

Name: _____

Klasse: _____

Datum: _____

Aufgabe 1

Wetterstation

In dieser Aufgabe wird zunächst die Sensorstation aufgebaut. Anschließend machen wir aus ihr eine Wetterstation, die ihre Messdaten via MQTT-Protokoll (Message Queuing Telemetry Transport) an einen Webserver überträgt. Sie nimmt eine Wettervorhersage vor und wird durch die regelmäßige Übertragung eines Kamerabilds zur Webcam.

Konstruktionsaufgabe

Baue die IoT-Station nach der Bauanleitung auf. Verbinde Sensoren und Aktoren nach dem Schaltplan mit dem TXT 4.0.

In den Experimentieraufgaben werden die Daten über das MQTT-Protokoll an einen Webserver übertragen, auf dem die übermittelten Werte visualisiert werden. Lege dir dafür einen Account in der fischertechnik-Cloud an (www.fischertechnik-cloud.com).

Programmieraufgaben

1. Messung von Luftfeuchtigkeit, Temperatur und Luftdruck

Der Umweltsensor, ein Bosch-Sensor mit der Typbezeichnung BME680, enthält (unter anderem) einen Temperatur-, einen Feuchtigkeits- und einen Drucksensor. Die Werte des Sensors werden über das I²C-Protokoll an den TXT übertragen. Abfragen kannst du die Sensorwerte über die entsprechenden Blockly-Befehle.

Schreibe ein Blockly-Programm, das die IoT-Station zu einer Wetterstation macht, die die gemessene Luftfeuchtigkeit (in %), die Temperatur (in °C) und den Luftdruck (in hPa) auf dem Display des TXT ausgibt und jede Sekunde aktualisiert.

2. Barometer

Mit dem gemessenen Luftdruck kannst du die Wetterstation um eine Wettervorhersage erweitern: ein sinkender Luftdruck kündigt Niederschlag an, ein steigender Luftdruck trockenes, sonniges Wetter.

Auf Meereshöhe liegt der Luftdruck zwischen etwa 950 und 1050 hPa, er schwankt ca. ±50 hPa um einen mittleren Wert von etwa 1000 hPa. Wenn du den gesamten

Messbereich in drei Teilbereiche unterteilt, kannst du aus dem gemessenen Luftdruck eine Wettervorhersage ableiten („regnerisch“, „wechselhaft“, „schön“).

Allerdings sinkt der Luftdruck mit zunehmender Höhe. Die von dem Sensor gemessenen Werte sind daher je nach Höhe unterschiedlich zu interpretieren. Das kannst du berücksichtigen, indem du den gemessenen Luftdruck $p(h)$ nach der „internationalen Höhenformel“ in einen „theoretischen“ Luftdruck p_0 auf Meereshöhe umrechnest [1]:

$$p_0 = p(h) \cdot \left(\frac{T_0}{T(h)} \right)^{5,255} \text{ hPa}$$

Dabei sind

- $p(h)$ der in der Höhe h gemessene Luftdruck,
- $T(h)$ die in dieser Höhe gemessene Temperatur (in Kelvin) und
- T_0 die (theoretische) Temperatur auf Meereshöhe, die je Höhenmeter näherungsweise 0,0065 K sinkt:

$$T_0 = T(h) + 0,0065 \cdot h \frac{\text{K}}{\text{m}}$$

Bestimme die Höhe h deiner Wetterstation in Meter über NN und berechne daraus, aus dem gemessenen Luftdruck $p(h)$ und der Temperatur $T(h)$ in deinem Blockly-Programm den „Luftdruck auf Meereshöhe“.

Erweitere anschließend deine Wetterstation um eine Wettervorhersage, die auf dem Display des TXT angezeigt wird.

Experimentieraufgaben

1. Temperaturbestimmung mit NTC-Widerstand

In Aufgabe 1 des Base Set wurde aus dem NTC-Widerstand (Heißleiter) mit Hilfe der Steinhart-Hart-Gleichung (siehe Begleitmaterial) die Temperatur bestimmt. Dazu mussten mit dem Heißleiter zunächst drei Widerstandswerte bei verschiedenen Temperaturen bestimmt werden. Die liefert dir jetzt der Umweltsensor mit hoher Genauigkeit.

Schließe den NTC-Widerstand an I8 an und erweitere dein Blockly-Programm um eine Messung des NTC-Widerstands. Trage die Ergebnisse dreier Messungen in die folgende Tabelle ein:

Widerstandswert	Temperatur

Hinweis: Der Umweltsensor reagiert wegen des Gehäuses sehr verzögert auf Temperaturänderungen. Die Messung sollte daher erst beendet werden, wenn sich der Temperaturwert des Sensors stabilisiert hat.

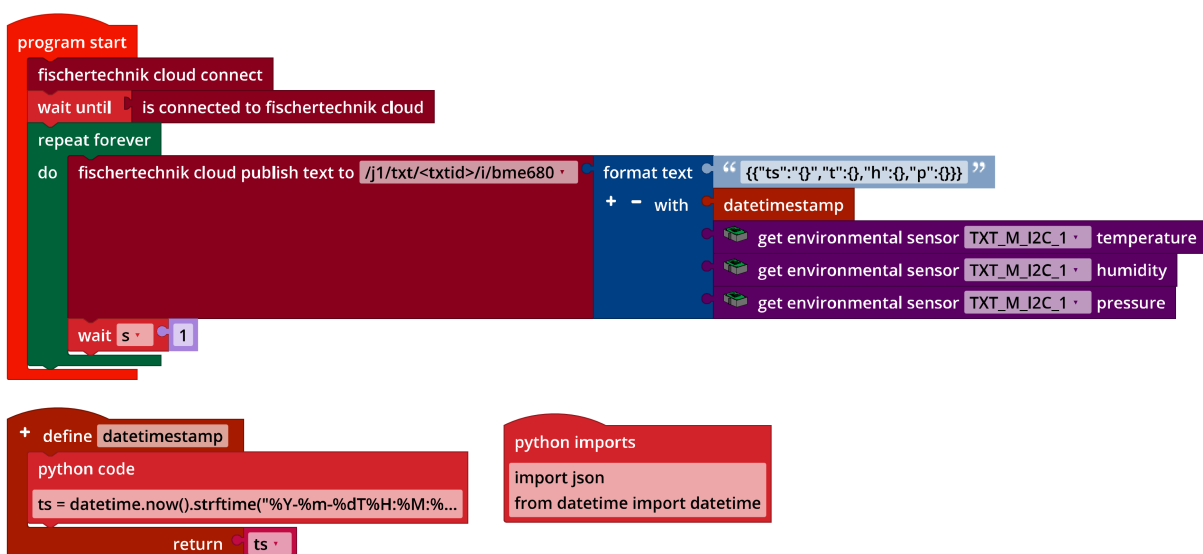
Die Koeffizienten der Steinhart-Hart-Gleichung kannst du nun über die im Begleitmaterial und im Anhang [4] angegebene Webseite bestimmen und die aus dem NTC-Widerstand berechnete Temperatur ebenfalls auf dem Display des TXT ausgeben.

2. Datenvisualisierung auf einem IoT-Server

Nun sollen die Messdaten deiner Wetterstation an einen Webserver übertragen und dort in einem (vorbereiteten) Dashboard angezeigt werden. Verbinde dazu deinen TXT mit deinem zuvor eingerichteten Account in der fischertechnik-Cloud.

2a. Konfiguriere zunächst das Dashboard so, dass Temperatur, Luftfeuchtigkeit und Luftdruck angezeigt werden. Klicke alle nicht benötigten Anzeigen weg.

2b. Das folgende Beispielprogramm demonstriert dir, wie die Daten des Sensors an den IoT-Server übermittelt werden: Zuerst öffnest du die Verbindung mit deinem Cloud-Account („MQTT connect“); dann werden in regelmäßigen Abständen die aktuellen Messwerte des Sensors in einem festen Format (Zeitstempel, Temperatur, Feuchtigkeit, Druck) an den Cloud-Server übertragen („MQTT publish“).



```

+ define timestamp
python code
ts = datetime.now().strftime("%Y-%m-%dT%H:%M:%S...")
return ts

python imports
import json
from datetime import datetime

program start
fischertechnik cloud connect
wait until is connected to fischertechnik cloud
repeat forever
do fischertechnik cloud publish text to /j1/txt/<txtid>/i/bme680
+ - with format text
    {"ts":0,"t":0,"h":0,"p":0}
    + - with timestamp
    + - with get environmental sensor TXT_M_I2C_1 temperature
    + - with get environmental sensor TXT_M_I2C_1 humidity
    + - with get environmental sensor TXT_M_I2C_1 pressure
wait 1s
    
```

IoT_MQTT.ft

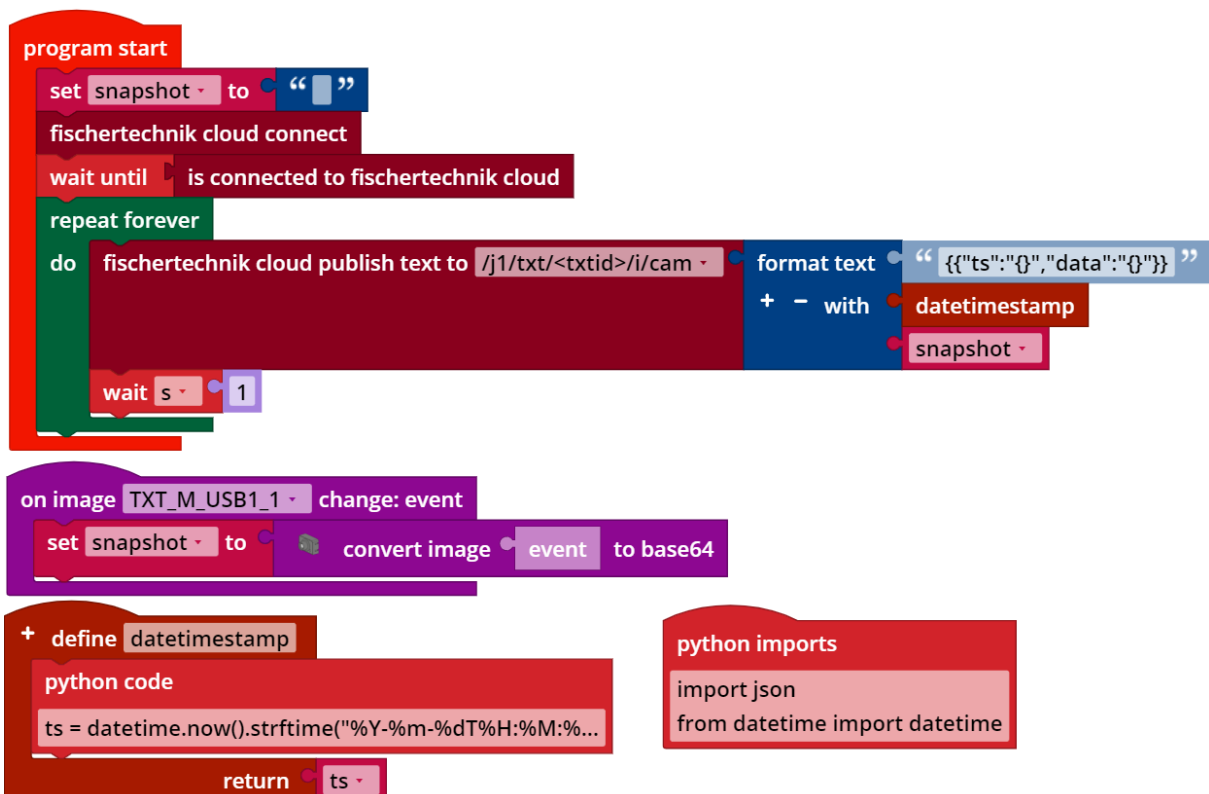
Ergänze dein Blockly-Programm des Barometers um die Anzeige auf dem IoT-Server. Jetzt kannst du Messreihen erzeugen, dir die Messwerte anzeigen lassen und sie als csv-Datei aus dem Dashboard herunterladen. Die csv-Datei kannst du in einer Tabellenkalkulation (bspw. Excel) auswerten und weiterbearbeiten.

3. Webcam

Was wäre eine Wetterstation ohne eine Webcam? Erweitere dein Programm um eine Übertragung des Kamerabilds an das Dashboard des IoT-Servers.

3a. Ergänze zunächst das Dashboard um eine Anzeige des Kamerabilds.

3b. Das folgende Beispielprogramm zeigt dir, wie die Übermittlung des Kamerabilds funktioniert: Jedes Bild, das die Kamera „einfängt“, wird Base64-kodiert. Einmal pro Sekunde wird dann das aktuelle Bild an den IoT-Server übermittelt („MQTT publish“).



The image shows a Scratch script for an IoT webcam. It starts with a 'program start' block, followed by 'set snapshot to ""', 'fischertechnik cloud connect', and 'wait until is connected to fischertechnik cloud'. A 'repeat forever' loop contains 'fischertechnik cloud publish text to /j1/txt/<txid>/i/cam', 'format text' with a JSON template, 'with' block containing 'datetimestamp' and 'snapshot', and 'wait s 1'. An 'on image TXT_M_USB1_1 change: event' block triggers 'set snapshot to' and 'convert image event to base64'. A 'define datetimestamp' block uses Python code to return a timestamp. A 'python imports' block imports 'json' and 'datetime'.

```

program start
set snapshot to ""
fischertechnik cloud connect
wait until is connected to fischertechnik cloud
repeat forever
do fischertechnik cloud publish text to /j1/txt/<txid>/i/cam
format text "" {"ts": "0", "data": "0"} ""
+ - with datetimestamp
snapshot
wait s 1

on image TXT_M_USB1_1 change: event
set snapshot to
convert image event to base64

+ define datetimestamp
python code
ts = datetime.now().strftime("%Y-%m-%dT%H:%M:%S...")
return ts

python imports
import json
from datetime import datetime
    
```

IoT_Webcam.ft

Ergänze in deinem Programm eine Übertragung des Kamerabildes an den IoT-Server. Im Dashboard kannst du Schnappschüsse in einer Galerie speichern und die Bilder einzeln herunterladen.

Anlagen

Aufgabe 1: Wetterstation

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- Beispielprogramme „IoT_MQTT.ft“ und „IoT_Webcam.ft“
- Account in der fischertechnik-cloud

Weiterführende Informationen

- [1] Wikipedia: [Internationale Höhenformel](#).
- [2] Online-Diagrammeditor zur Erstellung von Zustandsübergangsdigrammen (Format drawio): <https://www.diagrammeditor.de/>
- [3] fischertechnik: [NTC-Widerstand](#). Datenblatt, Art.-Nr. 36437.
- [4] Stanford Research Systems (SRS): [Thermistor Calculator](#). V1.1
- [5] Dirk Fox: [„Einmessen“ eines digitalen Messgeräts](#). ft:pedia 1/2013, S. 39-48.

Aufgabe 1: Wetterstation

Zunächst wird eine Sensorstation aufgebaut, die auch in allen folgenden Aufgaben als Modell dient. Dann werden mit dem Umweltsensor (einem Bosch Sensortec BME680) unterschiedliche Wetterdaten (Luftfeuchte, Temperatur und Luftdruck) erhoben, daraus eine Wettervorhersage berechnet und die Daten zur Visualisierung in einem Dashboard auf einen IoT-Server in der fischertechnik-Cloud übermittelt.

Themen

- Auswertung von Sensordaten zur Wetteranzeige
- Wettervorhersage über Luftdruckmessung und barometrische Höhenformel
- Visualisierung der Sensordaten auf einem Webserver
- Erweiterung der Wetterstation um eine Webcam

Lernziele

- Messwerterfassung, sinnvolle Visualisierung und Auswertung
- Nutzung der Barometrischen Höhenformel zur Erstellung einer Wetterprognose
- NTC-Widerstand und Temperaturberechnung (Steinhart-Hart-Gleichung)

Zeitaufwand

Der Aufbau der Sensorstation nach der Bauanleitung erfordert – etwas Erfahrung mit fischertechnik vorausgesetzt – etwa 60-90 Minuten.

Für die Lösung der Programmieraufgaben benötigen die Schülerinnen und Schüler ca. 45 bis 60 Minuten (mit Programmiererfahrung). Die Experimentieraufgaben bauen auf der Aufgabe 1 des Base Set und den Programmieraufgaben auf; ihre Bearbeitung sollte bei Verwendung der Beispielpprogramme maximal 60 Minuten beanspruchen. Die Bestimmung der für die Lösung von Experimentieraufgabe 1 erforderlichen Messwerte kann wegen der langsamen Anpassung des Sensors 30 Minuten Wartezeit erfordern.

Bezug Curriculum

Land	Stufe/Fächer	Bezüge
BW		
BY		
BE		
BB		
HB		
HH		
HE		
MV		
NI		
NW		
RP		
SL		
SN		
ST		
SH		
TH		

Anlagen

Aufgabe 1: Wetterstation

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- Beispielprogramme „IoT_MQTT.ft“ und „IoT_Webcam.ft“
- Account in der fischertechnik-cloud

Weiterführende Informationen

- [1] Wikipedia: [Internationale Höhenformel](#).
- [2] Online-Diagrammeditor zur Erstellung von Zustandsübergangsdigrammen (Format drawio): <https://www.diagrammeditor.de/>
- [3] fischertechnik: [NTC-Widerstand](#). Datenblatt, Art.-Nr. 36437.
- [4] Stanford Research Systems (SRS): [Thermistor Calculator](#). V1.1
- [5] Dirk Fox: [„Einmessen“ eines digitalen Messgeräts](#). ft:pedia 1/2013, S. 39-48.

Name: _____

Klasse: _____

Datum: _____

Lösungsblatt Aufgabe 1

Wetterstation

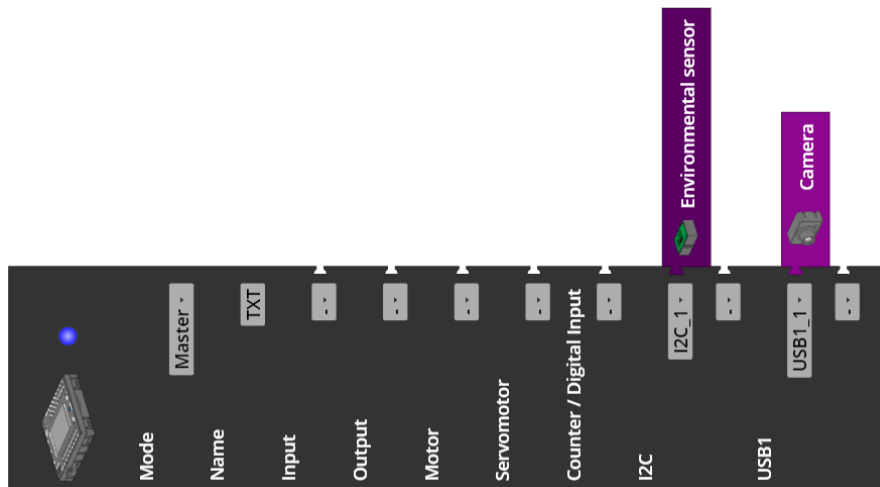
Aufgaben 1 und 2 machen die Sensorstation zu einer lokalen Wetterstation. In den Experimentieraufgaben wird die Wetterstation via MQTT mit einem Webserver verbunden, der die Messdaten und übertragenen Bilder in einem Dashboard anzeigt.

Konstruktionsaufgabe

Siehe Bauanleitung.

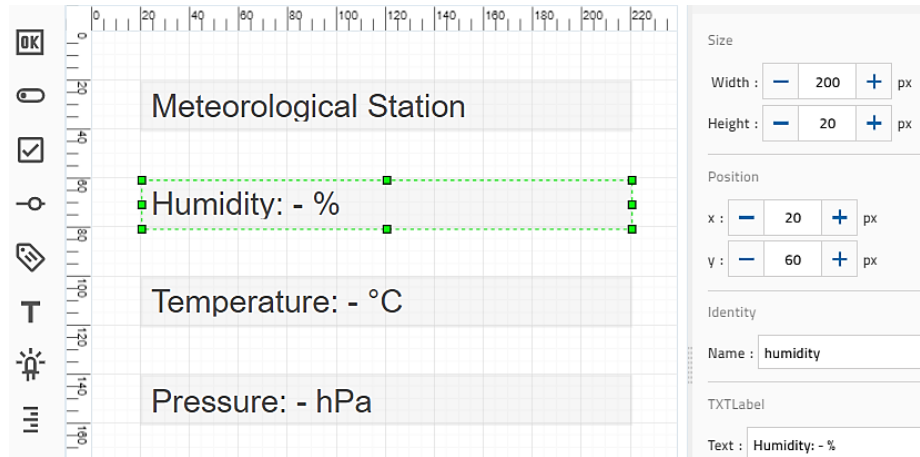
Programmieraufgaben

Konfiguration der Sensoren und Aktoren:



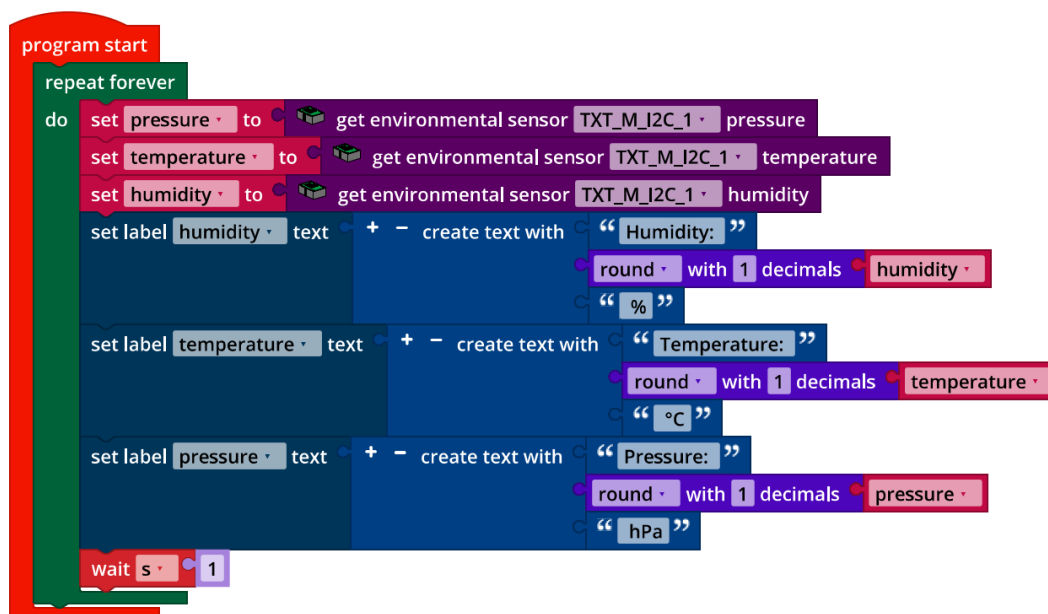
1. Messung von Luftfeuchtigkeit, Temperatur und Luftdruck

1a. Display-Konfiguration (Beispiel):



Konfiguration der Display-Ausgabe

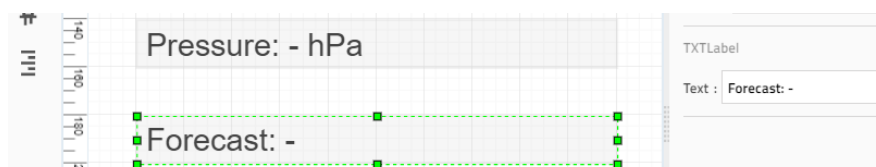
1b. Programm (Beispiel):



IoT_Meteorological_Station.ft

2. Barometer

2a. Erweiterung der Display-Konfiguration (Beispiel):



Erweiterung der Display-Ausgabe

2b: Programm (Beispiel):

Die Temperaturwerte müssen vor der Berechnung in Kelvin umgerechnet werden: $0^{\circ}\text{C} = 273,15\text{ K}$.

```

program start
  set h to 125
  repeat forever
    do
      set pressure to get environmental sensor TXT_M_I2C_1 pressure
      set temperature to get environmental sensor TXT_M_I2C_1 temperature
      set humidity to get environmental sensor TXT_M_I2C_1 humidity
      set label humidity text + - create text with " Humidity: "
      round with 1 decimals humidity
      "%"
      set label temperature text + - create text with " Temperature: "
      round with 1 decimals temperature
      " °C "
      set p0 to
        temperature + 273.15
        + 0.0065 x h
        ÷ 5.255
        x pressure
      set label pressure text + - create text with " Pressure: "
      round with 1 decimals pressure
      " hPa ( "
      round with 1 decimals p0
      " hPa to NN) "
      + if p0 < 980
      do set label forecast text " Forecast: Rain "
      else if p0 > 1020
      do set label forecast text " Forecast: Fair "
      else set label forecast text " Forecast: Change "
      wait s 1
  
```

IoT_Barometer.ft

Experimentieraufgaben

1. Temperaturbestimmung mit NTC-Widerstand

1a. Die Widerstandswerte des Heißleiters und die zugehörigen Temperatur-Messwerte des Umweltsensors lassen sich leicht durch eine Log-Ausgabe auf der Konsole bestimmen (siehe Programm-Beispiel unten).

Mit drei Messwerten lassen sich die Koeffizienten a, b und c der Steinhart-Hart-Gleichung bestimmen [4, 5] (T: Temperatur in Kelvin, R: NTC-Widerstand in Ohm):

$$\frac{1}{T} = a + b \cdot \ln(R) + c \cdot \ln(R)^3$$

Widerstandswert	Temperatur
1500 Ω	25,0 °C
2362 Ω	13,7 °C
2530 Ω	11,8 °C

Beispielmessung

Die Messwerte hängen vom NTC-Widerstand ab und können abweichen.

Koeffizienten: a = 0.004535418128, b = -0.0003767491105, c = 0.000004023802790

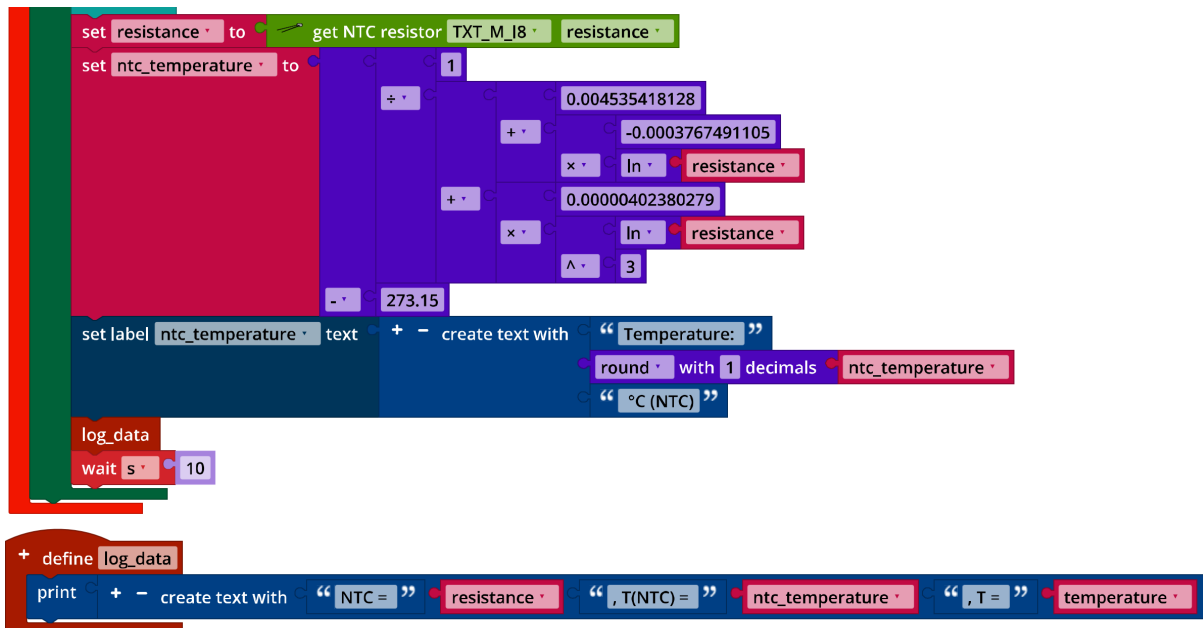
1b. Erweiterung der Display-Anzeige:

Konfiguration der Display-Ausgabe der NTC-Temperaturmessung

1c. Programmerweiterung (Beispiel):

Berechnung des Temperaturwerts aus dem NTC-Widerstand und Ausgabe der Messwerte des NTC-Widerstands und des Temperatursensors auf der Konsole.

Auch hier muss der aus dem Heißeleiter berechnete Temperaturwert abschließend in Kelvin umgerechnet werden: $0^{\circ}\text{C} = 273,15\text{ K}$.



```

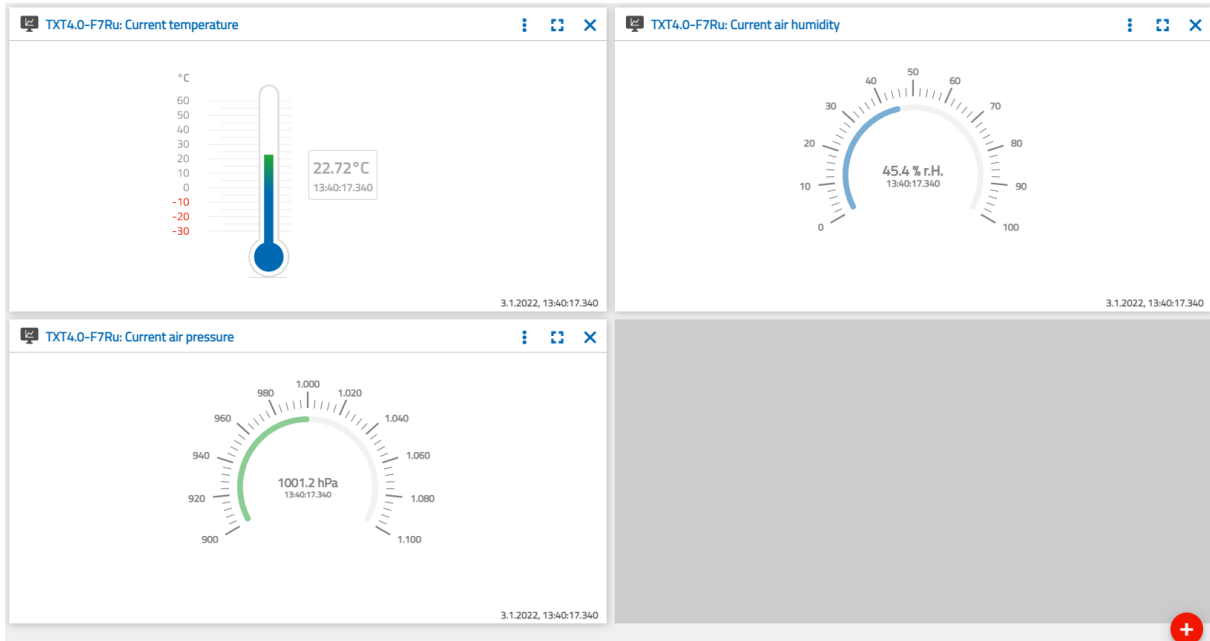
set resistance to get NTC resistor TXT_M_I8 resistance
set ntc_temperature to 1
÷ 0.004535418128
+ -0.0003767491105
× ln resistance
+ 0.00000402380279
× ln resistance
^ 3
- 273.15
set label ntc_temperature text + - create text with "Temperature:"
round with 1 decimals ntc_temperature
"C (NTC)"
log_data
wait s 10

+ define log_data
print + - create text with "NTC =" resistance ", T(NTC) =" ntc_temperature ", T =" temperature
    
```

IoT_Barometer_with_NTC_resistor.ft

2. Datenvisualisierung auf einem IoT-Server

2a. Konfiguration des Dashboards auf dem IoT-Server in der fischertechnik-Cloud:



Konfiguration der Anzeige im Dashboard

2b. Programmiererweiterung (Beispiel):

```

fischertechnik cloud publish text to /j1/txt/<txtid>/i/bme680
+ - with format text [{"ts":0,"t":0,"h":0,"p":0}]
+ - with timestamp
+ - with get environmental sensor TXT_M_I2C_1 temperature
+ - with get environmental sensor TXT_M_I2C_1 humidity
+ - with get environmental sensor TXT_M_I2C_1 pressure

wait s 1

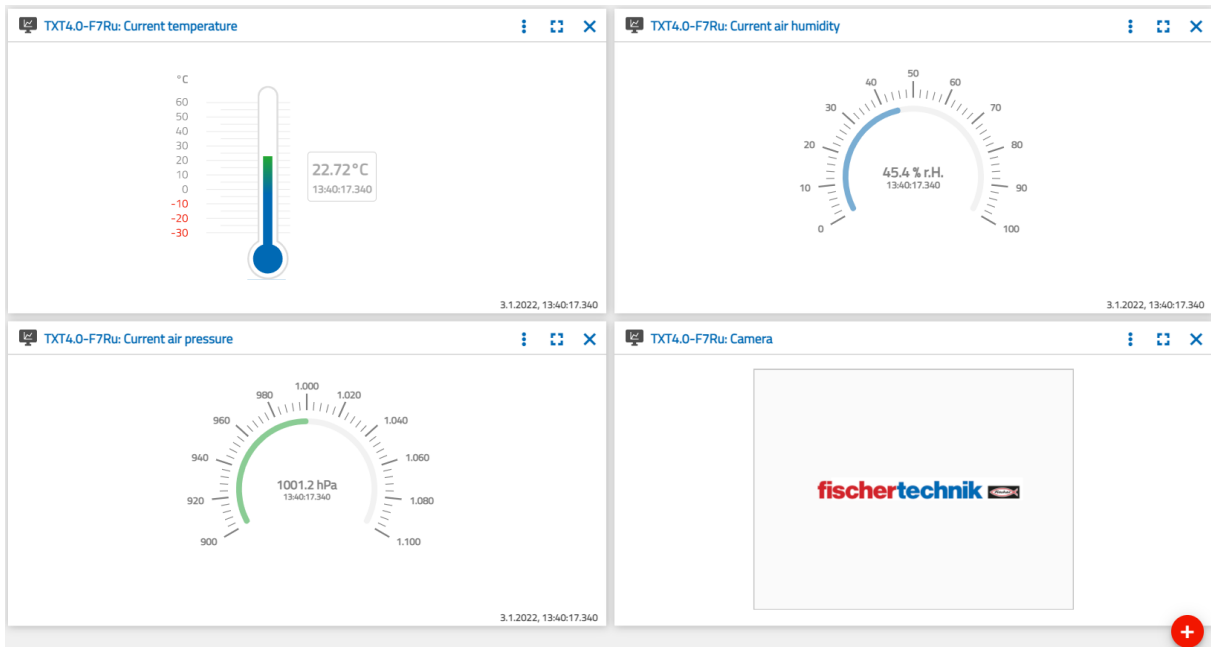
+ define timestamp
python code
ts = datetime.now().strftime("%Y-%m-%dT%H:%M:%S...")
return ts

python imports
import json
from datetime import datetime
    
```

IoT_MQTT_Barometer.ft

3. Webcam

3a. Erweiterung des Dashboards in der fischertechnik-Cloud:



Dashboard mit Webcam

3b. Programmerweiterung (Beispiel):

```

fischertechnik cloud publish text to /j1/txt/<txtid>/i/cam
format text “{{ts:"0", "data": "0"}}”
+ - with
snapshot
wait s 1

+ define timestamp
python code
ts = datetime.now().strftime("%Y-%m-%dT%H:%M:%S...")
return ts

python imports
import json
from datetime import datetime

on image TXT_M_USB1_1 change: event
set snapshot to
convert image event to base64
    
```

IoT_MQTT_Barometer_with_Webcam.ft

Anlagen

Aufgabe 1: Wetterstation

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- Beispielprogramme „IoT_MQTT.ft“ und „IoT_Webcam.ft“
- Account in der fischertechnik-cloud

Weiterführende Informationen

- [1] Wikipedia: [Internationale Höhenformel](#).
- [2] Online-Diagrammeditor zur Erstellung von Zustandsübergangsdigrammen (Format drawio): <https://www.diagrammeditor.de/>
- [3] fischertechnik: [NTC-Widerstand](#). Datenblatt, Art.-Nr. 36437.
- [4] Stanford Research Systems (SRS): [Thermistor Calculator](#). V1.1
- [5] Dirk Fox: [„Einmessen“ eines digitalen Messgeräts](#). ft:pedia 1/2013, S. 39-48.

Name: _____

Klasse: _____

Datum: _____

Aufgabe 2

Raumklima

In dieser Aufgabe überwacht unsere Sensorstation das Raumklima und warnt uns, wenn es zu heiß, zu kalt, zu feucht, zu dunkel oder zu laut wird – oder die Luftqualität (z. B. durch zu viel CO₂) sinkt.

Konstruktionsaufgabe

Die in Aufgabe 1 konstruierte Sensorstation kann unverändert genutzt werden.

Programmieraufgaben

1. Messung der Raumklima-Werte

Mit den in Aufgabe 1 erhobenen Messwerten „Luftfeuchtigkeit“ und „Temperatur“ kannst du nicht nur das Wetter, sondern auch das Raumklima bestimmen.

Luftfeuchtigkeit und Temperatur sind wesentliche Faktoren eines guten Raumklimas. Die (absolute) Luftfeuchtigkeit gibt die Menge an Wasser (in g) an, die sich als Wasserdampf in einem Kubikmeter (m³) Luft befindet.

Der Sensor misst die relative Luftfeuchte, also den Quotienten aus (absoluter) Luftfeuchtigkeit und maximal möglicher Luftfeuchte in %. Die maximale Luftfeuchte ändert sich mit der Temperatur: Luft kann bei 0°C maximal 5g Wasserdampf aufnehmen, 30°C warme Luft hingegen 30g – kalte Luft ist daher „trockener“ als warme.

Als optimales Raumklima für Wohn- und Arbeitsräume gilt eine relative Luftfeuchte von 45-55% bei einer Temperatur um 20°C.

1a. Passe die Display-Konfiguration der Wetterstation aus Aufgabe 1 für die Ausgabe von relativer Luftfeuchte und Temperatur an.

1b. Schreibe das Blockly-Programm der Wetterstation für die Raumklima-Station um.

Hinweis: Der Sensor benötigt nach dem Programmstart etwa 10 Sekunden, bis sich die Messwerte stabilisieren [1].

2. Messung der Luftqualität

Der Umwelt-Sensor kann verschiedene Gase in der Umgebungsluft detektieren, wie z. B. Formaldehyd, Alkohol, Lösungsmittel sowie Ausdünstungen von Lacken, Beize, Reinigungsmitteln oder Klebstoffen. Daraus bestimmt er einen Index für die Luftqualität (Index for Air Quality, IAQ) in einem Bereich von 0-500.

Ein IAQ-Wert von unter 50 signalisiert eine gute Raumluftqualität, ein Wert von über 200 weist auf eine deutlich verunreinigte Luft hin (siehe Tabelle).

IAQ Index	Air Quality	Impact (long-term exposure)	Suggested action
0 – 50	Excellent	Pure air; best for well-being	No measures needed
51 – 100	Good	No irritation or impact on well-being	No measures needed
101 – 150	Lightly polluted	Reduction of well-being possible	Ventilation suggested
151 – 200	Moderately polluted	More significant irritation possible	Increase ventilation with clean air
201 – 250 ⁹	Heavily polluted	Exposition might lead to effects like headache depending on type of VOCs	optimize ventilation
251 – 350	Severely polluted	More severe health issue possible if harmful VOC present	Contamination should be identified if level is reached even w/o presence of people; maximize ventilation & reduce attendance
> 351	Extremely polluted	Headaches, additional neurotoxic effects possible	Contamination needs to be identified; avoid presence in room and maximize ventilation

Index of Air Quality [2]

2a. Passe die Display-Konfiguration der Raumklima-Station für die Ausgabe der Luftqualität an.

2b. Erweitere das Blockly-Programm der Raumklima-Station entsprechend.

Hinweis: Beim Starten des Programms wird der Luftqualität-Sensor automatisch kalibriert. Die Kalibrierung dauert fünf Minuten; der Fortschritt wird auf der Konsole angezeigt. Solange wird als IAQ-Wert -1 zurückgeliefert.

Experimentieraufgaben

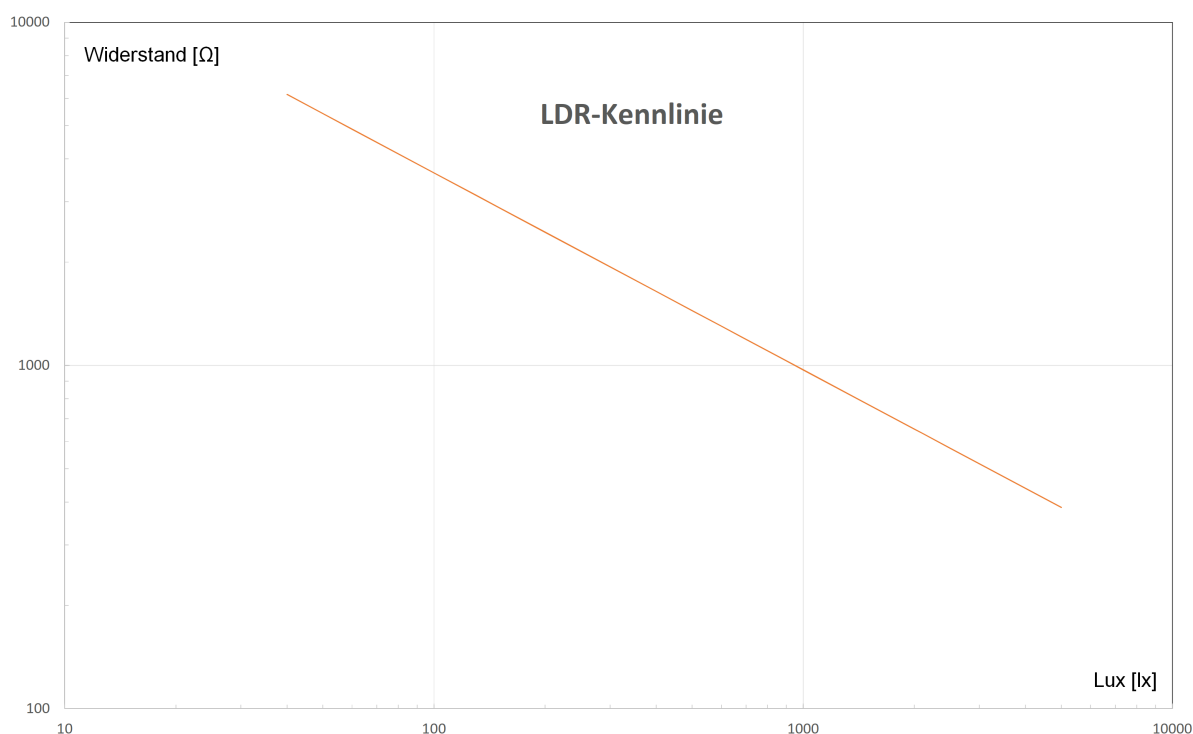
1. Helligkeit

Im Unterschied zum Fototransistor, der in Aufgabe 2 des Robotics Base Set als Sensor in einer Lichtschranke eingesetzt wurde, ist ein Fotowiderstand (LDR – *Light Dependent Resistor*) kein digitaler, sondern ein analoger Sensor. Mit ihm können wir die Umgebungshelligkeit bestimmen.

Lux (lx) ist die Einheit für die Beleuchtungsstärke – den Lichtstrom (gemessen in Lumen), der auf einer bestimmten Fläche auftrifft. Es gilt: $1 \text{ Lux} = 1 \text{ Lumen/m}^2$. Ein Lux entspricht etwa der Beleuchtungsstärke, die in einem Meter Abstand von einer Kerzenflamme gemessen werden kann.

Zum Vergleich: An einem wolkenlosen Sommertag kann die Beleuchtungsstärke bei direkter Sonneneinstrahlung 100.000 Lux erreichen, an einem bewölkten Wintertag kommt sie kaum über 2.000 Lux.

Der fischertechnik-Fotowiderstand liefert Widerstandswerte bis über 60 kOhm; je heller, desto kleiner der Widerstand. Dabei entspricht ein Widerstandswert von 1 kOhm etwa 1000 Lux (so genannter „Hellwiderstand“). Die Kennlinie des Widerstands gibt das Verhältnis von Widerstandswert zu Beleuchtungsstärke an. Sie verläuft logarithmisch (beachte die Skalen):



1a. Nimm nun mithilfe eines Luxmeters (bspw. einer App) mehrere Messungen vor und trage Widerstandswert und gemessenen Lux-Wert in eine Tabelle ein. Mit Hilfe eines Tabellenkalkulationsprogramms erhältst du zu deinen Messwerten eine einfache Formel der Gestalt $y = a \cdot x^b$, mit der du die Beleuchtungsstärke in Lux näherungsweise aus dem Widerstandswert berechnen kannst.

1b. Erweitere das Blockly-Programm aus Programmieraufgabe 2 um eine Anzeige der Beleuchtungsstärke in Lux.

1c. Lege einen geeigneten Schwellenwert für die minimale Beleuchtungsstärke im Klassenraum fest. Was schreibt z. B. die Arbeitsstättenverordnung [4] vor? Schließe die zweite LED an O7 und O8 an (auf korrekte Polung achten). Sie soll von deinem Blockly-Programm eingeschaltet werden, wenn es im Raum zu dunkel ist.

2. CO₂-Ampel

Der Umweltsensor kann den CO₂-Anteil der Luft nicht direkt bestimmen, wohl aber die von Menschen ausgeatmete Luft anhand von darin enthaltenen sonstigen Gasen detektieren. Somit steht die vom Sensor gemessene Luftqualität in einem unmittelbaren Zusammenhang zum CO₂-Gehalt der Luft, der durch Ausatmen entsteht.

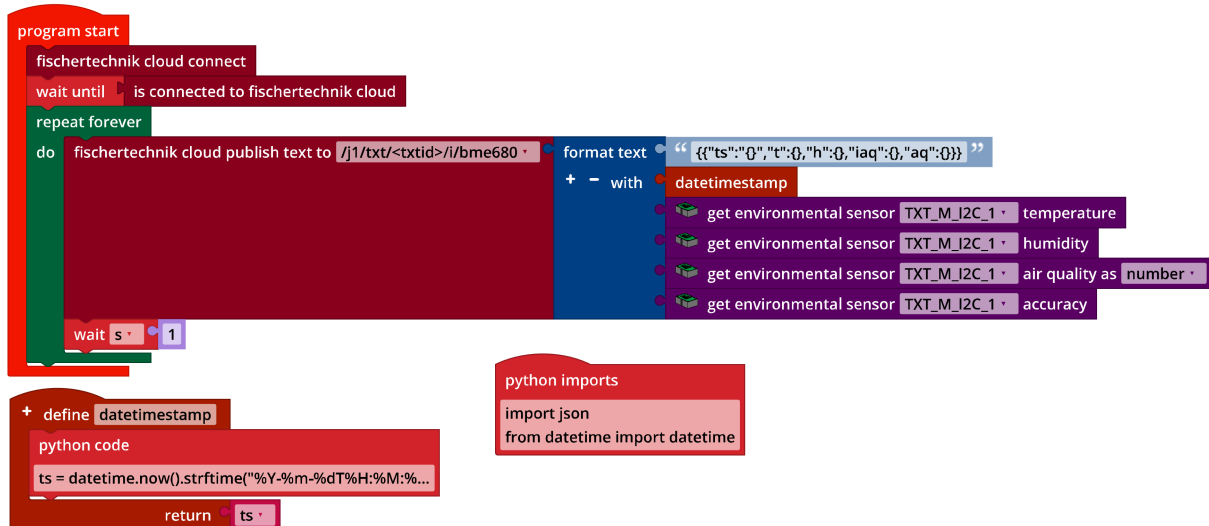
Der CO₂-Anteil der Luft liegt bei rund 400 ppm (*parts per million*). Da Menschen mit der Atemluft (8-10 Liter pro Minute) 40.000 ppm (= 4%) CO₂ ausstoßen, steigt der CO₂-Anteil der Luft in einem geschlossenen Raum mit mehreren Menschen kontinuierlich. Erreicht er 1.200 ppm, wurden 2% der Luft in einem Raum bereits einmal ausgeatmet – jeder 50ste Atemzug besteht damit aus ausgeatmeter Luft. Ein CO₂-Anteil von 1.000-2.000 ppm gilt daher nach Einschätzung des Umweltbundesamtes als bedenklich; insbesondere in Schulräumen sollten 1.000 ppm im Mittel nicht überschritten werden [5].

Mit der Atemluft werden auch kleine Tröpfchen (Aerosole) ausgestoßen, an denen Viren haften können, die sich lange in der Luft halten (Sinkgeschwindigkeit von 10 cm/h bei einer Halbwertszeit der Infektionsaktivität der Viren von 2,7 Stunden). Somit ist auch eine steigende Luftfeuchte ein Risikoindikator, der auf die Möglichkeit einer Verbreitung von Viren im Raum hinweist.

Durch kurzes Stoßlüften („Durchzug“) können CO₂-Anteil und Luftfeuchte schnell und effektiv reduziert werden. Zum Schutz vor Ansteckung wird für Klassenräume ein Lüften alle 20 Minuten empfohlen [6].

2a. Lege zunächst für die Luftfeuchte und die Luftqualität Schwellenwerte fest, ab denen ein Stoßlüften des Klassenzimmers erfolgen sollte. Dazu kannst du beispielsweise die Messwerte im Klassenraum nach 20 Minuten bestimmen. Zielwerte sind die Luftfeuchte und Luftqualität, die du durch Lüften erreichen kannst.

Den Verlauf der Messwerte der relativen Luftfeuchte und der Luftqualität kannst du dir mit dem folgenden Hilfsprogramm im Dashboard des IoT-Servers anzeigen lassen:



IoT_MQTT_Indoor_Air_Quality.ft

Leite aus deinen Schwellenwerten einen „Lüftungsindex“ (0-100 mit: 0 = Idealzustand nach Lüften, 100 = Lüften dringend erforderlich) ab, in dessen Berechnung die relative Luftfeuchte und die Luftqualität mit einer Gewichtung von 1:3 eingehen.

2b. Programmiere ein Unterprogramm, das die rote LED im Rhythmus von 0,5 Sekunden blinken lässt. Bei einem Lüftungsindex von 100 soll das rote Blinklicht aktiviert werden.

2c. Schließe die zweite LED (mit grüner Kappe) an O7 und O8 an (Polarität beachten) und aktiviere sie, wenn dein Lüftungsindex „0“ erreicht ist. Vergleiche die Lüftungsdauer, wenn nur ein Fenster, zwei Fenster, ..., alle Fenster und die Türe(n) geöffnet werden.

3. Lautstärke

Auch die Lautstärke in unserer Umgebung hat Einfluss auf unser Wohlbefinden. Lautstärke wird als Schalldruckpegel gemessen und in der Einheit Dezibel (dB) angegeben. Dabei entsprechen 0 dB der menschlichen Hörschwelle (Druckschwankungen um 20 Mikropascal); 120 dB sind bei einem Menschen üblicherweise die Schmerzgrenze.

Die (gefühlte) Lautstärke steigt dabei logarithmisch mit dem Dezibel-Wert: Sie verdoppelt sich etwa bei einer Zunahme des Schalldruckpegels um 10 dB. Der Schalldruckpegel hängt wiederum vom Abstand zur Geräuschquelle ab: Eine Verdreifachung

des Abstands bewirkt eine Halbierung der Lautstärke (Abnahme des Schalldruckpegels um 10 dB).

Um ein Gefühl für die Lautstärke bestimmter Dezibel-Werte zu gewinnen, helfen Vergleiche: Ein tickender Wecker (1 m Abstand) liegt bei etwa 30 dB, ein normales Gespräch bei etwa 60 dB, ein vorbeifahrender Bus bei 80 dB, ein Presslufthammer bei 100 dB und ein Düsenflugzeug bei 120 dB.

Die negativen Auswirkungen großer Lautstärken sind erheblich:

- Lern- und Konzentrationsstörungen treten bereits ab 40 dB auf
- Lautstärken ab 60 dB verursachen bei längerer Einwirkung Hörschäden
- Bei dauerhaften 65+ dB steigt das Risiko für Herz-Kreislauf-Erkrankungen
- Ab 120 dB sind Hörschäden schon nach kurzer Einwirkung möglich

3a. Erweitere die Anzeige deines Programms aus Programmieraufgabe 2 (bzw. Experimentieraufgabe 1) um eine Skalenanzeige für die Lautstärke (in dB).

3b. Das Mikrofon in der Kamera liefert dir die empfangene Lautstärke in dB. Erweitere dein Blockly-Programm um die Messung und Anzeige der Lautstärke.

3c. Lege einen Lautstärke-Schwellenwert fest, ab dem die rote LED blinken soll. Erweitere dein Programm entsprechend.

Anlagen

Aufgabe 2: Raumklima

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- Programm „IoT_MQTT_Indoor_Air_Quality.ft“

Weiterführende Informationen

- [1] Bosch Sensortec: *BME680 – Application Note*. Rev. 1.6, 20.09.2020.
- [2] Bosch Sensortec: *BME680 – Datasheet*. Rev. 1.7, 20.12.2021.
- [3] fischertechnik: [Photoresistor LDR03 \(32698\)](#). Datenblatt, 17.10.2018.
- [4] Bundesministerium für Arbeit und Soziales: [Beleuchtung](#). Technische Regeln für Arbeitsstätten, ASR A3.4, April 2011.
- [5] Umweltbundesamt: [Gesundheitliche Bewertung von Kohlendioxid in der Innenraumluft](#). Mitteilungen der Ad-hoc-Arbeitsgruppe Innenraumrichtwerte der Innenraumluftthygiene-Kommission des Umweltbundesamtes und der Obersten Landesgesundheitsbehörden, Bundesgesundheitsblatt – Gesundheitsforschung – Gesundheitsschutz 2008, 51, S. 1358–1369.
- [6] Deutsche Gesetzliche Unfallversicherung (DGUV): [Coronavirus SARS-CoV-2 – Ergänzende Empfehlungen der gesetzlichen Unfallversicherung für die Gefährdungsbeurteilung in Schulen](#). 03.12.2021.

Aufgabe 2: Raumklima

Die Sensorstation wird zur Analyse des Raumklimas eingesetzt und warnt bei der Über- oder Unterschreitung von vorgegebenen Grenzwerten. Die Messreihen können über den IoT-Server erhoben und anschließend ausgewertet werden.

Themen

- Messung von Raumklimafaktoren (relative Luftfeuchte, Temperatur, Luftqualität, Helligkeit, Lautstärke)
- Definition und Messung eines „Lüftungsindikators“ für Klassenräume („CO₂-Ampel“)
- Bestimmung der Beleuchtungsstärke

Lernziele

- Verständnis des Begriffs der relativen Luftfeuchtigkeit
- Verständnis für die Messung des Schallpegels (Dezibel)
- Verständnis des Begriffs der Beleuchtungsstärke (Lux) und deren Bestimmung aus den Werten eines Fotowiderstands
- Kenntnis typischer Grenzwerte für relative Luftfeuchtigkeit, Temperatur, Lautstärke und Beleuchtungsstärke
- Herleitung eines Index-Werts aus mehreren Messwerten nach vorgegebener Gewichtung

Zeitaufwand

Wurde die Sensorstation bereits in Aufgabe 1 aufgebaut, ist kein Zeitaufwand für die Konstruktion erforderlich.

Für die Lösung der Programmieraufgaben (aufbauend auf der Wetterstation aus Aufgabe 1) sollten 60 Minuten veranschlagt werden.

Die Lösung der Experimentieraufgaben 1 und 2 erfordert eine mathematische Modellbildung; hier ist ggf. eine Unterstützung der Schülerinnen und Schüler erforderlich. Der Zeitbedarf für die Programmentwicklung sollte (abhängig von der Programmiererfahrung) auf 60-120 Minuten veranschlagt werden.

Bezug Curriculum

Land	Stufe/Fächer	Bezüge
BW		
BY		
BE		
BB		
HB		
HH		
HE		
MV		
NI		
NW		
RP		
SL		
SN		
ST		
SH		
TH		

Anlagen

Aufgabe 2: Raumklima

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- Programm „IoT_MQTT_Indoor_Air_Quality.ft“

Weiterführende Informationen

- [1] Bosch Sensortec: *BME680 – Application Note*. Rev. 1.6, 20.09.2020.
- [2] Bosch Sensortec: *BME680 – Datasheet*. Rev. 1.7, 20.12.2021.
- [3] fischertechnik: [Photoresistor LDR03 \(32698\)](#). Datenblatt, 17.10.2018.
- [4] Bundesministerium für Arbeit und Soziales: [Beleuchtung](#). Technische Regeln für Arbeitsstätten, ASR A3.4, April 2011.
- [5] Umweltbundesamt: [Gesundheitliche Bewertung von Kohlendioxid in der Innenraumlufthygiene](#). Mitteilungen der Ad-hoc-Arbeitsgruppe Innenraumrichtwerte der Innenraumlufthygiene-Kommission des Umweltbundesamtes und der Obersten Landesgesundheitsbehörden, Bundesgesundheitsblatt – Gesundheitsforschung – Gesundheitsschutz 2008, 51, S. 1358–1369.
- [6] Deutsche Gesetzliche Unfallversicherung (DGUV): [Coronavirus SARS-CoV-2 – Ergänzende Empfehlungen der gesetzlichen Unfallversicherung für die Gefährdungsbeurteilung in Schulen](#). 03.12.2021.

Name: _____

Klasse: _____

Datum: _____

Lösungsblatt Aufgabe 2


Raumklima

Konstruktionsaufgabe

Siehe Bauanleitung.

Programmieraufgaben

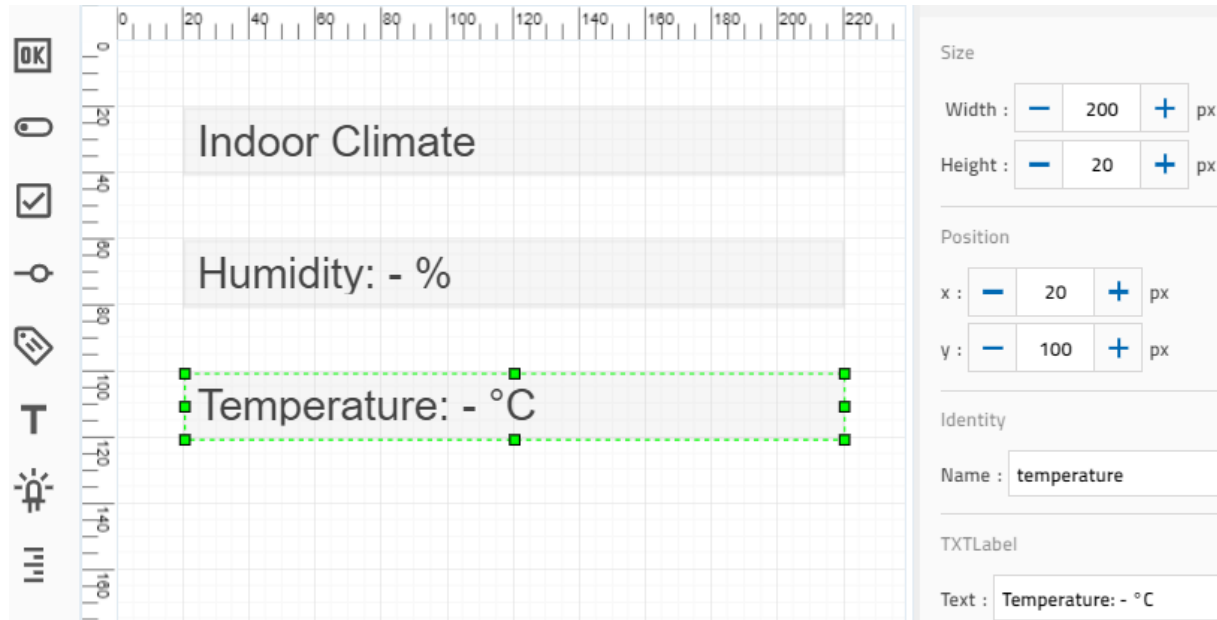
Konfiguration der Sensoren und Aktoren:



Mode	Name	Input	Output	Motor	Servomotor	Counter / Digital Input	I2C	USB1
Master	TXT	I3	O5				I2C_1	USB1_1
		Photo resistor	LED				Environmental sensor	Microphone

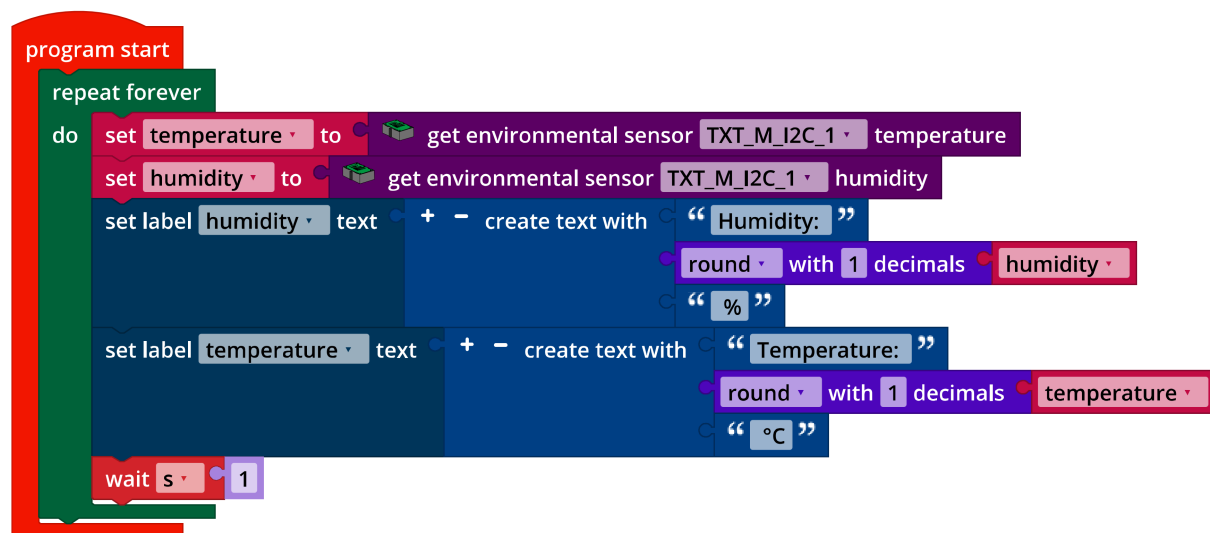
1. Messung der Raumklima-Werte

1a. Konfiguration des TXT-Displays:



Konfiguration der Anzeige auf dem TXT

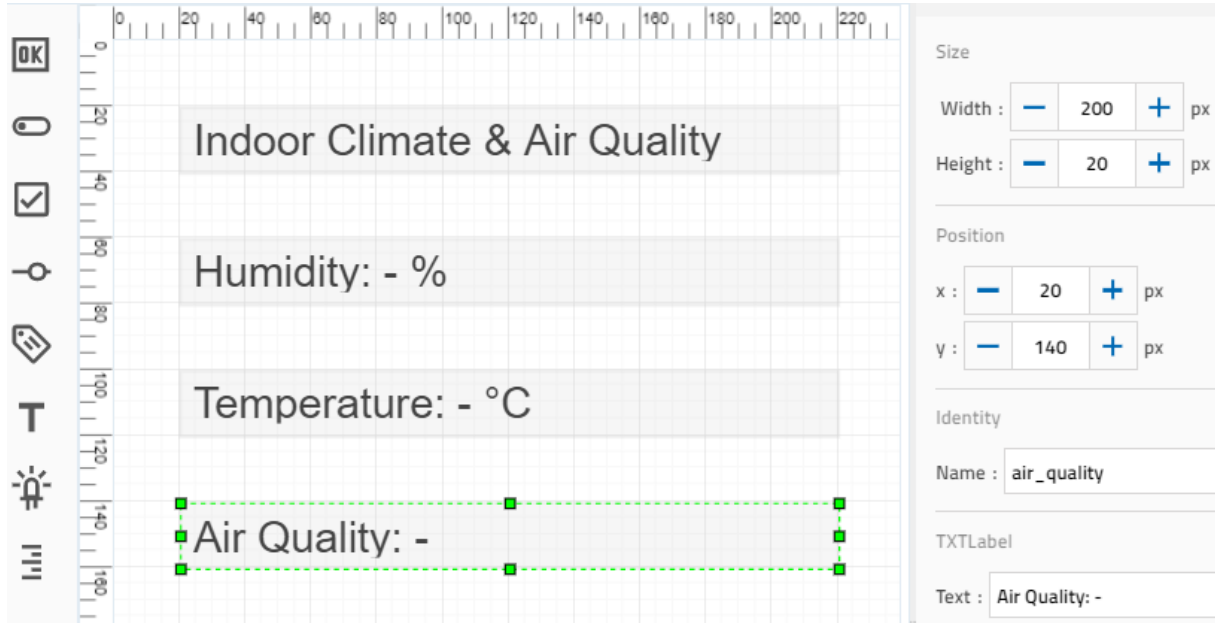
1b. Programm (Beispiel):



IoT_Indoor_Climate.ft

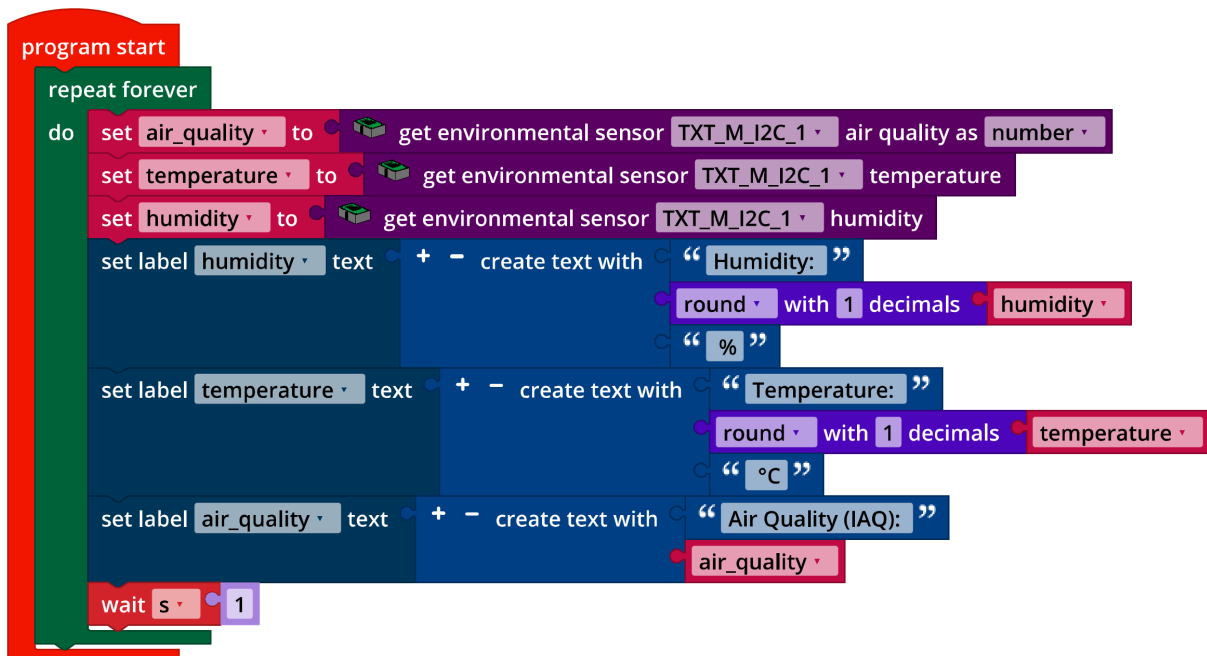
2. Messung der Luftqualität

2a. Konfiguration des TXT-Displays:



Erweiterung der Anzeige auf dem TXT

2b. Programm (Beispiel):



IoT_Indoor_Air_Quality.ft

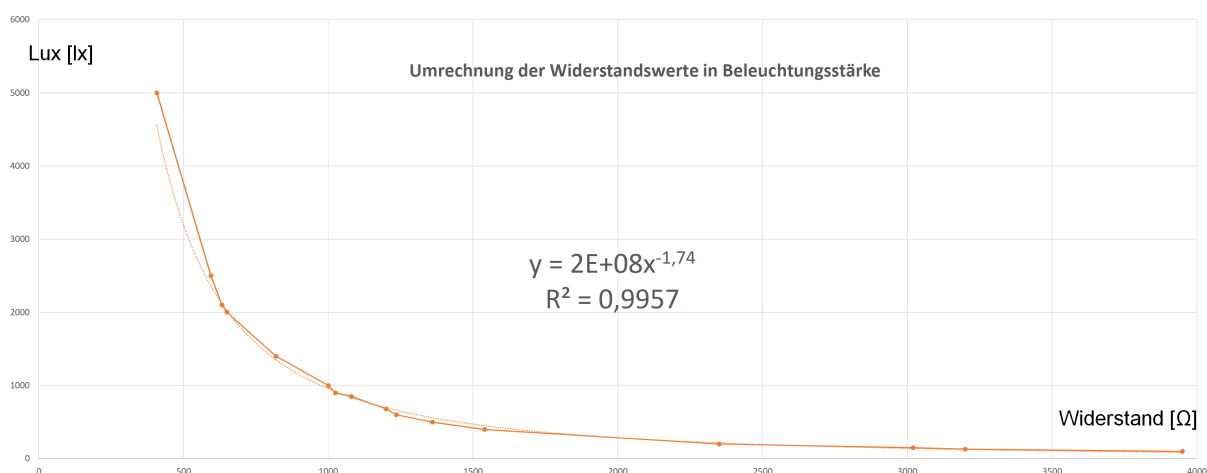
Experimentieraufgaben

1. Helligkeit

Die Messwerte (Widerstand/gemessene Beleuchtungsstärke) sollten in einem Tabellenkalkulationsprogramm mit einem Grafen veranschaulicht werden (x: Widerstand, y: Beleuchtungsstärke).

Die Umrechnung des Widerstandswerts r in die Beleuchtungsstärke i kann mit einer aus der Kennlinie ableitbaren Gleichung angenähert werden. Die Gleichung, die man bspw. in Excel zu einer hinzugefügten Trendlinie anzeigen lassen kann, sieht (bezogen auf Tageslicht-Messungen) etwa so aus:

$$i = 2 \cdot 10^8 \cdot r^{-1,74} \text{ lx}$$



Ein Arbeitsplatz sollte nach der Arbeitsstättenverordnung eine Beleuchtungsstärke von mindestens 500 lx aufweisen; das ist auch eine sinnvolle Mindestanforderung an einen Arbeitsplatz im Klassenraum.

Konfiguration des TXT-Displays:

Erweiterung der Display-Ausgabe am TXT

Programm (Beispiel):

```

program start
  set illuminance_threshold to 500
  repeat forever
    do
      set resistance to get photo resistor TXT_M_I3 value
      set illuminance to (2 * 10^8) * resistance^-1.74
      set air_quality to get environmental sensor TXT_M_I2C_1 air quality as number
      set temperature to get environmental sensor TXT_M_I2C_1 temperature
      set humidity to get environmental sensor TXT_M_I2C_1 humidity
      if illuminance < illuminance_threshold
        do
          set LED TXT_M_O7 on
        else
          set LED TXT_M_O7 off
      end if
      set label humidity text + - create text with " Humidity: "
      round with 1 decimals humidity
      " %"
      set label temperature text + - create text with " Temperature: "
      round with 1 decimals temperature
      " °C "
      set label air_quality text + - create text with " Air Quality: "
      air_quality
      set label illuminance text + - create text with " Illuminance: "
      round with 0 decimals illuminance
      " lx "
    end do
    wait s 1
  end repeat
  
```

IoT_Illuminance.ft

2. CO₂-Ampel

2a. Der „Lüftungsindex“ L soll Werte bis 100 einnehmen, dabei gleich Null sein, wenn Luftfeuchte und IAQ jeweils den gewünschten Zielwert erreichen, und gleich 100, wenn Luftfeuchte und IAQ die „Jetzt lüften!“-Schwelle erreichen.

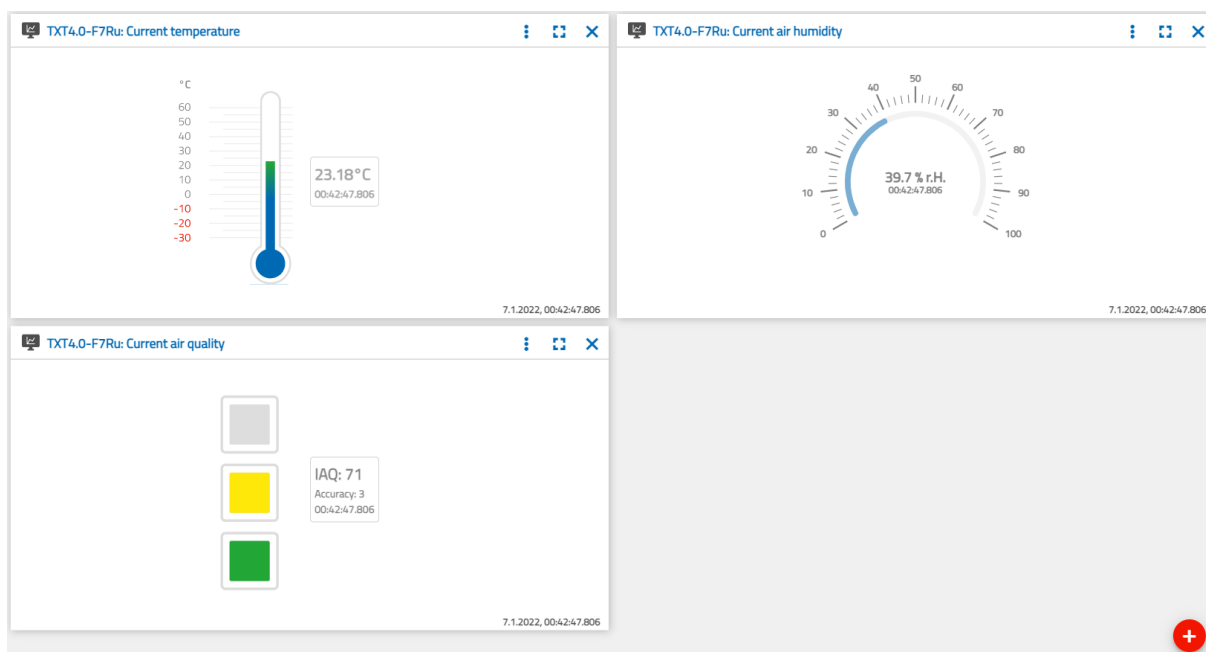
Seien H_0 der Zielwert und H_1 der Schwellenwert (obere Grenze) für die relative Luftfeuchte und H der gemessene Wert; und seien weiter Q_0 der Zielwert, Q_1 der Schwellenwert (obere Grenze) für die Luftqualität und Q der gemessene IAQ-Wert. Dann kann man den „Lüftungsindex“ L wie folgt definieren:

$$L = \left(\frac{H - H_0}{H_1 - H_0} + \frac{Q - Q_0}{Q_1 - Q_0} \cdot 3 \right) \cdot 25$$

Rechenbeispiel: Sollen nach den Stoßlüften die relative Luftfeuchte bei 30% und der IAQ bei 30 liegen (Zielwerte), und werden die Schwellenwerte (bspw. die Werte nach 20 Minuten im Klassenraum ohne Lüften) auf 55% und 100 festgelegt, dann gilt:

$$L = \left(\frac{H - 30}{25} + \frac{Q - 30}{70} \cdot 3 \right) \cdot 25$$

Konfiguration des IoT-Dashboards zur Anzeige der Messwerte (Temperatur, relative Luftfeuchte und Luftqualität):



Konfiguration des IoT-Dashboards

2b/c. Programmauszüge (Beispiel):

```

program start
  set air_quality_target_value to 30
  set humidity_target_value to 30
  set air_quality_range to 70
  set humidity_range to 25
  set ventilation_warning to 0
  execute function blink in a thread
  ...
  set ventilation_index to
    (humidity - humidity_target_value) / humidity_range +
    (air_quality - air_quality_target_value) / air_quality_range * 3
  ...
  if ventilation_index > 100
  do set ventilation_warning to 1
  else set ventilation_warning to 0
  if ventilation_index ≤ 0
  do set LED TXT_M_O7 on
  else set LED TXT_M_O7 off
  ...
  define blink
  repeat forever
  do if ventilation_warning = 1
  do set LED TXT_M_O5 on
  wait s 0.5
  set LED TXT_M_O5 off
  wait s 0.5
  
```

IoT_CO2_Signal_Light

3. Lautstärke

3a. Konfiguration des TXT-Displays:

Erweiterung des TXT-Displays um eine Lautstärke-Skala

3c. Als Grenzwert für den Lautstärkepegel können beispielsweise 70 dB gewählt werden.

Programm (Beispiel):

```

program start
  set illuminance_threshold to 500
  set loudness_threshold to 70
  set loudness_warning to 0
  execute function blink in a thread

  repeat forever
  do
    set loudness to microphone TXT_M_USB1_1 volume
    set resistance to get photo resistor TXT_M_I3 value
    set illuminance to (2 * 10^8) * resistance ^ -1.74
    set air_quality to get environmental sensor TXT_M_I2C_1 air quality as number
    set temperature to get environmental sensor TXT_M_I2C_1 temperature
    set humidity to get environmental sensor TXT_M_I2C_1 humidity

    + if illuminance < illuminance_threshold
    do
      set LED TXT_M_O7 on
    else
      set LED TXT_M_O7 off

    + if loudness > loudness_threshold
    do
      set loudness_warning to 1
    else
      set loudness_warning to 0

    set label humidity text + - create text with " Humidity: "
    + round with 1 decimals humidity
    + "%"

    set label temperature text + - create text with " Temperature: "
    + round with 1 decimals temperature
    + " °C"

    set label air_quality text + - create text with " Air Quality: "
    + air_quality

    set label illuminance text + - create text with " Illuminance: "
    + round with 0 decimals illuminance
    + " lx"

    set gauge loudness value loudness
  wait s 1

+ define blink
  repeat forever
  do
    + if loudness_warning = 1
    do
      set LED TXT_M_O5 on
      wait s 0.5
      set LED TXT_M_O5 off
      wait s 0.5
  
```

IoT_Loudness.ft

Anlagen

Aufgabe 2: Raumklima

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- Programm „IoT_MQTT_Indoor_Air_Quality.ft“

Weiterführende Informationen

- [1] Bosch Sensortec: *BME680 – Application Note*. Rev. 1.6, 20.09.2020.
- [2] Bosch Sensortec: *BME680 – Datasheet*. Rev. 1.7, 20.12.2021.
- [3] fischertechnik: [Photoresistor LDR03 \(32698\)](#). Datenblatt, 17.10.2018.
- [4] Bundesministerium für Arbeit und Soziales: [Beleuchtung](#). Technische Regeln für Arbeitsstätten, ASR A3.4, April 2011.
- [5] Umweltbundesamt: [Gesundheitliche Bewertung von Kohlendioxid in der Innenraumluft](#). Mitteilungen der Ad-hoc-Arbeitsgruppe Innenraumrichtwerte der Innenraumlufthygiene-Kommission des Umweltbundesamtes und der Obersten Landesgesundheitsbehörden, Bundesgesundheitsblatt – Gesundheitsforschung – Gesundheitsschutz 2008, 51, S. 1358–1369.
- [6] Deutsche Gesetzliche Unfallversicherung (DGUV): [Coronavirus SARS-CoV-2 – Ergänzende Empfehlungen der gesetzlichen Unfallversicherung für die Gefährdungsbeurteilung in Schulen](#). 03.12.2021.

Name: _____

Klasse: _____

Datum: _____

Aufgabe 3

Alarmanlage

Jetzt wird aus deiner Sensorstation eine Alarm- und Überwachungsanlage: Sie soll auf Geräusche und Bewegungen reagieren und es dir erlauben, den Raum über das IoT-Dashboard zu überwachen.

Konstruktionsaufgabe

Die in Aufgabe 1 konstruierte Sensorstation kann unverändert genutzt werden.

Vor dem Testen der Programme solltest du prüfen, ob die Verkabelung genügend Spiel lässt, um die Kamera in einem größeren Winkel nach links und rechts zu drehen.

Programmieraufgaben

1. Grundstellung

Die Kamera deiner Alarmanlage muss zunächst in eine Grundstellung gebracht werden. Dazu musst du die Kamera zunächst horizontal und vertikal – vom jeweiligen Antriebsmotor ausgesehen – gegen den Uhrzeigersinn bis zur Aktivierung des zugehörigen Endlagentasters drehen, damit du die genaue Position der Kamera kennst. Von dort kannst du sie in die gewünschte Grundstellung drehen.

1a. Um wie viele Impulse des Encoder-Motors muss die Kamera in der Vertikalen aus der Endlage nach oben gedreht werden, um sie horizontal nach vorne auszurichten? Bestimme den Wert zunächst anhand der Getriebeübersetzung.

Überprüfe das Ergebnis anschließend experimentell mit einem entsprechenden Blockly-Programm, das die Kamera erst in die Endlage dreht und dann horizontal ausrichtet.

Hinweis: Zur Erinnerung: Der Encoder-Motor liefert je Achsumdrehung 63,9 Impulse.

1b. Um wie viele Impulse des Encoder-Motors muss die Kamera in der Horizontalen aus der Endlage im Uhrzeigersinn gedreht werden, um sie „geradeaus“ auszurichten? Bestimme den Wert zunächst anhand der Getriebeübersetzung. Überprüfe das Ergebnis anschließend experimentell mit einem entsprechenden Blockly-Programm, das die Kamera erst in die Endlage dreht und dann nach vorne ausrichtet.

Hinweis: Der Drehkranz hat 58 Zähne.

1c. Führe beide Programme in einer Funktion zusammen, die die Ausrichtung der Kamera in beiden Dimensionen vornimmt. Wie kannst du die beiden Bewegungen parallelisieren?

2. Kameraüberwachung

In Aufgabe 1 hast du die Wetterstation sekundlich ein Bild von der Webcam an das Dashboard schicken lassen. Auf dieselbe Weise kannst du nun mit der Kamera den Raum überwachen.

Erweitere dein Programm aus Programmieraufgabe 1 um eine Bildübertragung an das Dashboard. Prüfe, wie viele Bilder du pro Sekunde maximal senden kannst, damit sie noch im Dashboard angezeigt werden.

3. Geräuschaktivierung

Für das Wachpersonal ist es ermüdend, ständig auf Bildschirme zu schauen, auf denen sich nichts Relevantes ereignet. Daher soll erst dann eine Bildübertragung erfolgen, wenn das Mikrofon ein ungewöhnliches Geräusch registriert.

3a. Teste zunächst mit einem einfachen Blockly-Programm den „normalen“ Geräuschpegel im Raum. Lass' dir die Werte auf dem Display des TXT anzeigen. Lege anschließend einen geeigneten Schwellenwert fest, ab dem eine Bildübertragung erfolgen soll.

Hinweis: Der Blockly-Befehl für das Mikrofon liefert dir die gemessene Lautstärke in Dezibel (dB).

3b. Schreibe ein Blockly-Programm, das mit der sekundlichen Übertragung des Kamerabildes erst dann beginnt, wenn der von dir definierte Schwellenwert für den Geräuschpegel überschritten wird. Bleibt es über eine Zeit von einer Minute ruhig, soll die Bildübertragung gestoppt werden.

4. Bewegungserkennung

Die Aktivierung der Kameraüberwachung kannst du auch von einer Bewegung im Raum abhängig machen. Konfiguriere dazu die Bewegungserkennung der Kamera.

4a. Erweitere dein Blockly-Programm aus Programmieraufgabe 3 so, dass die Bildübertragung auch dann aktiviert wird, wenn eine Bewegung erkannt wird.

4b. Während der Aktivierung der Kamera soll außerdem die rote LED blinken.

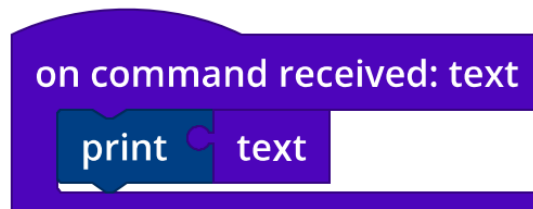
Experimentieraufgaben

1. Sprachsteuerung

Damit die Kamera einen größeren Teil des Raums überwachen kann, soll die Kamera nun über Sprachkommandos gesteuert werden. Lade dazu die App „Voice Control“ aus dem Apple-App-Store (für iOS) oder dem Google-Play-Store (für Android) herunter und verbinde sie mit dem TXT 4.0.

- Verbindung über WLAN: Der TXT 4.0 Controller und das Gerät (Smartphone oder Tablet) müssen mit demselben WLAN-Router verbunden werden. Der Router muss außerdem die Kommunikation der Geräte untereinander erlauben. Die IP-Adresse des TXT 4.0, mit der die App verbunden werden muss, ist dann über das Menü des Touch-Screen unter „Info“ / „WLAN“ abfragbar.
- Verbindung mit WLAN AP: Am TXT 4.0 kann statt „WLAN“ die Option „Access Point“ unter „Einstellungen“ / „Netzwerk“ aktiviert werden. Dann kann das Smartphone direkt mit dem Controller verbunden werden. Der für die WLAN-Verbindung benötigte WPA2-Key kann im TXT-Menü unter „Access Point“ abgelesen (oder geändert bzw. deaktiviert) werden.

Wenn du die App mit dem Controller verbunden hast, werden die Sprachkommandos als Text an den Controller übertragen und du kannst sie mit der folgenden Event-Funktion auswerten:



1a. Schreibe Funktionen für eine Drehung nach rechts, nach links, nach oben und nach unten. Da die Spracherkennung etwas verzögert reagiert, ist es sinnvoll, die Kamera bei jedem Aufruf um einen als Parameter übergebenen Winkel zu drehen.

Überlege: Wie kannst Du verhindern, dass die Kamera zu weit dreht?

1b. Wähle passende Sprachkommandos für die Kamerasteuerung. Ergänze dein Programm aus Programmieraufgabe 4 um die Steuerungsfunktionen aus Teilaufgabe 1a.

Steuere die Kamera nun über deine Sprachkommandos. Stelle dabei sicher, dass die Kamera nicht zu weit dreht.

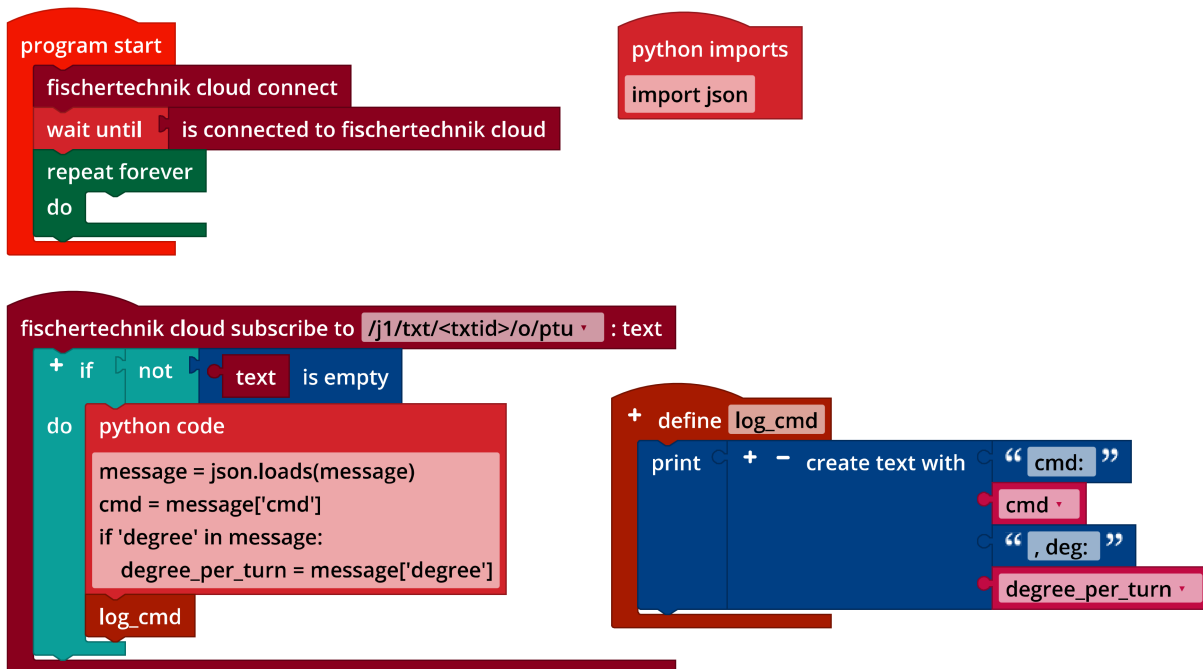
2. Steuerung über das Cloud-Dashboard

Auch über das IoT-Dashboard in der fischertechnik-Cloud kann die Kamera gesteuert werden.

Dazu musst du nach dem Aufbau der Verbindung mit dem IoT-Server den TXT mit der folgenden Funktion als „MQTT Subscriber“ anmelden:

fischertechnik cloud subscribe to `/j1/txt/<txtid>/o/ptu` : text

2a. Anschließend kannst du die mit der Maus angeklickten Kommandos mit dem folgenden Testprogramm auslesen:



```

program start
  fischertechnik cloud connect
  wait until is connected to fischertechnik cloud
  repeat forever
    do
      python code
        message = json.loads(message)
        cmd = message['cmd']
        if 'degree' in message:
          degree_per_turn = message['degree']
        log_cmd
  python imports
    import json
  define log_cmd
    print + - create text with " cmd: "
    cmd
    " , deg: "
    degree_per_turn
  
```

IoT_Test_Dashboard_Control.ft

2b. Neben den Steuerkommandos aus Experimentieraufgabe 1 gibt es hier noch ein Kommando, das die Kamera in die Grundstellung bewegen soll, und eines, das die Motoren sofort stoppt. Ergänze die Steuerkommandos entsprechend.

2c. Ersetze nun die Sprachsteuerung der Kamera durch die Steuerung über das Dashboard.

Anlagen

Aufgabe 3: Alarmanlage

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- fischertechnik-App „Voice Control“
- Hilfsprogramm „IoT_Test_Dashboard_Control.ft“
- Account in der fischertechnik-Cloud (www.fischertechnik-cloud.com)

Aufgabe 3: Alarmanlage

Aus der Sensorstation wird eine Alarmanlage mit Überwachungskamera, die auf unterschiedliche Auslöser reagiert (Geräusch, Bewegung) und das Kamerabild an einen Webserver überträgt. Die Kamera kann vom IoT-Dashboard aus gesteuert werden.

Thema

- Sprach- und Fernsteuerung einer Kamera („Pan-Tilt-Steuerung“)
- Bewegungs- und Geräuschaktivierung
- Nutzung eines Webserver zur Fernsteuerung eines IoT-Systems

Lernziele

- Präzise Steuerung über Encoder-Motoren
- Geeignete Strukturierung eines Programms durch Verwendung von Funktionen
- Verständnis des MQTT-Protokolls

Zeitaufwand

Wurde die Sensorstation bereits in Aufgabe 1 aufgebaut, ist kein Zeitaufwand für die Konstruktion erforderlich.

Für die Lösung der vier Programmieraufgaben sollten 90-120 Minuten veranschlagt werden. Die Lösung der Experimentieraufgaben sollte (abhängig von der Programmiererfahrung der Schülerinnen und Schüler) ebenfalls innerhalb von 90-120 Minuten gelingen.

Bezug Curriculum

Land	Stufe/Fächer	Bezüge
BW		
BY		
BE		
BB		
HB		
HH		
HE		
MV		
NI		
NW		
RP		
SL		
SN		
ST		
SH		
TH		

Anlagen

Aufgabe 3: Alarmanlage

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- fischertechnik-App „Voice Control“
- Hilfsprogramm „IoT_Test_Dashboard_Control.ft“
- Account in der fischertechnik-Cloud (www.fischertechnik-cloud.com)

Name: _____

Klasse: _____

Datum: _____

Lösungsblatt Aufgabe 3

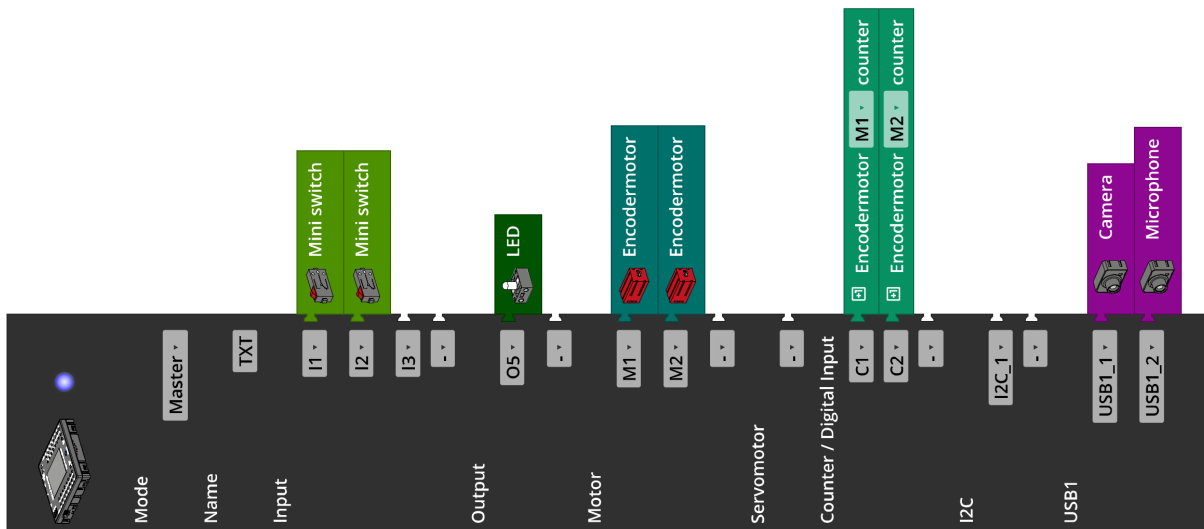
Alarmanlage

Konstruktionsaufgabe

Siehe Bauanleitung.

Programmieraufgaben

Konfiguration der Sensoren und Aktoren:



Für die Lösung von Experimentieraufgabe 2 wird die „Voice Control“-App (für iOS oder Android) benötigt. Die App muss für die Spracherkennung mit dem Internet verbunden sein und (über Bluetooth oder WLAN) mit dem Controller verbunden werden.

1. Grundstellung

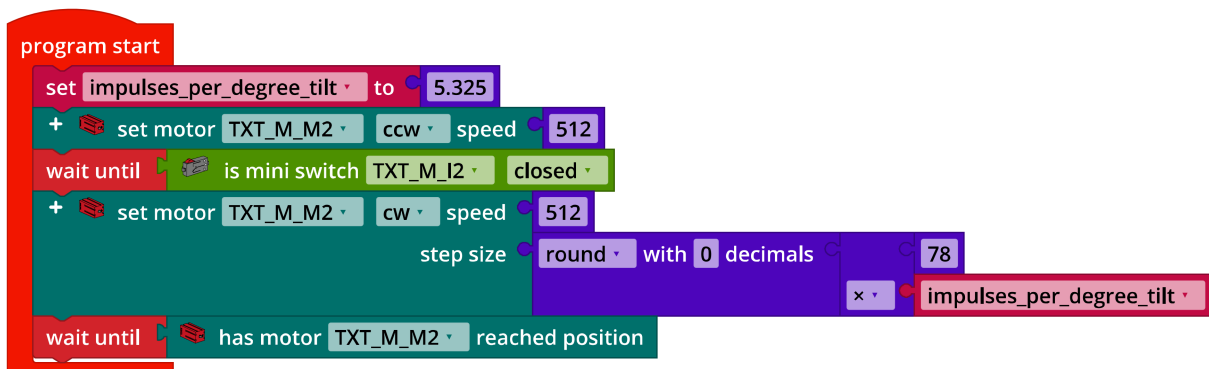
1a. Jede Umdrehung der Motorachse entspricht einer Umdrehung der Schnecke, und jede Umdrehung der Schnecke dreht das Z30 um einen Zahn weiter. Für eine 90°-Drehung des Z30 werden also 30/4 Umdrehungen der Motorachse benötigt. Das entspricht (bei 63,9 Encoder-Impulsen je Achsumdrehung)

$$30 \cdot \frac{63,9}{4} = 479,25$$

Encoder-Impulsen (oder 5,325 Impulsen je Winkelgrad).

Tatsächlich wird der Endlagentaster nicht genau nach einer Vierteldrehung ausgelöst; die Kamera muss lediglich etwa 78° gedreht werden. Das entspricht $78/90 \cdot 479,25 = 415,35$ Encoder-Impulsen.

Programm (Beispiel):



IoT_Init_Camera_Tilt.ft

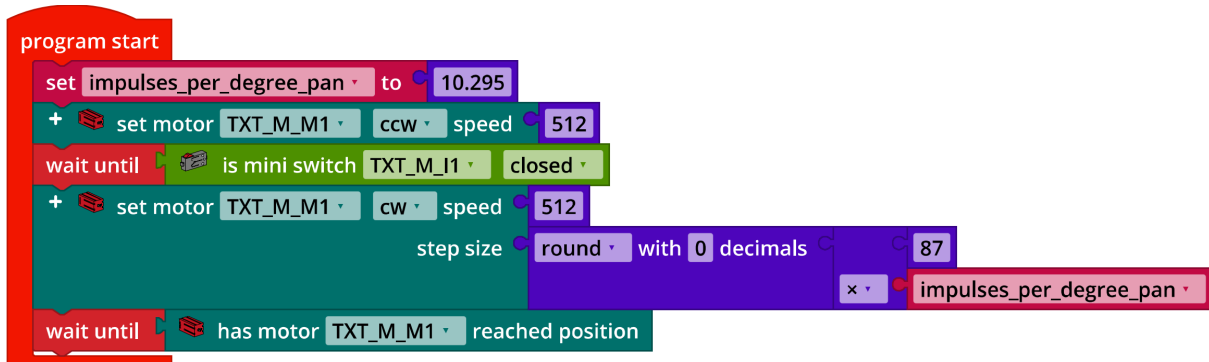
1b. Der Drehkranz hat 58 Zähne; eine 90°-Drehung benötigt also 58/4 Umdrehungen der Motorachse. Das entspricht

$$58 \cdot \frac{63,9}{4} = 926,55$$

Encoder-Impulsen (oder 10,295 Impulsen je Winkelgrad).

Auch hier wird der Endlagentaster nicht erst nach einer 90°-Drehung ausgelöst, bereits bei ca. 3° ausgelöst; die Kamera muss lediglich um etwa 87° gedreht werden. Das entspricht $87/90 \cdot 926,55 = 895,67$ Encoder-Impulsen.

Programm (Beispiel):



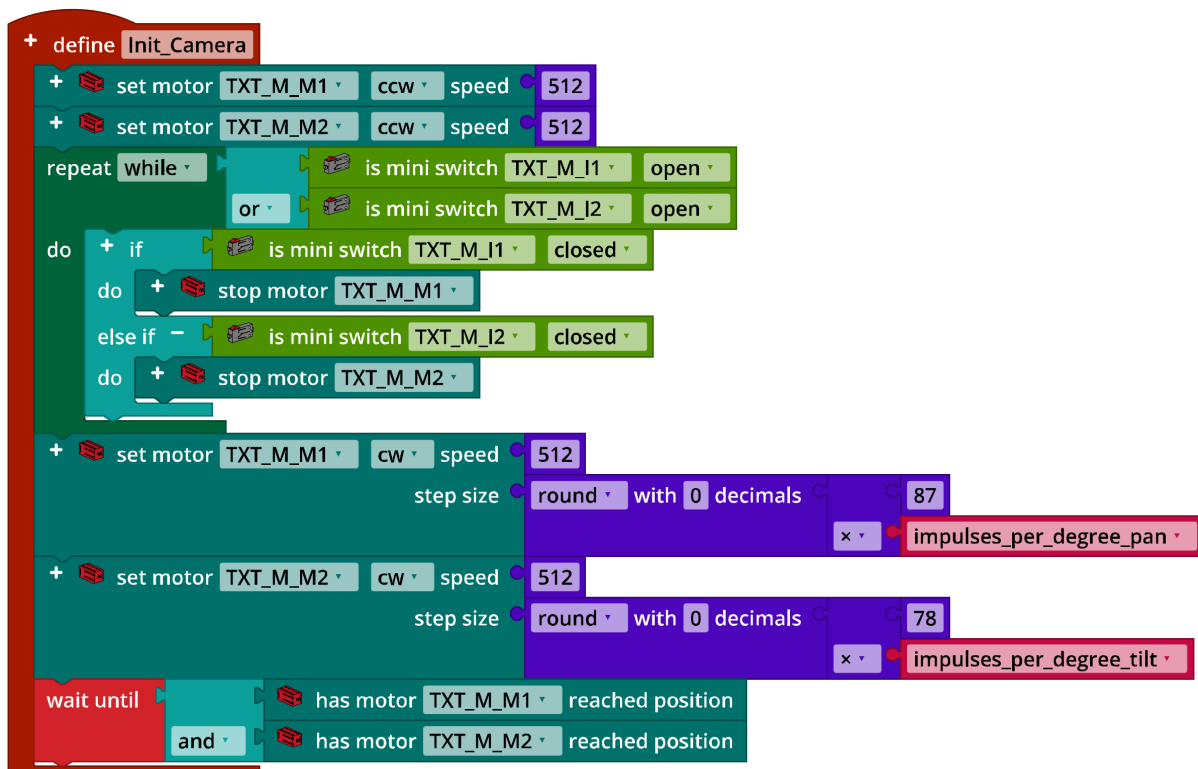
```

program start
  set impulses_per_degree_pan to 10.295
  + set motor TXT_M_M1 ccw speed 512
  wait until is mini switch TXT_M_I1 closed
  + set motor TXT_M_M1 cw speed 512
  step size round with 0 decimals 87
  × impulses_per_degree_pan
  wait until has motor TXT_M_M1 reached position
  
```

IoT_Init_Camera_Pan.ft

1c. Bei der Zusammenführung der beiden Programme können die beiden Bewegungen parallelisiert und die Initialisierungszeit damit erheblich verkürzt werden. Dafür gibt es zwei Lösungswege: Ausführung der beiden Funktionen als parallele Threads oder Integration in einer Funktion.

Programm Lösungsweg A – eine Funktion (Beispiel):



```

+ define Init_Camera
  + set motor TXT_M_M1 ccw speed 512
  + set motor TXT_M_M2 ccw speed 512
  repeat while
    is mini switch TXT_M_I1 open
    or
    is mini switch TXT_M_I2 open
  do
    + if
      is mini switch TXT_M_I1 closed
    do
      + stop motor TXT_M_M1
    else if
      is mini switch TXT_M_I2 closed
    do
      + stop motor TXT_M_M2
  + set motor TXT_M_M1 cw speed 512
  step size round with 0 decimals 87
  × impulses_per_degree_pan
  + set motor TXT_M_M2 cw speed 512
  step size round with 0 decimals 78
  × impulses_per_degree_tilt
  wait until
    has motor TXT_M_M1 reached position
    and
    has motor TXT_M_M2 reached position
  
```

IoT_Init_Camera_A.ft

Programm Lösungsweg B – parallele Threads (Beispiel):

```

program start
  set impulses_per_degree_pan to 10.295
  set impulses_per_degree_tilt to 5.325
  execute function Init_Pan in a thread
  execute function Init_Tilt in a thread
  repeat forever
  do

+ define Init_Pan
  + set motor TXT_M_M1 ccw speed 512
  wait until is mini switch TXT_M_I1 closed
  + set motor TXT_M_M1 cw speed 512
  step size round with 0 decimals 87
  x impulses_per_degree_pan
  wait until has motor TXT_M_M1 reached position

+ define Init_Tilt
  + set motor TXT_M_M2 ccw speed 512
  wait until is mini switch TXT_M_I2 closed
  + set motor TXT_M_M2 cw speed 512
  step size round with 0 decimals 78
  x impulses_per_degree_tilt
  wait until has motor TXT_M_M2 reached position
  
```

IoT_Init_Camera_B.ft

2. Kameraüberwachung

Programm (Beispiel):

```

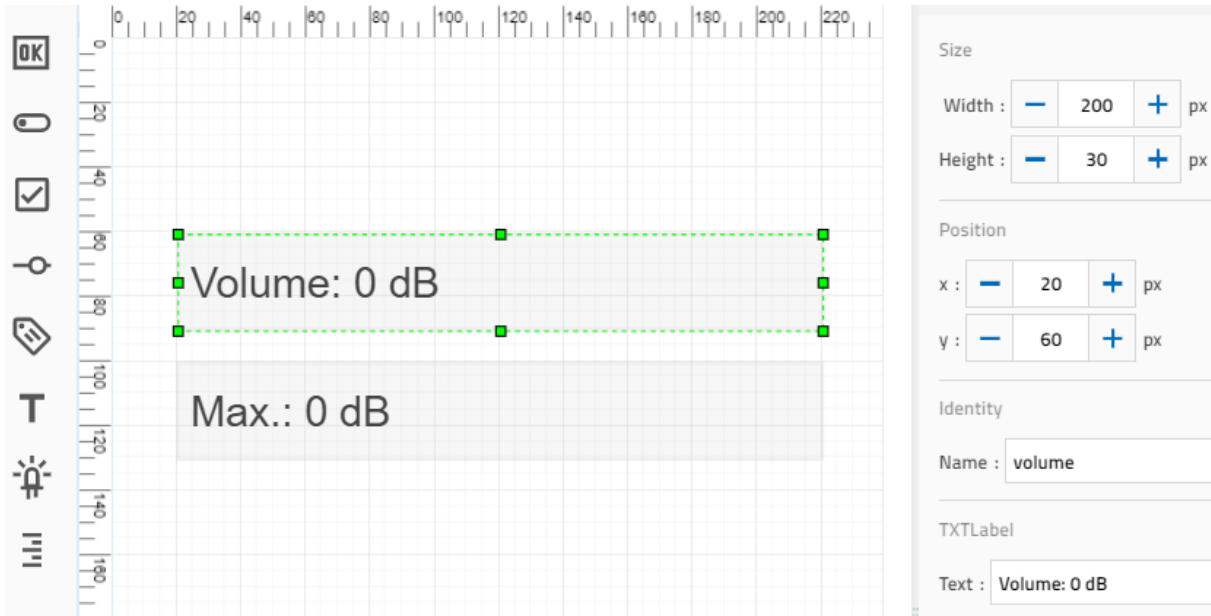
program start
  set impulses_per_degree_pan to 10.295
  set impulses_per_degree_tilt to 5.325
  set snapshot to ""
  execute function Init_Pan in a thread
  execute function Init_Tilt in a thread
  fischertechnik cloud connect
  wait until is connected to fischertechnik cloud
  repeat forever
    do fischertechnik cloud publish text to /j1/txt/<txtid>/i/cam
      format text
      + - with
      timestamp
      snapshot
    wait ms 100

on image TXT_M_USB1_1 change: event
  set snapshot to
  convert image event to base64
  
```

IoT_Surveillance_Camera.ft

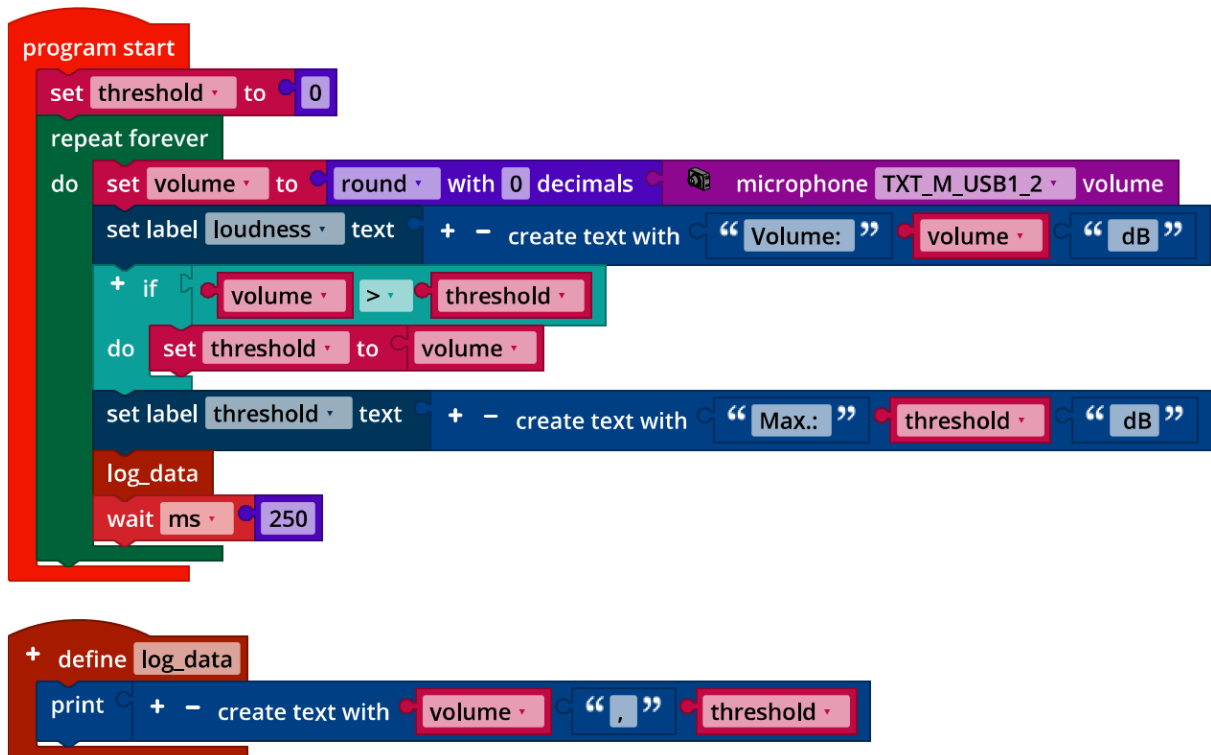
3. Geräuschaktivierung

3a. Konfiguration des TXT-Display (Beispiel):



Display-Konfiguration

Programm (Beispiel):



IoT_Microphone_Level.ft

3b. Programm mit Lautstärke-Schwelle 70 dB (Beispiel):

```

program start
  set impulses_per_degree_pan to 10.295
  set impulses_per_degree_tilt to 5.325
  set snapshot to ""
  set threshold to 70
  set surveillance_period to 60
  set surveillance_status to 0
  set start_time to Timestamp s
  execute function Init_Pan in a thread
  execute function Init_Tilt in a thread
  fischertechnik cloud connect
  wait until is connected to fischertechnik cloud
  repeat forever
    do + if microphone TXT_M_USB1_2 volume > threshold
      do set start_time to Timestamp s
        set surveillance_status to 1
      + if surveillance_status = 1
        and Timestamp s - start_time > surveillance_period
      do set surveillance_status to 0
      + if surveillance_status = 1
      do fischertechnik cloud publish text to /j1/txt/<txtid>/i/cam
        format text {"ts": "0", "data": "0"}
        + - with datetimestamp
        snapshot
      log_data
      wait s 1
  
```

```

on image TXT_M_USB1_1 change: event
  set snapshot to convert image event to base64
  
```

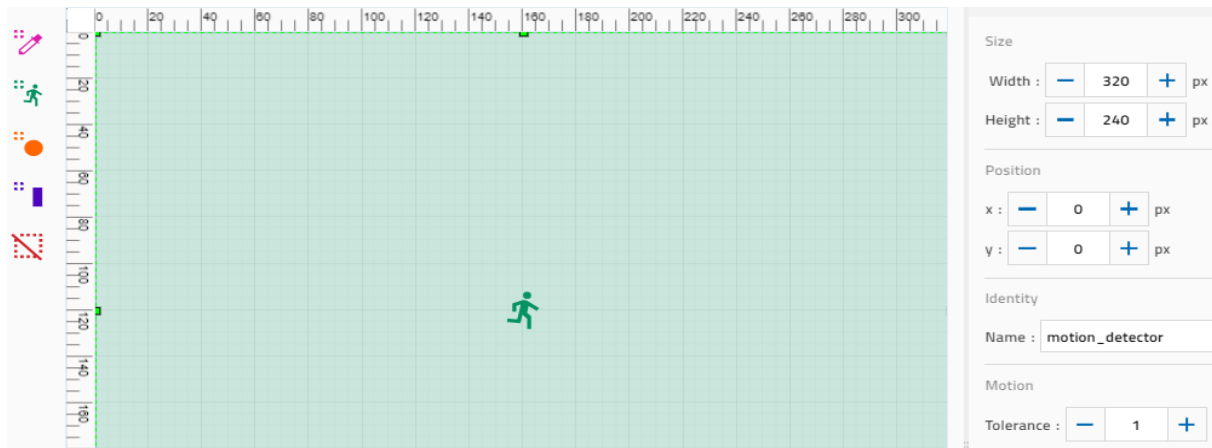
```

+ define log_data
  print + - create text with surveillance_status
  "("
  round with 0 decimals Timestamp s - start_time
  "s)"
  
```

IoT_Surveillance_Camera_Noise_Detection.ft

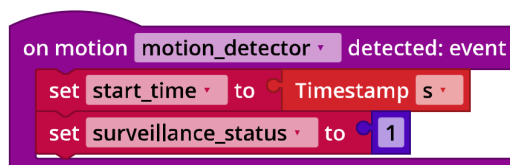
4. Bewegungserkennung

Die Bewegungserkennung wird über das gesamte Kamerafenster definiert. Über die „Toleranz“ kann im Inspektor die Empfindlichkeit der Erkennung auf einen Wert von 0 bis 1 eingestellt werden:



Konfiguration Bewegungserkennung

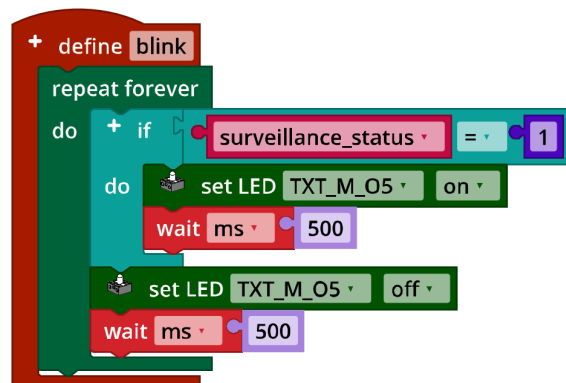
4a. Programmauszug (Beispiel):



IoT_Surveillance_Camera_Motion_Detection.ft

Das Erkennen einer Bewegung wird in einem Flag („surveillance_status“) gespeichert, ebenso der Zeitpunkt der Erkennung.

4b. Die Funktion „blink“ wird zu Beginn des Hauptprogramms als Thread gestartet:



IoT_Surveillance_Camera_Motion_Detection.ft

Experimentieraufgaben

1. Sprachsteuerung

1a. Steuerungsfunktionen (Beispiel):

```

+ define turn_right with:
- variable: degree
+ set motor TXT_M_M1 cw speed 512
  step size round with 0 decimals degree impulses_per_degree_pan
wait until has motor TXT_M_M1 reached position
    
```

```

+ define turn_left with:
- variable: degree
+ set motor TXT_M_M1 ccw speed 512
  step size round with 0 decimals degree impulses_per_degree_pan
wait until has motor TXT_M_M1 reached position
    
```

```

+ define turn_up with:
- variable: degree
+ set motor TXT_M_M2 cw speed 512
  step size round with 0 decimals degree impulses_per_degree_tilt
wait until has motor TXT_M_M2 reached position
    
```

```

+ define turn_down with:
- variable: degree
+ set motor TXT_M_M2 ccw speed 512
  step size round with 0 decimals degree impulses_per_degree_tilt
wait until has motor TXT_M_M2 reached position
    
```

IoT_Camera_Motion_Control.ft

Ein zu weites Drehen der Kamera lässt sich am einfachsten verhindern, indem der Drehwinkel in horizontaler und vertikaler Richtung auf jeweils insgesamt $\pm 90^\circ$ begrenzt wird. Setzt man beide Auslenkungswinkel nach der Initialisierung der Grundstellung auf 0° , so ist das im Programm leicht zu überprüfen.

1b. Programmauszug: Steuerung der Kamera (Beispiel):

```

on command received: text
  set command to text
  + if
    command = "links" and pan_angle + degree_per_turn ≤ max_angle
  do
    turn_left with:
      degree degree_per_turn
    set pan_angle to pan_angle + degree_per_turn
  else if
    command = "rechts" and pan_angle - degree_per_turn ≥ min_angle
  do
    turn_right with:
      degree degree_per_turn
    set pan_angle to pan_angle - degree_per_turn
  else if
    command = "hoch" and tilt_angle + degree_per_turn ≤ max_angle
  do
    turn_up with:
      degree degree_per_turn
    set tilt_angle to tilt_angle + degree_per_turn
  else if
    command = "runter" and tilt_angle - degree_per_turn ≥ min_angle
  do
    turn_down with:
      degree degree_per_turn
    set tilt_angle to tilt_angle - degree_per_turn
  
```

IoT_Surveillance_Camera_Voice_Control.ft

Im Hauptprogramm werden zu Beginn die folgenden Variablen initialisiert:

```

program start
  set impulses_per_degree_pan to 10.295
  set impulses_per_degree_tilt to 5.325
  set min_angle to -45
  set max_angle to 45
  set pan_angle to 0
  set tilt_angle to 0
  set degree_per_turn to 5
  
```

IoT_Surveillance_Camera_Voice_Control.ft

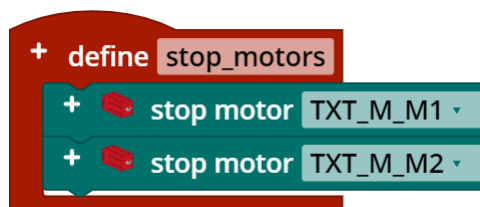
Der maximale und der minimale Auslenkungswinkel sind ggf. an die Konstruktion anzupassen (Reichweite der Verkabelung).

2. Steuerung über das Cloud-Dashboard

Die Steuerungsknöpfe des Dashboards liefern die Kommandos „stop“, „home“, „relmove_left“, „relmove_right“, „relmove_up“ und „relmove_down“. Die „relmove“-Kommandos entsprechen den vier Steuerfunktionen aus Experimentieraufgabe 1. Wird vom IoT-Server ein Wert „degree“ (2, 5, 10 oder 20°) übermittelt, muss die Drehwinkelvorgabe im Programm angepasst werden. Das Kommando „home“ bewegt die Kamera in die Grundstellung; für das Kommando „stop“ ist eine Funktion zu ergänzen, die die beiden Motoren stoppt.

Die Abfrage der Steuerkommandos kann aus der Lösung zu Experimentieraufgabe 1 übernommen werden.

Funktion „stop_motors“ (Beispiel):



IoT_Surveillance_Camera_Dashboard_Control.ft

Programmauszug: Steuerung der Kamera (Beispiel):

```
fischertechnik cloud subscribe to /j1/txt/<txtid>/o/ptu : text
+ if not text is empty
do python code
  message = json.loads(message)
  cmd = message['cmd']
  if 'degree' in message:
    degree_per_turn = message['degree']
camera_control
```

```
+ define camera_control
+ if cmd = "relmove_left" and pan_angle + degree_per_turn ≤ max_angle
do turn_left with:
  degree degree_per_turn
  set pan_angle to pan_angle + degree_per_turn
else if cmd = "relmove_right" and pan_angle - degree_per_turn ≥ min_angle
do turn_right with:
  degree degree_per_turn
  set pan_angle to pan_angle - degree_per_turn
else if cmd = "relmove_up" and tilt_angle + degree_per_turn ≤ max_angle
do turn_up with:
  degree degree_per_turn
  set tilt_angle to tilt_angle + degree_per_turn
else if cmd = "relmove_down" and tilt_angle - degree_per_turn ≥ min_angle
do turn_down with:
  degree degree_per_turn
  set tilt_angle to tilt_angle - degree_per_turn
else if cmd = "home"
do init_Camera
else if cmd = "stop"
do stop_motors
```

IoT_Surveillance_Camera_Dashboard_Control.ft

Anlagen

Aufgabe 3: Alarmanlage

Erforderliches Material

- PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
- USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
- fischertechnik-App „Voice Control“
- Hilfsprogramm „IoT_Test_Dashboard_Control.ft“
- Account in der fischertechnik-Cloud (www.fischertechnik-cloud.com)