

Eine Messreihe, die sich meldet

Workshoptag 3 • Abschlussprojekt

Lösungsblatt

Diese Fassung enthält die Lösungen.

Ein Messgerät, das nur einmal misst, ist keins. Bau ein Programm, das von allein weitermisst und sich meldet, wenn ein Wert aus dem Rahmen fällt. Es verbindet Schleife und Verzweigung von gestern mit dem Sensor und dem Dictionary von heute.

So läuft es

50 Minuten, allein oder zu zweit. Zu zweit könnt ihr weiterarbeiten, wenn ein Aufbau streikt.

Die **letzten 10 Minuten** gehören der Vorstellung: zwei Minuten pro Programm, kein Vortrag. Laufen lassen und eine Sache zeigen, die du selbst gebaut hast.

Erst die Grundversion, sie ist für alle gleich. Danach suchst du dir aus den Ausbaustufen aus, was dich interessiert. Sie bauen nicht aufeinander auf.

Die Grundversion

Das Ziel für alle. Dein Programm soll

- alle paar Sekunden neu messen und nicht von allein aufhören,
- die drei Messwerte in ein **Dictionary** legen, wie in Aufgabe 5,
- sie beschriftet und gerundet ausgeben, den Druck in hPa,
- einen Wert mit einer **Grenze** vergleichen und warnen, wenn er darüber liegt,
- Grenze und Wartezeit als Variablen ganz oben stehen haben.

Welche Grenze sinnvoll ist, entscheidest du. Sie soll sich beim Anhauchen des Sensors auflösen lassen, sonst siehst du nie, ob die Warnung funktioniert.

Lösung

projekt.py

```
from machine import Pin, I2C
import bme280
import time

grenze = 24.0
wartezeit = 3

i2c = I2C(0, sda=Pin(10), scl=Pin(11))
sensor_bme = bme280.BME280(i2c=i2c)

while True:
    temperatur, druck, feuchte = sensor_bme.read_compensated_data()

    messung = {"temperatur": round(temperatur, 1),
               "druck": round(druck / 100, 1),
               "feuchte": round(feuchte, 1)}

    for name in messung:
        print(name, messung[name])

    if messung["temperatur"] > grenze:
        print("Warnung: zu warm!")

    time.sleep(wartezeit)
```

Alles, was einmal passieren soll, steht **vor** der Schleife: `import`, `i2c`, `sensor_bme` und die beiden Einstellungen. Was sich wiederholt, steht eingerückt darin.

Die Warnung steht *in* der Schleife, aber *neben* der Ausgabe: Sie hängt am `if`, nicht am `for`. Wer sie versehentlich eine Stufe weiter einrückt, bekommt sie dreimal.

Andere Grenzen sind genauso richtig. Wichtig ist nur, dass sich die Warnung auslösen lässt. Bei `grenze = 40` passiert im Raum nie etwas.

Ausbaustufen

Such dir aus, was dich reizt. Jede geht für sich.

- Warnung mit Licht.** Statt einer Zeile auf dem Bildschirm leuchtet die LED von Tag 1: rot bei Überschreitung, sonst grün.
- Drei Fälle statt zwei.** Zu warm, zu kalt, in Ordnung. Dafür brauchst du `elif` und eine zweite Grenze.
- Ein Ende einbauen.** Nach zehn Messungen aufhören. Zähl mit, die Bedingung hinter `while` muss irgendwann falsch werden.
- Die Messreihe auswerten.** Häng jede Temperatur an eine Liste an und gib am Ende Mittelwert, Maximum und Minimum aus. Die Funktionen dafür stehen in Aufgabe 1.
- Sagen, woher der Wert kommt.** Nimm zwei Schlüssel dazu, etwa `"ort"` und `"nummer"`. An Tag 4 muss zu jedem verschickten Wert dabeistehen, was er ist und woher er kommt.

Lösung

Nur die entscheidende Stelle, der Rest bleibt wie in der Grundversion.

```
# 2 - drei Faelle
if messung["temperatur"] > obere_grenze:
    print("Warnung: zu warm!")
elif messung["temperatur"] < untere_grenze:
    print("Warnung: zu kalt!")
else:
    print("Alles in Ordnung.")

# 3 - nach zehn Messungen Schluss
nummer = 0
while nummer < 10:
    nummer = nummer + 1
    ...

# 4 - Messreihe sammeln
reihe = []
reihe.append(messung["temperatur"])
# ... und nach der Schleife:
print("Im Mittel:", round(sum(reihe) / len(reihe), 1), "Grad")
print("Wärmster Wert:", max(reihe), "Grad")

# 5 - woher der Wert kommt
messung["ort"] = "Raum 101"
```

Zu 3: Das Hochzählen muss *in* der Schleife stehen. Fehlt es, läuft sie ewig. Das ist der häufigste Fehler von gestern.

Zu 4: `reihe = []` gehört vor die Schleife, sonst ist die Liste bei jedem Durchlauf wieder leer. Die Auswertung gehört dahinter und braucht deshalb Ausbaustufe 3, sonst kommt das Programm nie dort an.

Zu 5: Ein Dictionary lässt sich nachträglich erweitern. Ein Schlüssel, den es noch nicht gibt, wird beim Zuweisen angelegt.

S033

S034