

openSenseMap

Dokumentation



Table of Contents

| | |
|-----------------------------------|-----|
| Introduction | 1.1 |
| Registration | 1.2 |
| Editing a station | 1.3 |
| Data download | 1.4 |
| Data analysis | 1.5 |
| REST API | 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](#)

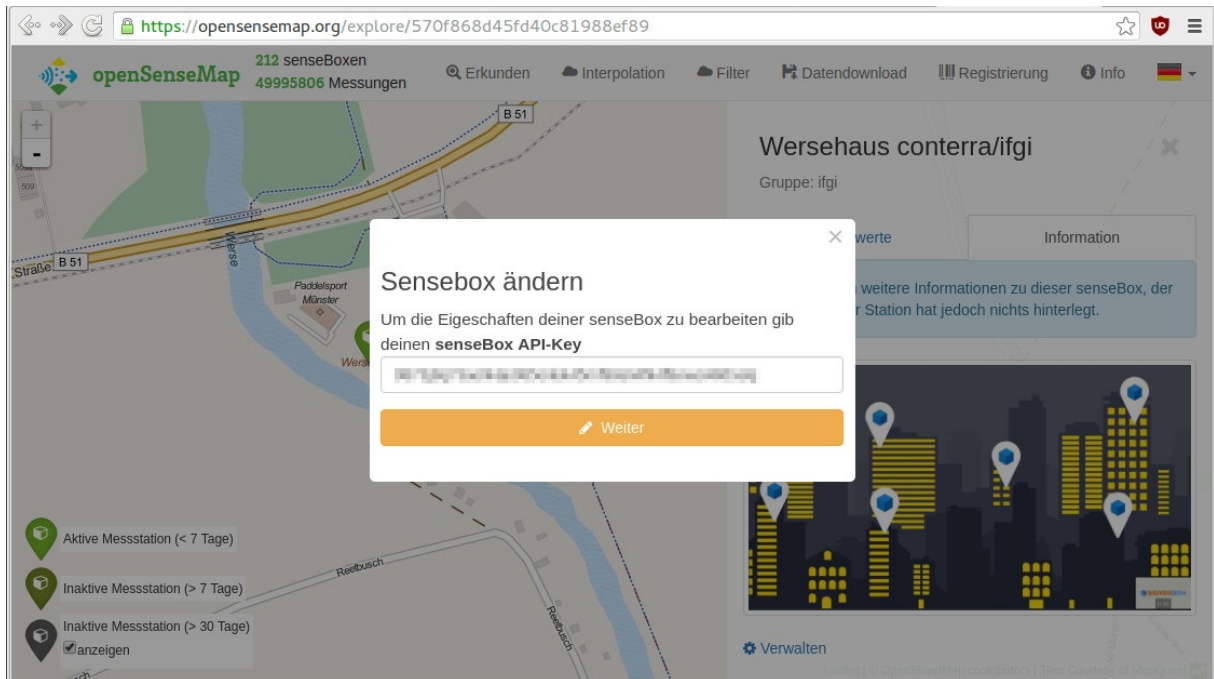
Registration on the OSeM

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

Interpolation

- **method** GET Get a Inverse Distance Weighting Interpolation as FeatureCollection

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** DELETE Delete measurements of a sensor
- **method** GET,POST Get latest measurements for a phenomenon as CSV
- **method** POST Post multiple new measurements

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 |
|-------------|--------|--|
| returnBox | String | optional if supplied and non-empty, returns the senseBox with the senseBoxId with hidden fields |
| :senseBoxId | String | the ID of the senseBox you are referring to. |

Success 200

| Name | Type | Description |
|----------|------|---|
| Response | json | <code>{"code": "Authorized", "message": "ApiKey is valid"}</code> |

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 | optional 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": "57000b8745fd40c8196ad04c",
  "boxType": "fixed",
  "createdAt": "2016-06-02T11:22:51.817Z",
  "exposure": "outdoor",
  "groupTag": "",
  "image": "57000b8745fd40c8196ad04c.png?1466435154159",
```

```
"loc": [
  {
    "geometry": {
      "coordinates": [
        7.64568,
        51.962372
      ],
      "type": "Point"
    },
    "type": "feature"
  }
],
"name": "Oststr/Mauritzsteinpfad",
"sensors": [
  {
    "_id": "57000b8745fd40c8196ad04e",
    "lastMeasurement": {
      "value": "0",
      "createdAt": "2016-11-11T21:22:01.675Z"
    },
    "sensorType": "VEML6070",
    "title": "UV-Intensität",
    "unit": "µW/cm²"
  },
  {
    "_id": "57000b8745fd40c8196ad04f",
    "lastMeasurement": {
      "value": "0",
      "createdAt": "2016-11-11T21:22:01.675Z"
    },
    "sensorType": "TSL45315",
    "title": "Beleuchtungsstärke",
    "unit": "lx"
  },
  {
    "_id": "57000b8745fd40c8196ad050",
    "lastMeasurement": {
      "value": "1019.21",
      "createdAt": "2016-11-11T21:22:01.675Z"
    },
    "sensorType": "BMP280",
    "title": "Luftdruck",
    "unit": "hPa"
  },
  {
    "_id": "57000b8745fd40c8196ad051",
    "lastMeasurement": {
      "value": "99.38",
      "createdAt": "2016-11-11T21:22:01.675Z"
    },
    "sensorType": "HDC1008",
    "title": "rel. Luftfeuchte",
    "unit": "%"
  },
  {
    "_id": "57000b8745fd40c8196ad052",
    "lastMeasurement": {
      "value": "0.21",
      "createdAt": "2016-11-11T21:22:01.675Z"
    },
    "sensorType": "HDC1008",
    "title": "Temperatur",
    "unit": "°C"
  },
  {
    "_id": "576996be6c521810002479dd",
    "sensorType": "WiFi",
    "unit": "dBm",
    "title": "Wifi-Stärke",
    "lastMeasurement": {
      "value": "-66",
      "createdAt": "2016-11-11T21:22:01.675Z"
    }
  }
]
```

```
    }
  },
  {
    "_id": "579f9eae68b4a2120069edc8",
    "sensorType": "VCC",
    "unit": "V",
    "title": "Eingangsspannung",
    "lastMeasurement": {
      "value": "2.73",
      "createdAt": "2016-11-11T21:22:01.675Z"
    },
    "icon": "osem-shock"
  }
],
"updatedAt": "2016-11-11T21:22:01.686Z"
}
```

Get all senseBoxes method GET

With the optional `date` and `phenomenon` parameters you can find senseBoxes that have submitted data around that time, +/- 4 hours, or specify two dates separated by a comma.

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

Parameter

| Name | Type | Description |
|------------|-------------|--|
| date | ISO8601Date | optional One or two ISO8601 timestamps at which boxes should provide measurements. Use in combination with <code>phenomenon</code> . |
| phenomenon | String | optional A sensor phenomenon (determined by sensor name) such as temperature, humidity or UV intensity. Use in combination with <code>date</code> . |
| format | String | optional the format the sensor data is returned in. Default value: json Allowed values: "json", "geojson" |
| exposure | String | optional (optional) only return sensors of boxes with the specified exposure. Can be indoor or outdoor Allowed values: "indoor", "outdoor" |

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`

Headers

| Name | Type | Description |
|--------------|--------|---|
| content-type | String | Should be <code>application/json</code> or <code>application/json; charset=utf-8</code> |

JSON request body

| Name | Type | Description |
|----------|----------|---|
| user | User | the user for this 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) |

Parameter for creating the user for a senseBox

| Name | Type | Description |
|-----------|--------|---|
| firstname | String | the firstname of the user. |
| lastname | String | the ths lastname of the user. |
| email | String | the email for the user. is used for sending the arduino sketch. |
| lang | String | the language of the user. |

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 | optional the visual representation for the openSenseMap of this sensor. |

Settings for a senseBox connected through MQTT

| Name | Type | Description |
|-------------------|---------|--|
| enabled | Boolean | enable or disable mqtt Default value: false |
| 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 (https://github.com/mqttjs/MQTT.js#client) |

Error Response

Error-Response:

```
HTTP/1.1 415 Unsupported Media Type
{"code": "NotAuthorized", "message": "Unsupported content-type. Try application/json"}
```

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. |
| content-type | String | Should be <code>application/json</code> or <code>application/json; charset=utf-8</code> |

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 | optional the visual representation for the openSenseMap of this sensor. |

Settings for a senseBox connected through MQTT

| Name | Type | Description |
|---------------|---------|---|
| enabled | Boolean | enable or disable mqtt Default value: false |
| 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 (https://github.com/mqttjs/MQTT.js#client) |
|-------------------|--------|---|

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"}
```

Error-Response:

```
HTTP/1.1 415 Unsupported Media Type
{"code":"NotAuthorized","message":"Unsupported content-type. Try application/json"}
```

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"}
```

Interpolation ---

Get a Inverse Distance Weighting Interpolation as FeatureCollection

method GET

Retrieve a JSON object containing

- `breaks` : an array containing equal distance breaks. use `numClasses` parameter to control how many breaks to return
- `featureCollection` : a GeoJSON FeatureCollection with a computed Inverse Distance Interpolation for a certain region of interest and phenomenon.

The properties of each feature in the `featureCollection` is an object with ISO8601 timestamps which are the `timeSteps`. The number of the timesteps can be controlled using the `numTimeSteps` parameter. Values falling inside each timestep are first averaged. Please be aware that requests with $(\text{areaSquareKilometers} / \text{cellWidth}) > 2500$ will be rejected.

```
GET /statistics/idw?bbox=7.6,51.2,7.8,51.4&phenomenon=Temperatur
```

Parameter

| Name | Type | Description |
|------------|-------------|---|
| phenomenon | String | the name of the phenomenon you want to download the data for. |
| from-date | ISO8601Date | optional Beginning date of measurement data (default: 2 days ago from now) |
| to-date | ISO8601Date | optional End date of measurement data (default: now) |
| exposure | String | optional only return sensors of boxes with the specified exposure. Can be indoor or outdoor. Default undecided. Allowed values: indoor,outdoor |

| | | |
|--------------|--------|--|
| gridType | String | optional The type of the grid for IDW calculation Default value: hex Allowed values: hex,square,triangle |
| cellWidth | Number | optional The width of the grid cells in kilometers. Must be positive Default value: 50 |
| power | Number | optional The power of the IDW calculation Default value: 1 Allowed values: 1-9 |
| numTimeSteps | Number | optional Return this many timesteps between <code>from-date</code> and <code>to-date</code> Default value: 6 Allowed values: 1-10 |
| numClasses | Number | optional Number of classes in the breaks array. Must be positive Default value: 6 |
| bbox | String | A bounding box containing 4 WGS84 coordinates separated by comata (.). Order is longitude, latitude and southwest, northeast. |

Measurements ---

Post new measurement method POST

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

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

Headers

| Name | Type | Description |
|--------------|--------|---|
| content-type | String | Should be <code>application/json</code> or <code>application/json; charset=utf-8</code> |

JSON request body

| Name | Type | Description |
|-----------|-------------|--|
| value | String | the measured value of the sensor. Also accepts JSON float numbers. |
| createdAt | ISO8601Date | optional 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. |
|-----------|--------|--|

Error Response

Error-Response:

```
HTTP/1.1 415 Unsupported Media Type
{"code": "NotAuthorized", "message": "Unsupported content-type. Try application/json"}
```

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 |
|-----------|-------------|--|
| from-date | ISO8601Date | optional Beginning date of measurement data (default: 48 hours ago from now) |
| to-date | ISO8601Date | optional End date of measurement data (default: now) |
| format | String | optional Can be 'json' (default) or 'csv' (default: json) Default value: json Allowed values: "json","csv" |
| download | Boolean | optional if specified, the api will set the <code>content-disposition</code> header thus forcing browsers to download instead of displaying. Is always true for format csv. Allowed values: "true","false" |
| | | |

| | | |
|-------------|--------|--|
| :senseBoxId | String | the ID of the senseBox you are referring to. |
| :sensorId | String | the ID of the sensor you are referring to. |
| separator | String | optional 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. Alternatively you can use <code>delimiter</code> as parameter name. Allowed values: "comma" |

Delete measurements of a sensor method DELETE

This method allows to delete measurements for the specified sensor. Use the request body to specify which measurements should be deleted.

```
DELETE /boxes/:senseBoxId/:sensorId/measurements
```

Headers

| Name | Type | Description |
|--------------|--------|---|
| x-apikey | String | the secret API key which corresponds to the <code>senseBoxId</code> parameter. |
| content-type | String | Should be <code>application/json</code> or <code>application/json; charset=utf-8</code> |

JSON request body

| Name | Type | Description |
|-----------------------|---------------|--|
| from-date | ISO8601Date | optional Beginning date of measurement data (no default) |
| to-date | ISO8601Date | optional End date of measurement data (no default) |
| timestamps | ISO8601Date[] | optional Allows to specify timestamps which should be deleted |
| deleteAllMeasurements | Boolean | optional Specify <code>deleteAllMeasurements</code> with a value of <code>true</code> to delete all measurements of this sensor Default value: false Allowed values: true,false |

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. |

Error Response

Error-Response:

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

Error-Response:

```
HTTP/1.1 415 Unsupported Media Type
{"code":"NotAuthorized","message":"Unsupported content-type. Try application/json"}
```

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 | ISO8601Date | optional Beginning date of measurement data (default: 2 days ago from now) |
| to-date | ISO8601Date | optional End date of measurement data (default: now) |
| columns | String | optional (optional) Comma separated list of columns to export. If omitted, columns createdAt, value, lat, lng are returned. Possible allowed values are createdAt, value, lat, lng, unit, boxId, sensorId, phenomenon, sensorType, boxName, exposure. The columns in the csv are like the order supplied in this parameter Default value: createdAt,value,lat,lng |
| exposure | String | optional (optional) only return sensors of boxes with the specified exposure. Can be indoor or outdoor Allowed values: "indoor","outdoor" |
| separator | String | optional Only for csv: the separator for csv. Possible values: comma for comma as separator, everything else: semicolon. Per default a semicolon is used. Alternatively you can use delimiter as parameter name. Allowed values: "comma" |
| bbox | String | A bounding box containing 4 WGS84 coordinates separated by comata (.). Order is longitude, latitude and southwest, northeast. |

Post multiple new measurements method POST

Post multiple new measurements in multiple formats to a box. Allows the use of csv, json array and json object notation.

CSV:

For data in csv format, first use `content-type: text/csv` as header, then submit multiple values as lines in `sensorId,value,[createdAt]` form. Timestamp is optional. Do not submit a header.

JSON Array:

You can submit your data as array. Your measurements should be objects with the keys `sensor` , `value` and optionally `createdAt` . Specify the header `content-type: application/json` .

JSON Object:

The third form is to encode your measurements in an object. Here, the keys of the object are the sensorIds, the values of the object are either just the `value` of your measurement or an array of the form `[value, createdAt]`

For all encodings, the maximum count of values in one request is 2500.

```
POST /boxes/:boxId/data
```

Parameter

| Name | Type | Description |
|--------------------------|--------|--|
| <code>:senseBoxId</code> | String | the ID of the senseBox you are referring to. |

Misc ---

Get some statistics about the database method GET

returns an array with three numbers which denominates the count of senseBoxes, the count of measurements and the count of measurements in the last minute.

```
GET /stats
```

Parameter

| Name | Type | Description |
|--------------------|---------|--|
| <code>human</code> | Boolean | optional if true, make numbers easier human readable. Default value: false Allowed values: true,false |

Success Response

[human=false]

```
[318,118241889,393]
```

[human=true]

```
["318", "118M", "393"]
```

