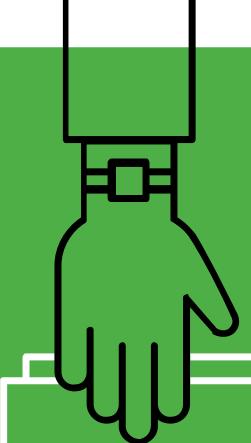
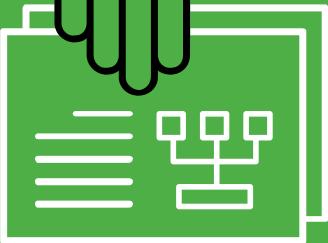
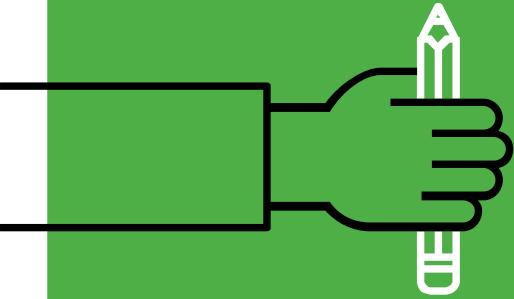
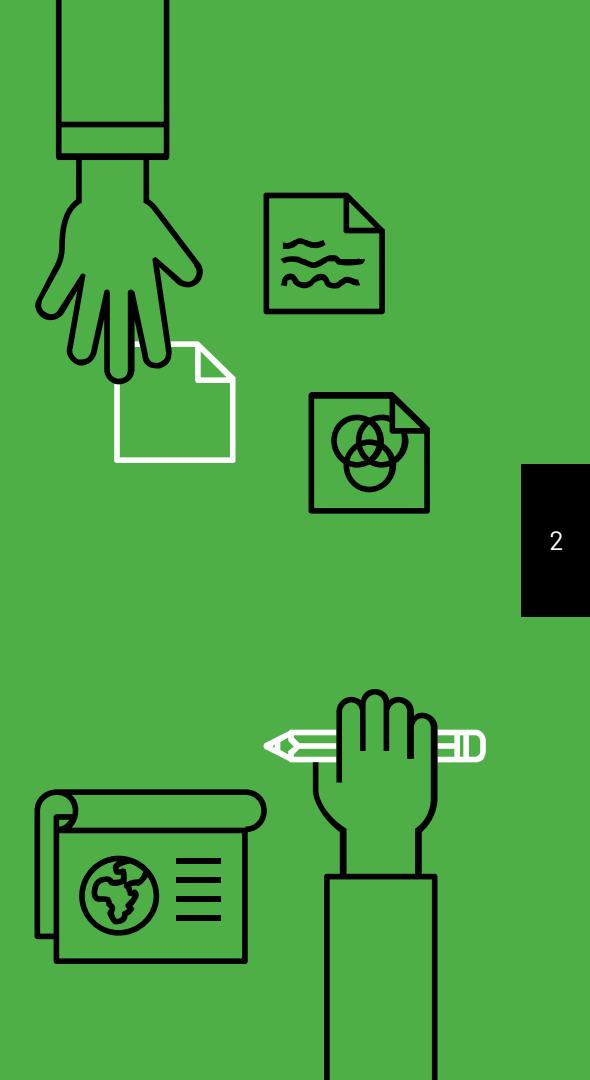
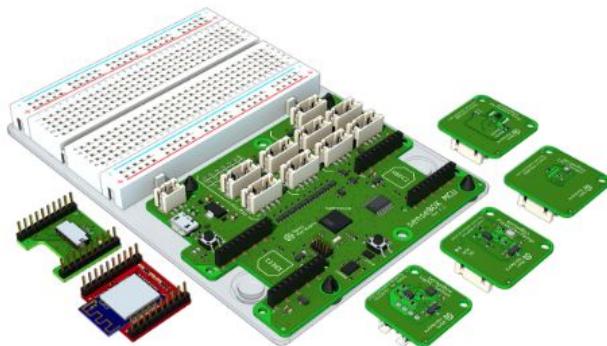


senseBox
Lehrerfortbildung



Übersicht

senseBox 

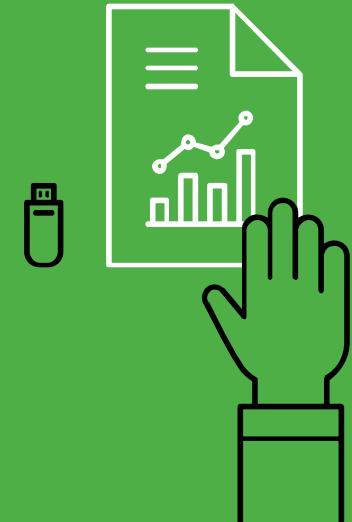
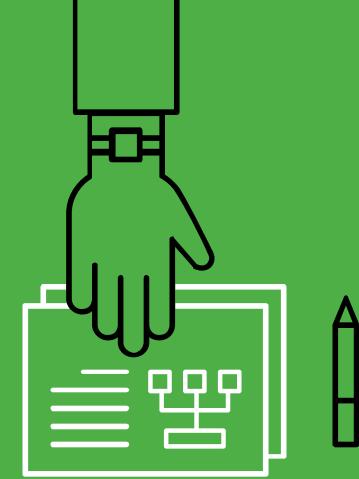


senseBox:edu

- ▷ Lernsetup für den MINT Unterricht
- ▷ Vollständig Arduino kompatibel
- ▷ Physical Computing

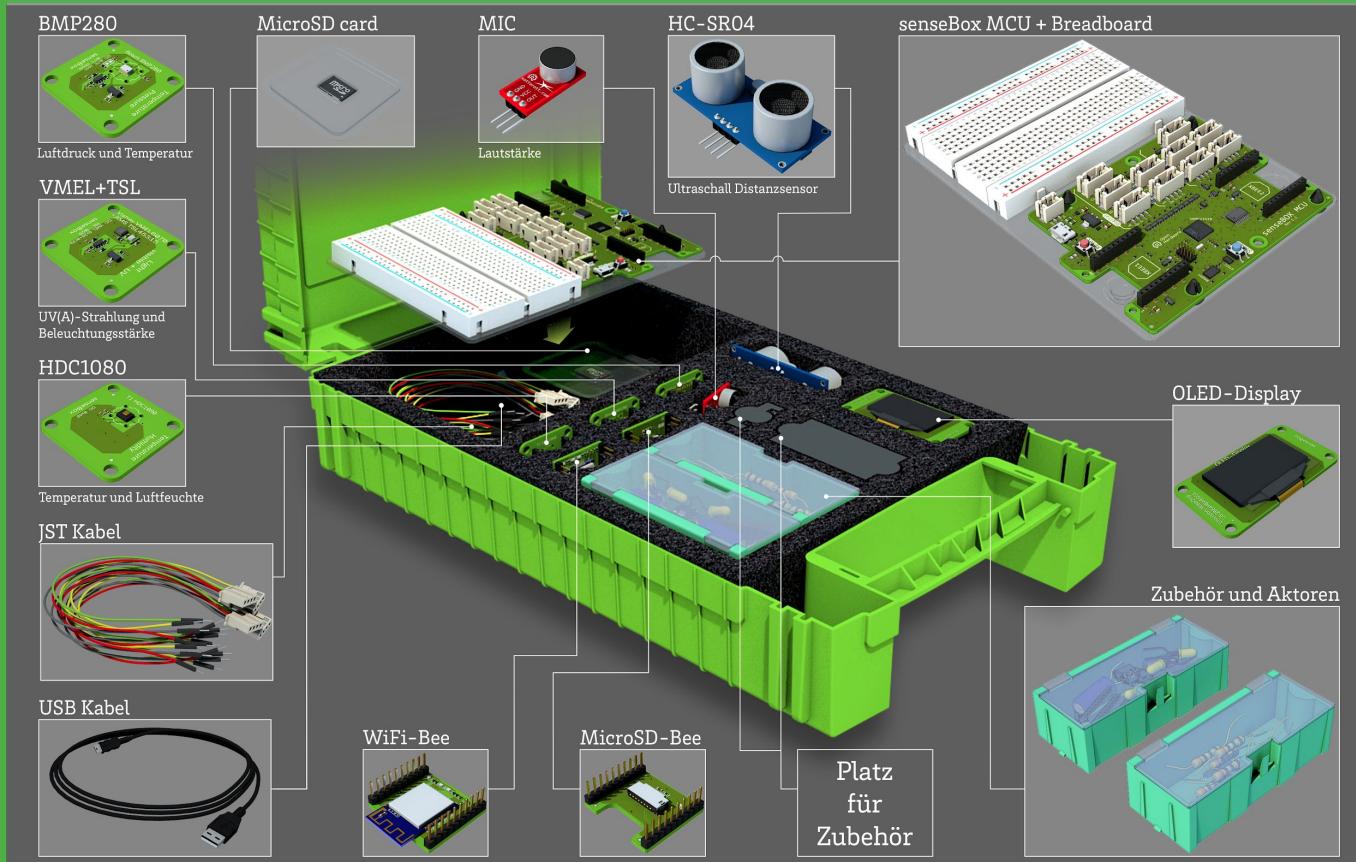
Ziele:

- ▷ Programmieren lernen
- ▷ Umweltphänomene messen und teilen
- ▷ Informatische Bildung nicht nur im Informatikunterricht
- ▷ Citizen Science





[Inhalt]



Physical Computing

- Entwicklung von **Systemen aus Soft- und Hardware**, die mit der physischen Welt interagieren. (Lärmampeln, Roboter, smarte Textilien)
- Verbinden von Materialien jeglicher Art (Pappe, Holz, Elektronik..)
- Öffnen der **Black Box** der digitalen Welt
- **Verbindung von verschiedenen Fächern** durch Physical Computing (Kunstwerke erstellen, Musikinstrumente erschaffen, Messgeräte bauen)
- Erlangen von Kompetenzen in verschiedenen Bereichen und übertragen auf andere Kontexte



Programmierumgebungen

```
#include <SenseboxLU.h>

GPS gps;
float loc[2]; //Geographische Breite
float long[2]; //Geographische Länge
float alt; //Höhe über Meeresspiegel in Meter
float temp;
float date;
float time;

void setup() {
  gps.begin();
}

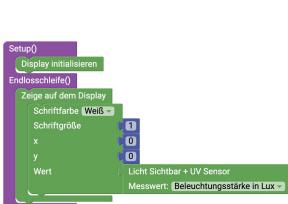
void loop() {
  loc[0] = gps.getLatitude();
  loc[1] = gps.getLongitude();
  alt = gps.getAltitude();
  temp = gps.getTemperature();
  date = gps.getDate();
  time = gps.getTime();
}
```

Arduino IDE

- Textbasierte Programmierung
- Erweiterungen
- Möglich → vollständige Arduino kompatibilität

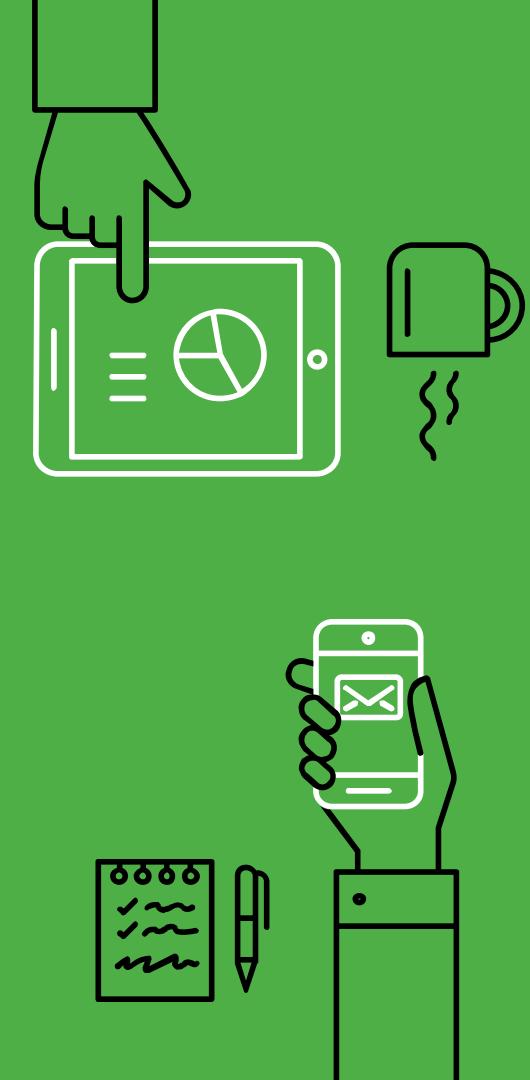
Blockly für senseBox

- Grafische Programmierung
- Großer Funktionsumfang
- Einfacher/schneller Einstieg ohne Software installation



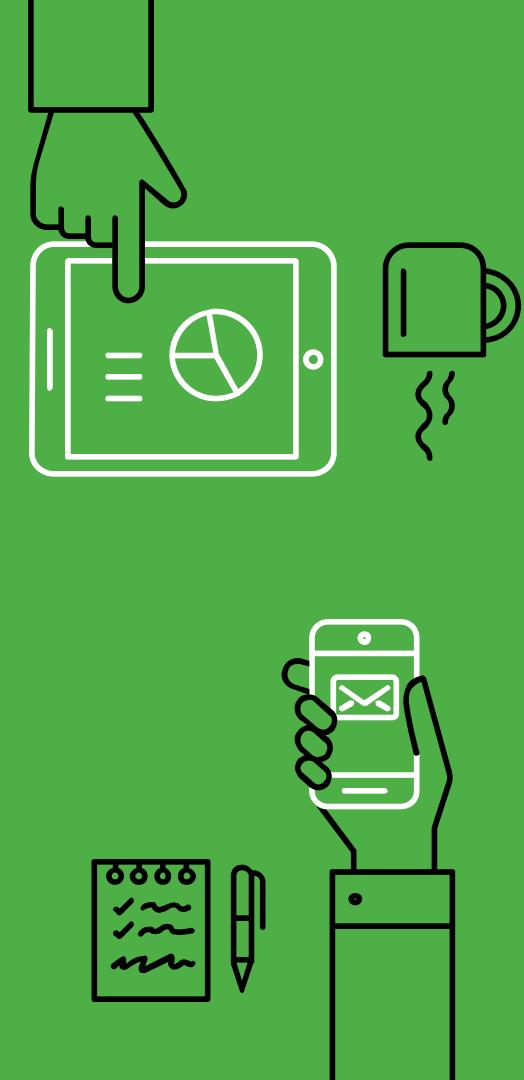
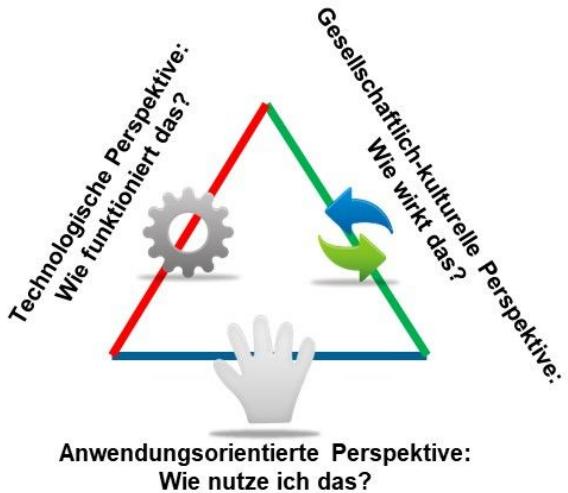
NEPO - openRoberta

- Grafische Programmierung
- Basis Funktionen
- aktuell im Beta Status



Das Dagstuhl-Dreieck

Digitale Bildung: Das Dagstuhl-Dreieck (2016)





1. BEDIENEN UND ANWENDEN	2. INFORMIEREN UND RECHERCHIEREN	3. KOMMUNIZIEREN UND KOOPERIEREN	4. PRODUZIEREN UND PRÄSENTIEREN	5. ANALYSIEREN UND REFLEKTIEREN	6. PROBLEMLÖSEN UND MODELLIEREN
1.1 Medienausstattung (Hardware)	2.1 Informationsrecherche	3.1 Kommunikations- und Kooperationsprozesse	4.1 Medienproduktion und Präsentation	5.1 Medienanalyse	6.1 Prinzipien der digitalen Welt
Medienausstattung (Hardware) kennen, auswählen und reflektiert anwenden; mit dieser verantwortungsvoll umgehen	Informationsrecherchen zielgerichtet durchführen und dabei Suchstrategien anwenden	Kommunikations- und Kooperationsprozesse mit digitalen Werkzeugen zielgerichtet gestalten sowie mediale Produkte und Informationen teilen	Medienprodukte adressatengerecht planen, gestalten und präsentieren; Möglichkeiten des Veröffentlichens und Teilens kennen und nutzen	Die Vielfalt der Medien, ihre Entwicklung und Bedeutungen kennen, analysieren und reflektieren	Grundlegende Prinzipien und Funktionsweisen der digitalen Welt identifizieren, kennen, verstehen und bewusst nutzen
1.2 Digitale Werkzeuge	2.2 Informationsauswertung	3.2 Kommunikations- und Kooperationsregeln	4.2 Gestaltungsmittel	5.2 Meinungsbildung	6.2 Algorithmen erkennen
Verschiedene digitale Werkzeuge und deren Funktionsumfang kennen, auswählen sowie diese kreativ, reflektiert und zielgerichtet einsetzen	Themenrelevante Informationen und Daten aus Medienangeboten filtern, strukturieren, umwandeln und aufbereiten	Regeln für digitale Kommunikation und Kooperation kennen, formulieren und einhalten	Gestaltungsmittel von Medienprodukten kennen, reflektiert anwenden sowie hinsichtlich ihrer Qualität, Wirkung und Aussageabsicht beurteilen	Die interessengeleitete Setzung und Verbreitung von Themen in Medien erkennen sowie in Bezug auf die Meinungsbildung beurteilen	Algorithmische Muster und Strukturen in verschiedenen Kontexten erkennen, nachvollziehen und reflektieren
1.3 Datenorganisation	2.3 Informationsbewertung	3.3 Kommunikation und Kooperation in der Gesellschaft	4.3 Quellendokumentation	5.3 Identitätsbildung	6.3 Modellieren und Programmieren
Informationen und Daten sicher speichern, wiederfinden und von verschiedenen Orten abrufen; Informationen und Daten zusammenfassen, organisieren und strukturiert aufbewahren	Informationen, Daten und ihre Quellen sowie dahinterliegende Strategien und Absichten erkennen und kritisch bewerten	Kommunikations- und Kooperationsprozesse im Sinne einer aktiven Teilhabe an der Gesellschaft gestalten und reflektieren; ethische Grundsätze sowie kulturell-gesellschaftliche Normen beachten	Standards der Quellenangaben beim Produzieren und Präsentieren von eigenen und fremden Inhalten kennen und anwenden	Chancen und Herausforderungen von Medien für die Realitätswahrnehmung erkennen und analysieren sowie für die eigene Identitätsbildung nutzen	Probleme formalisiert beschreiben, Problemlösestrategien entwickeln und dazu eine strukturierte, algorithmische Sequenz planen; diese auch durch Programme umsetzen und die gefundene Lösungsstrategie beurteilen
1.4 Datenschutz und Informationssicherheit	2.4 Informationskritik	3.4 Cybergewalt und -kriminalität	4.4 Rechtliche Grundlagen	5.4 Selbstregulierte Mediennutzung	6.4 Bedeutung von Algorithmen
Verantwortungsvoll mit persönlichen und fremden Daten umgehen; Datenschutz, Privatsphäre und Informationssicherheit beachten	Unangemessene und gefährdende Medieninhalte erkennen und hinsichtlich rechtlicher Grundlagen sowie gesellschaftlicher Normen und Werte einschätzen; Jugend- und Verbraucherschutz kennen und Hilfs- und Unterstützungsstrukturen nutzen	Persönliche, gesellschaftliche und wirtschaftliche Risiken und Auswirkungen von Cybergewalt und -kriminalität erkennen sowie Ansprechpartner und Reaktionsmöglichkeiten kennen und nutzen	Rechtliche Grundlagen des Persönlichkeit- (u.a. des Bildrechts), Urheber- und Nutzungsrechts (u.a. Lizenz) überprüfen, bewerten und beachten	Medien und ihre Wirkungen beschreiben, kritisch reflektieren und deren Nutzung selbstverantwortlich regulieren; andere bei ihrer Mediennutzung unterstützen	Einflüsse von Algorithmen und Auswirkung der Automatisierung von Prozessen in der digitalen Welt beschreiben und reflektieren



Einbindung in den Medienkompetenzrahmen

- Grundlegende Prinzipien und Funktionsweisen der digitalen Welt identifizieren, kennen, verstehen und bewusst nutzen
- Algorithmische Muster und Strukturen in verschiedenen Kontexten erkennen, nachvollziehen und reflektieren
- Probleme formalisiert beschreiben, Problemlösestrategien entwickeln und dazu eine strukturierte, algorithmische Sequenz planen; diese auch durch Programmieren umsetzen und die gefundenen Lösungstechniken beurteilen.
- Einflüsse von Algorithmen und Auswirkung der Automatisierung von Prozessen in der digitalen Welt beschreiben und reflektieren



(OER)-Begleitmaterialien



<https://sensebox.de/de/material>

Ampel

Es soll eine Ampel simuliert werden. Mit einem Button kann man die Ampel umschalten.

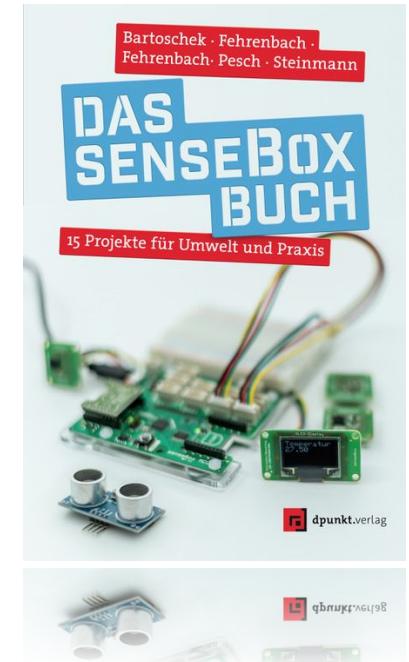
Materialien

- senseBox MCU
- rote LED
- gelbe LED
- grüne LED
- 3x 470Ω Widerstand
- Button
- 10kΩ (100000) Widerstand
- 2x senseBox JST-Adapterkabel

Aufbau

Hardwarekonfiguration

Um alle Komponenten anzuschließen benötigt Du zwei JST-Adapterkabel. Das erste wird an Digital A (also den digitalen Pins 1 und 2) angeschlossen, das zweite an Digital B (plus den digitalen Pins 3 und 4) angeschlossen. Am Kabel in Digital A werden die rote und die gelbe LED angeschlossen, am Kabel in Digital B die grüne LED und der Button.

A detailed circuit diagram on a breadboard. It shows a red LED connected to digital pin 1, a yellow LED connected to digital pin 2, and a green LED connected to digital pin 3. A 10kΩ pull-down resistor is connected between digital pin 4 and ground. A push-button switch is connected between digital pin 4 and digital pin 2. Three 470Ω resistors are connected between digital pins 1, 2, and 3 and ground. Two JST-adapter cables are used to connect the breadboard to the senseBox MCU.

Begleitmaterialien

senseBox Material

BOOKS

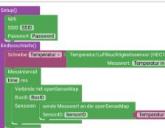
Unsere Books sind das Herzstück der Dokumentation. Hier findet man die Grundlagen, Hilfe und erste Beispiele zur senseBox. Im folgenden werden die aktuellen Ressourcen zur aktuellen Version der senseBox vorgestellt



:edu Book
senseBox - Team



:home Book
senseBox - Team



:blockly Book
senseBox - Team

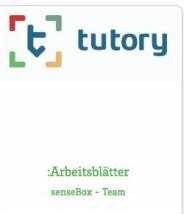


:openSenseMap Book
senseBox - Team

MATERIALIEN FÜR DEN EINSATZ IN DER LEHRE



:Lernkarten
senseBox - Team

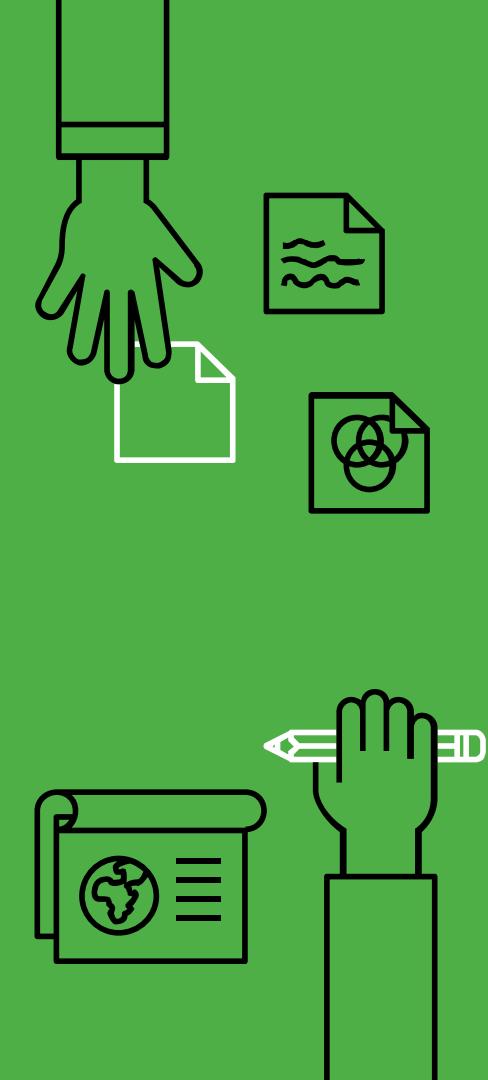


:Arbeitsblätter
senseBox - Team

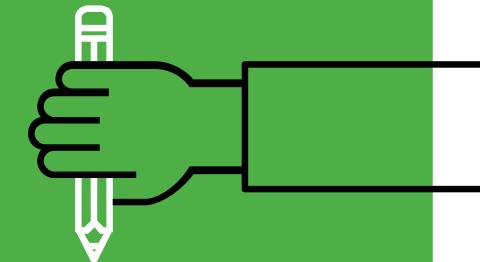
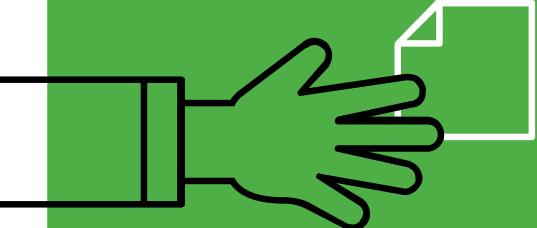


:ETH Zürich
ETH - Zürich

<https://sensebox.de/de/material>

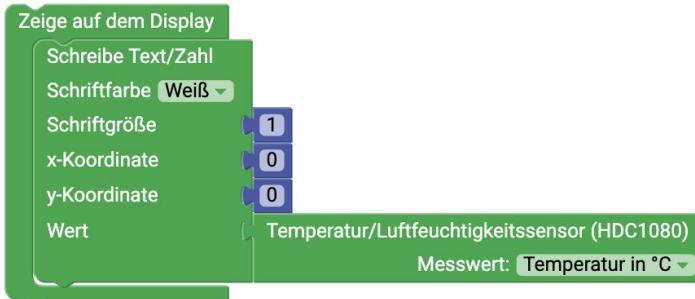


3. Blockly für senseBox

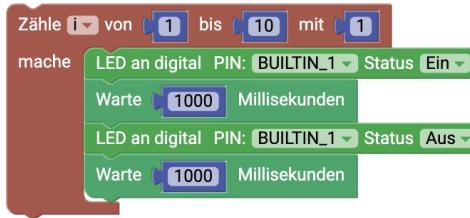


Blockly für senseBox

- Visuelle Programmieroberfläche
- Online, frei verfügbar
- Keine Installation notwendig
- So nah am Quellcode wie möglich
- Basierend auf Google Blockly



Initialisiere WLAN Access Point
SSID SSID



```
{ } Arduino Quellcode
#include "SenseBoxMCU.h"
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

int i;

Microphone microphone(A1);

#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

HDC1080 hdc;

void setup() {
    pinMode(7, OUTPUT);
    senseBox10.powerI2C(true);
    delay(2000);
    display.begin(SSD1306_SWITCHCAPVCC, 0x3D);
    display.display();
    delay(100);
    display.clearDisplay();
    hdc.begin();
}

void loop() {
    for (i = 1; i <= 10; i++) {
        digitalWrite(7,HIGH);
        delay(1000);
        digitalWrite(7,LOW);
        delay(1000);
    }

    if (microphone.getValue() < 0) {
        display.setCursor(0,0);
        display.setTextSize(1);
        display.setTextColor(WHITE,BLACK);
        display.println(hdc.getTemperature());
        display.display();
    }
}
```

```
{} Arduino Quellcode

#include "SenseBoxMCU.h"
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

int i;

Microphone microphone(A1);

#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

HDC1080 hdc;

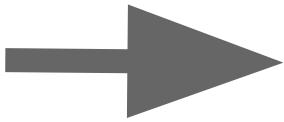
void setup() {
    pinMode(7, OUTPUT);
    senseBoxIO.powerI2C(true);
    delay(2000);
    display.begin(SSD1306_SWITCHCAPVCC, 0x3D);
    display.display();
    delay(100);
    display.clearDisplay();
    hdc.begin();
}

void loop() {
    for (i = 0; i <= 10; i++) {
        digitalWrite(7,HIGH);
        delay(1000);
        digitalWrite(7,LOW);
        delay(1000);
    }

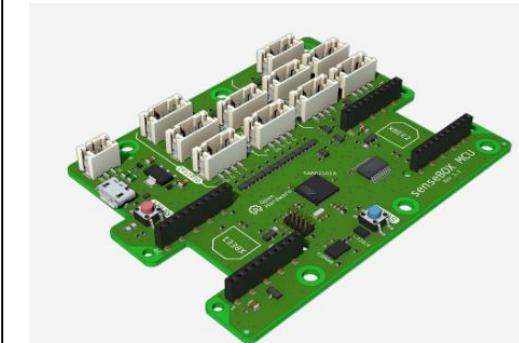
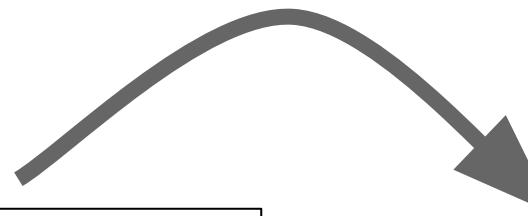
    if (microphone.getValue() < 0) {

        display.setCursor(0,0);
        display.setTextSize(1);
        display.setTextColor(WHITE,BLACK);
        display.println(hdc.getTemperature());
        display.display();
    }
}
```

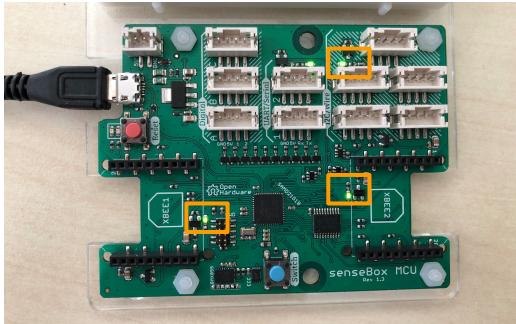
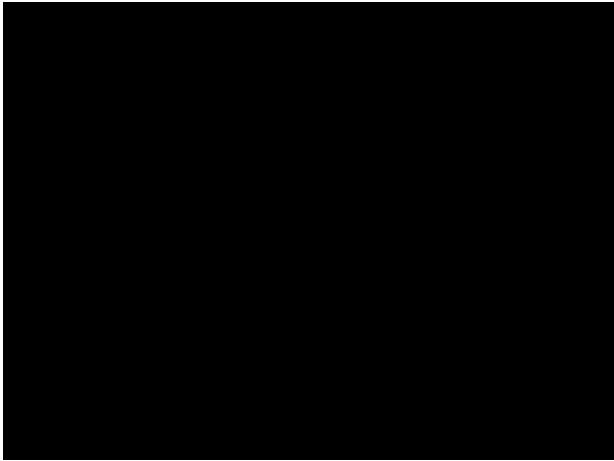
Compiler



1010101110101010101010111111
000011010101010001100011101
01011011100011101010101111010
1010001011001110110111010110
000101010101110101010100111
011010101100001010101010101
01011111010101010111011010101
0101010111101010101010100101
001101010001001001010101111
101010101010111110101010101
01111110101000000101010101



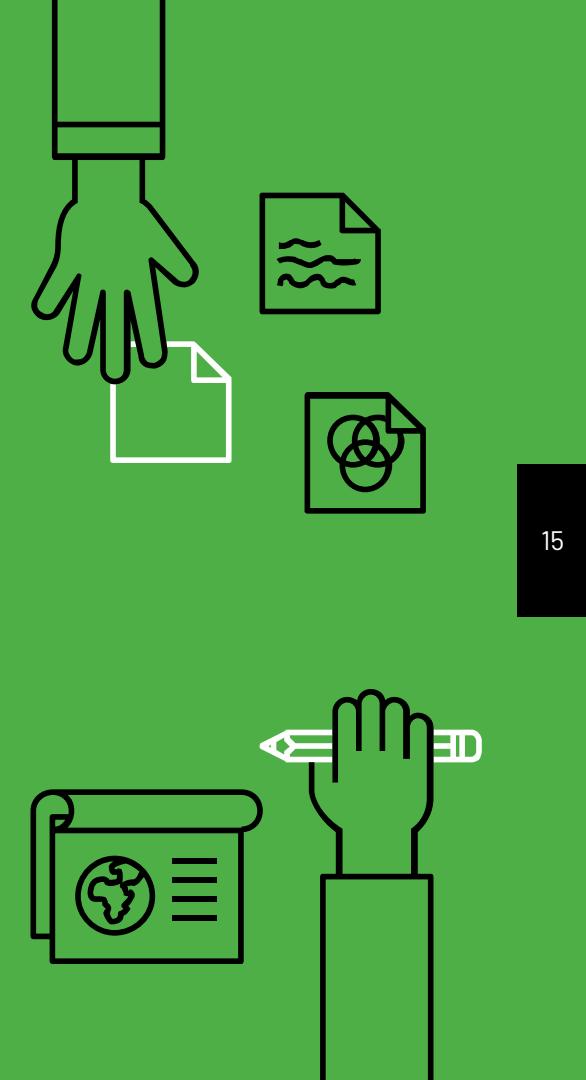
Anschließen und starten



Programm-Modus



Lern-Modus

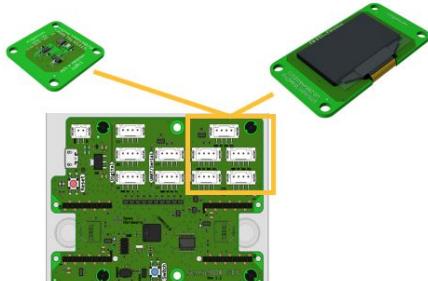


Und los geht's!

1. Gib blockly.sensebox.de in deinen Browser ein und wähle die senseBox MCU aus.
2. Verbinde die senseBox MCU mit Hilfe des USB-Kabels mit deinem Computer.
3. Nun kannst du mit der Programmierung starten.
4.  Mit dieser Schaltfläche überträgst du dein Programm an die MCU.

Anschließen von Display und Lichtsensor

Der Lichtsensor und das Display werden beide an die I2C/Wire-Ports auf der senseBox MCU angeschlossen.



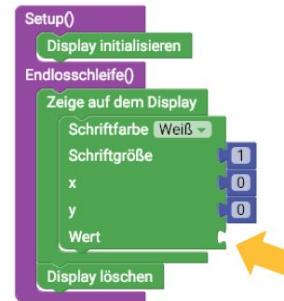
Messwerte auslesen und anzeigen

Um den Lichtsensor auszulesen benötigst du folgenden Block, der unter senseBox Sensoren zu finden ist:

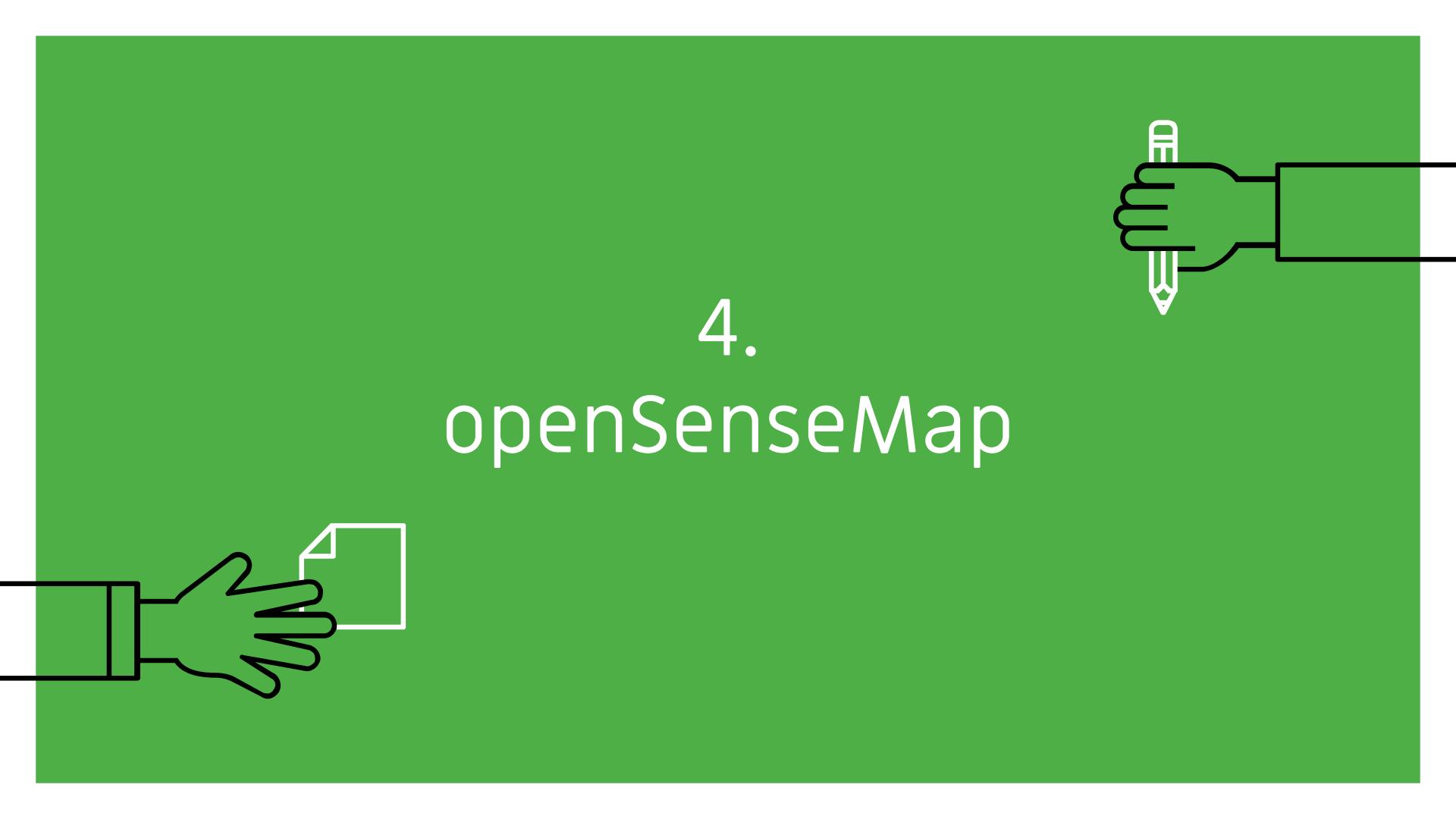
Licht Sichtbar + UV Sensor
Messwert: Beleuchtungsstärke in Lux ▾

Dieser Block funktioniert allerdings nicht allein. Er benötigt immer einen Empfänger, der den Messwert verarbeitet. Die einfachste Möglichkeit ist das Display.

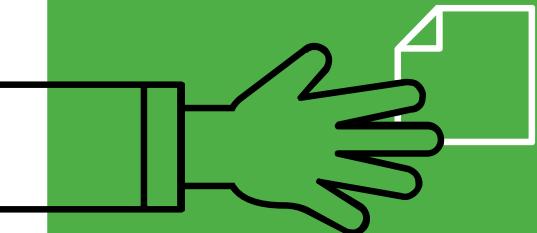
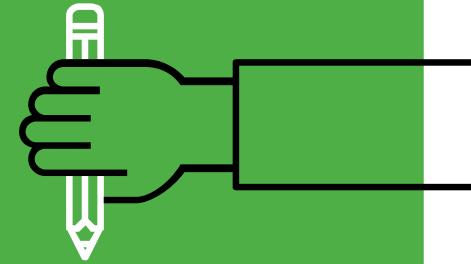
Um das Display zu programmieren benötigst du vier Blöcke die du unter senseBox Ausgabe → Display findest:



Verknüpfst du jetzt den Lichtsensor-Block mit der "Wert"-Schnittstelle des Display-Blocks werden deine Messwerte auf dem Display angezeigt.



4. openSenseMap

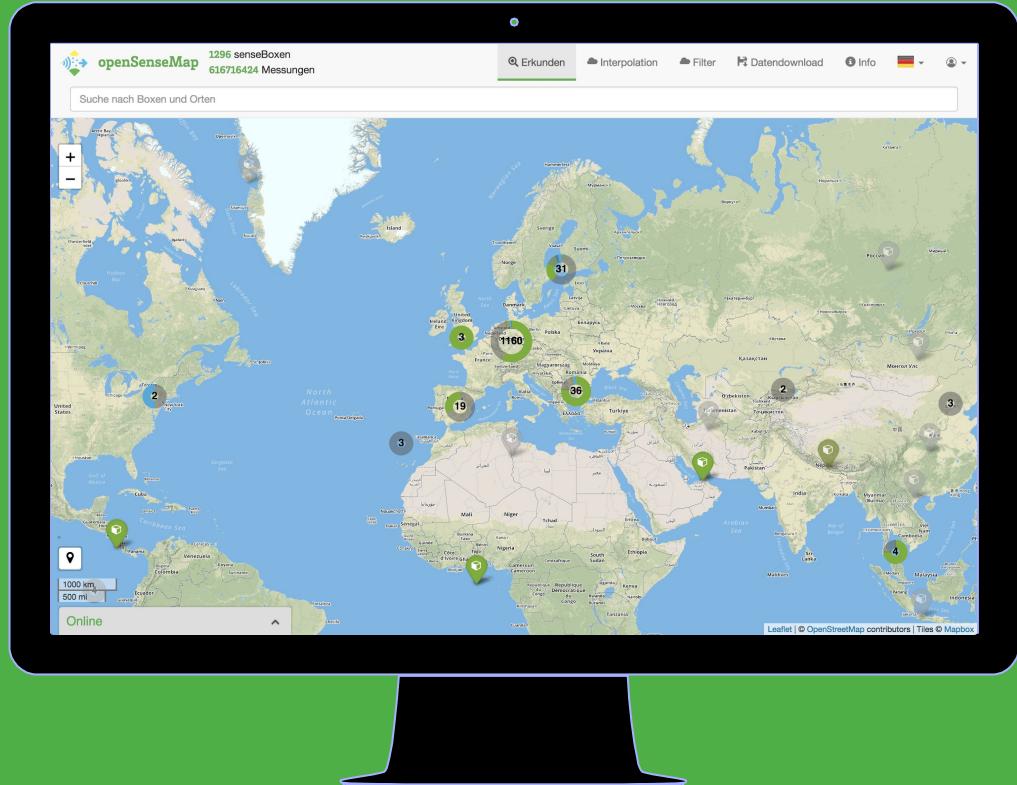


<https://opensensemap.org>

openSenseMap

- Plattform für offene Sensordaten, an der jeder teilnehmen kann
- REST API
- Ca. 4000 Messstationen
- Mobile Messstationen
- Download und Interpolationen möglich
- ArcGIS online kompatibel

Live Demo!



openSenseMap API

Get one senseBox
Get all senseBoxes
Post new senseBox
Update a senseBox
Mark a senseBox and its measurements for deletion
Download the Arduino script for your senseBox
Get locations of a senseBox
Interpolation
Get a Inverse Distance Weighting Interpolation as FeatureCollection
Measurements
Post new measurement
Get the 10000 latest measurements for a sensor
Delete measurements of a sensor
Get latest measurements for a phenomenon as CSV
Get latest measurements of a senseBox
Post multiple new measurements
Misc
Get some statistics about the database
print all routes
Statistics
Compute basic descriptive statistics over specified time windows
Users
Register new
Delete user, all of its boxes and all of its boxes measurements
Get details
Refresh Authorization
Sign in

Boxes

Boxes - Get one senseBox

GET

<https://api.opensensemaps.org/boxes/:senseBoxId?format=:format>

Parameter

Feld	Typ	Beschreibung
format	optional	String The format the sensor data is returned in. If <code>geojson</code> , a GeoJSON Point Feature is returned. Standardwert: <code>json</code> Erlaubte Werte: <code>json</code> , <code>geojson</code>
senseBoxId	String	the ID of the senseBox you are referring to.

Example data on success:

```
{  
    "_id": "57000b8745fd40c8196ad04c",  
    "createdAt": "2016-06-02T11:22:51.817Z",  
    "exposure": "outdoor",  
    "groupTag": "",  
    "image": "57000b8745fd40c8196ad04c.png?1466435154159",  
    "currentLocation": {  
        "coordinates": [  
            7.64568,  
            51.962372  
        ],  
        "timestamp": "2016-06-02T11:22:51.817Z",  
        "type": "Point"  
    },  
    "name": "0ststr/Mauritzsteinpfad",  
    "sensors": [  
        {  
            "_id": "57000b8745fd40c8196ad04e",  
            "lastMeasurement": {  
                "value": "0",  
                "createdAt": "2016-11-11T21:22:01.675Z"  
            },  
            "sensorType": "VEML6070",  
            "title": "UV-Intensität",  
            "unit": "µW/cm²"  
        }  
    ]  
}
```



Link Liste

senseBox:

<https://sensebox.de>

senseBox Dokumentation:

<https://sensebox.de/de/material>

Lernkarten:

https://sensebox.de/docs/senseBox_Karten.pdf

Blockly für senseBox:

<https://blockly.sensebox.de>

openSenseMap:

<https://opensensemap.org/>

openSenseMap Dokumentation:

<https://docs.opensensemap.org>

Arduino:

<https://www.arduino.cc/>

Board Support Package:

https://github.com/sensebox/senseBoxMCU-core/raw/master/package_sensebox_index.json

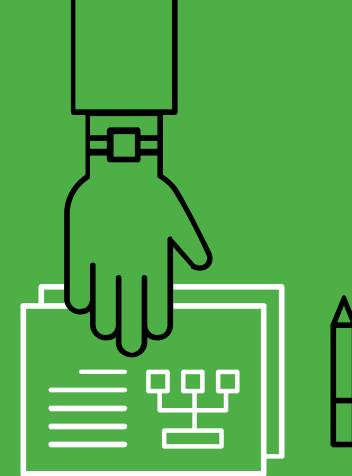
openRoberta:

<https://lab.open-roberta.org/>



Kontakt:

Institut für Geoinformatik
Heisenbergstr. 2
48149 Münster
info@sensebox.de



21

