

Softwareentwurf und Anwendungen verteilter Systeme

**BA Internet der Dinge – Gestaltung vernetzter Systeme
Semester 3
Hochschule für Gestaltung Schwäbisch Gmünd
Dozent: Yannick Schiele**

Arduino

Agenda

- Einführung
- NodeMCU ESP 8266
- Installation
- Programmierung
 - Sensoren
 - Aktoren
 - Webserver
 - Tbd.: Rest APIs

Mikrocontroller

- Chip mit einer bestimmten Aufgabe
 - Integrierter Prozessor (CPU)
 - Arbeitsspeicher (RAM)
 - Programmierbare Ein- und Ausgänge (Pins)
- Führt immer nur **ein** Programm aus
- Kleine praktische Größe
- Preiswert
- Einfach zu programmieren

<https://open.hpi.de/courses/mikrocontroller2019/>



Arduino

- eine Plattform für Elektronik- und Mikrocontroller
- Plattform umfasst Arduino Elektronik-Hardware-Boards und Arduino Software zur Programmierung
- Die gesamte Plattform ist „open Source“.

<https://www.elektrotechnik-einfach.de/was-ist-arduino/>

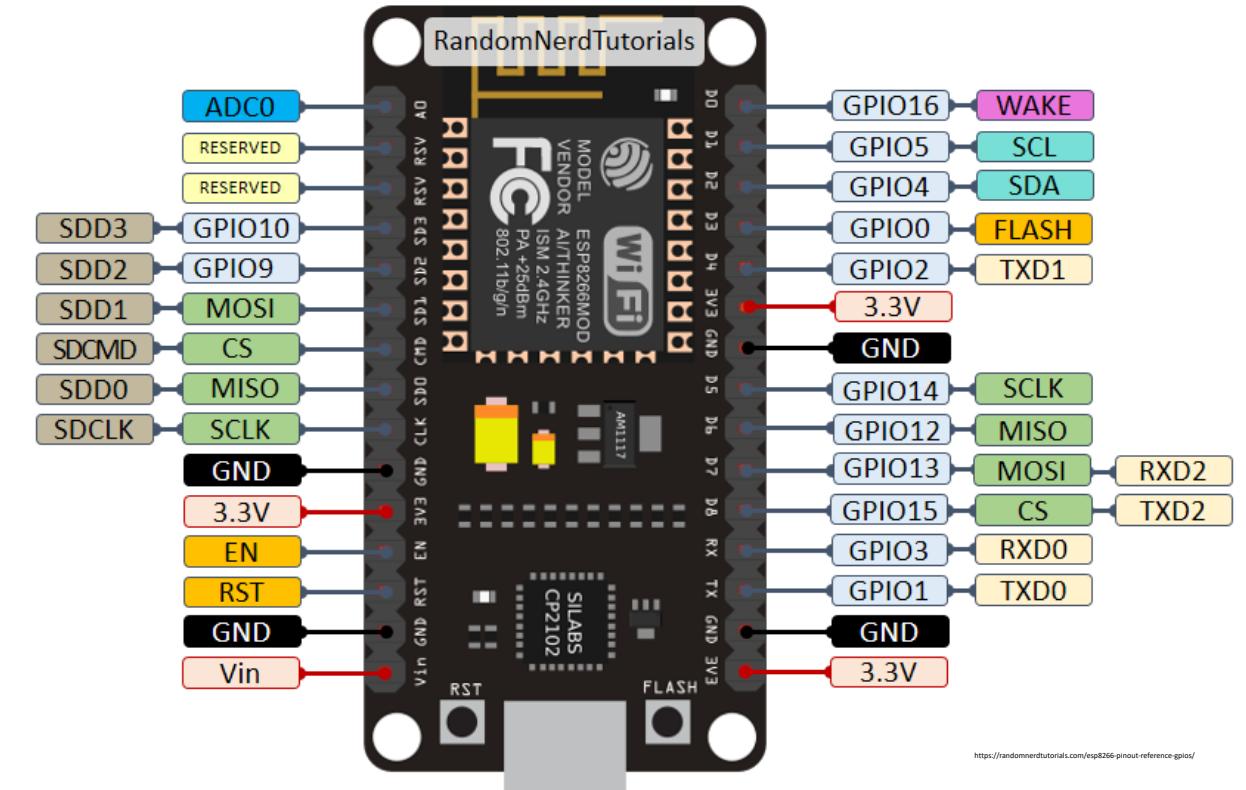
NodeMCU ESP 32 & 8266

| | ESP8266 | ESP32 |
|------------------|---|---|
| Prozessor | Tensilica Xtensa mit 80 MHz | Xtensa dual-core 32bit LX6 mit 160 bis 240 MHz |
| Flash Speicher | 4 MB | 4 MB |
| SRAM Speicher | 160KB | 520KB |
| WiFi / Bluetooth | IEEE 802.11 b/g/n bis max. 72,2 Mb/s | 802.11 b/g/n – 2.4 GHz bis max. 150 Mb/s Bluetooth Low Energy |
| Ein/Ausgänge | 16 GPIOs | 48 GPIO |
| PWM | 8 | |
| Schnittstellen | SPI, I2C, I2S, UART | SPI, I2C, I2S, UART, CAN bus 2.0 |
| Sensoren | | Hall Sensor (Magnetfeld Sensor) Temperatursensor Touch-Sensor |
| Betriebsspannung | | 2,3V .. 3,6V |
| Stromverbrauch | 80mA | 260mA |

<https://dragger.it/blog/esp8266-vs-esp32/>

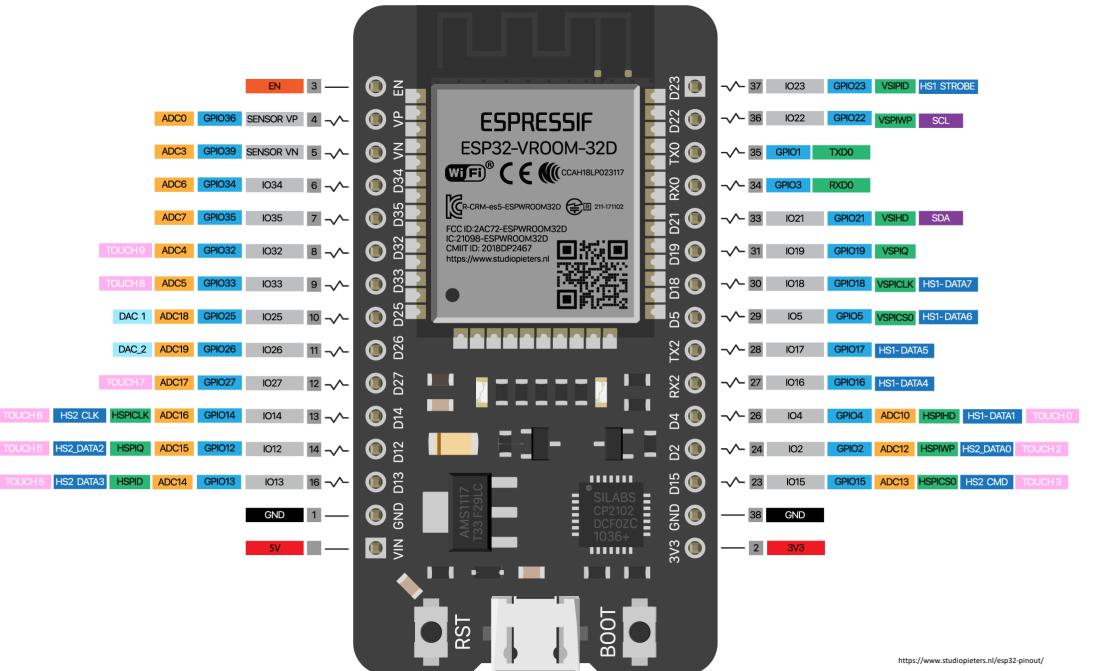
H f G

NodeMCU ESP8266



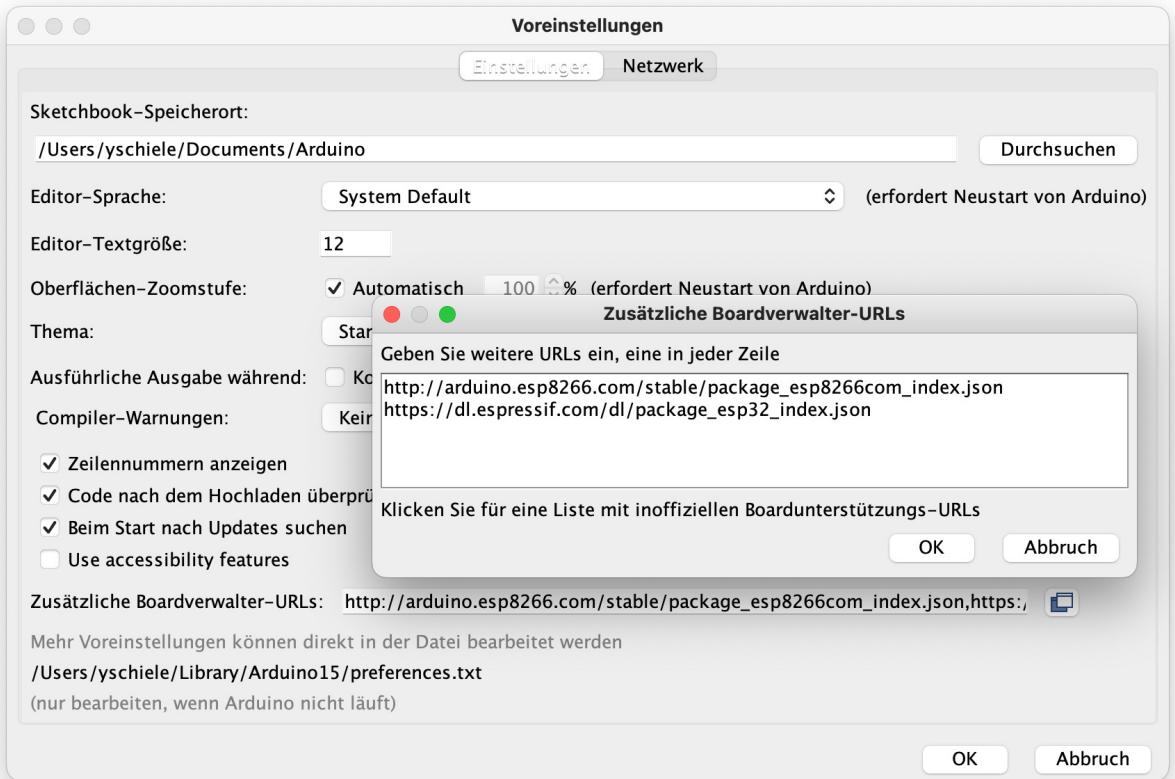
H f G

NodeMCU ESP32



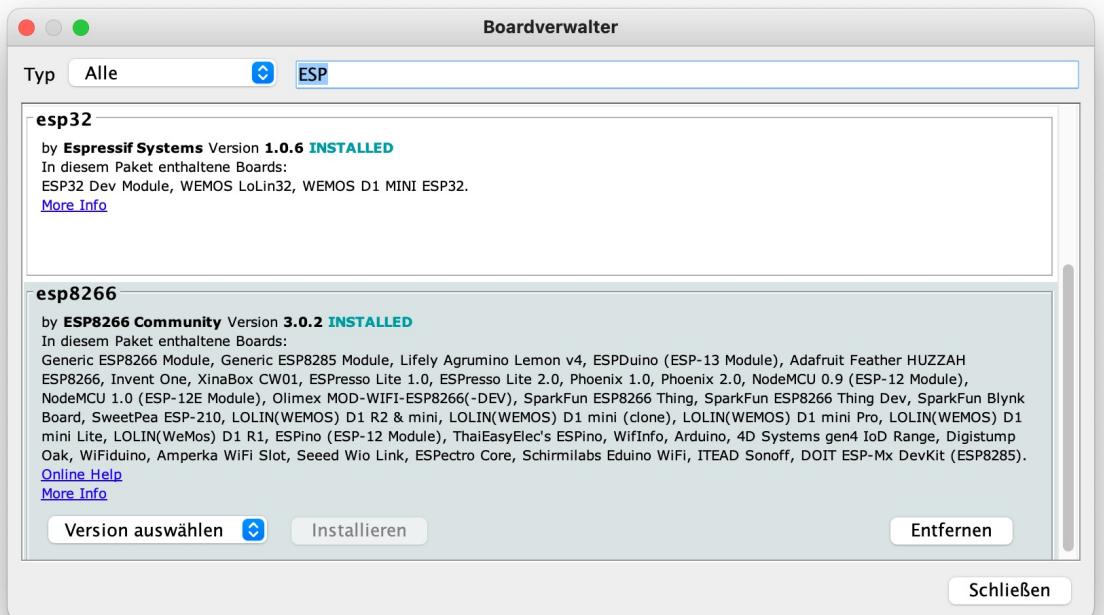
ESP32-Programmierung

- **Anschluss per USB – Verwendung des richtigen Kabels!**
 - Zum Anschluss des ESP32 wird ein **Datenkabel** mit Mikro-USB-Anschluss benötigt.
 - reine Ladekabel, die oft z.B. bei Powerbanks, etc. mitgeliefert werden, können ein angeschlossenes Gerät zwar mit Strom versorgen, aber es fehlen die Datenleitungen.
 - Dadurch kann keine Kommunikation über das Kabel erfolgen – der PC erkennt nicht, dass ein Gerät angeschlossen ist
- **Treiber-Installation**
 - <https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads>
 - verschiedene Treiber-Versionen für mehrere Windows-Versionen, Mac und Linux
- **Installation der Arduino-IDE**
 - <https://www.arduino.cc/en/software>



Installation der Arduino-IDE

- Weitere Werkzeuge und Bibliotheken müssen zusätzlich über die Boardverwaltung installiert werden
- Datei → Voreinstellungen → Zusätzliche Boardverwalter-URLs:
 - https://dl.espressif.com/dl/package_esp32_index.json
 - https://arduino.esp8266.com/stable/package_esp8266com_index.json

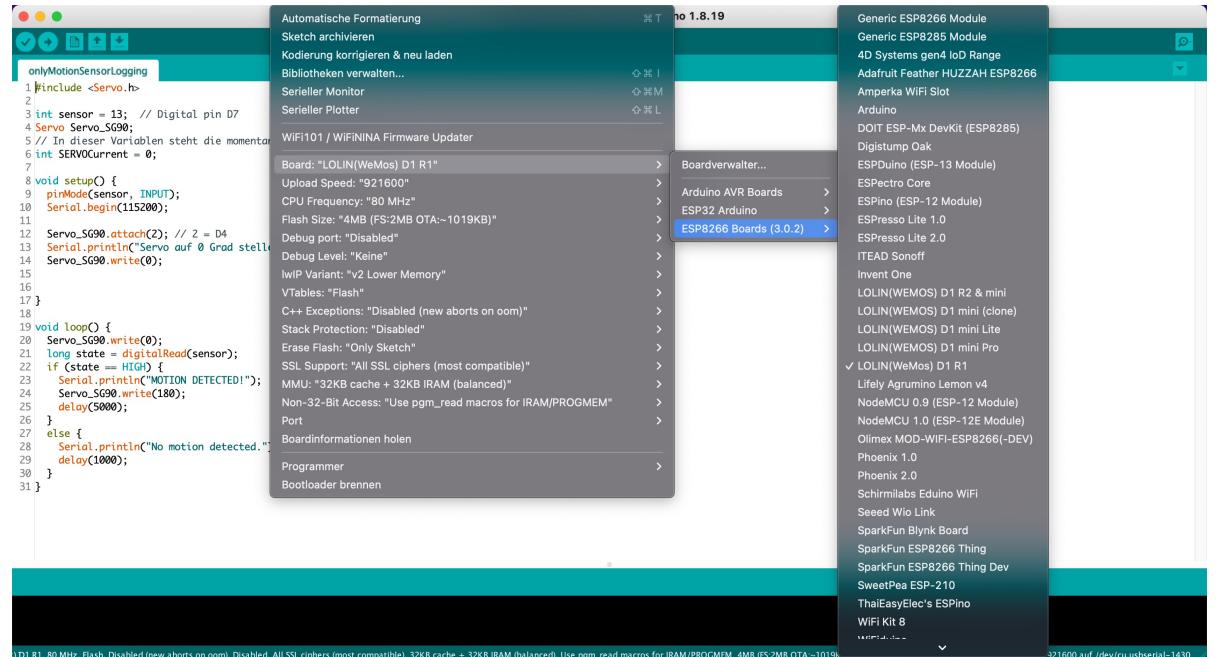


Installation der Arduino-IDE

- Weitere Werkzeuge und Bibliotheken müssen zusätzlich über die Boardverwaltung installiert werden
- Werkzeuge → Board → Boardverwalter:
 - esp32
 - esp8266

Installation der Arduino-IDE

- Weitere Werkzeuge und Bibliotheken müssen zusätzlich über die Boardverwaltung installiert werden
- Werkzeuge → Board →
 - ESP32 Arduino
 - ESP8266 Boards

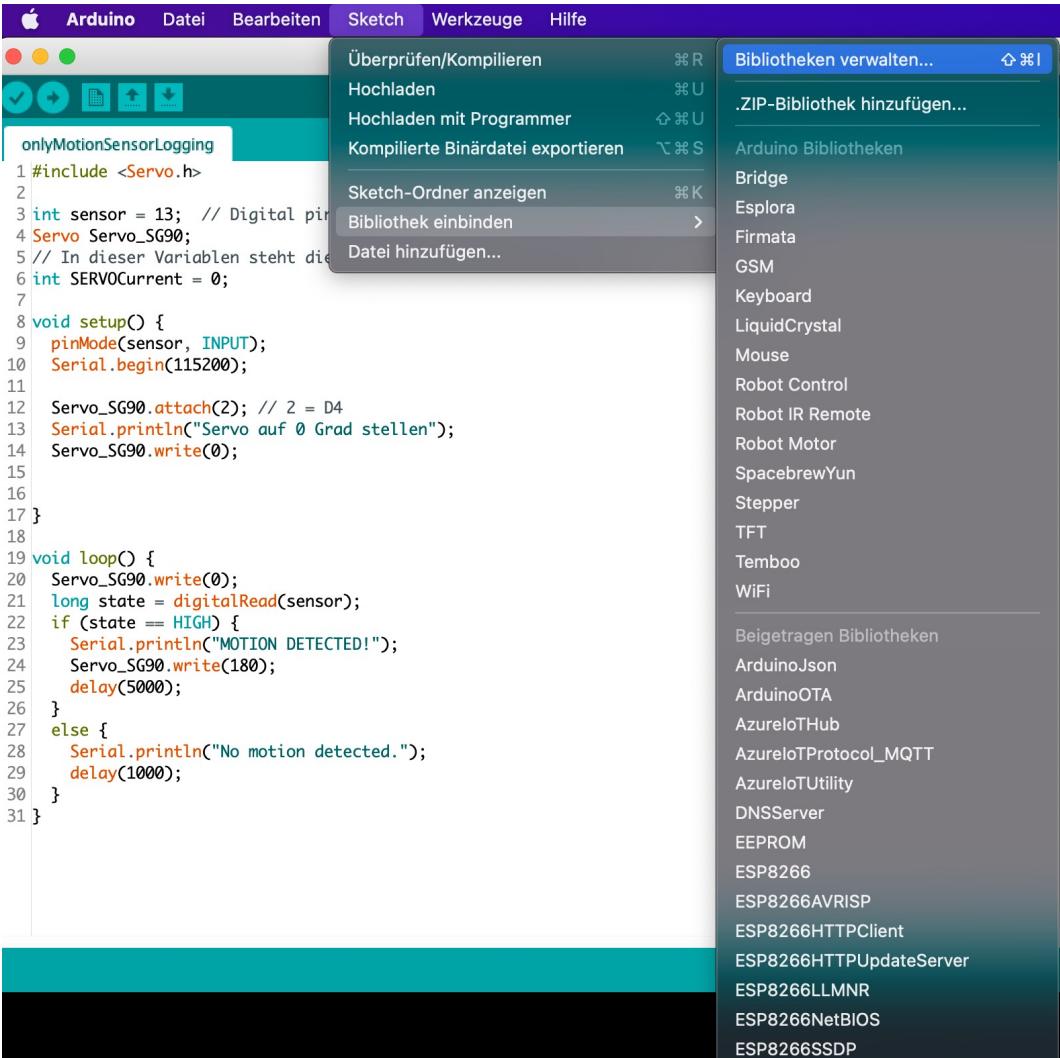


```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  // Initialize the LED_BUILTIN pin as an output
}

// the loop function runs over and over again
void loop() {
  digitalWrite(LED_BUILTIN, LOW);
  // Turn the LED on (LOW is the voltage level)
  // the LED is on;
  // because it is active on low
  delay(1000);
  // Wait for a second
  digitalWrite(LED_BUILTIN, HIGH);
  // Turn the LED off by making the voltage HIGH
  delay(2000);
  // Wait for two seconds
}
```

Programmierung

Skript zum Blinken der eingebauten LED
des ESP 8266



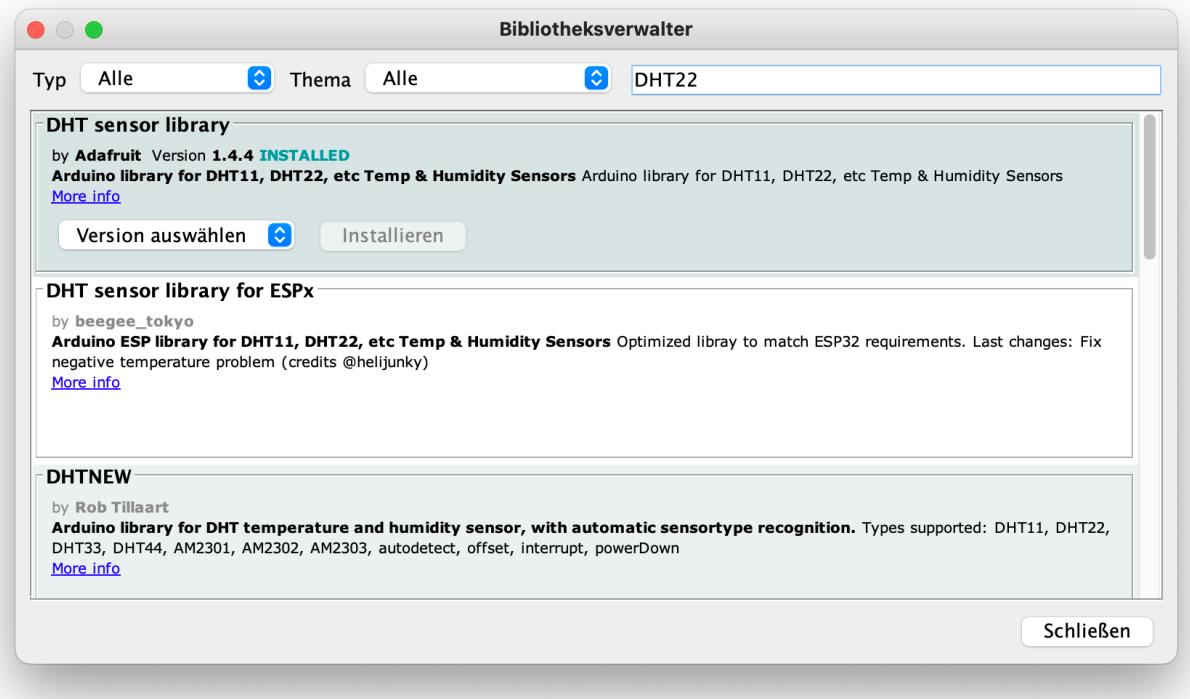
Einbindung einer Bibliothek

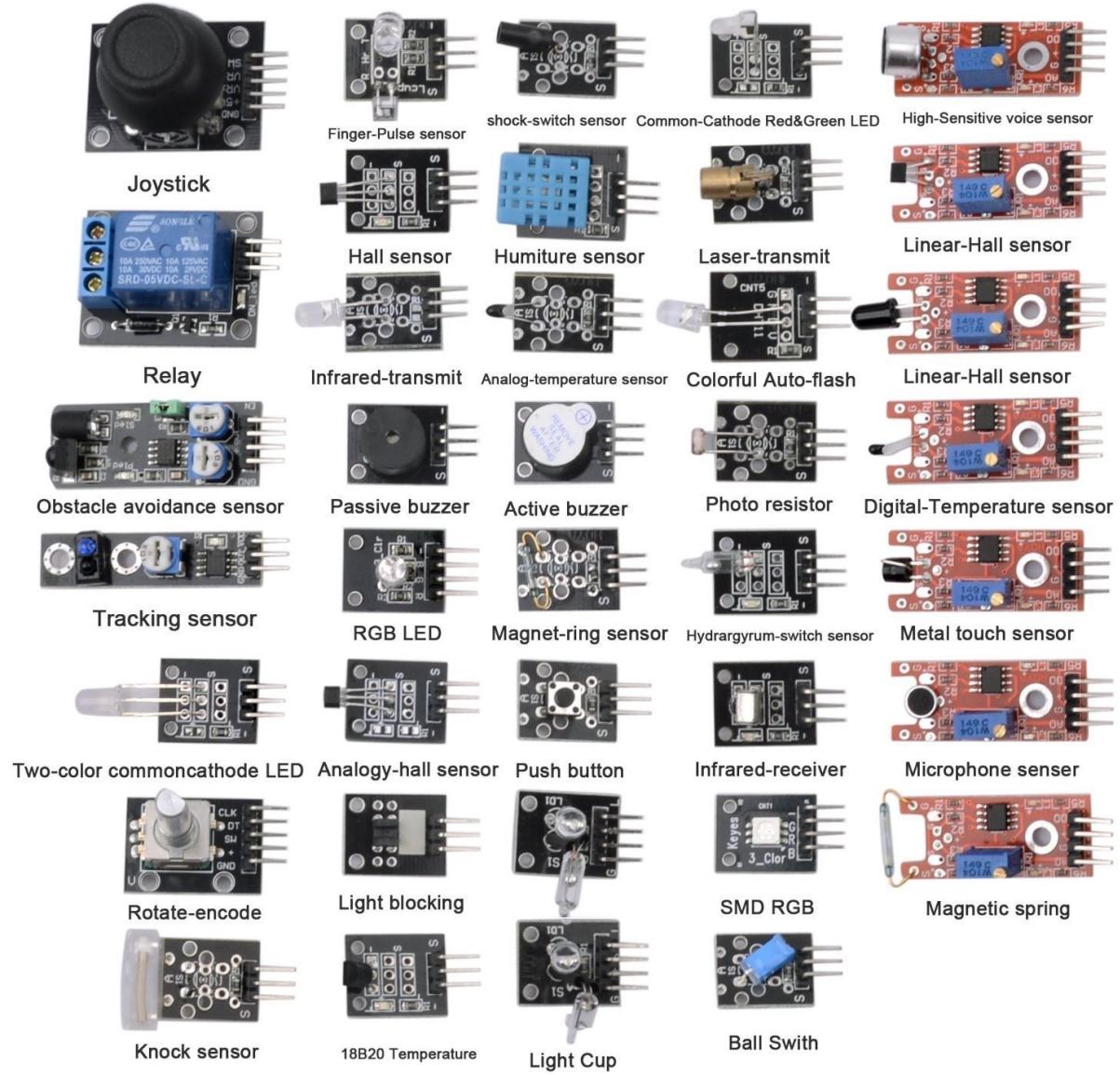
- Die Arduino-Entwicklungsumgebung kann durch die Verwendung von Bibliotheken erweitert werden.
- Bibliotheken bieten zusätzliche Funktionen für die Verwendung in Sketchen, z. B. für die Arbeit mit Hardware.
- Sketch → Bibliothek einbinden → Bibliothek verwalten

<https://www.arduino.cc/reference/en/libraries/>

Einbindung einer Bibliothek

Beispiel: Bibliothek des DHT 22 Sensors





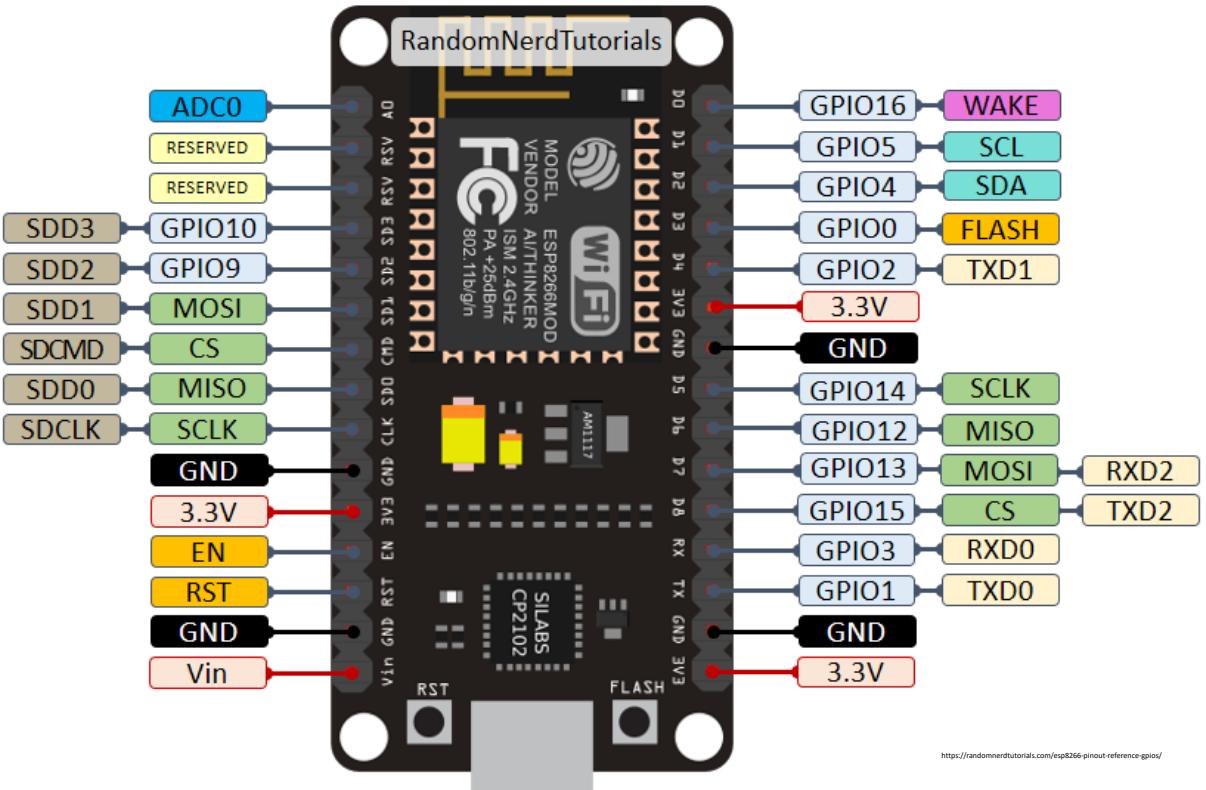
Sensoren

- Reize der Umwelt:
 - Bewegung
 - Töne
 - Helligkeit
 - Temperatur
 - ...
- Über Pins auslesbar
 - Digital
 - Analog
 - I2C, SPI, OneWire, Serial

digitalRead()/ Digitaler Input

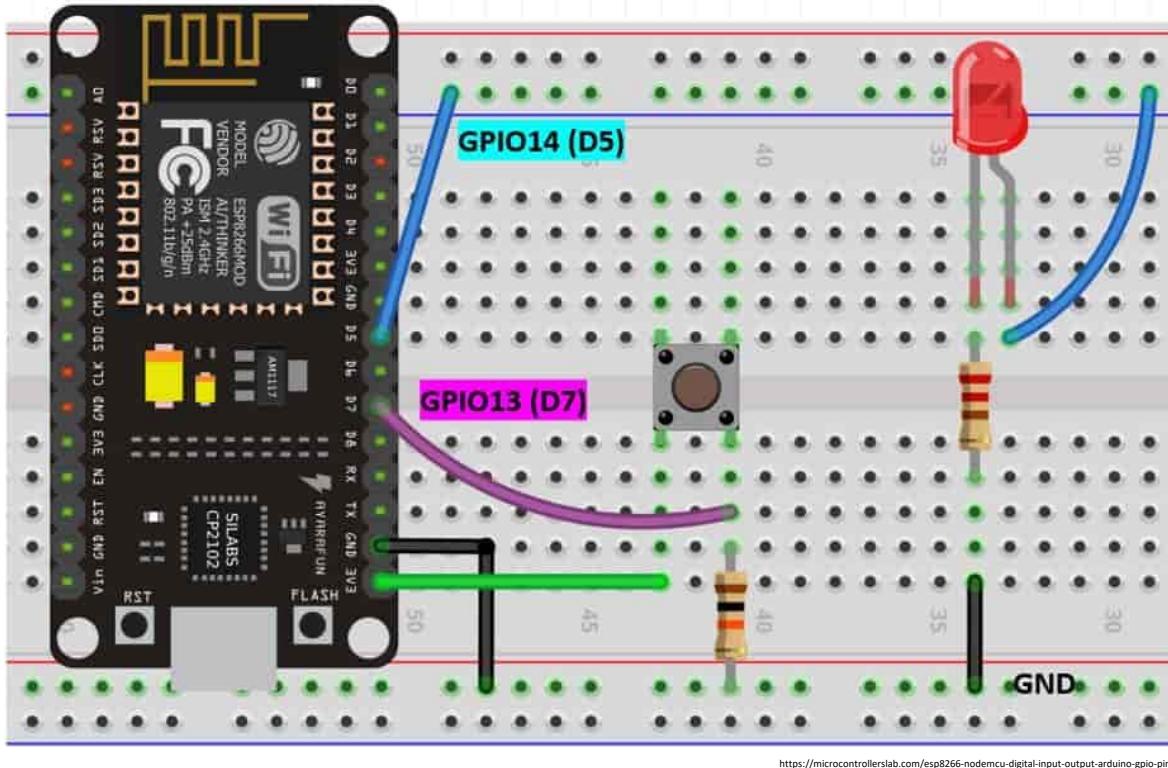
- D0 bis D8, RX, TX können als digitale Eingangs-/Ausgangs-Pins verwendet werden
- Mit `pinMode()` wird der entsprechende Pin als INPUT oder OUTPUT definiert
- Mit `digitalRead()` wird der Wert von einem vorgegebenen Pin ausgelesen
 - Gibt entweder HIGH oder LOW zurück

<https://www.arduino.cc/reference/de/language/functions/digital-io/digitalread/>



digitalRead() Digitaler Input

- GPIO 14 PIN ist mit dem Anoden-Pin der LED verbunden
- Kathoden-Pin der LED ist über den 220-Ohm-Widerstand mit der gemeinsamen Masse verbunden
- Button/Taster hat 4 Anschlüsse:
 - Versorgung mit 3,3 Volt
 - Digitaler Input über GPIO 13 mit 10k Ohm Widerstand (Pull Down) und gemeinsamer Masse (GND)
- Wird Button gedrückt, erhält GPIO13 den Zustand HIGH



<https://microcontrollerslab.com/esp8266-nodemcu-digital-input-output-arduino-gpio-pins/>

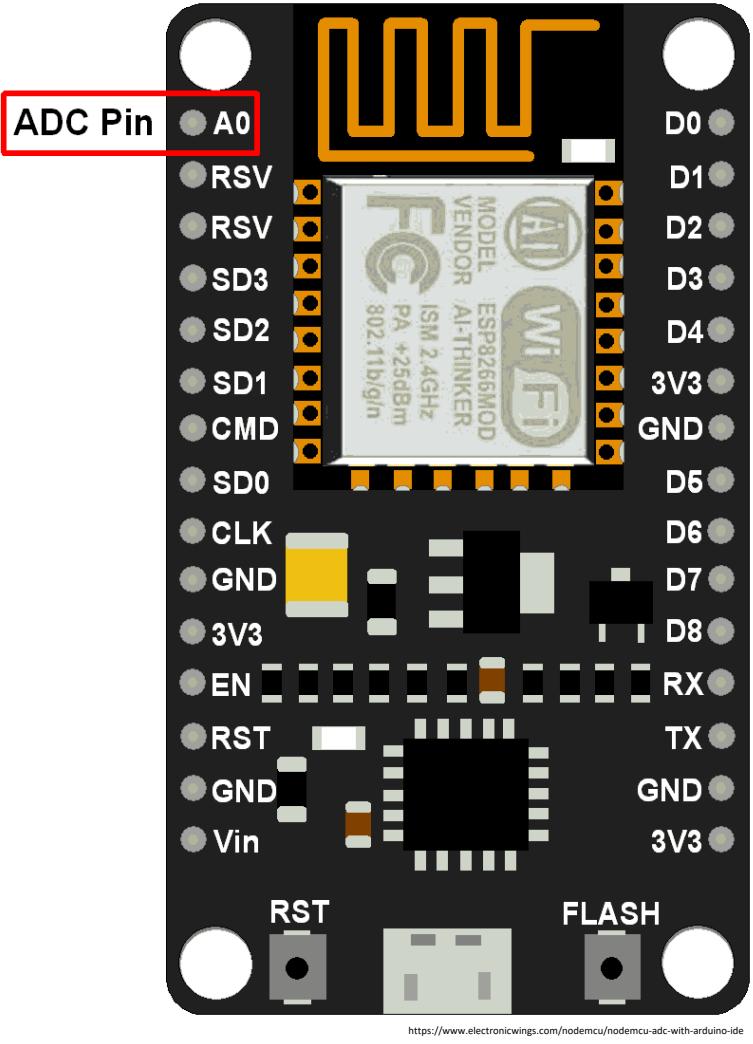
```
const int Push_button_pin = 13;
const int led = 14;
int Push_button_state = 0;

void setup(){
pinMode(Push_button_pin, INPUT);
pinMode(led, OUTPUT);
}

void loop(){
Push_button_state = digitalRead(Push_button_pin);
if (Push_button_state == HIGH) {
    digitalWrite(led, HIGH);
} else {
    digitalWrite(led, LOW);
}
}
```

digitalRead()/ Digitaler Input

Skript für das Auslesen des digitalen
Input Signals

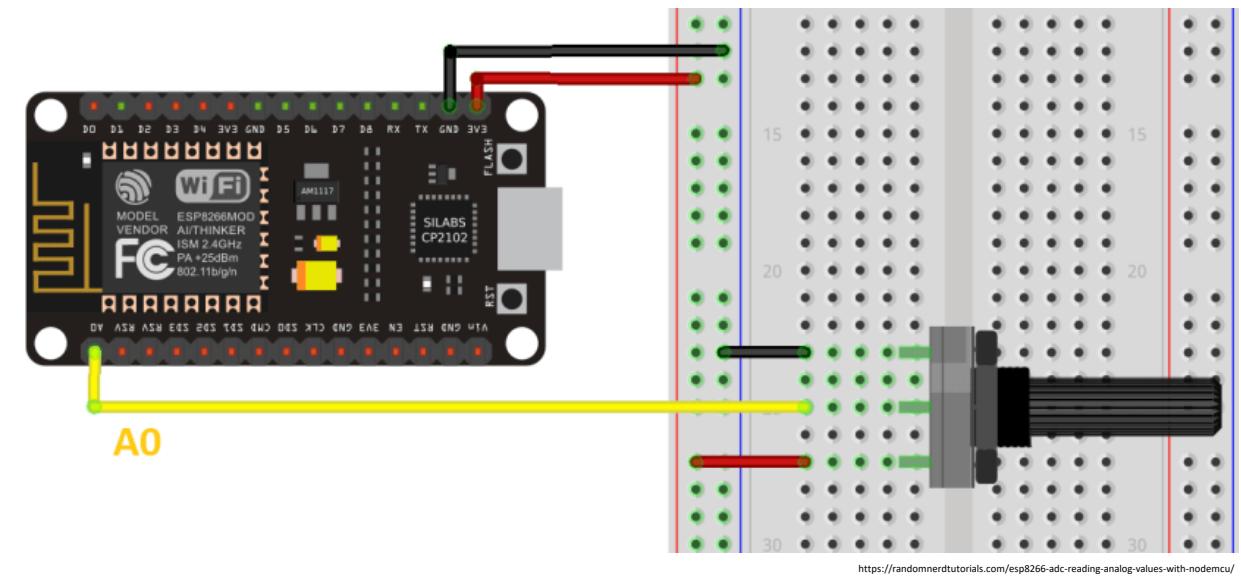


analogRead()/ Analoger Input

- Ein Analog-Digital-Wandler (ADC) wird verwendet, um das analoge Signal in eine digitale Form umzuwandeln.
- Der ESP8266 hat einen eingebauten 10-Bit-ADC mit nur einem ADC-Kanal, d.h. er hat nur einen ADC-Eingangspin, um die analoge Spannung von einem externen Gerät zu lesen.
- Es kann entweder die eingebaute Systemspannung oder eine externe Spannung gemessen werden.
- Der Eingangsspannungsbereich für den ADC-Pin beträgt 0-1,0 V, wenn eine externe Spannung gemessen wird.

analogRead()/ Analoger Input

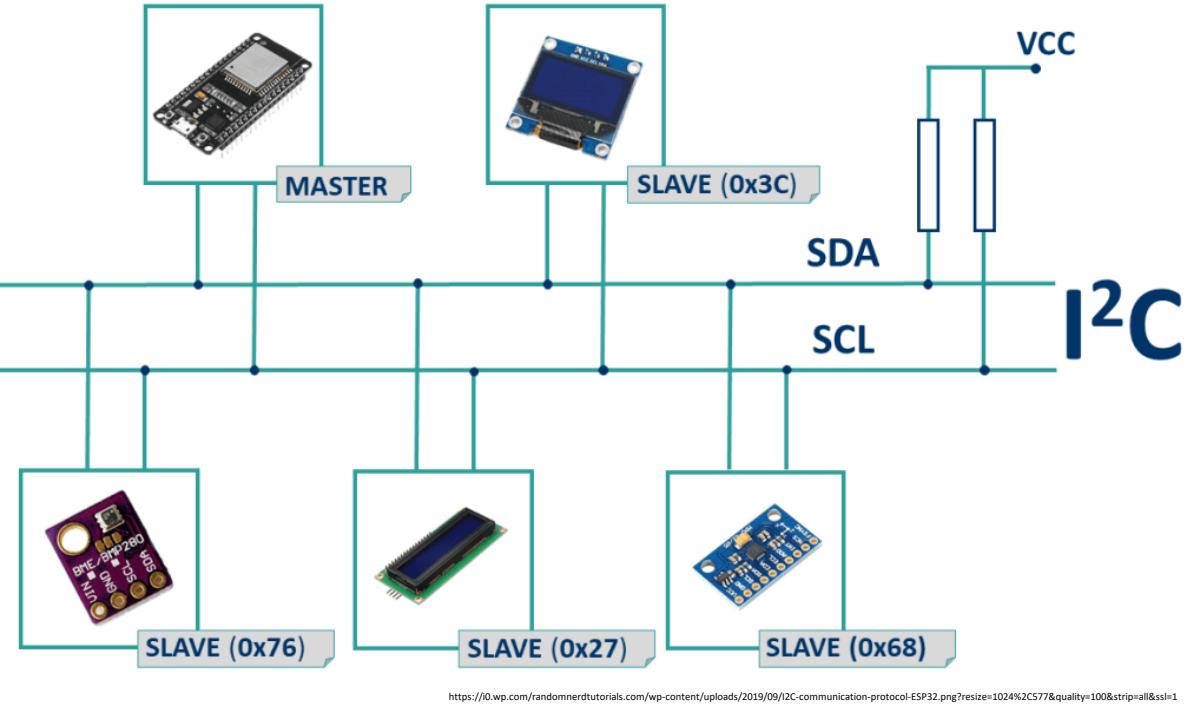
- Schaltplan ermöglicht das Auslesen des analogen Werts eines Potentiometers
 - Widerstandswerte sind hier mechanisch veränderbar
- `analogRead()`
 - Liest den Wert vom angegebenen analogen Pin ein.
 - Der ADC mappt die Eingangsspannungen zwischen 0 und 5 V auf Integer-Werte zwischen 0 und 1023



```
const int analogInPin = A0; // ESP8266 Analog Pin  
ADC0 = A0  
int sensorValue = 0; // value read from the pot  
  
void setup() { // initialize serial communication  
at 115200  
Serial.begin(115200);  
}  
  
void loop() { // read the analog in value  
sensorValue = analogRead(analogInPin); // print the  
readings in the Serial Monitor  
Serial.print("sensor = ");  
Serial.print(sensorValue);  
delay(1000);  
}
```

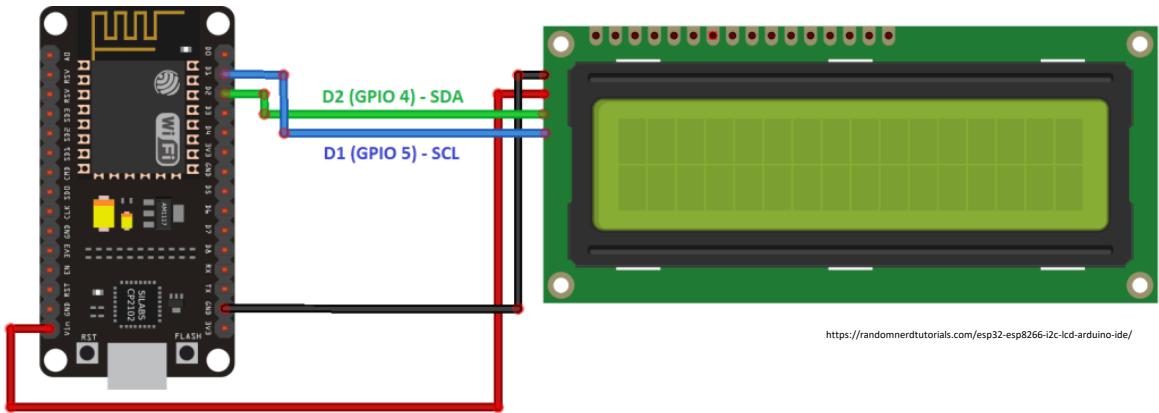
analogRead() Analoger Input

Skript für das Auslesen des analogen
Input Signals



I²C

- Das I²C-Protokoll (Inter Integrated Circuit) verwendet zwei Leitungen zum Senden und Empfangen von Daten:
 - einen seriellen Taktstift (SCL), den das Arduino Controller Board in regelmäßigen Abständen pulsiert
 - einen seriellen Datenstift (SDA), über den Daten zwischen den beiden Geräten gesendet werden
- Wenn die Takteleitung von niedrig auf hoch wechselt, wird ein einzelnes Bit an Informationen über die SDA-Leitung vom Board an das I²C-Gerät übertragen
- Wenn diese Information gesendet wird, führt das aufgerufene Gerät die Anfrage aus und sendet seine Daten über dieselbe Leitung an die Karte zurück



H f G

I2C

- I2C wird bspw. bei Liquid Crystal Displays verwendet
- ESP8266:
 - SDA: D2 (I2C → Data)
 - SCL: D1 (I2C → Clock)
- Zusätzlich wird LiquidCrystal_I2C Library benötigt

```

#include <Wire.h>

void setup() {
Wire.begin();
Serial.begin(115200); }

void loop() {
byte error, address; int nDevices;
nDevices = 0;
for(address = 1; address < 127; address++) {
    Wire.beginTransmission(address);
    error = Wire.endTransmission();
    if (error == 0) {
        Serial.print("I2C device found at address 0x");
        if (address<16) {
            Serial.print("0");
            Serial.println(address,HEX);
            nDevices++;
        }
        else if (error==4) {
            Serial.print("Unknow error at address 0x");
            if (address<16) {
                Serial.print("0");
                Serial.println(address,HEX);
            }
        }
    }
}

if (nDevices == 0) {
    Serial.println("No I2C devices found\n");
} else {
    Serial.println("done\n"); } delay(5000);
}

```

H f G

I2C

Bevor der Text auf dem LCD angezeigt werden kann, muss die I2C-Adresse des LCD ermittelt werden. Wenn das LCD richtig mit dem ESP32 verkabelt ist, kann folgender I2C-Scanner-Sketch ausgeführt werden.

<https://randomnerdtutorials.com/esp32-esp8266-i2c-lcd-arduino-ide/>

```

#include <LiquidCrystal_I2C.h> // set the LCD number of
columns and rows
int lcdColumns = 16;
int lcdRows = 2; // set LCD address, number of columns
and rows // if you don't know your display address, run
an I2C scanner sketch
LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);

void setup(){ // initialize LCD
lcd.init(); // turn on LCD backlight
lcd.backlight();
}

void loop(){ // set cursor to first column, first row
lcd.setCursor(0, 0); // print message
lcd.print("Hello, World!");
delay(1000); // clears the display to print new message
lcd.clear(); // set cursor to first column, second row
lcd.setCursor(0,1);
lcd.print("Hello, World!");
delay(1000);
lcd.clear();
}

```

H f G

I2C

Die Anzeige von statischem Text auf dem LCD-Display ist recht einfach. Es muss lediglich die Stelle ausgewählt werden, an der die Zeichen auf dem Bildschirm angezeigt werden sollen, und dann die Nachricht an das Display gesendet werden.

Hier ist ein sehr einfaches Beispiel für ein Script, das "Hello, World!" anzeigt.

<https://randomnerdtutorials.com/esp32-esp8266-i2c-lcd-arduino-ide/>



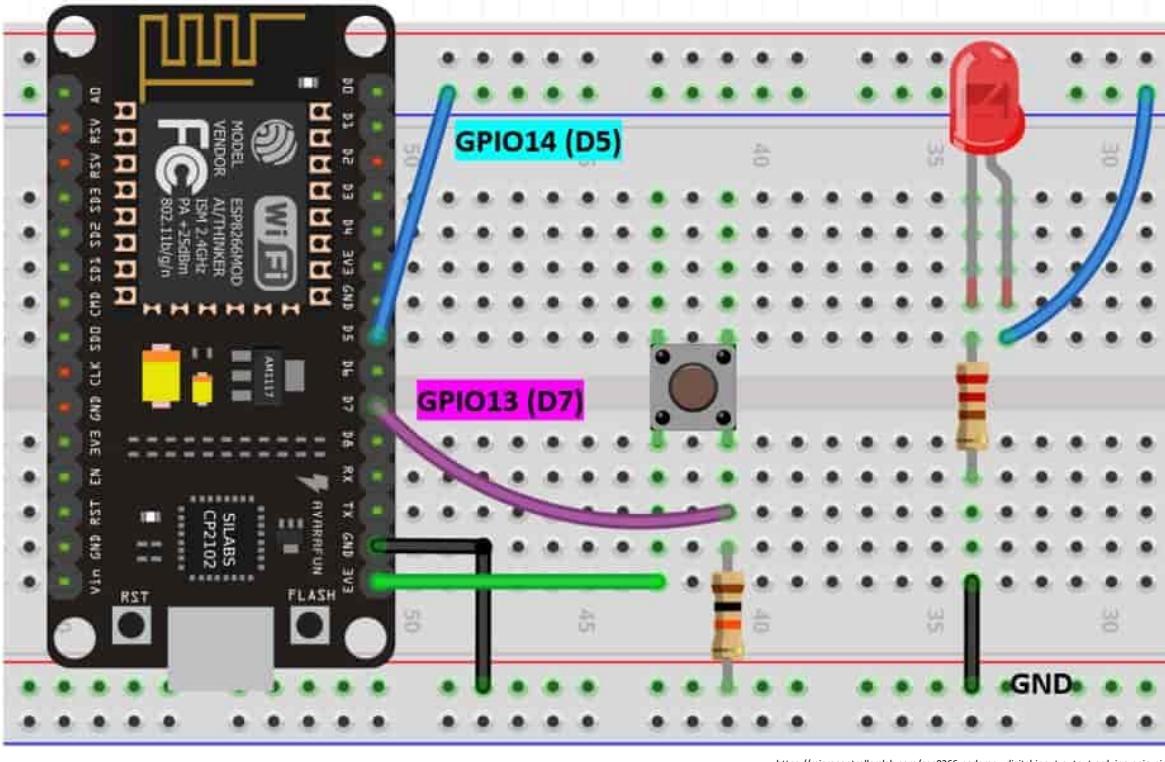
H f G

Aktoren

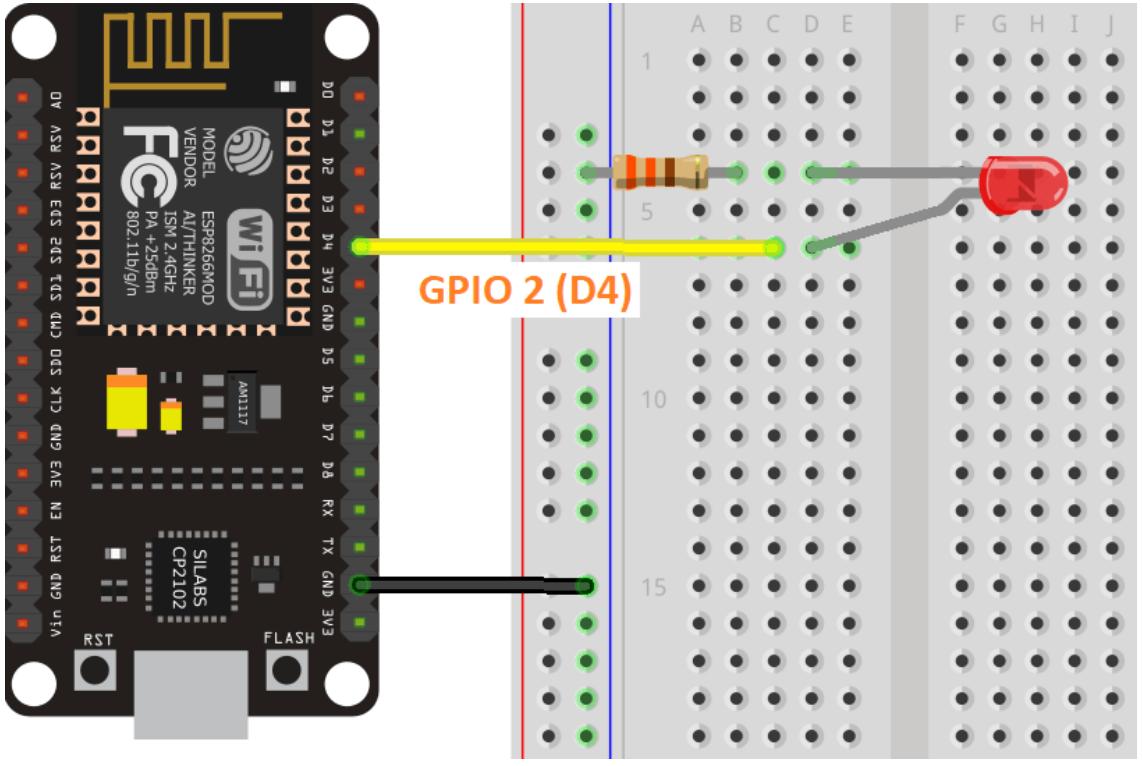
Mit Aktoren werden Signale in Bewegungen oder andere physikalische Größen, wie z. B. Druck und Temperatur umgewandelt und dadurch Kräfte ausgeübt oder mechanische Arbeit geleistet.

digitalWrite() / digitaler Output

- `digitalWrite()` schreibt einen HIGH- oder einen LOW-Wert an einen digitalen Pin
- Wenn der Pin mit `pinMode()` als OUTPUT konfiguriert wurde, wird seine Spannung auf den entsprechenden Wert gesetzt: 3,3 V für HIGH, 0 V für LOW
- Siehe Skript auf Folie 19



H f G



analogWrite() / Analoger Output

Beispiel: PWM (Pulse-Width Modulation)
Signale, um die LED-Helligkeit zu
dimmen, indem das Tastverhältnis über
die Zeit verändert wird.

```

const int ledPin = 2;

void setup() {
pinMode(ledPin, OUTPUT);
}

void loop() {

// increase the LED brightness
for(int dutyCycle = 0; dutyCycle < 255; dutyCycle++){
analogWrite(ledPin, dutyCycle);
// changing the LED brightness with PWM
delay(1);
}

// decrease the LED brightness
for(int dutyCycle = 255; dutyCycle > 0; dutyCycle--){
// changing the LED brightness with PWM
analogWrite(ledPin, dutyCycle);
delay(1);
}
}

```

H f G

analogWrite() / Analoger Output

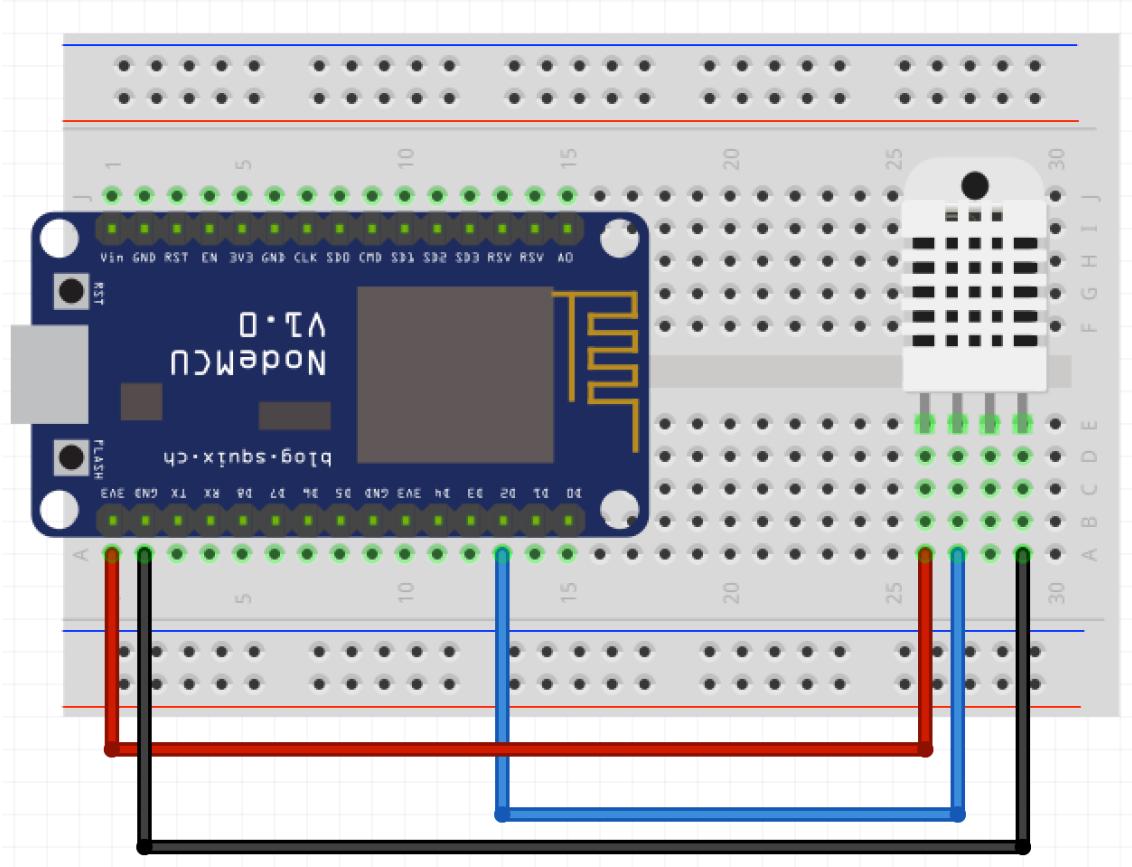
Um ein PWM-Signal zu erzeugen, wird folgende Funktion verwendet:

`analogWrite(pin, value);`

pin: PWM kann an den Pins 0 bis 16 verwendet werden

value: sollte im Bereich von 0 bis PWMRANGE liegen (standardmäßig 255)

Wenn der Wert 0 ist, ist die PWM an diesem Pin deaktiviert. Ein Wert von 255 entspricht einem Tastverhältnis von 100%



H f G

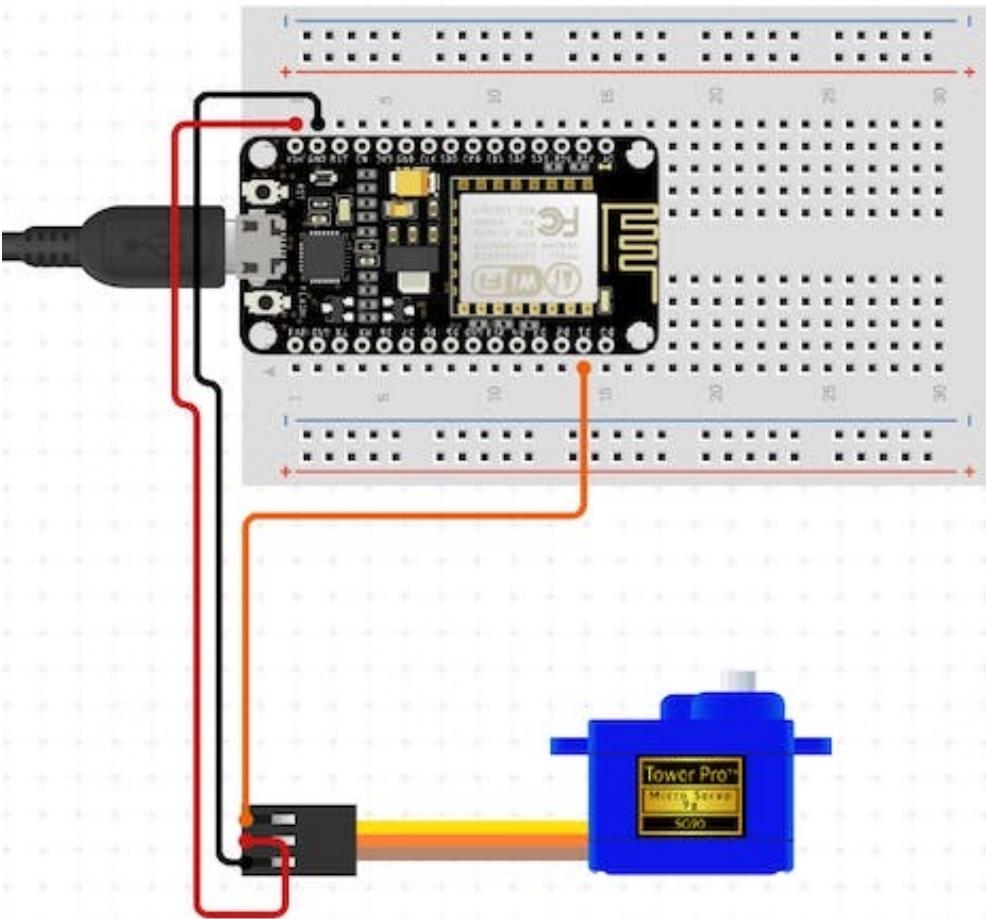
Sensor Beispiel

Der DHT22 ist ein sehr preiswerter Sensor.

Er besteht aus zwei Komponenten: einem kapazitiven Feuchtigkeitssensor und einem Thermistor, der die Temperatur misst.

Da es sich um einen digitalen Sensor handelt, können die Sensordaten über einen GPIO-Pin ausgelesen werden

Skript: DHTExample.ino



H f G

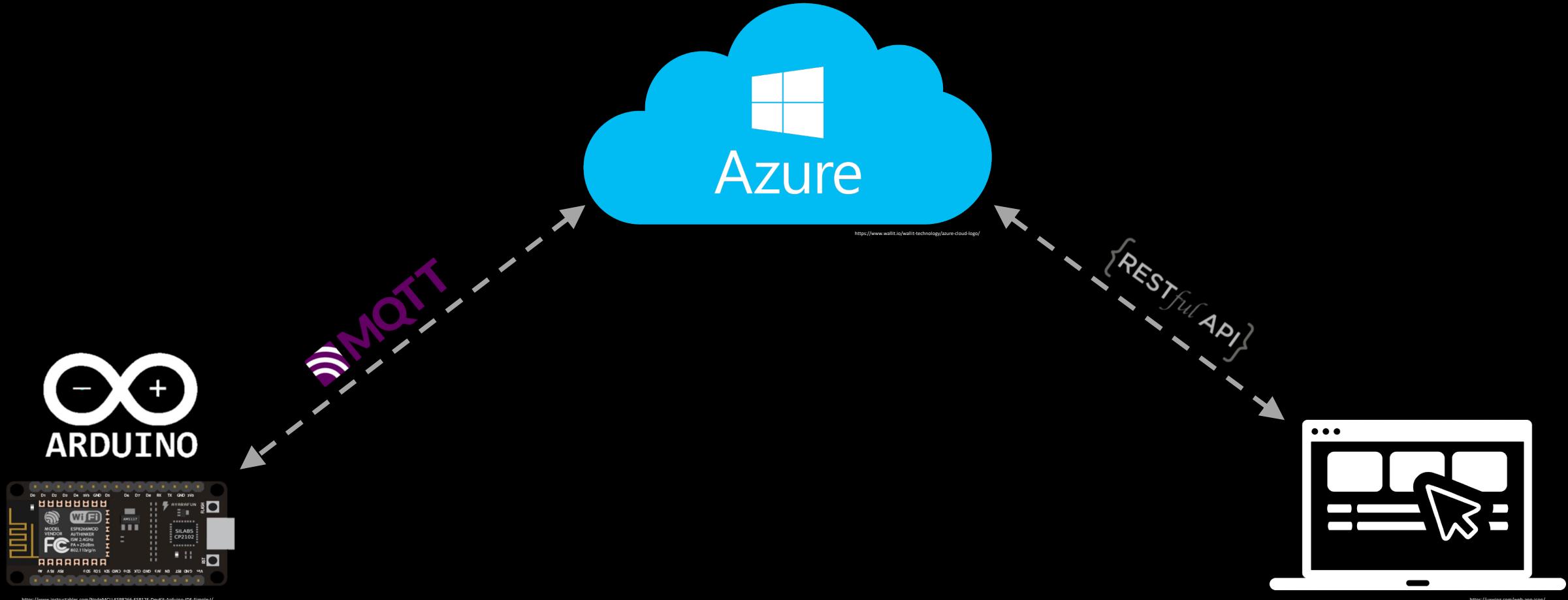
Aktor Beispiel

Ein Servo ist ein Aktor, der sich durch einen Befehl in einem bestimmten Winkel dreht.

Dieser Servo kann sich in jeden beliebigen Winkel zwischen 0-180 Grad bewegen.

Skript: ServoExample

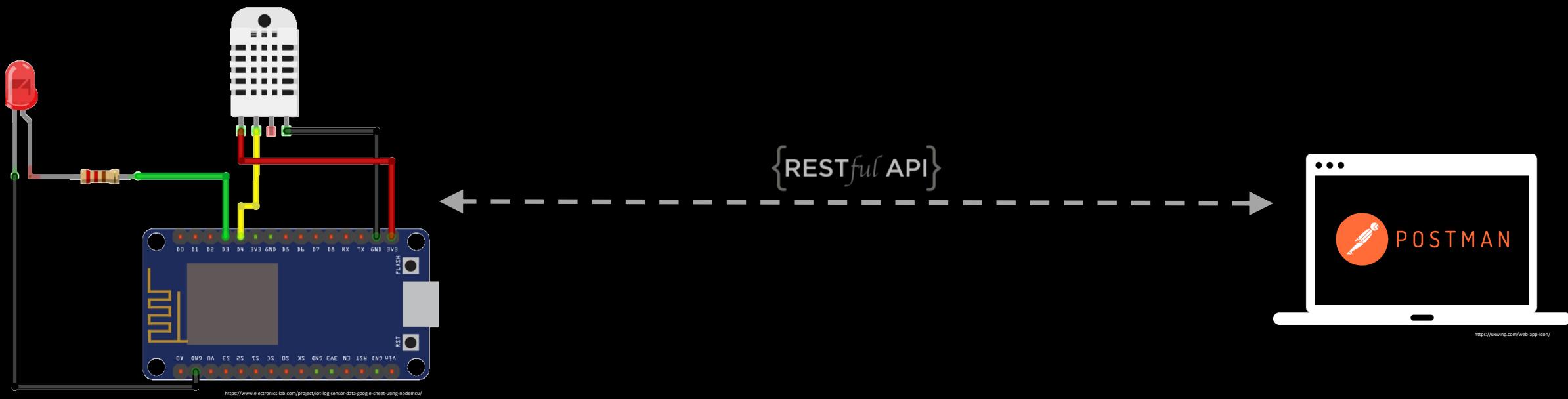
Ziel der Vorlesung



H f G

1. Projekt

- Arduino mit Sensor, Aktor und eigenem Webserver
 - Sensor Daten werden an Webserver gesendet
 - Webserver wird vom Arduino selbst gehostet und ist über das lokale Netz erreichbar
 - Über Rest APIs werden Sensor Daten abgerufen und der Aktor gesteuert



Recherche zu Sensor & Aktor

Ergebnis pro Team bitte mit euren Namen + Name des Sensors & Aktors am besten mit einem Link per Mail an: yannick.schiele@hfg.design