

Loopshore API

Version 0.6.0

Table of Contents

Introduction.....	3
Terminology.....	3
API location.....	3
Authentication.....	4
Username & password.....	4
Custom ingestion URL.....	4
API keys.....	4
Authorization.....	5
Restrictions.....	6
API descriptions.....	7
Introduction.....	7
Known quantities.....	7
Login.....	10
Description.....	10
Acquire apikey.....	11
Description.....	11
apikey usage.....	12
List apikeys.....	12
Description.....	12
Remove apikey.....	13
Description.....	13
Observation history.....	14
Description.....	14
Latest observations.....	16
Description.....	16
Subscribe observations.....	17
Description.....	17
Response.....	18
List subscriptions.....	19
Description.....	19
Cancel observation subscription.....	19
Description.....	19
Test subscriptions.....	20
Description.....	20
Send observations.....	21
Description.....	21
Revision history.....	22

Introduction

Loopshore API contains interfaces for reading and writing environmental sensor data from and to Loopshore cloud service.

APIs are HTTP based REST APIs. Only HTTPS is supported.

Terminology

Observation - a Physical or logical set of value, quantity and timestamp. Usually a measurement result.

Webhook - a way to deliver device observations to 3rd party servers.

API location

Base URL	https://service.loopshore.com/api
Port	443

Authentication

There are three ways to authenticate to the API.

- Username and password
- API keys
- Custom ingestion URL(only for sending observations)

Username & password

You can authenticate to the API using username and password. Obtaining these credentials is out of scope of this document. Workflow is as follows

- Post credentials to /token API
- Save received session token
- Use the session token to authenticate to other APIs

Custom ingestion URL

Ask URL from Loopshore

API keys

Users can use their credentials to acquire a API key that has same privileges as the username and password. Using API key might be easier, faster and less resource consuming than acquiring the session key for each request. Note that you have to handle the API keys with same precautions as you handle the password.

Authorization

Each user is only allowed to access the resources that are configured to the user by system admins. If you are trying to access a resource that you have not access to, 401 error code will be returned. Contact Loopshore if you think that you should have access to a resource.

Restrictions

A reasonable usage of APIs is expected from API users. Currently reasonable usage limit is *once/minute/device/day*, which should cover all the normal use cases. API access may be temporarily blocked if the user exceeds this amount.

If you want to exceed this restriction, contact Loopshore.

Example 1

User has access to two devices, that each report new measurements every 10 minutes. User polls the latest measurements for each device every 5 minutes.

API Usage

Devices: $\text{devices} \times \text{hours} \times \text{hourly_rate} = 2 \times 24 \times (60/10) = 288$

History: $\text{devices} \times \text{hours} \times \text{hourly_rate} = 2 \times 24 \times (60/5) = 576$

Total: $288 + 576 = 864$ requests/24h

Limit: $\text{devices} \times \text{hours} \times \text{minutes} = 2 \times 24 \times 60 = 2880$ requests/24h

24h requests(864) is well below the limit(2880) so API usage is ok.

Example 2

User has access to two devices, that each report new measurements every 10 minutes. User wants to load a year's worth of history data from both devices. User fetches 15 different quantities from each device.

API Usage

Devices: $\text{devices} \times \text{hours} \times \text{hourly_rate} = 2 \times 24 \times (60/10) = 288$

History:

Measurement in a API call(5000 limit): $(5000/15) = 333$

Hours in API call: $333 / (60/10) = 56$

API calls needed to get 1 year history: $(365 \times 24) / 56 = 158$

Total: $288 + 128 = 446$ requests

Limit: $\text{devices} \times \text{hours} \times \text{minutes} = 2 \times 24 \times 60 = 2880$

requests(446) is below the limit(2880) so API usage is ok.

API descriptions

Introduction

APIs are HTTP REST APIs. APIs accept two types of content. *JSON* and *transit+JSON*. If user does not give *Content-Type*(in POST) or *Accept header*(in GET), service assumes JSON.

Normal HTTP error codes are returned in case of failures. Success code is either 200 or 204.

Known quantities

APIs are designed so that you can practically send any kind of information/observations to the system and fetch those via APIs. There is however, some predefined and known data types that have special meaning. These should be used when possible, because then for example the user interface recognizes the types.

Quantities and units are case sensitive.

Some quantities are calculated by system if not sent by device and source data is available. These are marked with asterisk(*) in the list below.

Quantity	Unit	Datatype	Description
temperature	C	Number	Temperature
humidity	%	Number	Relative humidity
tvoc	ppb	Number	Total Volatile Organic Compounds
co2	ppm	Number	Carbon Dioxide
pm1p0	ug/m3	Number	Particles < 1.0um
pm2p5	ug/m3	Number	Particles < 2.5um
pm4p0	ug/m3	Number	Particles < 4.0um
pm10p0	ug/m3	Number	Particles < 10um
pressure	hPa	Number	Atmospheric pressure
light	lx	Number	Ambient light level
laeqx	dB	Number	A-weighted equivalent ambient sound pressure level since last measurement of same type
lafmin	dB	Number	A-weighted minimum sound pressure level since last measurement of same type
lafmax	dB	Number	A-weighted maximum sound pressure level since last measurement of same type
lzeqx	dB	Number	Z-weighted equivalent ambient sound pressure level since last measurement of same type
lzfmin	dB	Number	Z-weighted minimum sound pressure level since last measurement of same type
lzfmax	dB	Number	Z-weighted maximum sound pressure level since last measurement of same type
pressure_diff	Pa	Number	Pressure difference
accxposmax	m/s2	Number	Maximum acceleration along X-Axis in positive direction since last measurement of same type
accxnegmax	m/s2	Number	Maximum acceleration along X-Axis in negative direction since last measurement of same type



Quantity	Unit	Datatype	Description
accyposmax	m/s ²	Number	Maximum acceleration along Y-Axis in positive direction since last measurement of same type
accynegmax	m/s ²	Number	Maximum acceleration along Y-Axis in negative direction since last measurement of same type
acczposmax	m/s ²	Number	Maximum acceleration along Z-Axis in positive direction since last measurement of same type
accznegmax	m/s ²	Number	Maximum acceleration along Z-Axis in negative direction since last measurement of same type
latitude	d	Number	GPS coordinate in degree/angle
longitude	d	Number	GPS coordinate in degree/angle
altitude	m	Number	Altitude from sea level in meters
shock	m/s ²	Number	Acceleration shock
tvoc_density	ug/m ³	Number	Total Volatile Organic Compounds
absolute_humidity*	g/m ³	Number	Absolute humidity
dewpoint*	C	Number	Dewpoint

Login

Method	Location
POST	token

Body	Optional	Datatype
name	no	String
password	no	String

Description

Acquire a session cookie that can be used to authenticate following requests. On successful request, response sets a cookie using *Set-Cookie* header.

Example response header

```
Set-Cookie: jabster_token=<token>;Max-Age=1234
```

Example use of cookie in a request

```
Cookie: jabster_token=<token>
```

Acquire apikey

Method	Location
POST	api_key

Body	Optional	Datatype
key-name	no	String

Description

Acquire an apikey(random hextring) that can be used instead of session style login. **key-name** is a documentary string that can be used to identify different keys.

Response contains a **secret-key** that can be used in HTTP header to authenticate a user. Treat the **secret-key** with the same precautions as a password.

There is no way to recover a lost **secret-key**. In case a **secret-key** is lost, make a new one and preferably delete the lost key from the service.

id is used to identify a key in other APIs.

Example request

```
{
  "key-name": "my-api-key"
}
```

Example response

```
{
  "id": 123,
  "secret-key": "ccb22ee3fe90ce30d20654ee214307755806f53dbb84"
}
```

apikey usage

Apikey is used by putting the key to HTTP header with name **x-api-key**.

Example header

```
"x-api-key:ccb22ee3fe90ce30d20654ee214307755806f53dbb84"
```

List apikeys

Method	Location
GET	api_key

Description

List all the user's api-keys with **key-name** and **id**.

Example response

```
{  
  "id": 123,  
  "key-name": "my-api-key"  
}
```

Remove apikey

Method	Location
DELETE	api_key

Body	Optional	Datatype
id	no	Number

Description

Removes an api-key from the service.

Example request

```
{  
  "id": 123  
}
```

Observation history

Method	Location
GET	observation/read/device/{device-id}

Parameter	Optional	Type	Datatype
device-id	no	path	String
start	no	query	String(RFC3339)
end	yes	query	String(RFC3339)
quantity	yes	query	String

Description

Returns all observations for given **device-id** from the **start** timestamp to current time.

Result is limited to *5000 observations*.

Optionally, **end** timestamp and **quantity** filter can be given.

start time is inclusive and **end** time is exclusive.

Result is a JSON list of objects where each object presents a single quantity-value-time observation. *Timestamp*, *value* and *quantity* are always present. *Unit* is optional.

Response example

```
[
  {
    "timestamp": "2021-02-03T11:29:31.229Z",
    "value": 21.4,
    "quantity": "temperature",
    "unit": "C"
  },
  {
    "timestamp": "2021-02-03T11:23:12Z",
    "value": 442,
    "quantity": "co2",
    "unit": "PPM"
  }
]
```

Usage example

You want to get one year history of temperature from a device. You know that a device sends data mostly with once / 10 minute intervals. This means that you get $6 \times 24 = 144$ datapoints in a day and $144 \times 30 = 4320$ datapoints in a month. This falls within API 5000 result rows limit.

Because start time is inclusive and end time is exclusive, you can now call the API like this:

```
/read/device/my-device-id?start=2020-01-01T00%3A00%3A00Z&quantity=temperature&end=2020-02-00T00%3A00%3A00Z
```

```
/read/device/my-device-id?start=2020-02-01T00%3A00%3A00Z&quantity=temperature&end=2020-03-00T00%3A00%3A00Z
```

Latest observations

Method	Location
GET	observation/read/device/{device-id}/last-values

Parameter	Optional	Type	Datatype
device-id	no	path	String

Description

Returns the latest observations of each *quantity* for given ***device-id***.

This API is optimized for "poll latest observations" use case.

Response example

```
[
  {
    "timestamp": "2021-02-03T11:29:31.229Z",
    "value": 21.4,
    "quantity": "temperature",
    "unit": "C"
  }
]
```


Subscribe observations

Method	Location
POST	webhook

Body	Optional	Datatype
url	no	String ^{*1}
device-id	yes	String ^{*2}
scope	yes	String ^{*2}
x-headers	yes	JSON object ^{*3}

***1** This has to be a valid url that accepts data as described in Test subscriptions chapter. HTTP or HTTPS accepted as protocol.

***2** Either device-id or scope for the webhook has to be given, but not both.

***3** Any valid HTTP header in {key:value} format

Description

Posts a new webhook either given **device-id** or **scope**.

Webhooks or subscriptions are used to subscribe data to 3rd party server in real time. When observations from device arrive to Loopshore service, it immediately forwards them to all interested stakeholders.

Webhook must contain a valid **url** where the data is to be sent.

Webhooks can be targeted to single **device-ids** or all the devices that user has access to(**scope**). Only valid value for **scope** is "all".

Optionally HTTP message sent from Loopshore can contain custom headers(**x-headers**).

Simple example for single device

```
{
  "url": "http://my-insecure-server.io/my-api",
  "device-id": "my-device-id"
}
```

More complex example for all devices with custom headers

```
{
  "url": "https://my-secure-server.io/my-api",
  "scope": "all",
  "x-headers": {
    "apikey": "secret",
    "target": "all-loopshore-devices"
  }
}
```

Response

Response contains all the data that was posted to API plus **id** and **nonce** fields.

Body	Optional	Datatype
id	no	String(UUID)
nonce	no	String

id - can be used to identify this particular webhook in other APIs.

nonce - is a random string that is sent along with forwarded observations and it can be used to identify what hook generated the POST.

Example response

```
{
  "id": "a0c03d18-378e-4326-b1d5-b47d9248be1e",
  "url": "https://my-secure-server.io/my-api",
  "scope": "all",
  "nonce": "_izF930v",
  "x-headers": {
    "apikey": "secret",
    "target": "all-loopshore-devices"
  },
  "device-id": null
}
```

List subscriptions

Method	Location
GET	webhook

Description

This API can be used to list all the subscriptions/webhooks.

Example response

```
[{
  "id": "a0c03d18-378e-4326-b1d5-b47d9248be1e",
  "url": "https://my-secure-server.io/my-api",
  "scope": "all",
  "nonce": "_izF930v",
  "x-headers": {
    "apikey": "secret",
    "target": "all-loopshore-devices"
  },
  "device-id": null
}]
```

Cancel observation subscription

Method	Location
DELETE	webhook/{id}

Parameter	Optional	Type	Datatype
id	no	path	String(UUID)

Description

This API can be used to remove a subscription.

Test subscriptions

Method	Location
POST	webhook/{id}/test

Parameter	Optional	Type	Datatype
id	no	path	String(UUID)

Body	Optional	Datatype
nonce	no	String
data	no	JSON object

Description

Sends a test message to webhook url.

You can test your webhook counterpart by posting a custom message to this API. Service then forwards the message to your server.

nonce - is the nonce for the webhook **id**.

data - any valid JSON. When real message arrives, data contains the same data that came in to the system via *Send observations* API.

Example message sent to *url* server

```
{
  "nonce": "_izF930v",
  "data": {"my": "test data"}
}
```

Example of message identical to real data

```
{
  "nonce": "_izF930v",
  "data": {
    "device-id": "123456789",
    "observations": [
      {
        "unit": "C",
        "value": 19.24,
        "quantity": "temperature",
        "timestamp": "2021-02-03T11:23:12Z"
      }
    ],
  },
}
```

Send observations

Method	Location
POST	observation/{token}/v2

Body	Optional	Datatype
device-id	no	String
observations	no	List(observation Object)
observation.timestamp	no	String(RFC3339)
observation.quantity	no	String ^{*1}
observation.value	no	Number/String
observation.unit	no	String ^{*1}

^{*1} See *known quantities* chapter

Description

Posts new observations for given **device-id**. For getting your own **token**, contact loopshore.

Body example

```
[
  {
    "timestamp": "2021-02-03T11:29:31.229Z",
    "value": 21.4,
    "quantity": "temperature",
    "unit": "C"
  }
]
```

Revision history

Version	Date	Author	Description
0.1	4.2.2021	J. Ratilainen	Initial draft
0.2	11.2.2021	J. Ratilainen	Add api-keys
0.2.1	20.4.2021	J. Ratilainen	Clarify units
0.2.2	18.5.2021	J. Ratilainen	Add pressure difference
0.3.0	8.9.2021	J. Ratilainen	Add acceleration
0.4.0	20.3.2023	J. Ratilainen	Add GPS coordinates
0.5.0	2.4.2023	J. Ratilainen	Add shock
0.6.0	19.2.2024	J. Ratilainen	Add dewpoint, absolute_humidity and tvoc_density. Fixed