

# openSenseMap

## Dokumentation



---

# Table of Contents

<a href="#">Introduction</a>	1.1
<a href="#">Registration</a>	1.2
<a href="#">Editing a station</a>	1.3
<a href="#">Data download</a>	1.4
<a href="#">Data analysis</a>	1.5
<a href="#">REST API</a>	1.6

# senseBox



## openSenseMap

The openSenseMap (OSeM) is a webplatform which provides upload, visualisation and analysis of location-specific sensordata.

Stations may be registered on the platform, which host one or more Sensors on a specific location. Data up- & download is done via the restful [API](#).

## Features

- timeseries visualization for each phenomenon
- filtering by various parameters
- spatial interpolation
- data download with bounding box

All sensor data is available for download under the [Public Domain Dedication and License 1.0](#).

openSenseMap and it's API is open source software. Sourcecode and issuetracker are located here:

- [openSenseMap](#)
- [openSenseMap API](#)

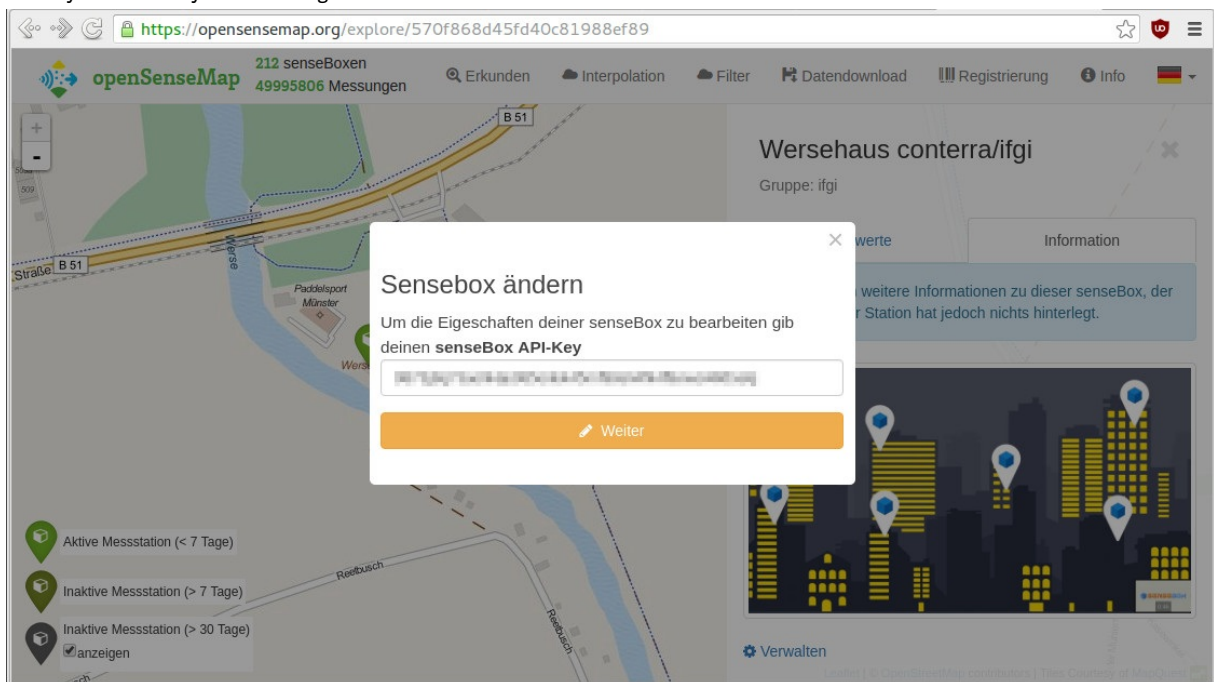
## Registration on the OSeM

## Modifying a station

All properties of a station may be changed after the registration.

To do this, authorization with the API-key is required, which was sent to you in the registration e-mail!

1. Select your station on the map by clicking on the marker on the map.
2. Select the tab "Info" in the sidebar and click "Manage".
3. Enter your API-key in the dialog.



4. Make your desired changes in the appearing form. You may edit metadata, geolocation, photo, as well as the stations sensor configuration.

Hint: If you have added a new sensor and want to download the updated arduino-sketch, a page-reload after saving is required.

5. Click "save" or "cancel" in the top of the dialog to apply or discard your changes.

## Deleting a station

Follow the steps under "[Modifying a station](#)", then type `DELETE` in the textfield "Delete senseBox" and confirm.

warning: All associated sensor data will be permanently deleted!

# Data download

Data analysis

Filter

Interpolation

The openSenseMap provides a REST API, which can be used to query & post senseBox metadata & measurements. The endpoint is <https://api.opensensemap.org/>.

This documentation can also be found [here](#) with an improved layout.

## openSenseMap API documentation

methods to manage senseBoxes and get/post measurements

### Boxes

- **method** GET Validate authorization
- **method** GET Get one senseBox
- **method** GET Get all senseBoxes
- **method** POST Post new senseBox
- **method** PUT Update a senseBox: Image and sensor names
- **method** DELETE Delete a senseBox and its measurements
- **method** GET Download the Arduino script for your senseBox

### Measurements

- **method** POST Post new measurement
- **method** GET Get latest measurements of a senseBox
- **method** GET Get the 10000 latest measurements for a sensor
- **method** GET,POST Get latest measurements for a phenomenon as CSV

### Misc

- **method** GET Get some statistics about the database

## Boxes

### Validate authorization **method** GET

Validate authorization through API key and senseBoxId. Will return status code 403 if invalid, 200 if valid.

GET /users/:senseBoxId

### Headers

Name	Type	Description
x-APIKey	String	the secret API key which corresponds to the <code>senseBoxId</code> parameter.

### Parameter



Name	Type	Description
:senseBoxId	String	the ID of the senseBox you are referring to.

## Success 200

Name	Type	Description
Response	String	ApiKey is valid

## Error Response

Error-Response:

```
HTTP/1.1 403 Forbidden  
{ "code": "NotAuthorized", "message": "ApiKey is invalid or missing" }
```

## Get one senseBox method GET

```
GET /boxes/:boxId
```

## Parameter

Name	Type	Description
format	String	the format the sensor data is returned in. Default value: json Allowed values: "json", "geojson"
:senseBoxId	String	the ID of the senseBox you are referring to.

## Success Response

Example data on success:

```
{
  "_id": "5386e44d5f08822009b8b614",
  "name": "PHOBOS",
  "boxType": "fixed",
  "sensors": [
    {
      "_id": "5386e44d5f08822009b8b615",
      "boxes_id": "5386e44d5f08822009b8b614",
      "lastMeasurement": {
        "_id": "5388d07f5f08822009b937b7",
        "createdAt": "2014-05-30T18:39:59.353Z",
        "updatedAt": "2014-05-30T18:39:59.353Z",
        "value": "584",
        "sensor_id": "5386e44d5f08822009b8b615",
      },
      "sensorType": "GL5528",
      "title": "Helligkeit",
      "unit": "Pegel"
    }
  ],
  "loc": [
    {
      "_id": "5386e44d5f08822009b8b61a",
      "geometry": {
        "coordinates": [
          10.5455893642828,
          49.61361673283691
        ],
        "type": "Point"
      },
      "type": "feature"
    }
  ]
}
```

## Get all senseBoxes method GET

With the optional `date` and `phenomenon` parameters you can find senseBoxes that have submitted data around that time, +/- 2 hours, or specify two dates separated by a comma.

```
GET /boxes?date=:date&phenomenon=:phenomenon&format=:format
```

### Parameter

Name	Type	Description
date	String	A date or datetime (UTC) where a station should provide measurements. Use in combination with <code>phenomenon</code> .
phenomenon	String	A sensor phenomenon (determined by sensor name) such as temperature, humidity or UV intensity. Use in combination with <code>date</code> .
format	String	the format the sensor data is returned in. Default value: json Allowed values: "json", "geojson"

## Post new senseBox method POST

Create a new senseBox. This method allows you to submit a new senseBox.

Along with the senseBox, an user is created which then owns the senseBox.

If you specify `mqtt` parameters, the openSenseMap API will try to connect to the MQTT broker specified by you. The parameter `messageFormat` tells the API in which format you are sending measurements in.

For `json`, the format is:

```
{  "sensorId": <value>,  "sensorId": [<value>,<createdAt>]  ... }
```

For `csv`, the format is:

```
sensorId,value sensorId,value,createdAt ...
```

POST /boxes

## JSON request body

Name	Type	Description
firstname	String	firstname of the user for the senseBox.
lastname	String	lastname of the user for the senseBox.
email	String	email of the user for the senseBox.
orderId	String	the apiKey of the user for the senseBox.
name	String	the name of this senseBox.
groupTag	String	the groupTag of this senseBox.
exposure	String	the exposure of this senseBox. Allowed values: "indoor","outdoor"
boxType	String	the type of the senseBox. Currently only 'fixed' is supported. Allowed values: "fixed"
sensors	Sensor[]	an array containing the sensors of this senseBox.
loc	Location	the location of this senseBox. Must be a GeoJSON Point Feature. (RFC7946)

## A single sensor for the nested Sensor parameter

Name	Type	Description
title	String	the title of the phenomenon the sensor observes.
unit	String	the unit of the phenomenon the sensor observes.
sensorType	String	the type of the sensor.
icon	String	the visual representation for the openSenseMap of this sensor.

## Settings for a senseBox connected through MQTT

Name	Type	Description
url	String	the url to the mqtt server.
topic	String	the topic to subscribe to.
messageFormat	String	the format the mqtt messages are in. Allowed values: "json", "csv"
decodeOptions	String	a json encoded string with options for decoding the message. 'jsonPath' for 'json' messageFormat.
connectionOptions	String	a json encoded string with options to supply to the mqtt client ( <a href="https://github.com/mqttjs/MQTT.js#client">https://github.com/mqttjs/MQTT.js#client</a> )

## Update a senseBox: Image and sensor names method PUT

Modify the specified senseBox.

```
PUT /boxes/:senseBoxId
```

### Headers

Name	Type	Description
x-APIKey	String	the secret API key which corresponds to the <code>senseBoxId</code> parameter.

### JSON request body

Name	Type	Description
description	String	the updated description of this senseBox.
image	String	the updated image of this senseBox encoded as base64 data uri.
name	String	the name of this senseBox.
groupTag	String	the groupTag of this senseBox.
exposure	String	the exposure of this senseBox. Allowed values: "indoor","outdoor"
boxType	String	the type of the senseBox. Currently only 'fixed' is supported. Allowed values: "fixed"
sensors	Sensor[]	an array containing the sensors of this senseBox.
loc	Location	the location of this senseBox. Must be a GeoJSON Point Feature. (RFC7946)

### A single sensor for the nested Sensor parameter

Name	Type	Description
title	String	the title of the phenomenon the sensor observes.
unit	String	the unit of the phenomenon the sensor observes.
sensorType	String	the type of the sensor.
icon	String	the visual representation for the openSenseMap of this sensor.

### Settings for a senseBox connected through MQTT

Name	Type	Description
url	String	the url to the mqtt server.
topic	String	the topic to subscribe to.
messageFormat	String	the format the mqtt messages are in. Allowed values: "json","csv"
decodeOptions	String	a json encoded string with options for decoding the message. 'jsonPath' for 'json' messageFormat.
connectionOptions	String	a json encoded string with options to supply to the mqtt client ( <a href="https://github.com/mqttjs/MQTT.js#client">https://github.com/mqttjs/MQTT.js#client</a> )

### Parameter

Name	Type	Description
:senseBoxId	String	the ID of the senseBox you are referring to.

## Error Response

Error-Response:

```
HTTP/1.1 403 Forbidden
{"code": "NotAuthorized", "message": "ApiKey is invalid or missing"}
```

## Delete a senseBox and its measurements method DELETE

```
DELETE /boxes/:senseBoxId
```

### Headers

Name	Type	Description
x-APIKey	String	the secret API key which corresponds to the <code>senseBoxId</code> parameter.

### Parameter

Name	Type	Description
:senseBoxId	String	the ID of the senseBox you are referring to.

## Error Response

Error-Response:

```
HTTP/1.1 403 Forbidden
{"code": "NotAuthorized", "message": "ApiKey is invalid or missing"}
```

## Download the Arduino script for your senseBox method GET

```
GET /boxes/:senseBoxId/script
```

### Headers

Name	Type	Description
x-APIKey	String	the secret API key which corresponds to the <code>senseBoxId</code> parameter.

### Parameter

Name	Type	Description
:senseBoxId	String	the ID of the senseBox you are referring to.

## Error Response

Error-Response:

```
HTTP/1.1 403 Forbidden
{"code": "NotAuthorized", "message": "ApiKey is invalid or missing"}
```

## Measurements

### Post new measurement method POST

Posts a new measurement to a specific sensor of a box.

```
POST /boxes/:senseBoxId/:sensorId
```

#### JSON request body

Name	Type	Description
value	String	the measured value of the sensor. Also accepts JSON float numbers.
createdAt	String	the timestamp of the measurement. Should be parseable by JavaScript.

#### Parameter

Name	Type	Description
:senseBoxId	String	the ID of the senseBox you are referring to.
:sensorId	String	the ID of the sensor you are referring to.

### Get latest measurements of a senseBox method GET

Get the latest measurements of all sensors of the specified senseBox.

```
GET /boxes/:senseBoxId/sensors
```

#### Parameter

Name	Type	Description
:senseBoxId	String	the ID of the senseBox you are referring to.

## Get the 10000 latest measurements for a sensor method GET

Get up to 10000 measurements from a sensor for a specific time frame, parameters `from-date` and `to-date` are optional. If not set, the last 48 hours are used. The maximum time frame is 1 month. If `download=true` `Content-disposition` headers will be set. Allows for JSON or CSV format.

```
GET /boxes/:senseBoxId/data/:sensorId?from-date=fromDate&to-date=toDate&download=true&format=json
```

### Parameter

Name	Type	Description
<code>from-date</code>	String	Beginning date of measurement data (default: 48 hours ago from now)
<code>to-date</code>	String	End date of measurement data (default: now)
<code>download</code>	String	If set, offer download to the user (default: false, always on if CSV is used) Allowed values: "true", "false"
<code>format</code>	String	Can be 'json' (default) or 'csv' (default: json) Default value: json Allowed values: "json", "csv"
<code>:senseBoxId</code>	String	the ID of the senseBox you are referring to.
<code>:sensorId</code>	String	the ID of the sensor you are referring to.
<code>separator</code>	String	Only for csv: the separator for csv. Possible values: <code>comma</code> for comma as separator, everything else: semicolon. Per default a semicolon is used.

## Get latest measurements for a phenomenon as CSV method GET, POST

Download data of a given phenomenon from multiple selected senseBoxes as CSV

```
GET, POST /boxes/data?boxid=:senseBoxIds&from-date=:fromDate&to-date:toDate&phenomenon=:phenomenon
```

### Parameter



Name	Type	Description
senseBoxIds	String	Comma separated list of senseBox IDs.
phenomenon	String	the name of the phenomenon you want to download the data for.
from-date	String	Beginning date of measurement data (default: 15 days ago from now)
to-date	String	End date of measurement data (default: now)
separator	String	Only for csv: the separator for csv. Possible values: <code>comma</code> for comma as separator, everything else: semicolon. Per default a semicolon is used.

## Misc

### Get some statistics about the database method GET

8 boxes, 13 measurements in the database, 2 measurements in the last minute

GET /stats

### Success Response

[8,13, 2]

[8, 13, 2]