boot.py und main.py

Recherchieren Sie, was das `boot.py` auf dem ESP32 bedeutet/macht. Suchen Sie auch, was mit einer Datei passiert, welche als `main.py` auf dem ESP32 abgespeichert wird.

Z.B. hier: Boot-Scripts

Drahtlos-Verbindung

Der ESP32-Chip ist Bluetooth- und WLAN-fähig. Er kann sich mit einem WLAN-Netzwerk verbinden oder selbst eines erzeugen. Diese Funktion muss jedoch noch aktiviert werden:

- Verbinden Sie den ESP32 mit ihrem Laptop und öffnen Sie Thonny.
- Im unteren Fenster (Shell) sollten nun wieder die Zeichen >>> erscheinen. Drücken Sie ansonsten auf 'STOP' oder stecken Sie den ESP32 aus und wieder ein.
- Führen Sie nun folgenden Befehl aus:

```
import webrepl_setup
```

- Bestätigen Sie das Aktivieren der WLAN-Funktion mit E.
- Führen Sie mit y einen Reboot des ESP32 durch.
- Nun kann ein WLAN-Netzwerk erstellt werden. Z.B. mit:

```
import network
ap = network.WLAN(network.AP_IF)
ap.active(True)
ap.config(essid = 'ESP32-Wifi')
ap.config(authmode=3, password ='hallo123')
```

- Der Befehl `network.WLAN(network.AP_IF)` aktiviert einen sogenannten Access-Point, also ein WLAN-Netzwerk. Mit `ap.active(True)` wird dieser aktiviert. Mit `ap.config(…)` kann dann der Name (`essid`) und das Passwort des WLANs konfiguriert werden.
- Per Smartphone oder Notebook können Sie sich nun ins ESP32-Wifi einloggen.
- Nach der Verbindung passiert nichts, da der ESP32 lediglich ein Netzwerk erstellt hat, wir haben aber keine weiteren Befehle verwendet, um z.B. Text oder eine Webseite auszugeben… Im Folgenden können Sie aber einen Web-Server mit einfacher Webseite mit dem ESP32 erstellen.

Web-Server

Versuchen Sie den Code von oben (Drahtlos-Verbindung) zu ergänzen, dass Sie mit dem Smartphone/Notebook eine Webseite auf ihrem ESP32 abrufen können. Auf der Webseite soll ein Text (z.B. „Hallo Welt!“) erscheinen. Verwenden Sie ChatGPT oder folgen Sie dieser Anleitung.

Mögliche ChatGPT-Prompts:

- Programmiere in Micropython eine Webseite auf einem ESP32 mit dem Text „Hallo Welt!“. Man soll sich mit dem WLAN, welches vom ESP32 erzeugt wird, verbinden können und dann soll

„Hallo Welt!“ erscheinen.

Beispiel Lösung / Code

```
import socket, network

ap = network.WLAN(network.AP_IF)
ap.active(True)
ap.config(essid = 'ESP32-Wifi') #Wählen Sie den Namen für das WLAN
ap.config(authmode=3, password ='sensorsensor') #Wählen Sie ein Passwort


def webseite(): #Diese Funktion definiert ihre HTML-Webseite
  html = """<html><head><meta name="viewport" content="width=device-width,
initial-scale=1"</head><body><h1>Hallo Welt!</h1></body></html>"""
  return html

addr = socket.getaddrinfo('0.0.0.0', 80)[0][-1]
s = socket.socket()
s.bind(addr)
s.listen(5)

while True:
  conn, addr = s.accept()
  print('Verbunden mit %s' % str(addr))
  request = conn.recv(1024)
  print('Inhalt = %s' % str(request))
  response = webseite()
  conn.send(response)
  conn.close()
```

Neopixel per Smartphone steuern

Die Steuerung von der Neopixel-LED kann mit der Funktion des Webservers kombiniert werden, um die LED per Smartphone zu steuern.

- Versuchen Sie den ESP32 so zu programmieren, dass die Neopixel-LED per WLAN über das Smartphone gesteuert werden kann.

Beispiel Lösung / Code

```
import socket, machine, neopixel
import network

np = neopixel.NeoPixel(machine.Pin(7),1)

np[0] = (255,0,0)
```

```
np.write()

ap = network.WLAN(network.AP_IF)
ap.active(True)
ap.config(essid = 'Sensor10')
ap.config(authmode=3, password ='sensorsensor')
addr = socket.getaddrinfo('0.0.0.0', 80)[0][-1]
s = socket.socket()
s.bind(addr)
s.listen(1)
ledstatus = "ROT"

print('listening on', addr)


def web_page():
  html = """<html><head> <title>Mini Web-Server</title> <meta
name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" href="data:,"> <style>html{font-family: Helvetica;
display:inline-block; margin: 0px auto; text-align: center;}
  h1{color: #0F3376; padding: 2vh;}p{font-size: 1.5rem;}.button{display:
inline-block; background-color: #d60c0c; border: none;
  border-radius: 4px; color: white; padding: 16px 40px; text-decoration:
none; font-size: 30px; margin: 2px; cursor: pointer;}
  .button2{background-color: #47a730;}</style></head><body> <h1>Mini Web-
Server</h1>
  <p>Neopixel-Status: <strong>""" + ledstatus + """</strong></p><p><a
href="/?led=rot"><button class="button">ROT</button></a></p>
  <p><a href="/?led=grun"><button class="button
button2">GR&Uuml;N</button></a></p></body></html>"""
  return html


while True:
  conn, addr = s.accept()
  print('Verbunden mit %s' % str(addr))
  request = conn.recv(1024)
  request = str(request)
  print('Content = %s' % request)
  ledr = request.find('/?led=rot')
  ledg = request.find('/?led=grun')
  if ledr == 6:
    print('Neopixel rot')
    np[0] = (255, 0, 0)
    ledstatus="ROT"
    np.write()
  if ledg == 6:
    print('Neopixel grün')
    np[0] = (0, 100, 0)
    ledstatus="GR&Uuml;N"
    np.write()
```

```
  response = web_page()
  conn.send('HTTP/1.1 200 OK\n')
  conn.send('Content-Type: text/html\n')
  conn.send('Connection: close\n\n')
  conn.sendall(response)
  conn.close()
```

* Webserver mit BME280 und OLED

Bonus: Versuchen Sie eine Webseite auf dem ESP32 zu programmieren, die die Messwerte des BME280-Sensors anzeigt. Zudem sollen die Messwerte auf dem OLED-Display ausgegeben werden. Die Messwerte sollen sich jeweils beim Neuladen der Webseite aktualisieren bzw. neu gemessen werden.