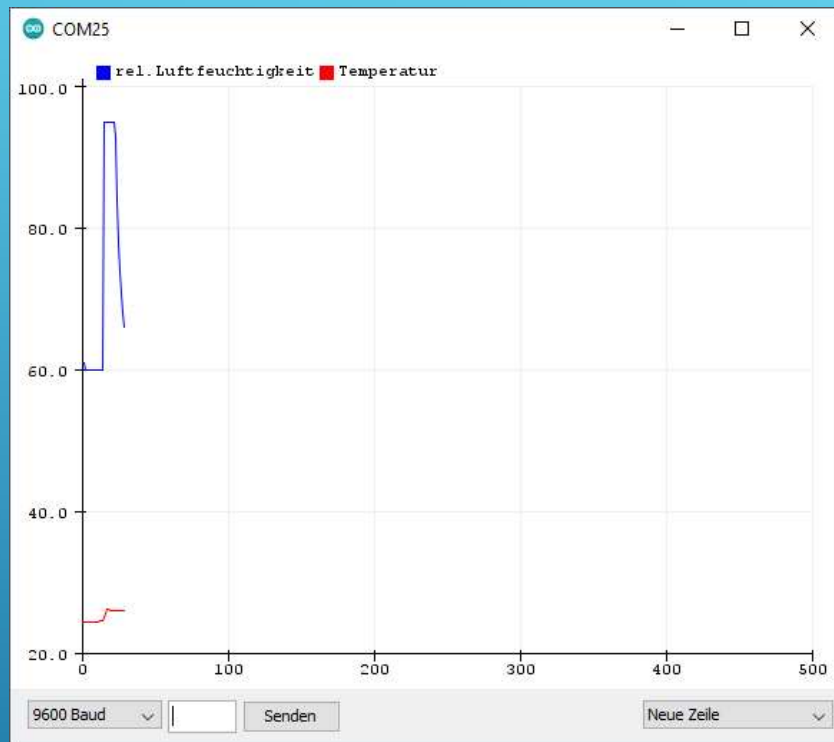
Ausgabe auf
dem seriellen
Monitor

Anzeigen auf
dem OLED
Display

WEMOS D1 MINI (ESP8266)

Beispiele

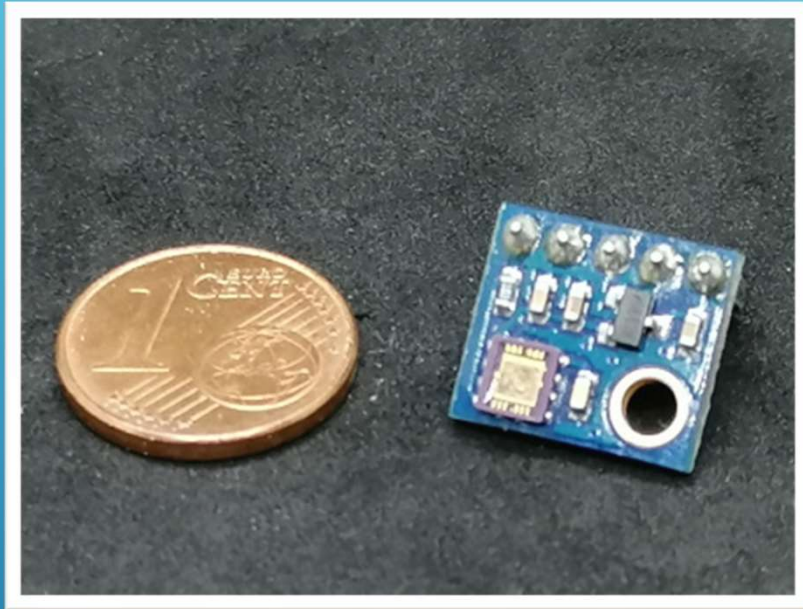




- ▶ visualisieren von Daten eines oder mehrerer Sensoren

```
String line1 = "rel.Luftfeuchtigkeit Temperatur";  
String line2 = String(luftfeuchtigkeit, DEC) + " " + String(temperatur, DEC);  
Serial.println(line1);  
Serial.println(line2);
```

ARDUINO IDE
serieller Plotter



ARDUINO IDE

analoger UV-Sensor

- ▶ Technische Daten
 - ▶ Betriebsspannung 3,3 V
 - ▶ maximale Stromaufnahme 0,3 mA
 - ▶ Ruhestrom 0,1 μ A

```
uv_sensor_serial (Arduino 1.8.19)
File: Bearbeiten Sketch Werkzeuge Hilfe
uv_sensor_serial
pinMode(uvSensorIn, INPUT);
Serial.begin(9600); //Starten der Seriellen Kommunikation mit 9600 baud
}

void loop() {
  int uvLevel = averageAnalogRead(
  float outputVoltage = 3.3 * uvLe
  float uvIntensity = mapfloat(out
  Serial.print(" UV Intensität: ")
  Serial.println(uvIntensity);
  Serial.println(" mW/cm^2");
  delay(200); //Pause von 200ms.
}

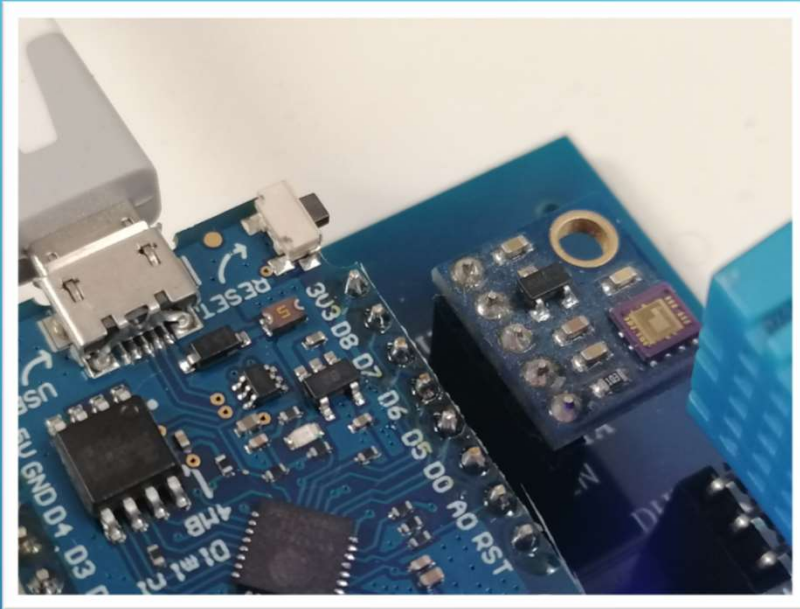
//Ließt den PIN x mal und berechne
//Durchschnittswert und liefert di
int averageAnalogRead(int pinToRea
  unsigned int runningValue = 0;
  for(int x = 0 ; x < numberOfRead
    runningValue += analogRead(pin
  }
  return(runningValue / numberOfRe
}

//Arduino von Sketch unterstützt kein float daher wurde diese neu implementiert
```

```
COM25
17:19:53.848 -> UV Intensität: 1.18 mW/cm^2
17:19:54.023 -> UV Intensität: 1.18 mW/cm^2
17:19:54.251 -> UV Intensität: 1.18 mW/cm^2
17:19:54.436 -> UV Intensität: 1.21 mW/cm^2
17:19:54.666 -> UV Intensität: 1.18 mW/cm^2
17:19:54.850 -> UV Intensität: 1.18 mW/cm^2
17:19:55.067 -> UV Intensität: 1.18 mW/cm^2
17:19:55.233 -> UV Intensität: 1.18 mW/cm^2
17:19:55.457 -> UV Intensität: 1.18 mW/cm^2
17:19:55.644 -> UV Intensität: 1.18 mW/cm^2
17:19:55.831 -> UV Intensität: 1.18 mW/cm^2
17:19:56.061 -> UV Intensität: 1.18 mW/cm^2
17:19:56.233 -> UV Intensität: 1.18 mW/cm^2
17:19:56.451 -> UV Intensität: 1.18 mW/cm^2
17:19:56.636 -> UV Intensität: 1.18 mW/cm^2
Autoscroll Zeitstempel anzeigen
```

- Komplizierte Berechnung aus einem analogen Wert (0..1023) zu einem UV Wert

ARDUINO DIE Berechnung des UV-Wertes

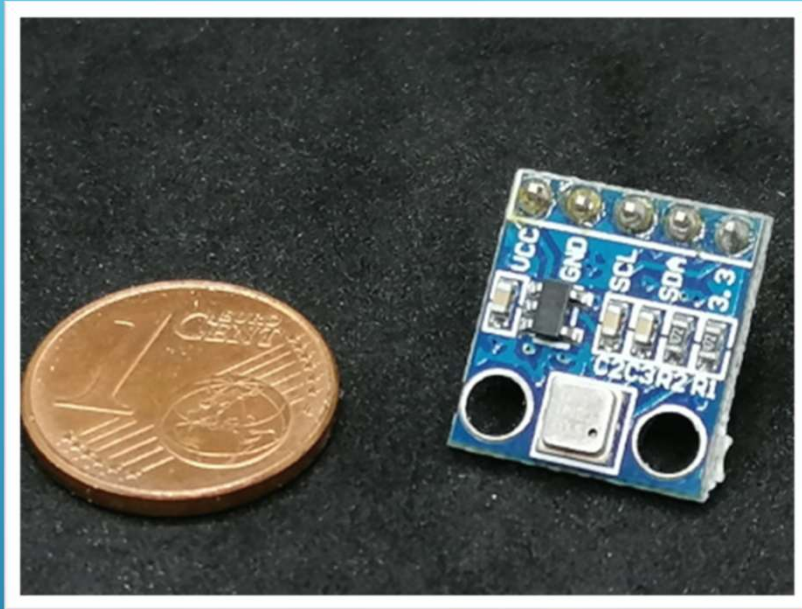


Anzeigen auf
dem OLED
Display

Ausgabe von
Sound bei
erreichen eines
Schwellwertes

WEMOS D1 MINI (ESP8266)

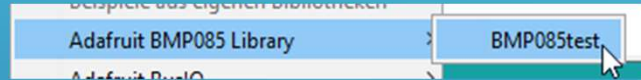
Beispiele



ARDUINO IDE
BMP180

► Technische Daten

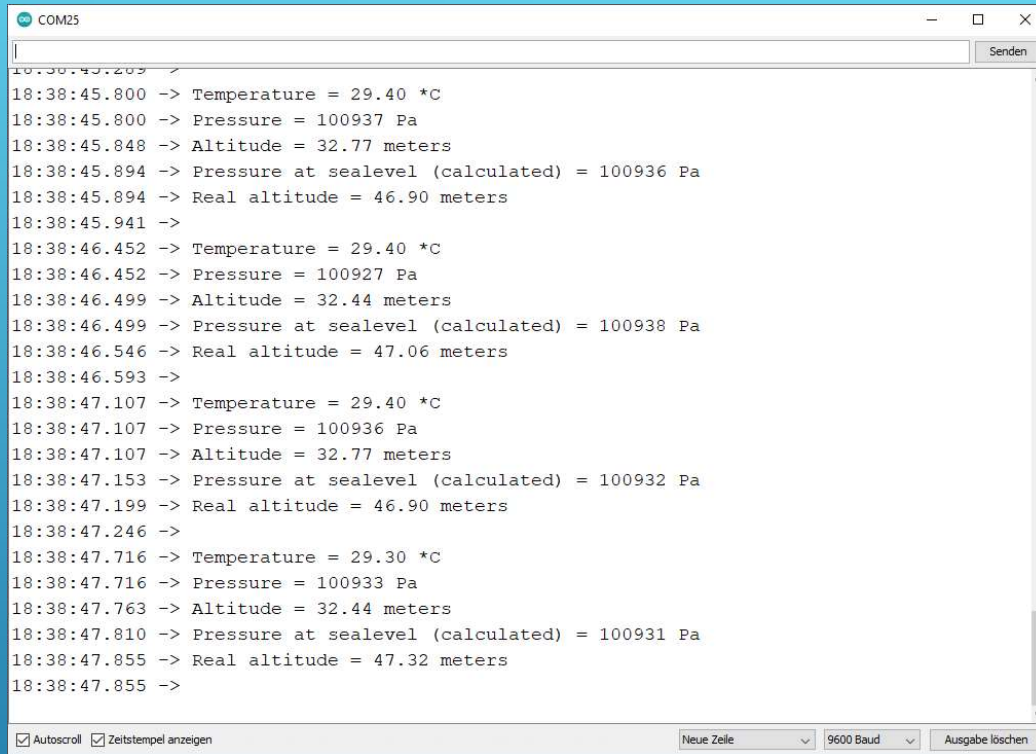
- Betriebsspannung 1.8 V bis 3.6 V
- Leistungsaufnahme 0,5 μ A bei 1Hz
- Genauigkeit 0.02 hPa
- Messbereich 300 hPa bis 1100 hPa



► Bibliothek Adafruit-BMP085-Library

ARDUINO IDE

Installieren der benötigten Bibliothek

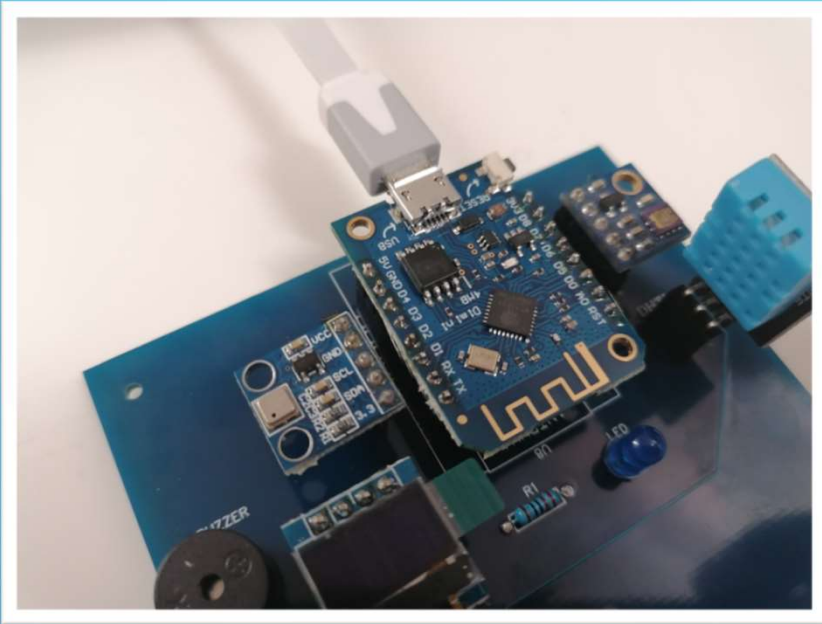


The screenshot shows a serial monitor window titled 'COM25'. The window contains a stream of text representing sensor data. Each line starts with a timestamp in the format 'HH:MM:SS.SSS' followed by a right-pointing arrow and a line of sensor data. The data includes temperature in degrees Celsius, pressure in Pascals (Pa), altitude in meters, and calculated sea level pressure in Pa. The window also features a 'Senden' button at the top right and a status bar at the bottom with options for 'Autoscroll', 'Zeitstempel anzeigen', 'Neue Zeile', '9600 Baud', and 'Ausgabe löschen'.

```
18:38:45.800 -> Temperature = 29.40 *C
18:38:45.800 -> Pressure = 100937 Pa
18:38:45.848 -> Altitude = 32.77 meters
18:38:45.894 -> Pressure at sealevel (calculated) = 100936 Pa
18:38:45.894 -> Real altitude = 46.90 meters
18:38:45.941 ->
18:38:46.452 -> Temperature = 29.40 *C
18:38:46.452 -> Pressure = 100927 Pa
18:38:46.499 -> Altitude = 32.44 meters
18:38:46.499 -> Pressure at sealevel (calculated) = 100938 Pa
18:38:46.546 -> Real altitude = 47.06 meters
18:38:46.593 ->
18:38:47.107 -> Temperature = 29.40 *C
18:38:47.107 -> Pressure = 100936 Pa
18:38:47.107 -> Altitude = 32.77 meters
18:38:47.153 -> Pressure at sealevel (calculated) = 100932 Pa
18:38:47.199 -> Real altitude = 46.90 meters
18:38:47.246 ->
18:38:47.716 -> Temperature = 29.30 *C
18:38:47.716 -> Pressure = 100933 Pa
18:38:47.763 -> Altitude = 32.44 meters
18:38:47.810 -> Pressure at sealevel (calculated) = 100931 Pa
18:38:47.855 -> Real altitude = 47.32 meters
18:38:47.855 ->
```

- ▶ Sensor liefert Werte für
 - ▶ Temperatur,
 - ▶ Luftdruck,
 - ▶ Höhe über den Meeresspiegel

ARDUINO DIE Sensorwerte



Anzeigen auf
dem seriellen
Monitor

Anzeigen auf
dem OLED
Display

WEMOS D1 MINI (ESP8266)

Beispiele

COM25

```
18:11:47.046 -> Starting WiFi scan...
```

```
18:11:49.254 -> 6 networks found:
```

```
18:11:49.254 -> 00: [CH 04] [28:EE:52:22:C6:46] -83dBm * V FRITZBox7590GI24
```

```
18:11:49.254 -> 01: [CH 06] [C0:D7:AA:2F:96:5E] -87dBm * V MagentaWLAN-AZX4
```

```
18:11:49.254 -> 02: [CH 06] [1C:ED:6F:F4:D2:00] -88dBm * V WL-GON
```

```
18:11:49.254 -> 03: [CH 01] [44:4E:6D:17:EF:3E] -86dBm * V WLAN Flur 10 Fritz
```

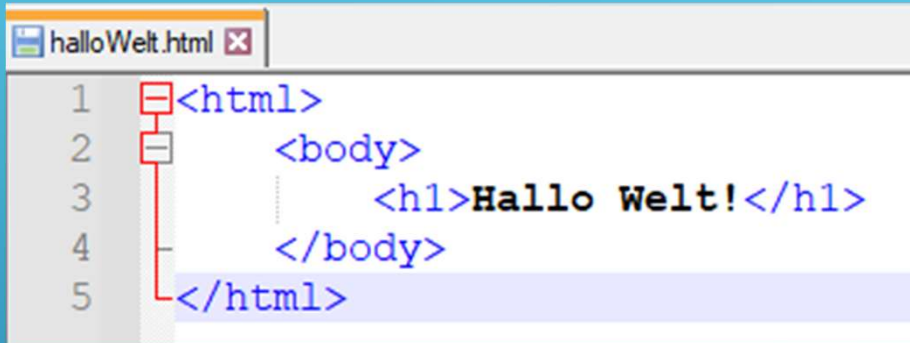
```
18:11:49.254 -> 04: [CH 11] [90:9A:4A:C8:C0:60] -24dBm * V FRITZBox7590GI_EXT
```

```
18:11:49.254 -> 05: [CH 11] [44:4E:6D:BD:F5:9A] -59dBm * V FRITZBox7590GI
```

WIFI SCANNER

```
COM25
|
18:06:16.318 -> rllr$nnnnl?l?b|??r?b?b?nnl?b?b?
18:06:16.404 -> Aufbau der Verbindung zu: FRITZBox7590GI_EXT
18:06:16.404 ->
18:06:16.911 -> .....
18:06:20.636 -> Mit FRITZBox7590GI_EXT erfolgreich verbunden!
18:06:20.636 -> Server gestartet
18:06:20.636 -> Adresse : http://192.168.178.72/
```

AUFBAU EINER WI-FI VERBINDUNG

A screenshot of a code editor window titled 'halloWelt.html'. The editor shows five lines of HTML code. Line 1: <html>. Line 2: <body>. Line 3: <h1>Hallo Welt!</h1>. Line 4: </body>. Line 5: </html>. The code is color-coded: <html> is blue, <body> is purple, <h1> is blue, and the text 'Hallo Welt!' is black. The closing tags </body> and </html> are purple. A red tree view on the left shows the structure of the document.

```
1 <html>
2   <body>
3     <h1>Hallo Welt!</h1>
4   </body>
5 </html>
```

- ▶ HTML ist eine Beschreibungssprache keine Programmiersprache!
- ▶ Webseiten können
 - ▶ dynamischen Code mit Php, JavaScript,
 - ▶ Bilder, Videos enthalten

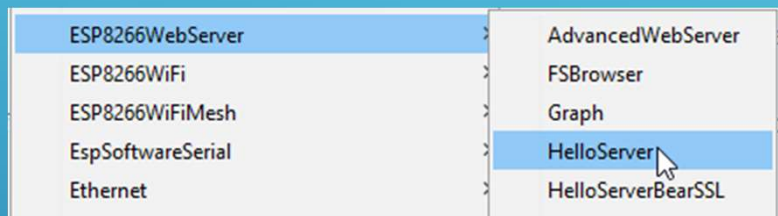
ERSTELLEN EINER WEBSEITE IN HTML

```
1 <html>
2   <body>
3     <h1>Hallo Welt!</h1>
4   </body>
5 </html>
6
7 <html><body><h1>Hallo Welt!</h1></body></html>
```

- ▶ Reduziert Ladezeiten,
- ▶ spart auf dem Mikrocontroller Speicher,
- ▶ Internet Service
 - ▶ [Google.de](#) > „html komprimieren“

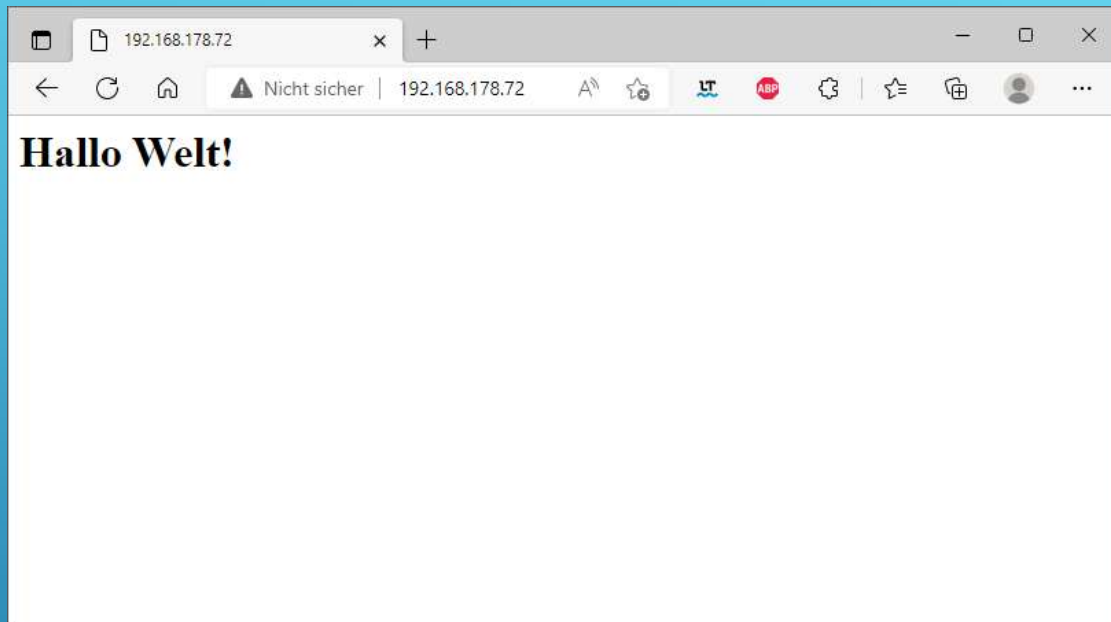
HTML KOMPRIMIEREN

▶ „Datei“ > „Beispiele“ > „ESP8266WebServer“ > „HelloServer“



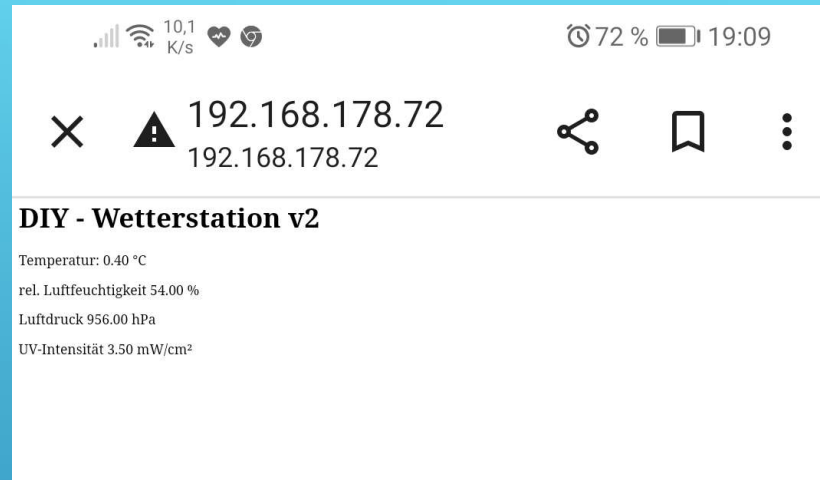
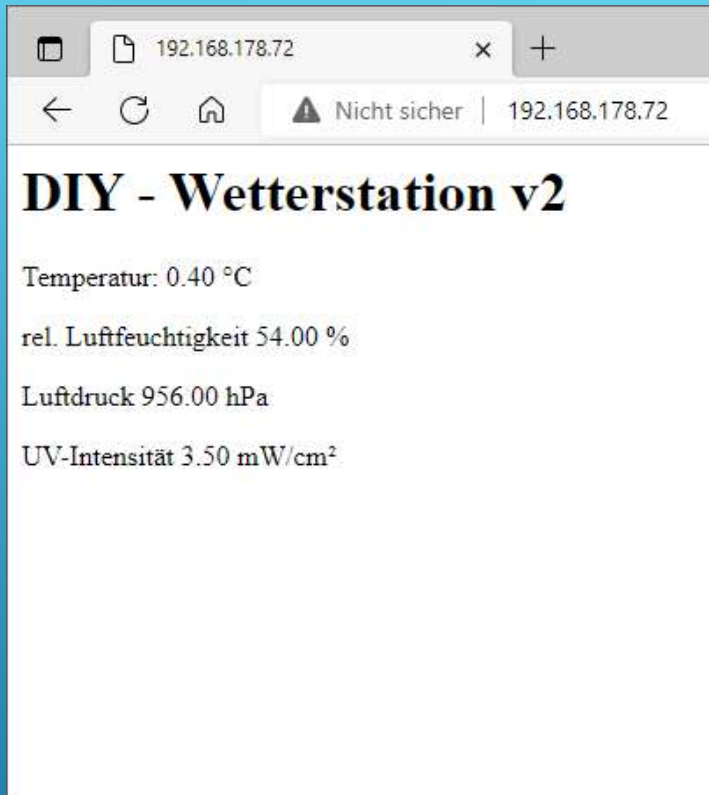
```
server.send(200, "text/plain", "hello from esp8266!\r\n");  
server.send(200, "text/html", " <html><body><h1>Hallo Welt!</h1></body></html> ");
```

ERSTELLEN EINES WEBSERVERS



- ▶ öffnen der IP-Adresse im Browser
- ▶ Anzeige des Textes „Hallo Welt!“

ANZEIGE DER WEBSEITE IM BROWSER



- ▶ Adresse im Browser aufrufen Handy & Computer

ERSTELLEN EINER HTML SEITE FÜR SENSORWERTE

DIY Weatherstation V2

Sensordaten

Sensor	Wert
DHT11	
Temperatur	24.10 °C
rel. Luftfeuchtigkeit	64.00 %
BMP180	
Temperatur	26.80 °C
Luftdruck	100946.00 hPa
UV-Sensor	
	1.06 mW

DIY Weatherstation V2

Sensordaten

Sensor	Wert
DHT11	
Temperatur	23.90 °C
rel. Luftfeuchtigkeit	64.00 %
BMP180	
Temperatur	27.40 °C
Luftdruck	100939.00 Pa
UV-Sensor	
	1.06 mW/cm ²

AUSLESEN DER SENSORWERTE UND
EINFÜGEN IN DIE SEITE

ThingSpeak for IoT Projects

Data collection in the cloud with advanced data analysis using MATLAB

Get Started For Free

Learn More



► Adresse: <https://thingspeak.com/>

THINGSSPEAK

Create MathWorks Account

Email Address

Missing required information

i To access your organization's MATLAB license, use your school or work email.

Location

United States

First Name

Last Name

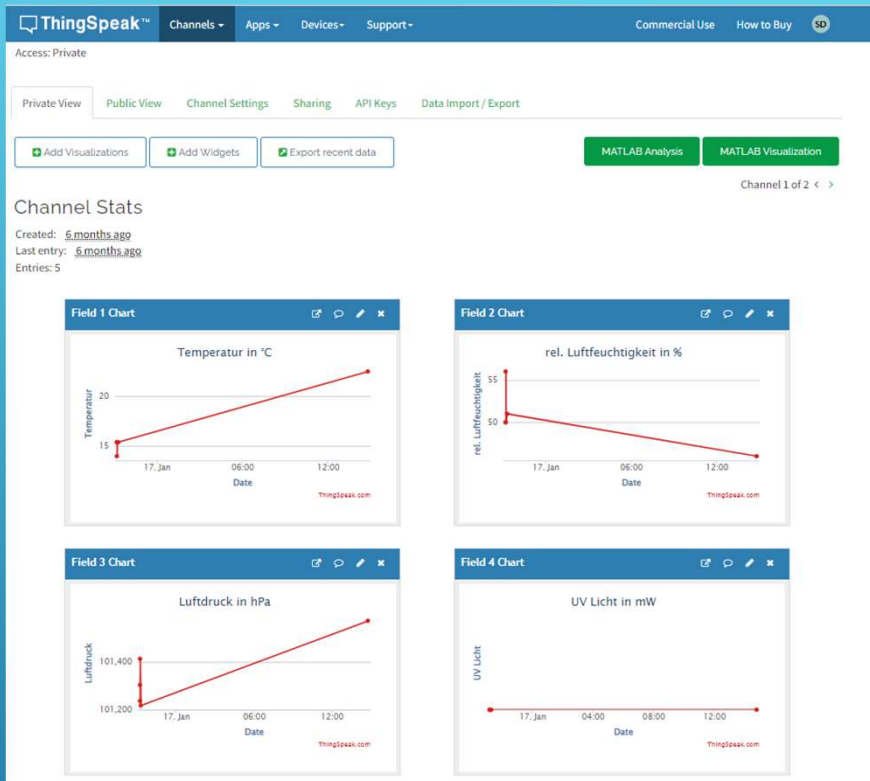
Continue

Cancel

This site is protected by reCAPTCHA Enterprise and the Google Privacy Policy and Terms of Service apply.

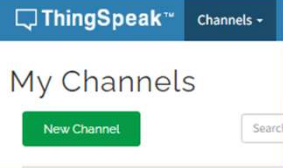
- ▶ Email Address – eine echte E-Mail Adresse
- ▶ Location – Germany
- ▶ First Name – Vorname
- ▶ Last Name - Nachname

THINGSPEAK
Benutzer anlegen



- ▶ Liniendiagramme für Sensorwerte
- ▶ Export als CSV-Datei
- ▶ Einfaches hinzufügen per HTTP-Request (ohne Login)

THINGSPEAK
Dashboard



New Channel

Name

Description

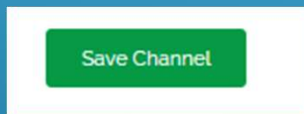
Field 1

Field 2

Field 3

Field 4

Field 5



- ▶ neuen Channel über „New Channel“ erstellen
- ▶ Name – Name des neuen Dashboards,
- ▶ Description – Beschreibung (optional)
- ▶ Field1 – Temperatur
- ▶ Field2 – rel. Luftfeuchtigkeit
- ▶ Field3 – Luftdruck
- ▶ Field4 - uvIntensitaet

THINGSPEAK
neues Dashboard anlegen

Wetterstation v2

Channel ID: 1809252
Author: stefandraeger
Access: Private

Private View Public View Channel Settings Sharing API Keys Data Import / Export

Write API Key

Key: 7GX9605BZGE7YLMG

Generate New Write API Key

Read API Keys

Key: QAFKP07A46ZSRJ3V

Note:

Save Note Delete API Key

Add New Read API Key

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key**.
- Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

Write a Channel Feed

```
GET https://api.thingspeak.com/update?api_key=7GX9605BZGE7YLMG&field=
```

Read a Channel Feed

```
GET https://api.thingspeak.com/channels/1809252/feeds.json?api_key=
```

Read a Channel Field

```
GET https://api.thingspeak.com/channels/1809252/fields/1.json?api_key=
```

Read Channel Status Updates

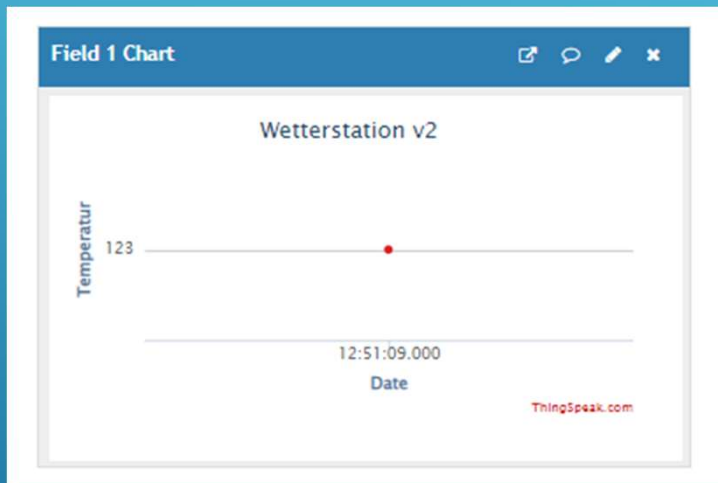
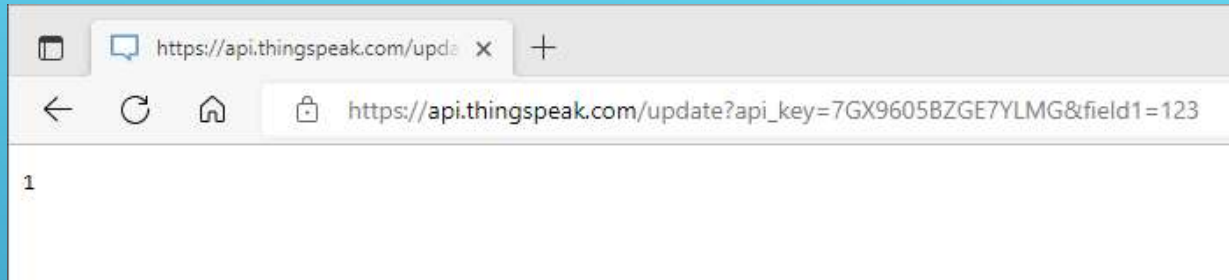
```
GET https://api.thingspeak.com/channels/1809252/status.json?api_key=
```

Learn More

- ▶ API Key wird benötigt zum validieren des Benutzers
- ▶ Daten lesen & schreiben nur mit Key möglich

THINGSPEAK

API Key für den Zugriff



- ▶ API Key wird benötigt zum validieren des Benutzers
- ▶ Daten lesen & schreiben nur mit Key möglich

BROWSER

Beispiel URL zum einfügen von Daten

Field 1 Chart Options ? x

Title:	<input type="text"/>	Timescale:	<input type="text"/>
X-Axis:	<input type="text"/>	Average:	<input type="text"/>
Y-Axis:	<input type="text"/>	Median:	<input type="text"/>
Color:	<input type="text" value="#d62020"/>	Sum:	<input type="text"/>
Background:	<input type="text" value="#ffffff"/>	Rounding:	<input type="text"/>
Type:	<input type="text" value="line"/>	Data Min:	<input type="text"/>
Dynamic?:	<input type="text" value="true"/>	Data Max:	<input type="text"/>
Days:	<input type="text"/>	Y-Axis Min:	<input type="text"/>
Results:	<input type="text" value="60"/>	Y-Axis Max:	<input type="text"/>

Save Cancel

- ▶ Title – Titel des Diagramms
- ▶ X-Axis – Text auf der X-Achse
- ▶ Y-Axis – Text auf der Y-Achse
- ▶ Color – HTML HEX-Color Wert für die Linie
- ▶ Background - HTML HEX-Color Wert für den Hintergrund
- ▶ Type: Art des Diagramms
- ▶ Data Min: minimaler Wert
 - ▶ kann vom Sensor entnommen werden
- ▶ Data Max: maximaler Wert
 - ▶ kann vom Sensor entnommen werden

THINGSPEAK

Konfiguration der Diagramme

```
String API_KEY = "";  
String thingSpeakAPIUrl = "http://api.thingspeak.com/update?api_key="+API_KEY+"&;
```

```
void loop() {  
  if ((WiFiMulti.run() == WL_CONNECTED)) {  
    String url = thingSpeakAPIUrl;  
    url = url + "field1=" + getDht11TempValue();  
    url = url + "&field2=" + getDht11HumidityValue();  
    url = url + "&field3=" + getBmp180Pressure();  
    url = url + "&field4=" + getUvSensorValue();  
    http.begin(client, url);  
  
    int httpCode = http.GET();  
    Serial.println("HTTP Code:"+httpCode);  
    Serial.println("URL:"+url);  
  
    delay(PAUSE);  
  }  
}
```

- ▶ HTTP Request = Anfrage
- ▶ HTTP Response = Antwort
- ▶ Adresse wird zusammengefügt aus der Basis und den Sensordaten

ARDUINO IDE

Absenden eines Requests



VIELEN DANK!

info@draeger-it.blog