openSenseMap

Dokumentation

Inhaltsverzeichnis

openSenseMap

Die openSenseMap (OSeM) ist eine Webplattform, auf welcher diverse standortbezogene Sensordaten hochgeladen und visualisiert werden können. Auf der Plattform lassen sich Stationen registrieren, welche die Daten eines oder mehrerer Sensoren übertragen.

Neben einer Zeitreihenvisualisierung der Daten, ist es auch möglich diese nach verschiedenen Kriterien zu Filtern und räumliche Interpolationen zu errechnen.

Sämtliche Sensordaten stehen unter der Public Domain Dedication and License 1.0[1] zum Download zur Verfügung, und können frei verwendet werden.

Sowohl die openSenseMap als auch die zugehörige API ist Open Source Software. Quellcode und Issuetracker sind hier zu finden:

- openSenseMap[2]
- openSenseMap API[3]

---

1. http://opendatacommons.org/licenses/pddl/summary/ ↩
2. https://github.com/sensebox/OpenSenseMap ↩
3. https://github.com/sensebox/OpenSenseMap-API ↩

Registrierung auf der OSeM

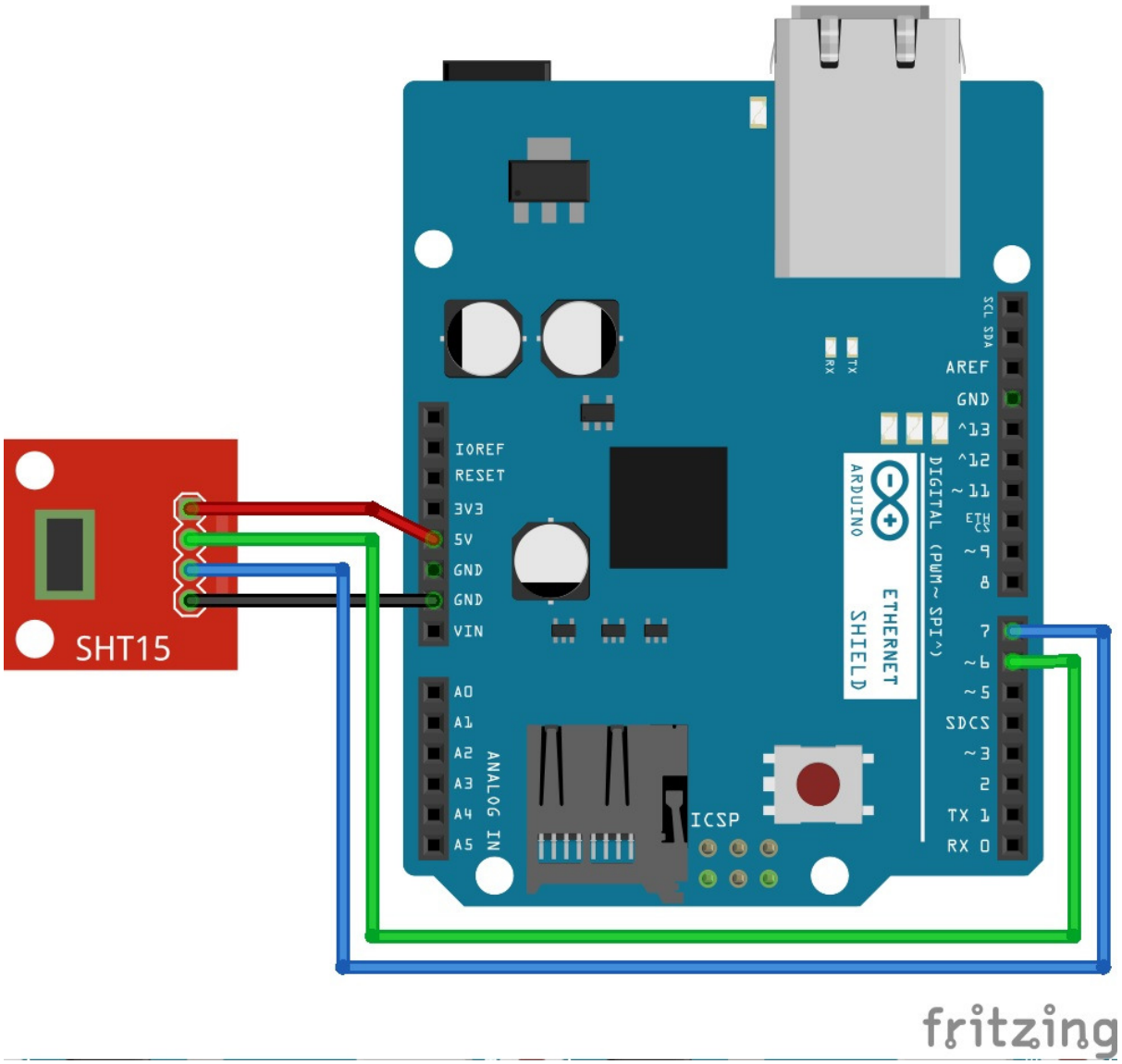
Manuelle Konfiguration eines Sensors auf der OSeM

In dieser Anleitung wird beispielhaft die Anbindung eines SHT15 Sensors in das OSeM Netzwerk beschrieben. Für die Datenverarbeitung bzw. -übertragung nutzen wir einen Arduino Uno mit Arduino Ethernet Shield. Die REST Schnittstelle bietet aber auch jeder anderen programmierbaren und mit dem Internet verbundenen Messstation die Möglichkeit, Messungen auf der OSeM zu veröffentlichen und zu teilen.

Materialien

- Arduino Uno R3
- Arduino Ethernet Shield R3
- Sparkfun SHT15 Breakout[1]

Aufbau



- VCC zu 5V

- DATA zu Pin 6
- SCK zu Pin 7
- GND zu Arduino GND

SHT1x Bibliothek

Für Sensoren der SHT1x Serie von Sensirion gibt es bereits eine einfache Arduino-Schnittstelle. Lade die Bibliothek von Github herunter[2] und entpacke sie in deinen `Arduino/libraries` Ordner. Importiert die Bibliothek wie gehabt in deinen Arduino Sketch, und definiere zusätzlich den Daten- und Taktpin entsprechend der Verkabelung in obiger Abbildung. Danach kannst du eine Verbindung zum Sensor herstellen:

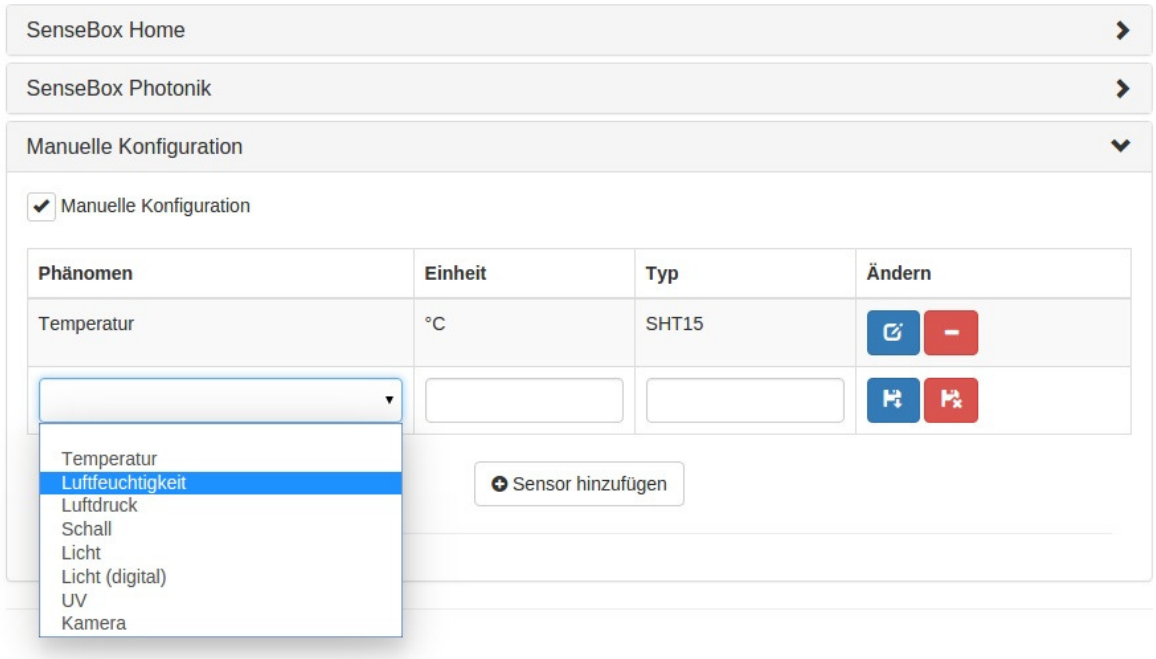
```
#include<sht1x.h>
#define dataPin 6
#define clockPin 7
SHT1x sht1x(dataPin, clockPin);
```

Nun lassen sich über zwei Funktionen die Temperatur in Grad Celsius, sowie die relative Luftfeuchte in Prozent als Gleitkommazahl abspeichern:

```
float temp = sht1x.readTemperatureC();
float humi = sht1x.readHumidity();
```

Manuelle Registrierung

Um einen Sensor mit der OSeM zu verbinden, musst du ihn zuerst dort registrieren[3]. Falls du dabei keinen der senseBox-Bausätze nutzt, muss in Schritt 4 der Registrierung die manuelle Konfiguration gewählt werden. Wie unten in der Abbildung dargestellt, wird dort für jedes gemessene Phänomen ein neuer Sensor angelegt:



openSenseMap API

Eine REST Schnittstelle regelt den Zugang zur Datenbank auf dem OSeM Server. Intern ist jede Messstation mit ihren Sensoren (bzw. Phänomenen) verknüpft, die bei der Registrierung angegeben wurden. In unserem Falle haben wir eine senseBox ID für die Station, sowie jeweils eine Sensor ID für Temperatur- und Luftfeuchtigkeitsmessungen bei der Registrierung generiert. Die IDs werden dir nach der Registrierung per Mail zugeschickt. Jede Messung wird dann über das HTTP Protokoll mit der `POST` Operation an den Server gesendet. Dazu muss eine eindeutige URI angegeben werden die wie folgt aufgebaut ist:

```
https://api.opensensemap.org/senseBoxID/SensorID
```

> Hinweis: Sollte der verwendete Microcontroller nicht HTTPS-kompatibel sein, gibt es derzeit noch eine HTTP Schnittstelle: *http://opensensemap.org:8000/senseBoxID/sensorID*

Jede Messung wird einzeln im JSON Format über das `value` -Attribut an den Server gesendet. Angenommen, wir wollen von unserer Station (ID 1234) einen Messwert des Thermometers (ID abcd) von `22,5` an den OSeM Server schicken, dann sähe der vollständige HTTP POST Request folgendermaßen aus:

```
POST /boxes/1234/abcd HTTP/1.1
Host:opensensemap.org
Content-Type: application/json
Connection: close
Content-Length: 14

{“value“:22.5}
```

> Achtung: Ab Zeile 7 werden die JSON-Daten gesendet. Der Zeilenumbruch (`\n`) in Zeile 6 ist notwendig um die Operation korrekt auszuführen.

Arduino OSeM Client

Nach der Registrierung wird ein Arduino Sketch generiert, den du als Anhang in einer Bestätigungsmail zugeschickt bekommst. Diesen Sketch musst du noch anpassen, indem die SHT1x Bibliothek eingefügt, sowie die benötigen Variablen und eine Sensorinstanz erstellt werden:

```
#include <SPI.h>
#include <Ethernet.h>

#include<sht1x.h>
#define dataPin 6
#define clockPin 7
SHT1x sht1x(dataPin, clockPin);

//senseBox ID
#define senseBox_ID "1234"
//Sensor IDs
#define TEMPERATURESENSOR_ID "abcd"
#define HUMIDITYSENSOR_ID "efgh"
```

Innerhalb der if-Anweisung in der `loop` -Funktion, musst du nacheinander die Sensoren auslesen und mit der Hilfsfunktion `postFloatValue()` hochladen.

```
void loop()
{
 //Upload der Daten mit konstanter Frequenz
 if (millis() - oldTime >= postInterval)
 {
  oldTime = millis();
  temperature = sht1x.readTemperatureC();
```

```
    postFloatValue(temperature, 1, temperatureSensorID);
    humidity = sht1x.readHumidity();
    postFloatValue(humidity, 0, humiditySensorID);
  }
}
```

Falls du ein Ethernet Modul nutzt, welches nicht mit der Ethernet Bibliothek kompatibel ist muss der Sketch entsprechend angepasst werden. Solltest du weitere Fragen dazu haben, kannst du dich auch direkt an unseren Support wenden[4].

---

1. https://www.sparkfun.com/products/8257 ↩
2. https://github.com/practicalarduino/SHT1x ↩
3. https://opensensemap.org/register ↩
4. mailto:support@sensebox.de ↩

Bearbeiten einer Station

Von bereits registrierten Stationen lassen sich sämtliche Angaben nachträglich ändern.

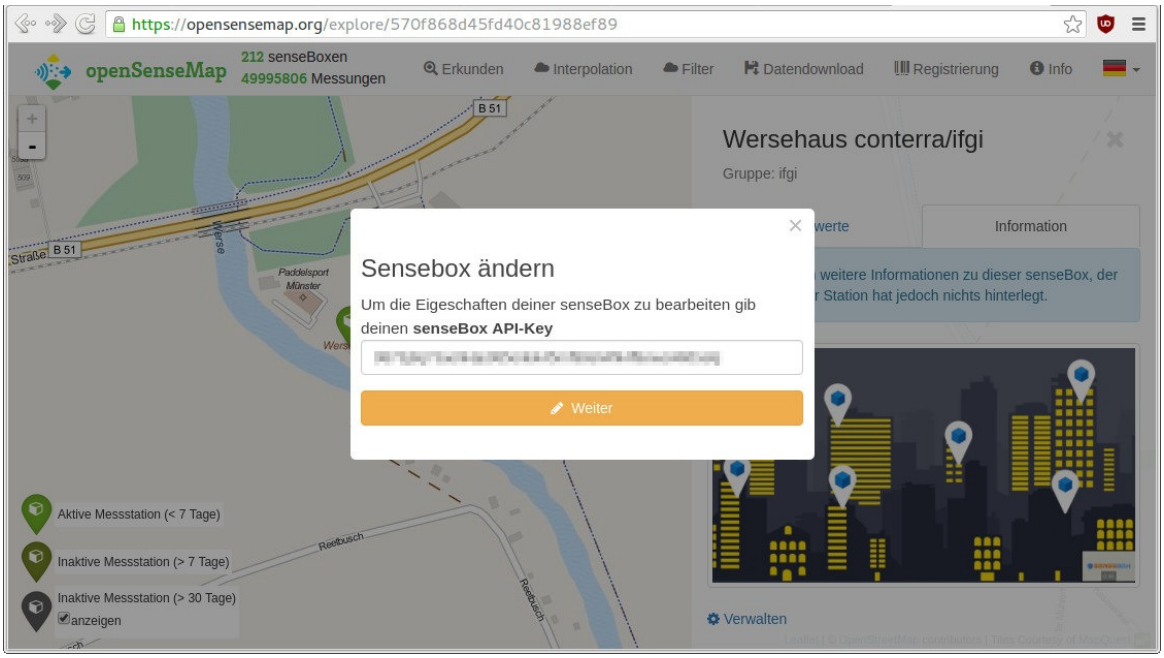
Hierzu wird der bei der Registrierung erhaltene API-Key benötigt!

1. Wähle deine Station auf der openSenseMap durch Klick auf den entsprechenden Marker auf der Karte aus.

   Alternativ kannst du auch in der folgenden URL `<senseBox-ID>` durch deine senseBox-ID (nicht der API-Key!) ersetzen: `https://opensensemap.org/explore/<senseBox-ID>`

2. Wähle in der rechten Sidebar den Tab "Information", und klicke unten den Button "Verwalten".

3. Gebe deinen API-Key im erscheinenden Dialog ein.



4. Gebe deine Änderungen im sich öffnenden Formular ein. Neben Änderungen an Metadaten, Standort und Foto ist es auch möglich die Sensor-Konfiguration zu ändern.

   > Hinweis: Falls du einen neuen Sensor hinzugefügt hast und den aktualisierten Arduino-Sketch herunterladen willst, ist dies erst möglich, wenn die Seite nach dem Speichern neu geladen, und der API-Key erneut eingegeben wurde.

5. Klicke im oberen Teil des Dialogs auf "Speichern" um deine Änderungen zu speichern, oder "Abbrechen" um sie zu verwerfen.

Löschen einer Station

Folge den Schritten unter "Bearbeiten einer Station"[1] und gebe unter "Sensebox löschen" `DELETE` in das Textfeld ein.

> Achtung: Hierdurch werden neben deiner senseBox alle hinterlegten Sensordaten unwiderruflich entfernt!

> [1]. Siehe 1.3 Bearbeiten einer Station > bearbeiten-einer-station ↩

Datendownload

Es bestehen mehrere Möglichkeiten Sensordaten von der openSenseMap herunterzuladen. Abhängig von der Fragestellung bietet sich je eine Option an.

Daten zu einer Box

Unter https://archive.opensensemap.org wird ein Archiv für sämtliche Messungen in der openSenseMap Datenbank geführt. Hier sind nach Tag und Box gegliederte Messwerte als CSV beziehungsweise ZIP-Archiv verfügbar.

Daten zu einem Phänomen

Unter dem Reiter "Datendownload" sind Funktionen zum Herunterladen der Sensordaten zu finden.

Der Datendownload bezieht sich immer auf ein ausgewähltes Phänomen (z.B. Lufttemperatur), einen Zeitraum und eine Boundingbox. Die Boundingbox bezeichnet die räumliche Auswahl der Stationen, und wird automatisch durch den aktuell sichtbaren Kartenausschnitt bestimmt.

> Achtung: Je nach Auswahl der Filterparameter kann der Download sehr groß werden (mehrere 100MB)!

Erweitertes Filtern

Zusätzlich zu den zeitlichen und räumlichen Filtern unter "Datendownload" lässt sich die Stationsauswahl weiter unter dem Reiter "Filter" einschränken. Wie das geht ist im Kapitel Datenanalyse[1] beschrieben.

Formate

Derzeit wird nur das Datenformat CSV unterstützt, welches problemlos mit Tabellenkalkulations-Tools wie Excel verarbeitet werden kann.

Jede Zeile enthält eine Messung einer senseBox mit dem ausgewählten Phänomen. Der Messwert (`value`), Standort des Sensors (`lat`, `lng`, Referenzsystem WGS84) und ein Zeitstempel (`createdAt`) sind in je einer Spalte angegeben:

```
createdAt;value;lat;lng
2016-09-20T10:05:49.581Z;18.70;7.64568;51.962372
2016-09-20T10:00:52.689Z;18.62;7.64568;51.962372
2016-09-20T09:55:54.282Z;18.47;7.64568;51.962372
....
```

API-Download

Falls die beiden genannten Möglichkeiten nicht flexibel genug sind, können über die REST API unter `/boxes/data`[2] auch komplexe Anfragen gestellt werden.

Für solche anfragen bietet sich das Kommandozeilenwerkzeug `curl` an. Unter Linux ein Terminal öffnen und beispielsweise folgenden Befehl eingeben, um sämtliche Temperatur-Messungen im geographischen Bereich 51°N - 52°N, 7°E - 8°E in die Datei `measurements.csv` herunterzuladen:

```
curl "https://api.opensensemap.org/boxes/data?phenomenon=Temperatur&bbox=7,51,8,52" > measurements.csv
```

Andere geeignete Parameter (Zeitraum, Box-IDs, ...) lassen sich der verlinkten API-Dokumentation entnehmen.

---

[1]. Siehe 1.5 Datenanalyse ↩

[2]. Siehe 1.6 REST API > get-latest-measurements-for-a-phenomenon-as-csv- ↩

Datenanalyse

Filter

Die angezeigten senseBoxen lassen sich nach verschiedenen Kriterien auswählen.

Hierzu können unter dem Reiter "Filter" die entsprechenden Angaben in der Sidebar gemacht werden. Nach einem Klick auf "Filter anwenden" werden die Boxen gefiltert (dies kann je nach Auswahl einen Augenblick dauern).

Anschließend wird unterhalb der Filtereinstellungen eine Auflistung der auf die Kriterien zutreffenden senseBoxen angezeigt.

Dieser Filter bezieht sich auch auf die anderen Datenanalyse-Funktionen Interpolation[1] und Datendownload[2]!

Interpolation

Im Reiter "Interpolation" lassen sich die Daten mehrerer senseBoxen räumlich interpolieren.

Dies ist nützlich um die räumlichen Unterschiede eines Phänomens auf der Karte sichtbar zu machen, oder um ungefähre (!) Werte in Regionen ableiten zu können, in welchen keine Sensoren vorhanden sind.

Vorraussetzung für diese Funktion ist, dass zuvor ein Filter auf einen Zeitraum und ein bestimmtes Mess-Phänomen gewählt wurde.

Es stehen zwei Interpolationsverfahren zur Verfügung: Inverse Distance Weighting (IDW) und Thiessen Polygone.

Nach Einstellung der Parameter (s.U.) wird die Interpolation auf unserem Server berechnet. Wenn die Kalkulation abgeschlossen ist, wird das Ergebnis als Heatmap in der Karte angezeigt.

Inverse Distance Weighting

`TODO`

Thiessen Polygone

`TODO`

---

1. Siehe 1.5 Datenanalyse > interpolation ↩

2. Siehe 1.4 Datendownload ↩

openSenseMap RESTful API

Die openSenseMap stellt eine API zur Abfrage von Daten zu Boxen und Messungen sowie zum Upload von Messungen unter der Adresse https://api.opensensemap.org/ zur Verfügung.

Die Dokumentation der API-Routen ist hier[1] zu finden.

---

[1]. https://docs.opensensemap.org ↩

Datenübertragung über MQTT

Diese Anleitung beschreibt die Möglichkeit, Messwerte über MQTT an die openSenseMap zu senden. Die openSenseMap ist in der Lage, sich als MQTT Client mit einem öffentlichen MQTT Broker zu verbinden. Einen eigenen MQTT Broker bietet die openSenseMap nicht an. Der openSenseMap MQTT Client verbindet sich, wenn nicht anders in den Verbindungseinstellungen angegeben, mit einer 13 Stelligen Id mit prefix `osem_` gefolgt von 8 zufälligen Ziffern und Buchstaben von A bis F.

Je registrierter senseBox müssen separate MQTT Einstellungen vorgenommen werden. Für eine Verbindung mit einem Broker können die folgenden Parameter angegeben werden. Alle angegeben Einstellungen werden in der Datenbank der openSenseMap gespeichert. Es bietet sich also an, eigene Zugangsdaten einzurichten.

URL

Die Adresse zum MQTT Broker. Sollte mit `mqtt://` oder `ws://` beginnen. Sollte der MQTT Broker Authentifizierung mittels Nutzername und Passwort benötigen, kann dieser in der URL kodiert werden. Die URL sollte dann wie folgt aussehen: `mqtt://username:password@hostname.of.mqtt.broker`

Topic

Das MQTT Topic unter dem die openSenseMap Nachrichten empfangen soll. Zum Beispiel `home/temperatures/outside`

Nachrichtenformat

Hier sollte zwischen `json` und `csv` ausgewählt werden. Die Formate entsprechen JSON-Array und csv dokumentiert in docs.opensensemap.org[1].

Dekodierungsoptionen

Erwartet ein JSON Objekt. Nur für Nachrichtenformat json: Erlaubt es, unter dem Schlüssel jsonPath einen JSONPath Ausdruck anzugeben, welches die Position der JSON kodierten Daten angibt. Beispiel: `{"jsonPath":"$.payload_fields"}`

Verbindungsoptionen

Erwartet ein JSON Objekt. Erlaubt es, dem MQTT Client Verbindungsoptionen zu übergeben. Die Schlüssel `keepAlive`, `reschedulePings`, `clientId`, `username` und `password` von https://github.com/mqttjs/MQTT.js#client[2] sind erlaubt.

---

[1]. https://docs.opensensemap.org/#api-Measurements-postNewMeasurements ↩

[2]. https://github.com/mqttjs/MQTT.js#client ↩