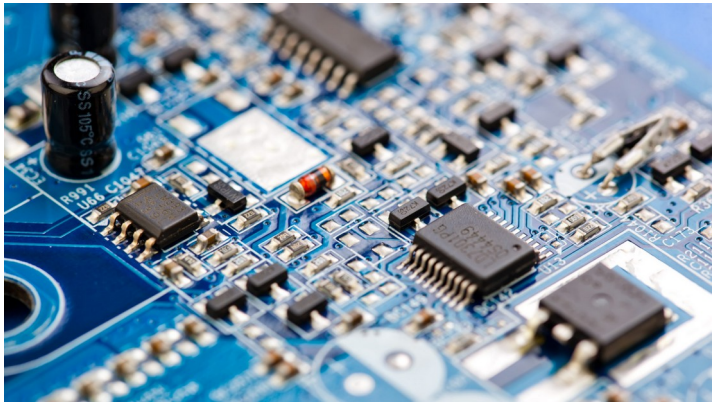
Informatik

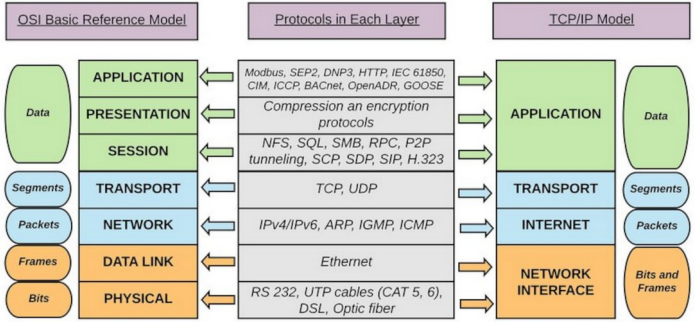
Grundkurs

Technische Informatik

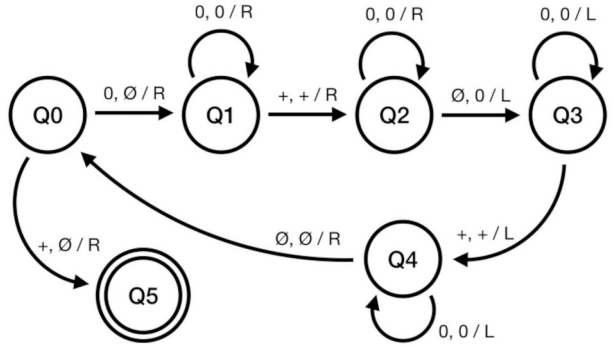


Praktische Informatik

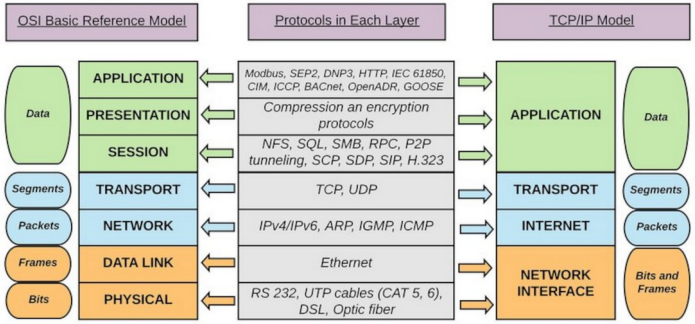
Angewandte Informatik



Theoretische Informatik



Angewandte Informatik



→ **Angewandte Informatik**
beschäftigt sich mit der praktischen Nutzung von Informatikmethoden, z. B. Netzwerken, Verschlüsselung und Datenkomprimierung, sowie ethischen und gesellschaftlichen Aspekten der IT.

Angewandte Informatik

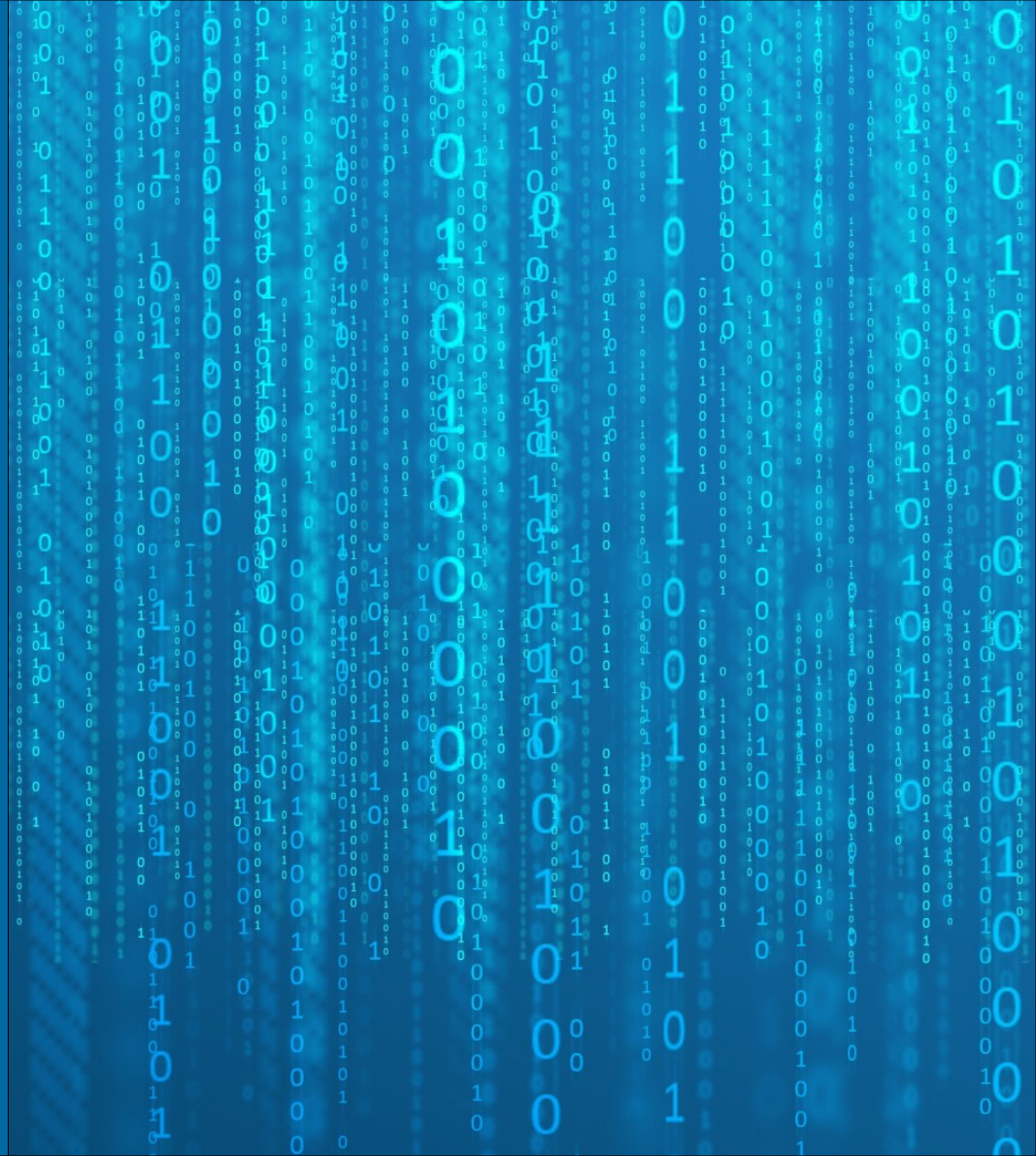
1. Das Client-Server-Modell
2. Datenaustausch im Schichtenmodell
3. Aufbau einer IP-Adresse (inkl. IPv4 & IPv6)
4. Namensauflösung durch DNS
5. Funktionsweise von URLs
6. Routing von Datenpaketen
7. Verlustfreie Datenkomprimierung
8. Verlustbehaftete Datenkomprimierung

Osterferien

9. Symmetrische & Asymmetrische Verschlüsselung
10. Sicherheit von Verschlüsselungsalgorithmen
11. Datenschutz und Datensicherheit
12. Gesellschaftliche und ethische Aspekte sozialer Netzwerke
13. Nutzen und Risiken aktueller Internetdienste
14. Fake News und Bewertung von Informationen

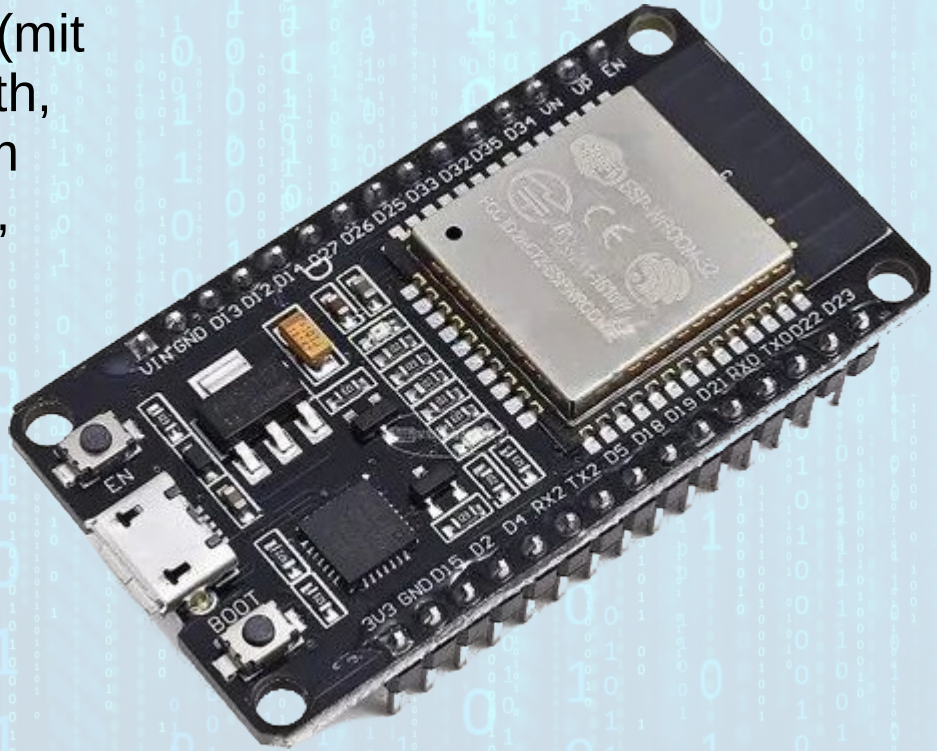
Heute:

- ESP-32-Projekt



ESP32

Der ESP32 ist ein günstiger Mikrocontroller von Espressif Systems aus Shanghai (mit Dual-Core-Prozessor, WLAN, Bluetooth, vielfältigen Anschlüssen und niedrigem Stromverbrauch. Er eignet sich für IoT, Wearables und mobile Geräte und unterstützt zahlreiche Entwicklungsplattformen.



ESP-WROOM-32 Module

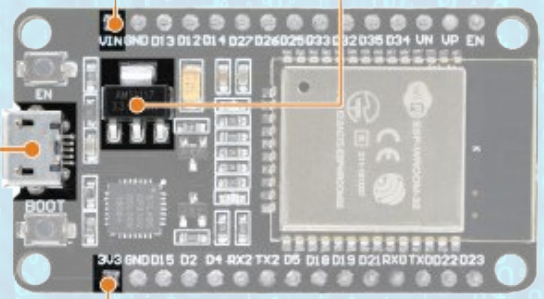
Switches & Indicators

- EN – Reset the ESP32 chip
- Boot – Download new programs
- Red LED – Power Indicator
- Blue LED – User Programmable

External Power Supply

3.3V LDO Voltage Regulator

Micro USB Connector



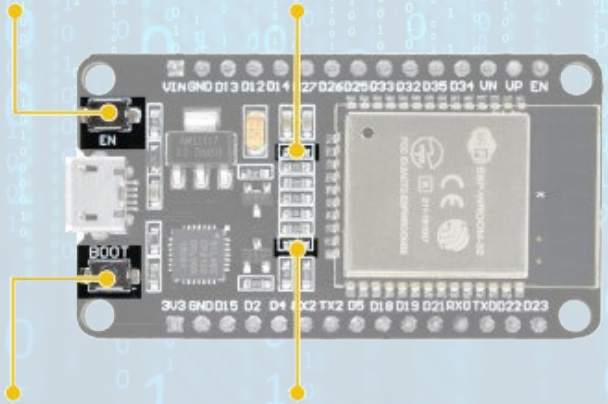
3.3V Output

ESP-WROOM-32 Chip

- Xtensa® Dual-Core 32-bit LX6
- Upto 240MHz Clock Freq.
- 520kB internal SRAM
- 4MB external flash
- 802.11b/g/n Wi-Fi transceiver
- Bluetooth 4.2/BLE

EN/Reset Button

Power Led



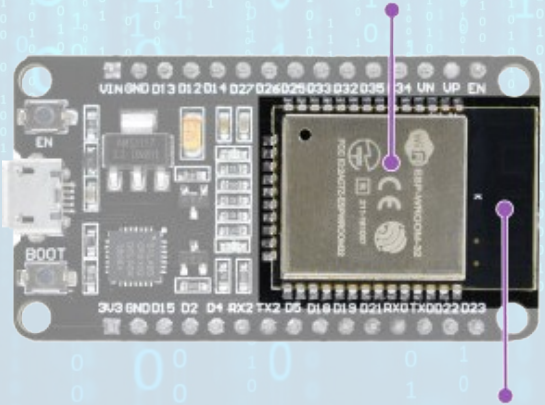
Boot Button

On Board Led D2 Pin

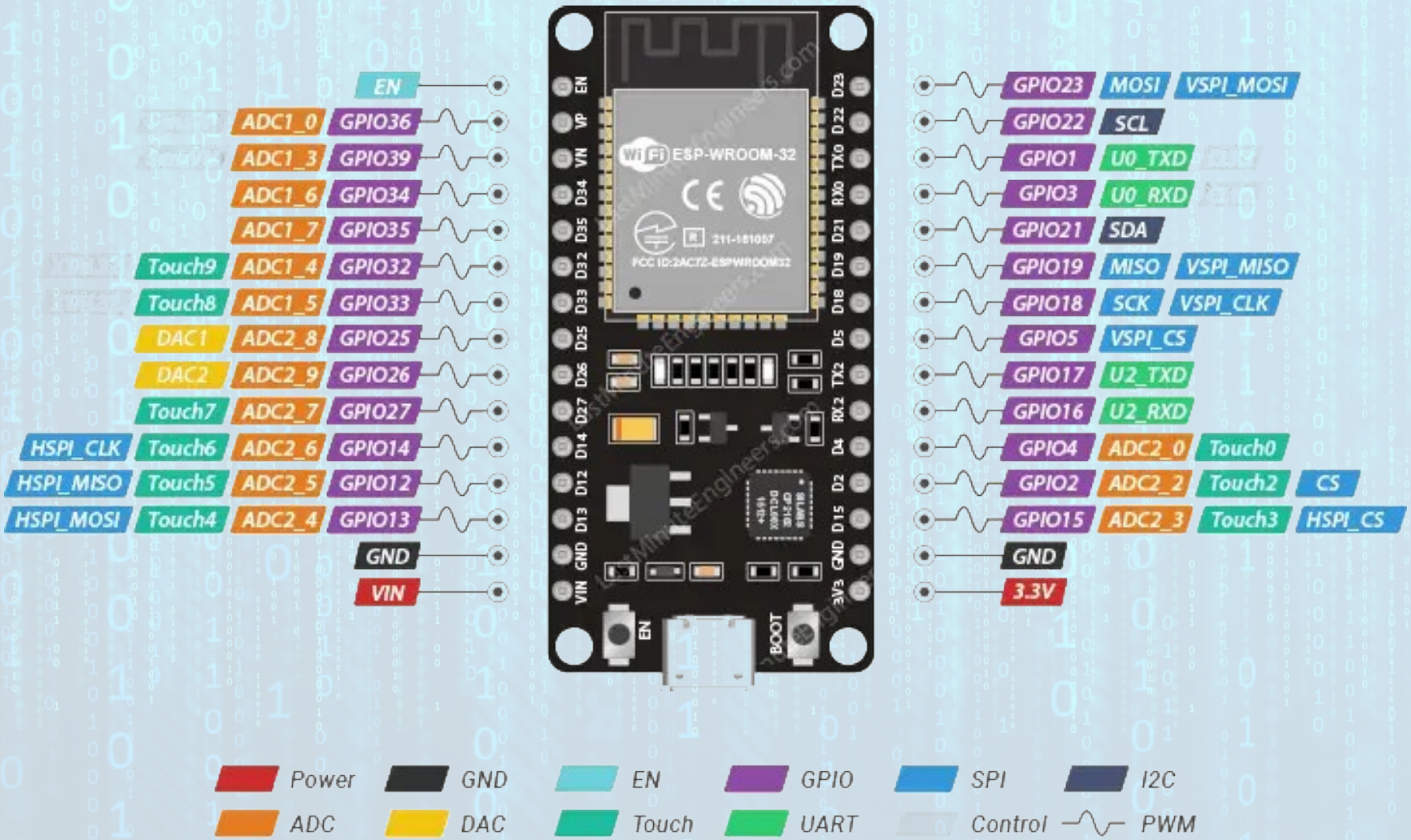
Power Requirement

- Operating Voltage: 2.2V to 3.6V
- On-board 3.3V 600mA regulator
- 5 µA during Sleep Mode
- 250mA during RF transmissions

ESP-WROOM-32 Chip



2.4GHz Antenna


EN
ADC1_0 GPIO36
ADC1_3 GPIO39
ADC1_6 GPIO34
ADC1_7 GPIO35
Touch9 ADC1_4 GPIO32
Touch8 ADC1_5 GPIO33
DAC1 ADC2_8 GPIO25
DAC2 ADC2_9 GPIO26
Touch7 ADC2_7 GPIO27
HSPI_CLK Touch6 ADC2_6 GPIO14
HSPI_MISO Touch5 ADC2_5 GPIO12
HSPI_MOSI Touch4 ADC2_4 GPIO13
GND
VIN
EN VP VN D34 D35 D32 D33 D25 D26 D27 D14 D12 D13 GND VIN
D23 D22 TX0 RX0 D21 D19 D18 D5 TX2 RX2 D4 D2 D15 GND 3V3
Wi Fi ESP-WROOM-32
211-181057
FCC ID:2AC7Z-ESPWROOM32
EN
BOOT
GPIO23 MOSI VSPI_MOSI
GPIO22 SCL
GPIO1 U0_TXD
GPIO3 U0_RXD
GPIO21 SDA
GPIO19 MISO VSPI_MISO
GPIO18 SCK VSPI_CLK
GPIO5 VSPI_CS
GPIO17 U2_TXD
GPIO16 U2_RXD
GPIO4 ADC2_0 Touch0
GPIO2 ADC2_2 Touch2 CS
GPIO15 ADC2_3 Touch3 HSPI_CS
GND
3.3V
Power
GND
EN
GPIO
SPI
I2C
ADC
DAC
Touch
UART
Control
PWM

Das ESP32-Projekt

In den kommenden Wochen wirst du mit einem ESP32-Mikrocontroller ein eigenes Projekt entwickeln. Dabei hast du die Freiheit, kreativ zu sein oder dich von bestehenden Projekten inspirieren zu lassen. Dein Ziel ist es, den ESP32 für eine spannende Anwendung zu nutzen, die du am Ende präsentieren kannst.

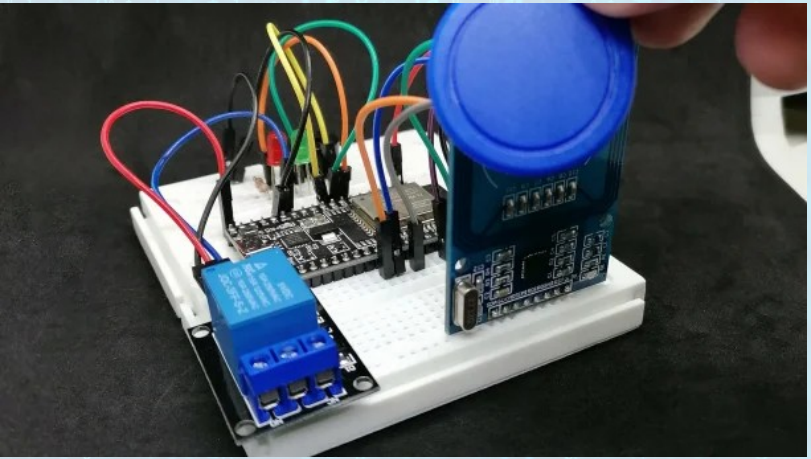
So gehst du vor:

1. Ideenfindung: Überlege dir, welches Projekt du umsetzen möchtest. Beispiele könnten eine Wetterstation, eine Steuerung für LEDs oder ein kleines IoT-Gerät sein.
2. Planung: Erstelle einen Plan, was du für dein Projekt brauchst (Hardware, Software) und welche Schritte nötig sind.
3. Programmierung: Lerne die Grundlagen von Python und bringe deinem ESP32 bei, was er tun soll. Nutze dabei Tutorials, Dokumentationen oder ChatGPT für Unterstützung.
4. Umsetzung: Setze dein Projekt Schritt für Schritt um. Teste regelmäßig, ob alles funktioniert.
5. Dokumentation: Halte fest, wie du vorgegangen bist, welche Probleme aufgetreten sind und wie du sie gelöst hast.
6. Präsentation: Stelle dein Projekt der Klasse vor und erkläre, wie es funktioniert.

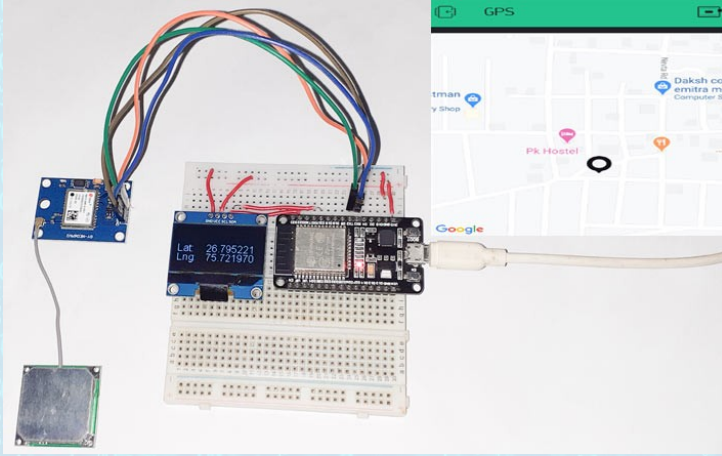
Voraussetzungen:

- Du erhältst einen ESP32 als Leihgabe. Bitte behandle ihn sorgfältig.
- Du kannst bestehende Projekte nutzen, musst aber erklären können, was du daran angepasst oder gelernt hast.

Das ESP32-Projekt: Ideenfindung

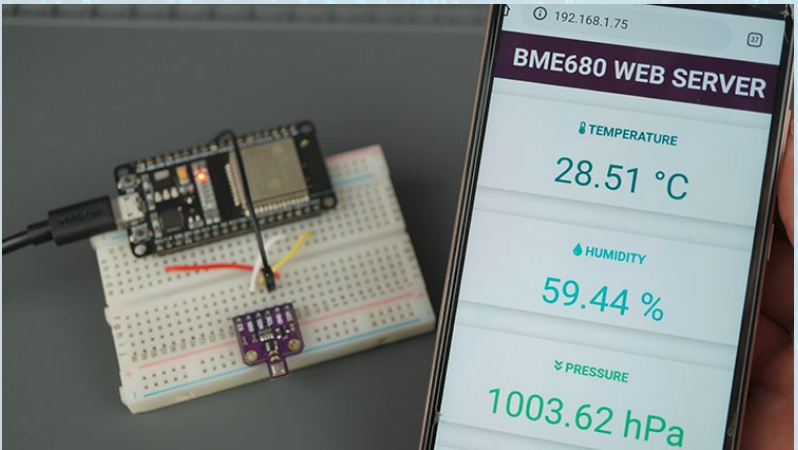


RFID-Zugangskontrolle

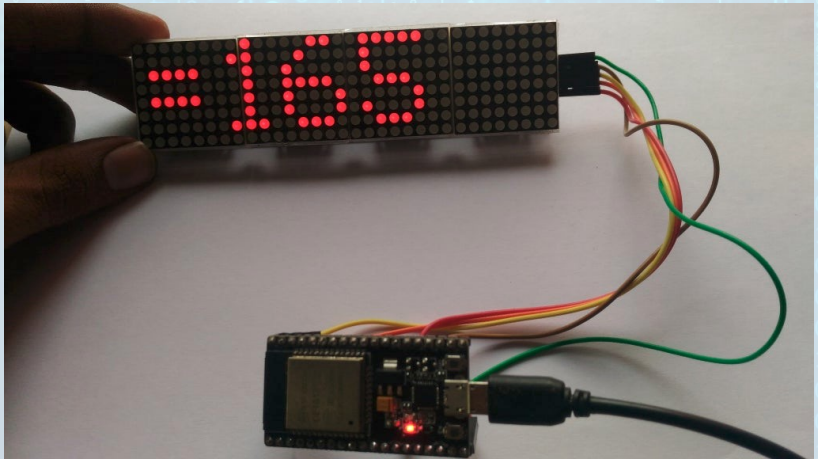


GPS-Tracker

Wetterstation mit Webserver

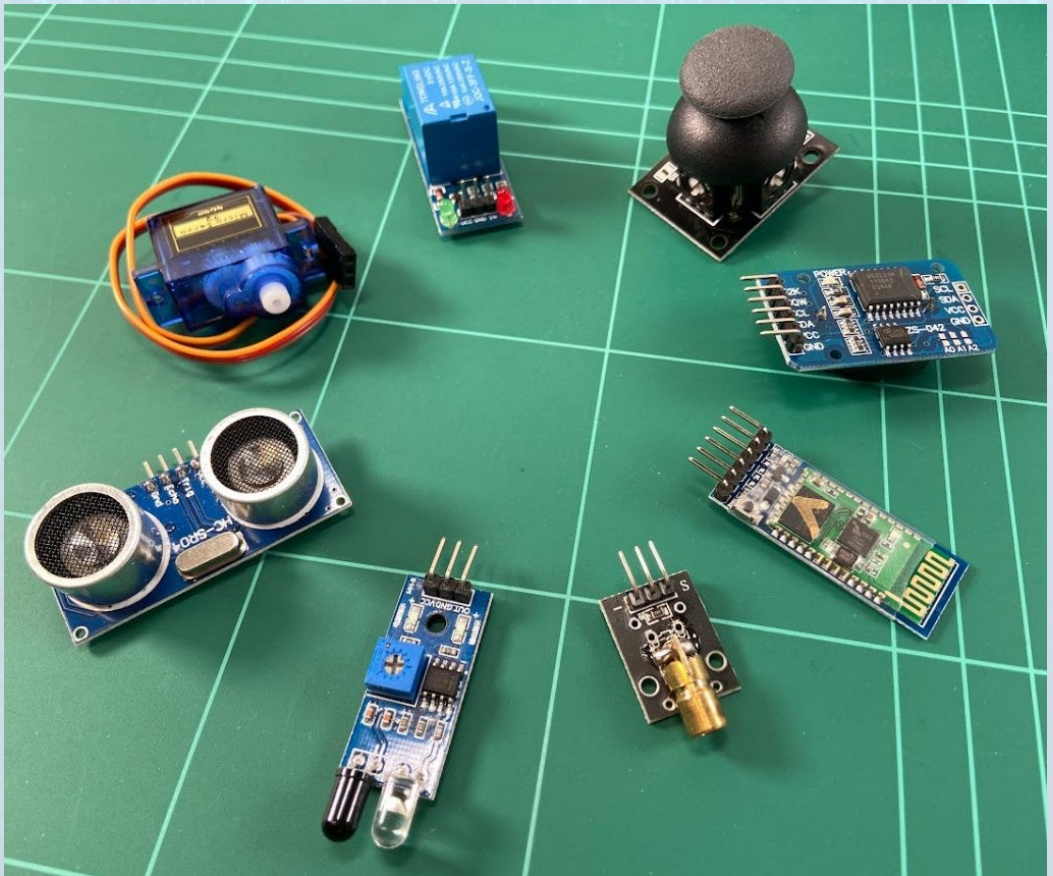


Follower-Zähler



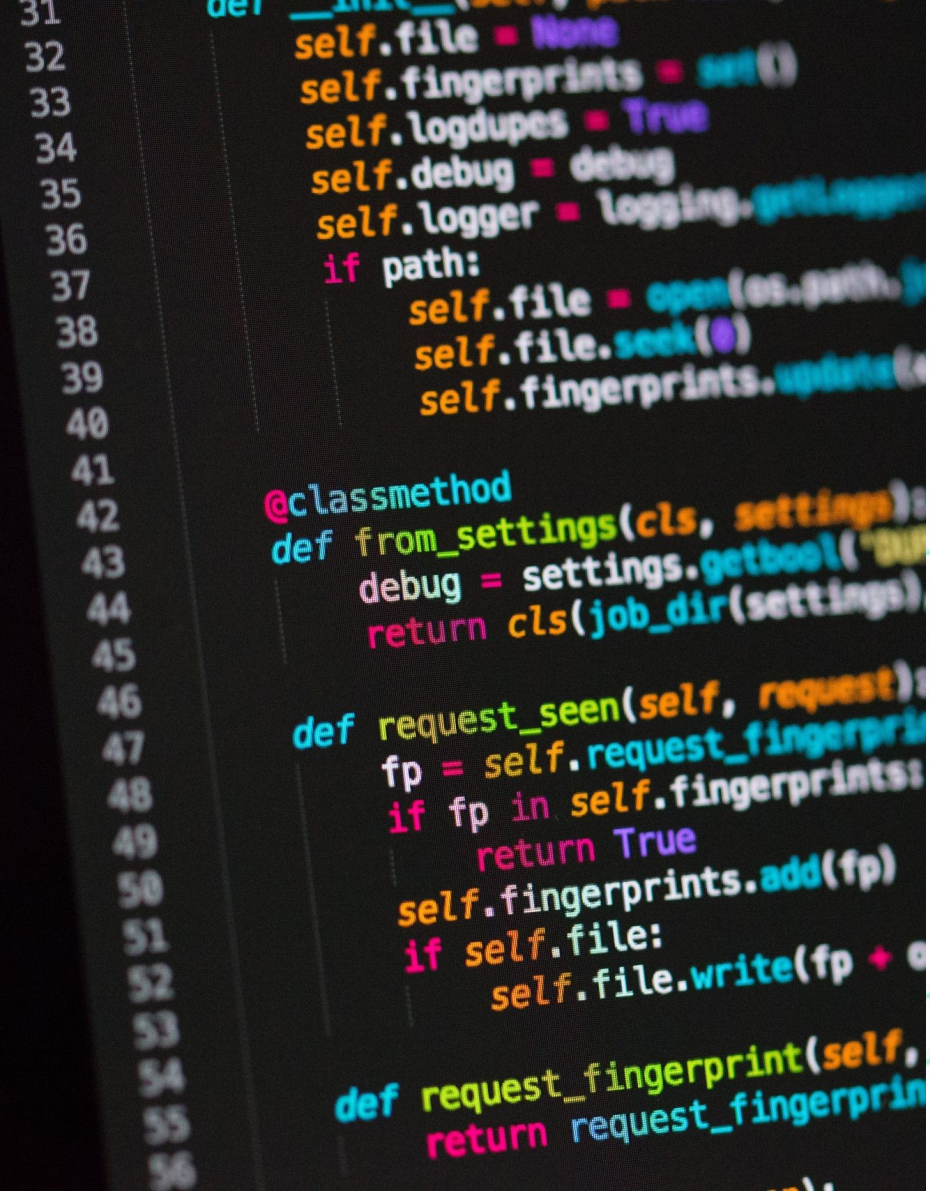
Sensoren

- Beschleunigungssensor
- Farbsensor
- Funkmodul
- Gassensor (z. B. CO2, Methan, Alkohol)
- GPS-Modul
- Infrarotsensor
- Kraftsensor
- Lichtsensor
- Magnetsensor
- Mikrowellensensor
- Temperatursensor



Aktoren

- Relaismodule
- Servomotoren
- Schrittmotoren
- DC-Motoren
- LEDs
- Piezo-Summer
- Heizelemente
- Displays
- Lautsprecher

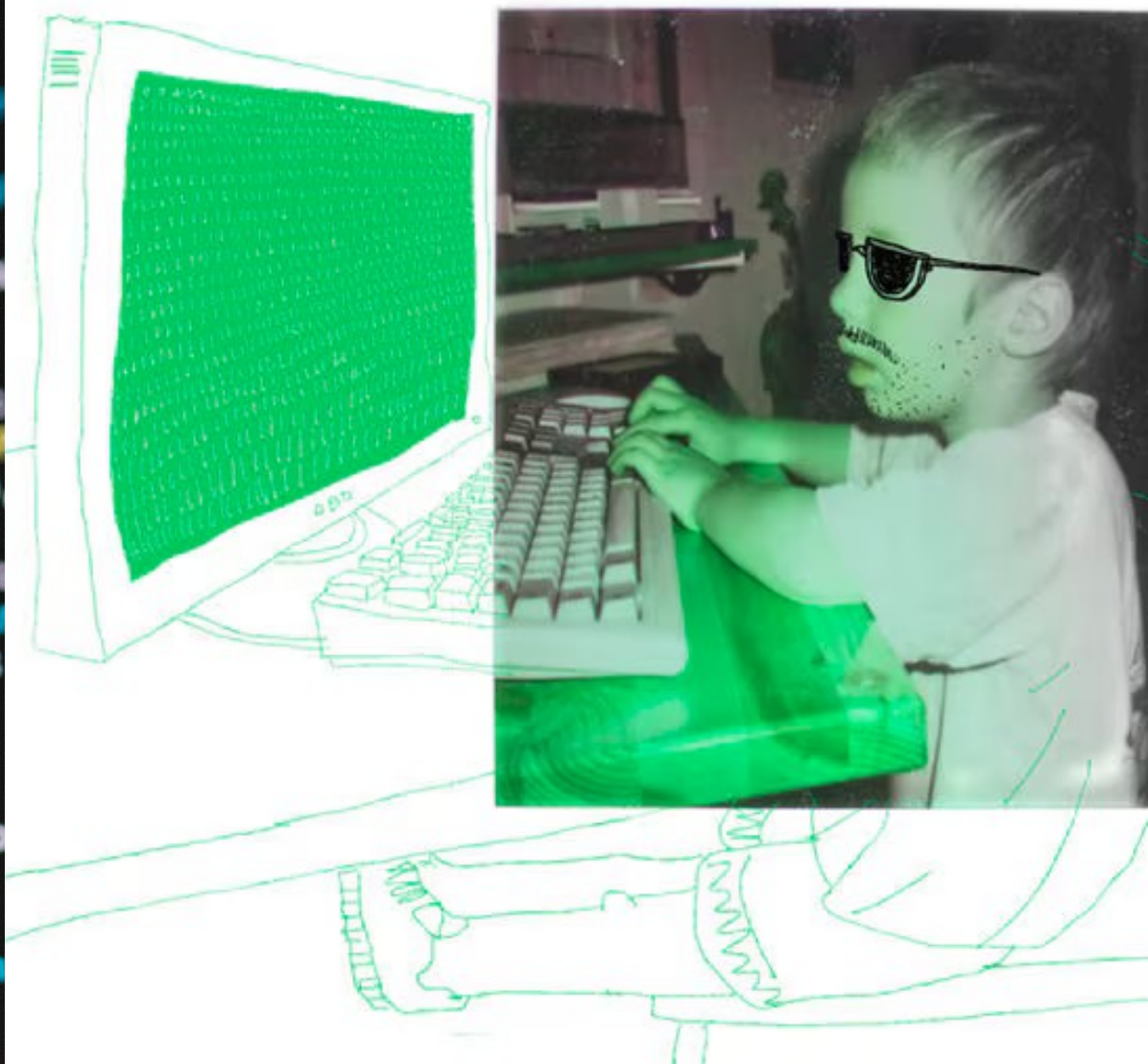


```
        self.file = None
        self.fingerprints = set()
        self.logdupes = True
        self.debug = debug
        self.logger = logging.
        if path:
            self.file = open(os.path.
            self.file.seek(0)
            self.fingerprints.

    @classmethod
    def from_settings(cls, settings):
        debug = settings.getbool("
        return cls(job_dir(settings)

    def request_seen(self, request):
        fp = self.request_fingerpri
        if fp in self.fingerprints:
            return True
        self.fingerprints.add(fp)
        if self.file:
            self.file.write(fp + o

    def request_fingerprint(self,
        return request_fingerprin
```

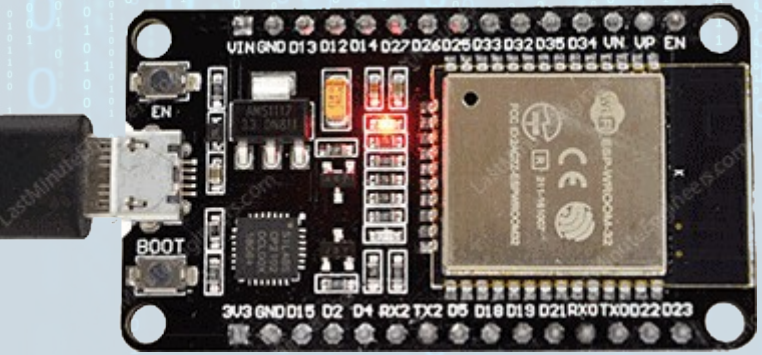


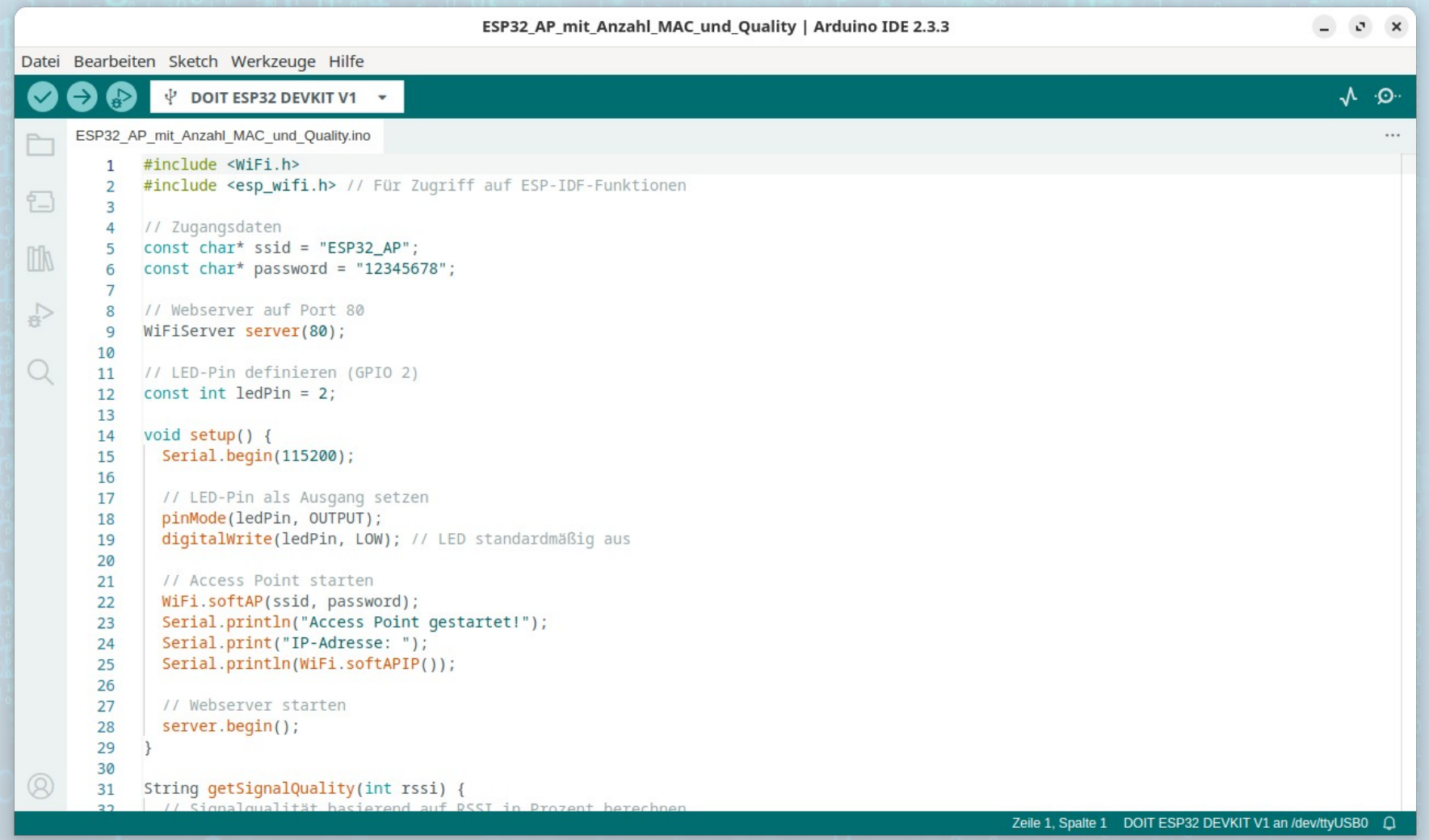
PW: 12345678

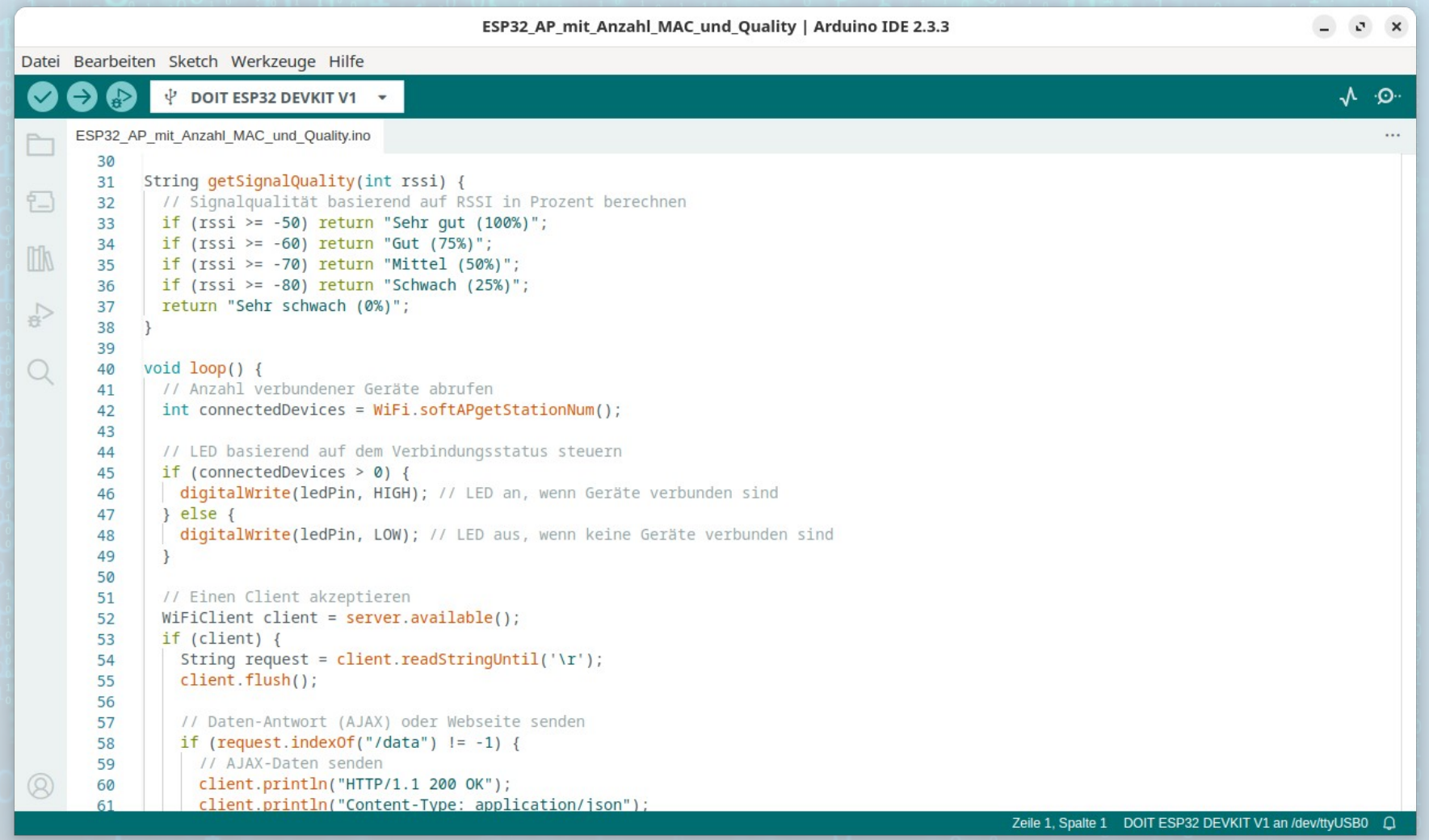
ESP32 Access Point

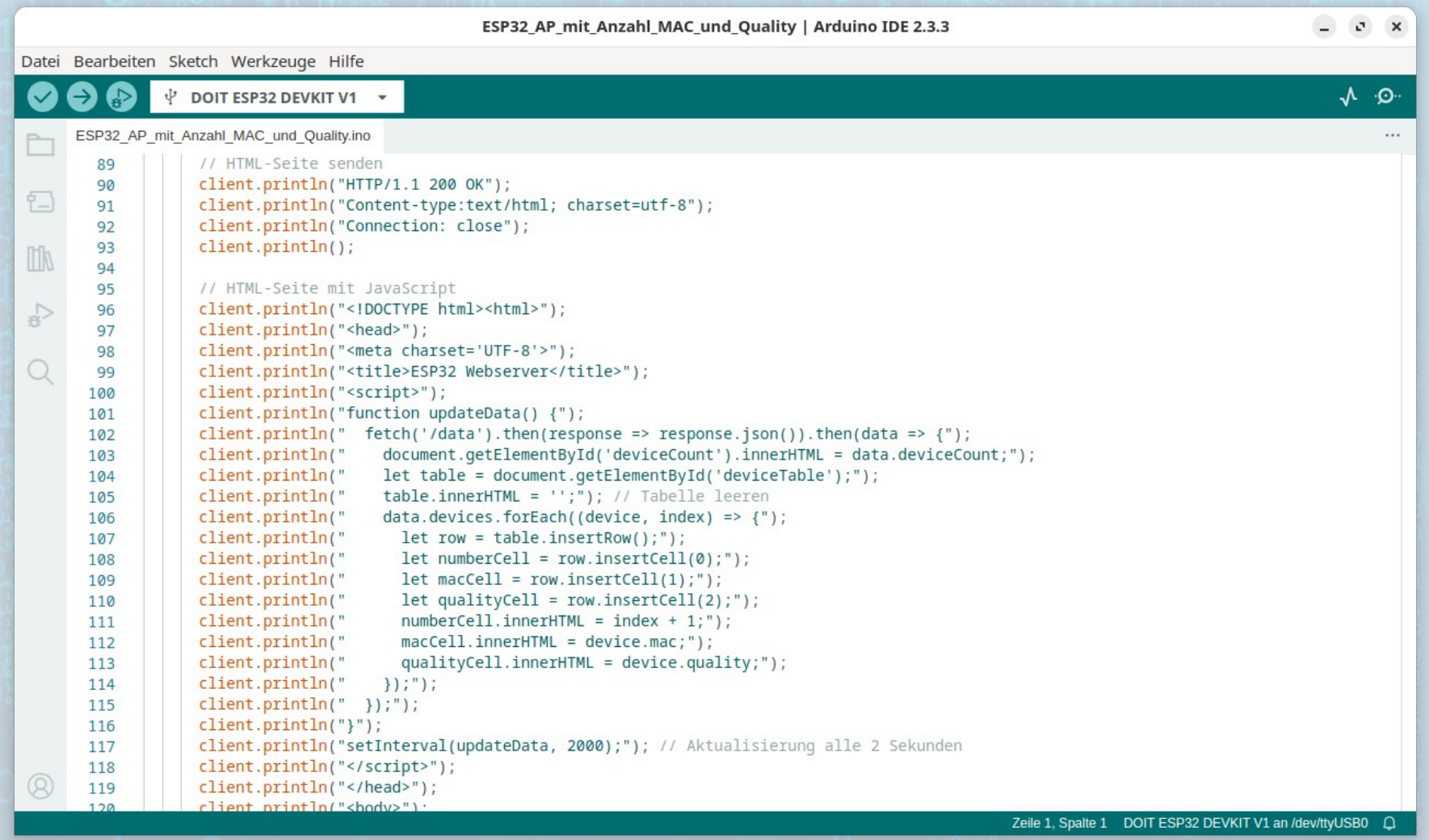
Verbunden: **2** Gerät(e)

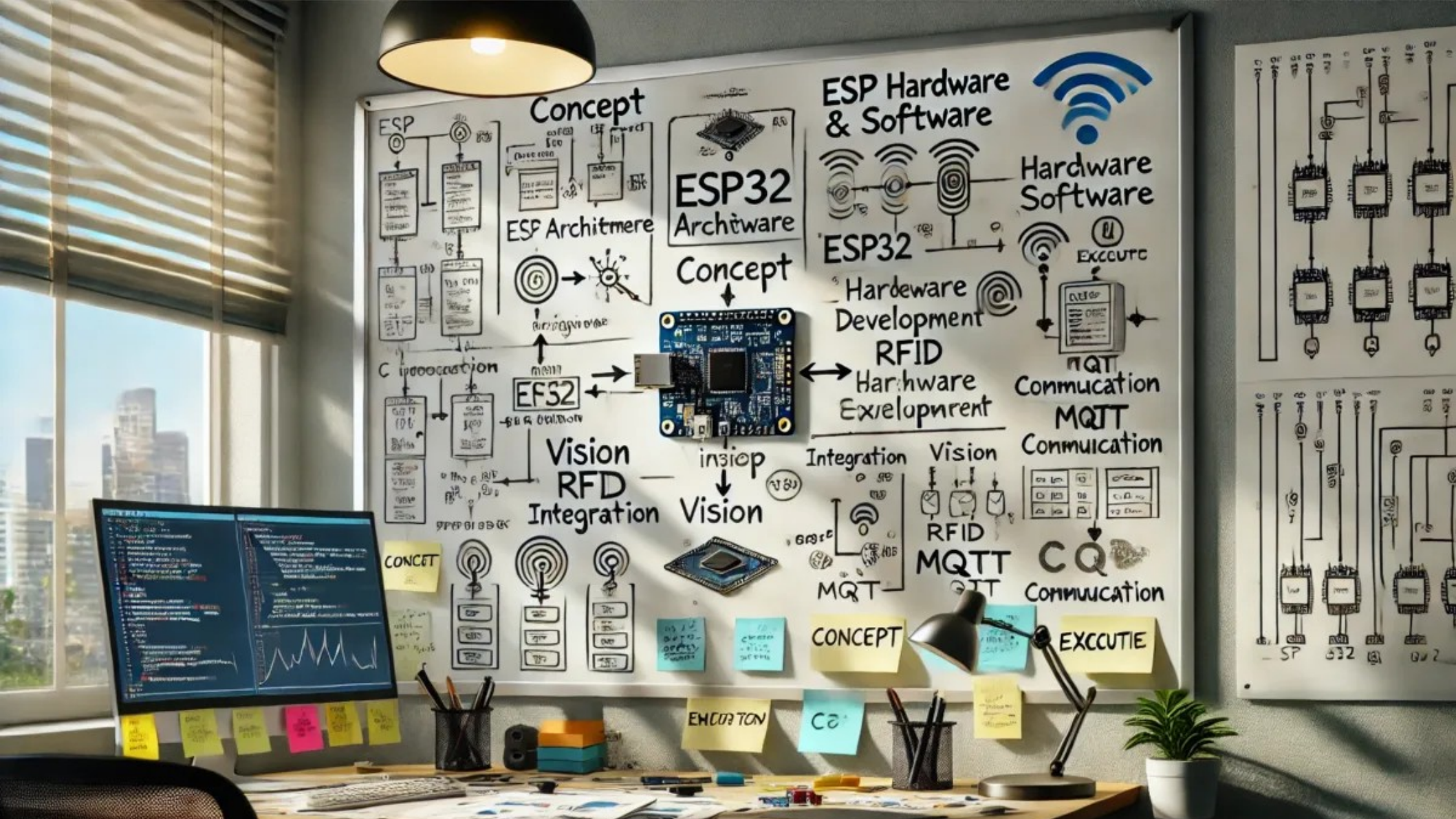
#	MAC-Adresse	Signalqualität
1	[illegible]	Sehr gut (100%)
2	[illegible]	Sehr gut (100%)




ESP32_AP_mit_Anzahl_MAC_und_Quality | Arduino IDE 2.3.3
Datei Bearbeiten Sketch Werkzeuge Hilfe
DOIT ESP32 DEVKIT V1
ESP32_AP_mit_Anzahl_MAC_und_Quality.ino
#include <WiFi.h>
#include <esp_wifi.h> // Für Zugriff auf ESP-IDF-Funktionen
// Zugangsdaten
const char* ssid = "ESP32_AP";
const char* password = "12345678";
// Webserver auf Port 80
WiFiServer server(80);
// LED-Pin definieren (GPIO 2)
const int ledPin = 2;
void setup() {
Serial.begin(115200);
// LED-Pin als Ausgang setzen
pinMode(ledPin, OUTPUT);
digitalWrite(ledPin, LOW); // LED standardmäßig aus
// Access Point starten
WiFi.softAP(ssid, password);
Serial.println("Access Point gestartet!");
Serial.print("IP-Adresse: ");
Serial.println(WiFi.softAPIP());
// Webserver starten
server.begin();
}
String getSignalQuality(int rssi) {
Zeile 1, Spalte 1
DOIT ESP32 DEVKIT V1 an /dev/ttyUSB0


ESP32_AP_mit_Anzahl_MAC_und_Quality | Arduino IDE 2.3.3
Datei Bearbeiten Sketch Werkzeuge Hilfe
DOIT ESP32 DEVKIT V1
ESP32_AP_mit_Anzahl_MAC_und_Quality.ino
30
31 String getSignalQuality(int rssi) {
32 // Signalqualität basierend auf RSSI in Prozent berechnen
33 if (rssi >= -50) return "Sehr gut (100%)";
34 if (rssi >= -60) return "Gut (75%)";
35 if (rssi >= -70) return "Mittel (50%)";
36 if (rssi >= -80) return "Schwach (25%)";
37 return "Sehr schwach (0%)";
38 }
39
40 void loop() {
41 // Anzahl verbundener Geräte abrufen
42 int connectedDevices = WiFi.softAPgetStationNum();
43
44 // LED basierend auf dem Verbindungsstatus steuern
45 if (connectedDevices > 0) {
46 digitalWrite(ledPin, HIGH); // LED an, wenn Geräte verbunden sind
47 } else {
48 digitalWrite(ledPin, LOW); // LED aus, wenn keine Geräte verbunden sind
49 }
50
51 // Einen Client akzeptieren
52 WiFiClient client = server.available();
53 if (client) {
54 String request = client.readStringUntil('\r');
55 client.flush();
56
57 // Daten-Antwort (AJAX) oder Webseite senden
58 if (request.indexOf("/data") != -1) {
59 // AJAX-Daten senden
60 client.println("HTTP/1.1 200 OK");
61 client.println("Content-Type: application/json");
Zeile 1, Spalte 1
DOIT ESP32 DEVKIT V1 an /dev/ttyUSB0


ESP32_AP_mit_Anzahl_MAC_und_Quality | Arduino IDE 2.3.3
Datei Bearbeiten Sketch Werkzeuge Hilfe
DOIT ESP32 DEVKIT V1
ESP32_AP_mit_Anzahl_MAC_und_Quality.ino
// HTML-Seite senden
client.println("HTTP/1.1 200 OK");
client.println("Content-type:text/html; charset=utf-8");
client.println("Connection: close");
client.println();
// HTML-Seite mit JavaScript
client.println("<!DOCTYPE html><html>");
client.println("<head>");
client.println("<meta charset='UTF-8'>");
client.println("<title>ESP32 Webserver</title>");
client.println("<script>");
client.println("function updateData() {");
client.println(" fetch('/data').then(response => response.json()).then(data => {");
client.println("   document.getElementById('deviceCount').innerHTML = data.deviceCount;");
client.println("   let table = document.getElementById('deviceTable');");
client.println("   table.innerHTML = '';"); // Tabelle leeren
client.println("   data.devices.forEach((device, index) => {");
client.println("   let row = table.insertRow();");
client.println("   let numberCell = row.insertCell(0);");
client.println("   let macCell = row.insertCell(1);");
client.println("   let qualityCell = row.insertCell(2);");
client.println("   numberCell.innerHTML = index + 1;");
client.println("   macCell.innerHTML = device.mac;");
client.println("   qualityCell.innerHTML = device.quality;");
client.println("   });");
client.println(" });");
client.println("}");
client.println("setInterval(updateData, 2000);"); // Aktualisierung alle 2 Sekunden
client.println("</script>");
client.println("</head>");
client.println("<body>");
Zeile 1, Spalte 1
DOIT ESP32 DEVKIT V1 an /dev/ttyUSB0

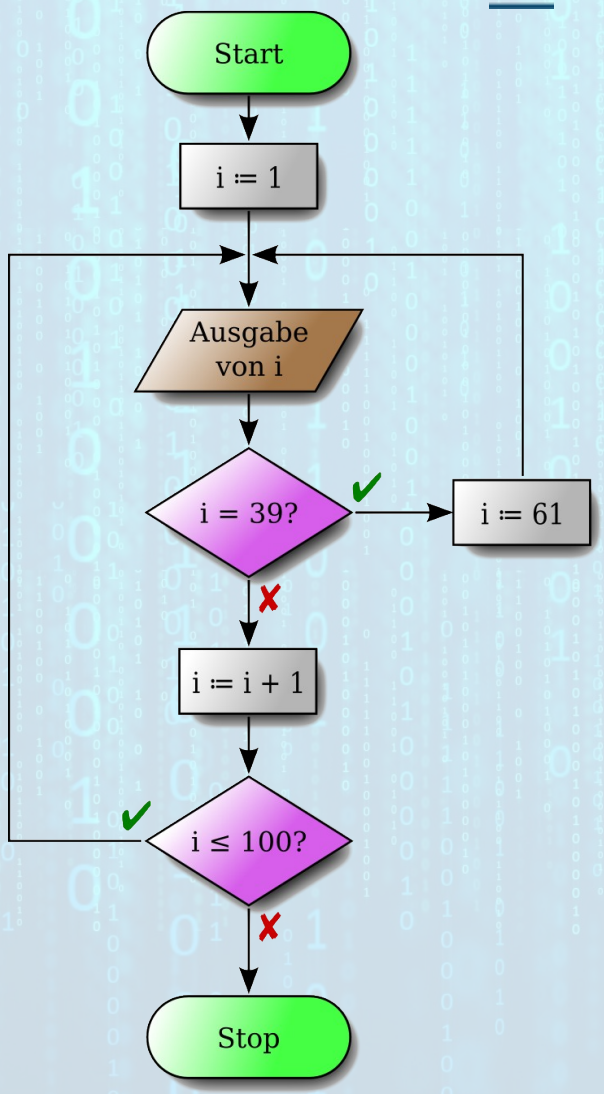

Concept
ESP
ESP Architmere
ESP32
Architware
Concept
ESP Hardware
& Software
Hardware
Software
ESP32
Hardeware
Development
RFID
Hardware
Exvelopment
EXCCUTC
MQT
Commucation
MQTT
Commucation
EFS2
Vision
RFD
Integration
Vision
Integration
Vision
RFID
MQTT
MQT
CONCET
CONCEPT
EXCCUTIE
Commucation

DIN 66001

Programmablaufplan

Ein Programmablaufplan (PAP) ist eine grafische Darstellung, die den Ablauf eines Programms mithilfe von standardisierten Symbolen zeigt. Er stellt die Reihenfolge und Verzweigungen von Befehlen übersichtlich dar und hilft bei der Planung und Umsetzung von Programmen.

1. Start: Setze i = 1.
2. Ausgabe: Gib i aus.
3. Prüfung: Ist i == 39?
4. Ja: Setze i = 61.
5. Nein: Erhöhe i um 1.
6. Schleifenbedingung: Ist i ≤ 100?
7. Ja: Wiederhole ab Schritt 2.
8. Nein: Beende das Programm.

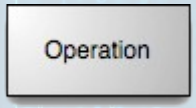


Programmablaufplan: Elemente

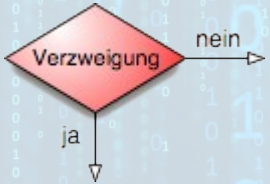
Kreis; Oval / Rechteck mit gerundeten Ecken: Terminator

Pfeil, Linie: Verbindung zum nächstfolgenden Element

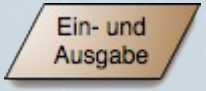
Rechteck: Operation (Tätigkeit)

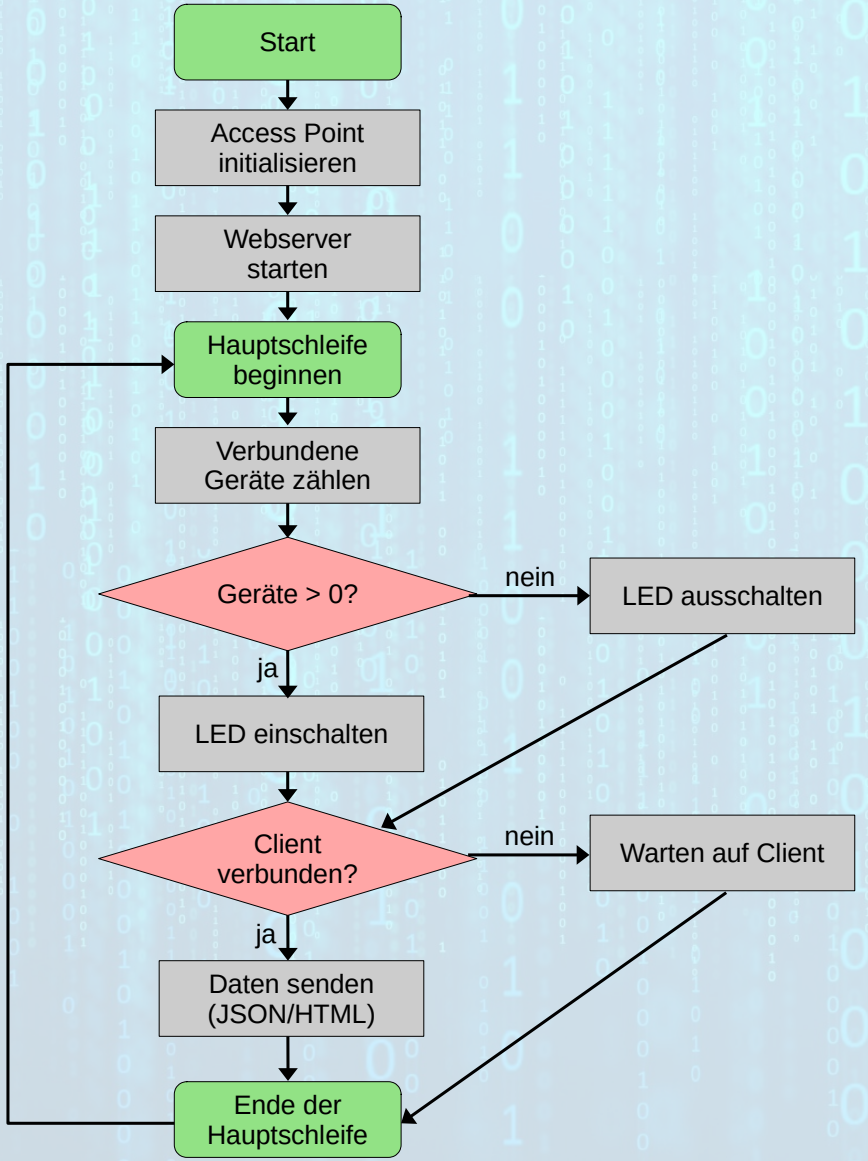


Raute: Verzweigung / Entscheidungen



Parallelogramm: Ein- und Ausgabe





Aufgabenstellung: Planung und Programmablauf für dein ESP32-Projekt

In dieser Phase geht es darum, dein Projekt genau zu planen und eine erste Struktur für die Umsetzung zu entwickeln. Dein Ziel ist es, eine klare Idee deines Projekts zu haben, die benötigte Hardware festzulegen und einen Programmablaufplan zu erstellen.

Deine Aufgaben:

1. Idee beschreiben:

Schreibe in 3–5 Sätzen, was dein Projekt machen soll und warum es sinnvoll oder spannend ist. Denke dabei daran, dass du deine Idee später präsentieren musst.

2. Hardware planen:

Liste die Sensoren und Aktoren auf, die du für dein Projekt benötigst (z. B. Temperatur- oder Lichtsensor, LEDs, Motoren). Notiere dir auch, welche zusätzlichen Bauteile wie Kabel, Widerstände oder Module du brauchst.

3. Programmablauf planen:

Erstelle einen Programmablaufplan, der die grundlegende Logik deines Programms zeigt. Überlege dir:

- Was soll dein Programm zuerst machen (z. B. Initialisierung der Hardware)?
- Welche Schritte folgen dann? (z. B. Daten von Sensoren auslesen, LEDs steuern)
- Wie wird dein Programm enden oder auf Benutzereingaben reagieren?

Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe

esp32-arduino.ino - Wok

https://wokwi.com/projects/420793132234787841

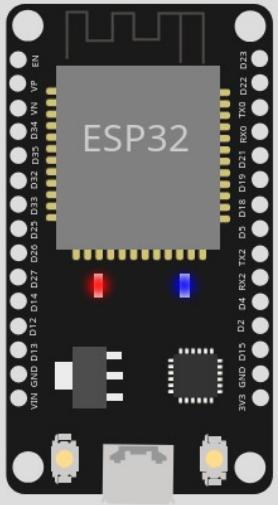
WOKWI SAVE SHARE Docs SIGN IN

esp32-arduino.ino diagram.json Library Manager

```
1  int ledPin = 2;
2
3  void setup() {
4    pinMode(ledPin, OUTPUT);
5    Serial.begin(115200);
6    Serial.println("Hello, ESP32!");
7  }
8
9  void loop() {
10   digitalWrite(ledPin, HIGH);
11   delay(500);
12   digitalWrite(ledPin, LOW);
13   delay(500);
14 }
15
```

Simulation

00:46.796 36%




```
p_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
```