Public API manual

embion

Table of contents

1 Introduction

The HUB public API offers a programmatic interface to the HUB platform, enabling integration with both the HUB portal and Embion EMS Controllers. Through this API, users can retrieve data such as plant information, meter readings, and inverter data. Additionally, the API provides endpoints for controlling EMS Controllers, allowing users to remotely manage them.

2 About this document

2.1 Purpose

The purpose of this document is to provide comprehensive documentation for the HUB public API, including detailed information about authentication methods, API endpoints, request and response formats, and usage examples.

2.2 Intended audience

This document is intended for developers and integrators who need to interact with the HUB public API, as well as for technical stakeholders who require an understanding of the API's capabilities and usage.

2.3 Callouts

Note

Used for notes in this documentation

Warning

Used for warnings in this documentation

Important

Used for important notes in this documentation

Tip

Used for tips in this documentation

Caution

Used for caution notes in this documentation

3 General information

3.1 HTTP status codes

The HUB public API uses standard HTTP status codes to indicate the success or failure of API requests. The following status codes are commonly used:

HTTP Status Codes

Code	Status	Description
200	OK	The request was successful, and the response contains the requested data.
400	Bad Request	The request was invalid or malformed.
401	Unauthorized	Authentication is required, and the request has not been applied because it lacks valid authentication credentials.
403	Forbidden	The server understood the request but refuses to authorize it.
404	Not Found	The requested resource could not be found.
500	Internal Server Error	An error occurred on the server while processing the request.

3.2 ISO 8601

The HUB public API uses the ISO 8601 standard for date and time representation. Dates and times are formatted as follows:

Timestamp formats

Notation	Date	Timezone
2022-12-14T08:00Z	14-12-2022 8:00:00	UTC
2022-12-14T08:00	14-12-2022 8:00:00	UTC
2022-12-14T08Z	14-12-2022 8:00:00	UTC
2022-12-14T08:00:00.000+0100	14-12-2022 8:00:00	GMT+1
2022-12-14	14-12-2022 00:00:00	UTC
2022-12-14GMT+0100	14-12-2022 00:00:00	GMT+1
2022-12	01-12-2022 00:00:00	UTC
2022	01-01-2022 00:00:00	UTC

3.3 Content type

The HUB public API exclusively supports the JSON format for both requests and responses. All data exchanged with the API must be encoded as JSON, and clients should set the Content-Type: application/json header when sending requests.

3.4 GZIP compression

The HUB public API supports GZIP compression for request and response bodies. Clients can request GZIP compression by including the Accept-Encoding: gzip header in their requests. If the server supports GZIP compression, it will respond with a Content-Encoding: gzip header, and the response body will be compressed using GZIP.

3.5 Rate limiting

The API enforces daily request limits that reset at midnight UTC. When using device access tokens, responses include headers indicating your current rate limit status:

Rate Limit Headers

Header	Description
RateLimit-Limit	Maximum number of requests allowed per day.
RateLimit-Remaining	Number of requests remaining in the current period.
RateLimit-Reset	Seconds until the rate limit resets (at midnight UTC).

Your specific limit depends on your token configuration. When the limit is exceeded, the API returns HTTP status code 429 Too Many Requests.

4 Authentication

All API endpoints require authentication via access tokens. The public API supports two authentication methods: device access tokens and third party access tokens.

4.1 Device access tokens

Device access tokens can be generated within the HUB portal and always correspond to a specific Embion EMS Controller. These tokens can be used to access data from the controller and to control it. Multiple tokens can be created for a single controller.

4.1.1 Creation

Device access tokens can be created within the Devices app of the HUB portal. For more information on creating device access tokens, please refer to the device access tokens section within the documentation of the HUB portal.

4.1.2 Permissions and limits

4.1.2.1 Request limits

> **Note**
>
> If more than 10,000 calls per device per day are required, please contact Embion for further assistance and to discuss specific needs.

There is a default limit of 10,000 requests per device per day, with each request counting towards this limit. Device tokens can also be configured to a specific maximum request limit, which has to be lower or equal to the daily device limit since all requests still contribute to this limit.

4.1.2.2 Expiration and revocation

Device access tokens do not expire by default, but an expiration date can be configured within the HUB portal. Additionally, device tokens can be temporarily disabled or revoked there.

4.1.2.3 Permissions

Device access tokens can be configured with specific permissions, determining the actions that can be performed with the token. These permissions can be set when creating the token and can be modified later if needed.

Control access: This permission allows the token to control the associated EMS Controller, e.g. by sending plant control commands to it.

Allow all UIDs: Embion EMS Controllers monitor and control assets like PV inverters, batteries, and meters using unique identifiers (UIDs). This permission allows the token to monitor all assets associated with the controller. If this permission is not granted, the token can only monitor assets that are explicitly assigned by their UIDs.

4.1.3 Usage

Device access tokens consist of two parts, an id and the actual token value. The id is a unique identifier for the token, while the token value is a long string of characters that is used to authenticate API requests. Both the id and the token value are required for authentication. If there is a mismatch between them, the request will be rejected and a response with status code 401 - Unauthorized will be returned.

In order to correctly authenticate API requests using a device access token, the token value must be included in the API-KEY header of the request. For example: API-KEY: 1nsq2001is51dXmaz6xmc8Xmla10rf109lsyb1Xyopape.

The id must also be included in within the request. The way this is done differs per request. For GET requests, it must be passed as a query parameter. For other request types, the id should be included in the request body.

Example with a query parameter (appended to the URL of the request):

https://api.hub.embion.nl/v1/status?id=1nsq2001is51d

Example request body:

```
{
  "id": "1nsq2001is51d"
}
```

4.2 Third party access tokens

Third party access tokens allow an authorized third party to monitor and/or control EMS Controllers that users have granted access to. Compared to device access tokens, these are easier to use and manage, since the same token value can be used to access all connected devices. Additionally, users can grant access to specific devices without creating and sharing device access tokens, simplifying the process.

4.2.1 Creation

> **!** Important
>
> Third party access tokens are created and managed by Embion. If you wish to create a third party access token, please contact us for further assistance.

After your third party is registered, users can grant device access to you within the Devices app of the HUB portal.

4.2.2 Permissions and limits

4.2.2.1 Request limits

Third party access tokens have no explicit request limits.

4.2.2.2 Expiration and revocation

Third party access tokens do not expire by default, but an expiration date can be configured by Embion if necessary. Additionally, third party tokens can be temporarily disabled or revoked.

4.2.2.3 Permissions

Third parties and their corresponding tokens can have the following permissions:

Monitoring: This permission allows the third party to monitor the associated EMS Controller and its assets (always active).

Control: This permission allows the third party to control the associated EMS Controller and its assets (optional).

Only one third party with Control access is allowed per EMS Controller. There is no limit on the number of third parties with Monitoring access.

4.2.3 Usage

Third party access tokens consist of two parts, an id and the actual token value. The id is a unique identifier for either the token itself or a connected EMS Controller, while the token value is a long string of characters that is used to authenticate API requests. Both the id and the token value are required for authentication. If there is a mismatch between them, the request will be rejected and a response with status code 401 - Unauthorized will be returned.

In order to fetch the ids of connected EMS Controllers, please refer to the third party information API endpoints.

In order to correctly authenticate API requests using a third party access token, the token value must be included in the API-KEY-THIRD-PARTY header of the request. For example: API-KEY-THIRD-PARTY: 1nsq2001is51dXmaz6xmc8Xmla10rf109lsyb1Xyopape.

The id must also be included in within the request. The way this is done differs per request. For GET requests, it must be passed as a query parameter. For other request types, the id should be included in the request body.

Example with a query parameter (appended to the URL of the request):

https://api.hub.embion.nl/v1/status?id=1nsq2001is51d

Example request body:

```
{
  "id": "1nsq2001is51d"
}
```

5 API endpoints

5.1 General device information

5.1.1 GET device status

Method: GET | URL: https://api.hub.embion.nl/v1/status

This endpoint returns information about the current status of a specific device.

5.1.1.1 Required permissions

Device tokens: (all allowed)

Third party tokens: Monitoring

5.1.1.2 Request

5.1.1.2.1 Query parameters

GET device status query parameters

Var	Description	Mandatory
id	The ID of the token (device tokens) or the device (third party tokens) to retrieve status information for.	Yes

5.1.1.2.2 Example request

https://api.hub.embion.nl/v1/status?id=[id]

5.1.1.3 Response

var	description	format

5.1.1.3.1 Response body parameters

var	description	format
status	actual status of the plant	string
online	true if plant is online, false if offline (no data or status received in the last 2 minutes)	bool
last_contact	last contact in ISO 8601 layout	string
serial	serial number of device	string
version	actual software version of device	string
pn	product number of device	string
name	reference name of device	string
namespace	namespace location of device	string
status_message	returns the actual status message of the device	string
support_status	returns the actual support status, disabled or support ID when enabled	string
safe_state	true if safe_state is enabled on the device	bool
plant_control	idle, pending, sent, accepted, failed	string
epex_configured	True if the device has energy price control rules (defined in the energy pricing app), false if not	bool
token	token data (only included when a device token was used, see table below)	**token**

Definition of token:

var	description	format
control_allowed	true if control access is enabled for the given token	bool
expire_date	expiriation date of the given token if set	string
request_limit	maximum daily requests for the token when configured, otherwise the configured maximum device requests (default 3000).	Integer
requests_today_token	number of requests done for the token today	Integer

5.1.1.3.2 Example response body

```
{
   "status": "ok",
   "online": true,
   "last_contact": "2022-12-14T12:48:13.000Z",
   "serial": "0100211001090B",
   "version": "1.3.1",
   "pn": "GSE-A010-POE",
   "name": "main-solar",
   "namespace": "Embion",
   "status_message": "reducing inverters",
   "support_status": "A291D88",
   "safe_state": false,
   "plant_control": "idle",
   "epex_configured": false,
   "token": {
      "control_allowed": true,
      "expire_date": "2024-09-30T00:00:00.000Z",
      "request_limit": 3000,
      "requests_today_token": 40
   }
}
```

5.1.2 GET UIDs

Method: GET | URL: https://api.hub.embion.nl/v1/uids

This endpoint returns all UIDs available for the given token. The result can include either all UIDs or a subset of UIDs, depending on the token's configuration.

5.1.2.1 Required permissions

Device tokens: (all allowed)

Third party tokens: Monitoring

Please note that third party tokens always have access to all UIDs of the device.

5.1.2.2 Request

5.1.2.2.1 Query parameters

GET uids query parameters

Var	Description	Mandatory
id	The ID of the token (device tokens) or the device (third party tokens) to retrieve status information for.	Yes
isonline	When 'true', it will only return UIDs that were online in the last two days. Default is 'false'.	No

5.1.2.2.2 Example request

https://api.hub.embion.nl/v1/uids?id=[id]&isonline=true

5.1.2.3 Response

5.1.2.3.1 Response body parameters

var	description	format
allowall	Indicates whether the token has access to all UIDs.	boolean
isonline	Indicates whether online or offline UIDs are returned.	boolean
uids	The list of UIDs associated with the device.	array

5.1.2.3.2 Example response body

```
{
  "allowall": true,
  "isonline": false,
  "uids": [
    "ev1_Parkeerplaats P1:21",
    "inverter1_Dak:31",
    "meter1_grid:1"
  ]
}
```

5.2 Data retrieval

5.2.1 GET plant data

Method: GET | URL: https://api.hub.embion.nl/v1/plant

This API endpoint returns data from the given plant.

5.2.1.1 Request

5.2.1.1.1 Required permissions

Device tokens: (all allowed)

Third party tokens: Monitoring

5.2.1.1.2 Query parameters

> **Last value**
>
> Note that the last data sample is only available within the last two hours. If both the start date and period are not defined in the call, only the last data sample is returned if data is available; otherwise, no data will be returned.

> **Data time range**
>
> - Users can optionally include a time with the start_date, which shifts the day interval. However, the total number of returned entries remains unchanged. If no time is specified, a day is considered from 00:00:00 to 23:59:59 in the selected timezone.
> - The start_date determines the beginning of the data range. The period defines the interval between returned dates, while the range specifies the end date/time relative to start_date, thereby determining the number of entries returned.

> **Period limitations**
>
> Please note there are some limitations when combining period and range
>
> - For period q (quarter-hourly) and h (hourly), the maximum range is d (one day).
> - For period d (daily), the maximum range is w (one week).

- For period w (weekly), the maximum range is m (one month).
- For period m (monthly), the maximum range is y (one year.)

Get plant data query parameters

Var	Description	Mandatory	Format
ID	ID of the token (generated in the HUB)	Yes	String
period	Select data return period: • **l** last sample (default) • **q** 15 minute • **h** hourly • **w** weekly • **d** daily • **m** monthly • **y** yearly	No	String
range	Time range to show: • **d** day (default) • **w** week • **m** month • **y** year	No	String
type	Combination type of multiple datapoints: • **min** minimum value in timerange • **max** maximum value in timerange (default) • **avg** average value in timerange	No	String
start_date	Date of the first sample in ISO 8601 layout, if not set current day is used	No	String

5.2.1.1.3 Example request

https://api.hub.embion.nl/v1/plant?id=[id]&period=q&range=d&type=max&start_date=2022-12-14

5.2.1.2 Response

5.2.1.2.1 Response body parameters

The plant data is included in the JSON body of the response. The actual lay-out of the body varies depending on the query parameters given in the request.

The following parameters can be present, if the data is available:

var	description	units	format
timestamp	Timestamp of the measurement	ISO 8601	string
psol	Actual solar power	1 W	Integer
kdy	Cumulative daily yield	1 Wh	Integer
kty	Cumulative total yield	1 Wh	Integer
soc	Average state of charge	0.1%	Integer
evku	Cumulative EV charger consumption	1 Wh	Integer
pev	Total EV charger power	1 W	Integer
pbat	Total battery power	1 W	Integer
run	# inverters in RUN state	-	Integer
warn	# inverters in WARN state	-	Integer
err	# inverters in ERR state	-	Integer
red	Actual reduction value (10000 == 100% => no reduction) Represents power limit	%	Integer
var1	Free to use variable	-	Integer
var2	Free to use variable	-	Integer
var3	Free to use variable	-	Integer
var4	Free to use variable	-	Integer
in1	State of digital input 1	-	Integer 0 or 1 (bool)
in2	State of digital input 2	-	Integer 0 or 1 (bool)
out1	State of digital output 1	-	Integer 0 or 1 (bool)
out2	State of digital output 2	-	Integer 0 or 1 (bool)
con	# of inverters connected to the gateway	-	Integer
pgrid	gridpower	1 W	Integer
egi	Grid import energy	1 Wh	Integer
ege	Grid export energy	1 Wh	Integer
gil1	Grid phase 1 current	0.1 A	Integer
gil2	Grid phase 2 current	0.1 A	Integer
gil3	Grid phase 3 current	0.1 A	Integer
gul1	Grid phase 1 voltage	0.1 V	Integer
gul2	Grid phase 2 voltage	0.1 V	Integer
gul3	Grid phase 3 voltage	0.1 V	Integer

5.2.1.2.2 Example response body


```
[
  {
    "timestamp": "2022-12-14T10:00:00.000Z",
    "con": 3,
    "ege": 3500,
    "egi": 2000,
    "err": 0,
    "in1": 1,
    "in2": 0,
    "out1": 0,
    "out2": 0,
    "gil1": 5,
    "gil2": 6,
    "gil3": 7,
    "gul1": 220,
    "gul2": 230,
    "gul3": 240,
    "kdy": 1010,
    "kty": 7200,
    "pgrid": 1000,
    "psol": 1750,
    "soc": 550,
    "pbat": 1550,
    "pev": 750,
    "evku": 2450,
    "red": 10000,
    "run": 2,
    "var1": 1,
    "var2": 2,
    "var3": 3,
    "var4": 4,
    "warn": 1
  },
  {
    "timestamp": "2022-12-14T10:15:00.000Z",
    "con": 3,
    "ege": 4100,
    "egi": 2000,
    "err": 0,
    "in1": 1,
    "in2": 0,
```

```
    "out1": 0,
    "out2": 0,
    "gil1": 56,
    "gil2": 63,
    "gil3": 78,
    "gul1": 2218,
    "gul2": 2301,
    "gul3": 2368,
    "kdy": 12010,
    "kty": 7200,
    "pgrid": -11600,
    "psol": 2000,
    "soc": 550,
    "pbat": -15000,
    "pev": 750,
    "evku": 2450,
    "red": 10000,
    "run": 3,
    "var1": 1,
    "var2": 2,
    "var3": 3,
    "var4": 4,
    "warn": 0
  }
]
```

5.2.2 GET meter data

Method: GET | URL: https://api.hub.embion.nl/v1/meter

This API endpoint returns individual meter data.

5.2.2.1 Required permissions

Device tokens: (all allowed)

Third party tokens: Monitoring

5.2.2.2 Request

5.2.2.2.1 Query parameters

> **Last value**
>
> Note that the last data sample is only available within the last two hours. If both the start date and period are not defined in the call, only the last data sample is returned if data is available; otherwise, no data will be returned.

> **Data time range**
>
> - Users can optionally include a time with the start_date, which shifts the day interval. However, the total number of returned entries remains unchanged. If no time is specified, a day is considered from 00:00:00 to 23:59:59 in the selected timezone.
> - The start_date determines the beginning of the data range. The period defines the interval between returned dates, while the range specifies the end date/time relative to start_date, thereby determining the number of entries returned.

> **Period limitations**
>
> Please note there are some limitations when combining period and range
>
> - For period q (quarter-hourly) and h (hourly), the maximum range is d (one day).
> - For period d (daily), the maximum range is w (one week).
> - For period w (weekly), the maximum range is m (one month).
> - For period m (monthly), the maximum range is y (one year.)

GET meter data query parameters

Var	Description	Mandatory	Format
ID	ID of the token (generated in the HUB)	Yes	String
uid	the uid of the meter to read, only one uid can be entered	Yes	String
period	Select data return period: • **l** last sample (default) • **q** 15 minute • **h** hourly • **w** weekly • **d** daily • **m** monthly • **y** yearly	No	String
range	Time range to show: • **d** day (default) • **w** week • **m** month • **y** year	No	String
type	Combination type of multiple datapoints: • **min** minimum value in timerange • **max** maximum value in timerange (default) • **avg** average value in timerange	No	String
start_date	Date of the first sample in ISO 8601 layout, if not set current day is used	No	String

5.2.2.2.2 Example request

https://api.hub.embion.nl/v1/meter?id=[id]&uid=[uid]&period=q&range=d&type=max&start_date=2022-12

5.2.2.3 Response

5.2.2.3.1 Response body parameters

The meter data is included in the JSON body of the response. The actual lay-out of the body varies depending on the query parameters given in the request. Data that is not used by the given meter is left out from the response body.

The following parameters can be present, if the data is available:

var	description	units	format
timestamp	Timestamp of the measurement	ISO 8601	string
actpow	Total active power	1 W	Integer
apparpow	Total apparent power	1 VA	Integer
reactpow	Total reactive power	1 VAR	Integer
pf	Total powerfactor	0.01 $cos(\varphi)$	Integer
pfl1	Phase 1 powerfactor	0.01 $cos(\varphi)$	Integer
pfl2	Phase 2 powerfactor	0.01 $cos(\varphi)$	Integer
pfl3	Phase 3 powerfactor	0.01 $cos(\varphi)$	Integer
actpowl1	Phase 1 active power	1 W	Integer
actpowl2	Phase 2 active power	1 W	Integer
actpowl3	Phase 3 active power	1 W	Integer
il1	Phase 1 current	0.1 A	Integer
il2	Phase 2 current	0.1 A	Integer
il3	Phase 3 current	0.1 A	Integer
vll12	Phase1-2 line-line voltage	0.1 V	Integer
vll13	Phase1-3 line-line voltage	0.1 V	Integer
vll23	Phase2-3 line-line voltage	0.1 V	Integer
vl1	Phase1 to neutral voltage	0.1 V	Integer
vl2	Phase2 to neutral voltage	0.1 V	Integer
vl3	Phase3 to neutral voltage	0.1 V	Integer
eimp	imported energy counter	1 Wh	Integer
eexp	exported energy counter	1 Wh	Integer
esolar	used solar energy counter	1 Wh	Integer
egrid	used grid energy counter	1 Wh	Integer
fgrid	Measured grid frequency	0.01 Hz	Integer
thdul1	Phase 1 voltage THD	0.01 %	Integer
thdul2	Phase 2 voltage THD	0.01 %	Integer
thdul3	Phase 3 voltage THD	0.01 %	Integer
thdil1	Phase 1 current THD	0.01 %	Integer
thdil2	Phase 2 current THD	0.01 %	Integer
thdil3	Phase 3 current THD	0.01 %	Integer
gas	Used gas counter	0.01 m3	Integer
water	Used water counter	0.01 m3	Integer
heat	Used heat counter	100 J	Integer
radi	Measured radiation	0.1 W/m2	Integer
temp	Measured temperature	0.1 C	Integer
humi	Measured humidity	0.01 %	Integer
pres	Measured pressure	1000 Pa	Integer
flow	Measured flow	0.01 liter/min	Integer
weight	Measured weigth	1 gram	Integer

5.2.2.3.2 Example response body


```
[
  {
    "timestamp": "2022-12-14T08:00:00.000Z",
    "actpow": 1000,
    "actpowl1": 100,
    "actpowl2": 1200,
    "actpowl3": -300,
    "apparpow": 1005,
    "eexp": 0,
    "egrid": 13541,
    "eimp": 36578912,
    "esolar": 31575661,
    "fgrid": 5011,
    "gas": 12300,
    "il1": 1000,
    "il2": 2000,
    "il3": 500,
    "pf": 30,
    "pfl1": 50,
    "pfl2": -50,
    "pfl3": 100,
    "reactpow": 100,
    "thdil1": 100,
    "thdil2": 200,
    "thdil3": 140,
    "thdul1": 111,
    "thdul2": 15,
    "thdul3": 109,
    "ul1": 23011,
    "ul2": 24011,
    "ul3": 23544,
    "ull12": 39821,
    "ull13": 40201,
    "ull23": 39098
  },
  {
    "timestamp": "2022-12-14T08:15:00.000Z",
    "actpow": 1000,
    "actpowl1": 100,
    "actpowl2": 1200,
    "actpowl3": -300,
```

```
    "apparpow": 1005,
    "eexp": 0,
    "egrid": 13541,
    "eimp": 36578912,
    "esolar": 31575661,
    "fgrid": 5011,
    "gas": 15300,
    "il1": 1000,
    "il2": 2000,
    "il3": 500,
    "pf": 30,
    "pfl1": 50,
    "pfl2": -50,
    "pfl3": 100,
    "reactpow": 100,
    "thdil1": 100,
    "thdil2": 200,
    "thdil3": 140,
    "thdul1": 111,
    "thdul2": 15,
    "thdul3": 109,
    "ul1": 23011,
    "ul2": 24011,
    "ul3": 23544,
    "ull12": 39821,
    "ull13": 40201,
    "ull23": 39098
  }
]
```

5.2.3 GET inverter data

Method: GET | URL: https://api.hub.embion.nl/v1/inverter

This API endpoint returns individual inverter data.

5.2.3.1 Required permissions

Device tokens: (all allowed)

Third party tokens: Monitoring

5.2.3.2 Request

5.2.3.2.1 Query parameters

> **Last value**
>
> Note that the last data sample is only available within the last two hours. If both the start date and period are not defined in the call, only the last data sample is returned if data is available; otherwise, no data will be returned.

> **Data time range**
>
> - Users can optionally include a time with the start_date, which shifts the day interval. However, the total number of returned entries remains unchanged. If no time is specified, a day is considered from 00:00:00 to 23:59:59 in the selected timezone.
> - The start_date determines the beginning of the data range. The period defines the interval between returned dates, while the range specifies the end date/time relative to start_date, thereby determining the number of entries returned.

> **Period limitations**
>
> Please note there are some limitations when combining period and range
>
> - For period q (quarter-hourly) and h (hourly), the maximum range is d (one day).
> - For period d (daily), the maximum range is w (one week).
> - For period w (weekly), the maximum range is m (one month).
> - For period m (monthly), the maximum range is y (one year.)

GET inverter data query parameters

Var	Description	Mandatory	Format
ID	ID of the token (generated in the HUB)	Yes	String
uid	the uid of the inverter to read, only one uid can be entered	Yes	String
period	Select data return period: • **l** last sample (default) • **q** 15 minute • **h** hourly • **w** weekly • **d** daily • **m** monthly • **y** yearly	No	String
range	Time range to show: • **d** day (default) • **w** week • **m** month • **y** year	No	String
type	Combination type of multiple datapoints: • **min** minimum value in timerange • **max** maximum value in timerange (default) • **avg** average value in timerange	No	String
start_date	Date of the first sample in ISO 8601 layout, if not set current day is used	No	String

5.2.3.2.2 Example request

https://api.hub.embion.nl/v1/inverter?id=[id]&uid=[uid]&period=h&range=d&type=max&start_date=2022-

5.2.3.3 Response

5.2.3.3.1 Response body parameters

The inverter data is included in the JSON body of the response. The actual lay-out of the body varies depending on the query parameters given in the request.

The body can contain the following parameters, if the data is available:

var	description	units	format
timestamp	Timestamp of the measurement	ISO 8601	string
stat	Inverter status		Integer
kdy	Inverter daily yield	1 Wh	Integer
kty	Inverter total yield	1 Wh	Integer
pac	Inverter AC power	1 W	Integer
psol	Inverter solar power	1 W	Integer
ul1	Inverter phase 1 voltage	0.1 V	Integer
ul2	Inverter phase 2 voltage	0.1 V	Integer
ul3	Inverter phase 3 voltage	0.1 V	Integer
il1	Inverter phase 1 current	0.1 A	Integer
il2	Inverter phase 2 current	0.1 A	Integer
il3	Inverter phase 3 current	0.1 A	Integer
tmp1	Inverter internal temperature 1	0.1 C	Integer
tmp2	Inverter internal temperature 2	0.1 C	Integer
ilk	Inverter leakge current or isolation resistance	0.0001 A	Integer
arc	Inverter arc detection status		Integer
batpow	Battery power (+charge, -discharge)	1 W	Integer
batcap	Remaining battery capacity	1 Wh	Integer
batsoc	Battery State Of Charge	0.1 %	Integer
batsoh	Battery State Of Health	0.1 %	Integer
battemp	Battery temperature	0.1 C	Integer
string_data	Individual string data (see table below)	**stringdata**	

Definition of stringdata:

var	description	units	format
sid	string number of inverter uid		string
idc	String current	0.1 A	Integer
udc	String voltage	0.1 V	Integer
pdc	String power	1 W	Integer
ydc	String daily yield	1 Wh	Integer
sarc	String arc detection status		Integer

5.2.3.3.2 Example response body


```
{
  "timestamp": "2022-12-13T23:00:00.000Z",
```

```
    "arc": 0,
    "batcap": 0,
    "batpower": 0,
    "batsoc": 0,
    "batsoh": 0,
    "battemp": 0,
    "il1": 56,
    "il2": 63,
    "il3": 77,
    "ilk": 3,
    "kdy": 1100000,
    "kty": 6100000,
    "pac": 10000,
    "psol": 20000,
    "string_data": [
      {
        "sid": "1",
        "idc": 50,
        "udc": 5000,
        "pdc": 2500,
        "sarc": 0
      }
    ],
    "stat": 1,
    "tmp1": 531,
    "tmp2": 366,
    "ul1": 2301,
    "ul2": 2405,
    "ul3": 2508
  },
  {
    "timestamp": "2022-12-14T00:00:00.000Z",
    "arc": 0,
    "batcap": 0,
    "batpower": 0,
    "batsoc": 0,
    "batsoh": 0,
    "battemp": 0,
    "il1": 120,
    "il2": 130,
    "il3": 120,
    "ilk": 3,
```

```
    "kdy": 1200000,
    "pac": 14000,
    "string_data": [
      {
        "sid": "1",
        "idc": 50,
        "udc": 5000,
        "pdc": 2500,
        "sarc": 0
      }
    ],
    "stat": 1,
    "tmp1": 551,
    "tmp2": 346,
    "ul1": 2301,
    "ul2": 2405,
    "ul3": 2508
  }
```

5.3 Plant control

5.3.1 Introduction

The plant control API endpoints allow users to manage and schedule plant control commands for their EMS Controllers. With these plant control commands, it is possibly to remotely control various processes of the connected devices, such as limiting power import/export, controlling battery charging/discharging, and managing the operation of controllable loads.

5.3.1.1 Triggering methods

> **Note**
>
> User-defined and energy pricing based scheduled commands are only available on Embion EMS Controllers running software version 5.2.0 or higher.

There are three different triggering methods for plant control commands:

1. **Instantaneous commands (via the public API)**: These commands are sent via the POST plant control command endpoint and are executed immediately on the device. They are not stored in the plant control schedule.
2. **User-defined scheduled commands (via the public API)**: These commands are scheduled for a specific time in the future and are created via the POST plant control schedule endpoint. They allow users to plan and automate actions based on their requirements. Their source is set to public_api in the plant control schedule.
3. **Energy pricing based scheduled commands (automatically generated)**: These commands are triggered based on the plant control rules and pricing scheme set in the Energy pricing app of the HUB. They are automatically generated and added to the plant control schedule with energy_pricing as their source.

5.3.1.2 Activation and prioritization

Since there are multiple ways to trigger plant control commands, it is important to understand how these commands are activated and prioritized on the device. The following rules apply:

- **Instantaneous commands** are always executed immediately and take precedence over any scheduled commands. They override any currently active scheduled commands. If the merge parameter is set to true when sending an instantaneous command, the command will be merged with any currently active scheduled commands instead of overriding them.
- **Scheduled commands** contain a single plant control command action (unlike the instantaneous commands) and are activated based on their defined start (dtstart) and end (dtend) times. If multiple scheduled commands are active at the same time, the command with the highest priority will be executed. The priority is determined by the source of the command, with user-defined scheduled commands (public_api) having a higher priority than energy pricing based scheduled commands (energy_pricing). Schedule items with different command actions (e.g. control_generation and p_export_limit) can be active at the same time, as they control different aspects of the plant operation.

The diagram below illustrates the activation and prioritization of plant control commands on the device:

Plant control activation priorities

5.3.2 GET plant control

Method: GET | URL: https://api.hub.embion.nl/v1/read_plantcontrol

This endpoint returns the currently active plant control command (if one is active on the device).

5.3.2.1 Required permissions

Device tokens: (all allowed)

Third party tokens: Monitoring

5.3.2.2 Request

5.3.2.2.1 Query parameters

GET plant control query parameters

Var	Description	Mandatory
ID	ID of the token (generated in the HUB)	Yes

5.3.2.2.2 Example request

https://api.hub.embion.nl/v1/read_plantcontrol?id=[id]

5.3.2.3 Response

5.3.2.3.1 Response body parameters

Please note that the dtcreated, dtupdated and valid_time fields are always present. Other fields may be present depending on the active control command. If no command is active, the string "No plantcontrol command active" will be returned instead.

var	description	Units	Format
dtcreated	Time that the control command is created	ISO 8601	String
dtupdated	Time that the control command is updated	ISO 8601	String

var	description	Units	Format
valid_time	Time in seconds that the control commando is being active	Seconds	Int
p_import_limit	Grid import limit	W	Int
p_export_limit	Grid export limit	W	Int
rel_p_import_limit	Relative grid export limit	W	Int
rel_p_export_limit	Relative grid export limit	W	Int
control_generation	Control generation	-	String
control_consumption	Control consumption	-	String
control_pv_limit	Control plant PV limit	%	Int
control_battery_setpoint	Control battery setpoint	%	Int
control_ev_limit	Control plant EV limit	%	Int

5.3.2.3.2 Example response body

A plant control command is active:


```
{
  "dtcreated": "2025-08-14T14:35:30.056Z",
  "dtupdated": "2025-08-14T14:35:30.056Z",
  "valid_time": 90,
  "p_export_limit": 100,
  "p_import_limit": 100,
  "control_ev_limit": 100,
  "control_pv_limit": 100,
  "control_generation": "min",
  "rel_p_export_limit": 100,
  "rel_p_import_limit": 100,
  "control_consumption": "nom",
  "control_battery_setpoint": 100
}
```


No command is active:


```
"No plantcontrol command active"
```

5.3.3 POST plant control command

Method: POST | URL: https://api.hub.embion.nl/v1/plantcontrol

This endpoint allows you to send a plant control command to the device, allowing you to control various aspects of the plant's operation. This includes adjusting import/export limits, controlling generation and consumption, and setting battery and EV limits.

While it is possible to specify all parameters in a single command, it is recommended to only include the parameters that need to be changed. This helps to reduce the complexity of the command and makes it easier to understand. The Embion EMS Controller will also check if certain limits are exceeded and adjust them accordingly.

If a plant control action is still active when a new command is sent, the previous command will be overwritten or merged based on the request, and the return message will be updated accordingly.

5.3.3.1 Limits

The following limits are in place regarding the parameters that can be sent in a plant control command:

- valid_time:
  - If omitted or set to 0 the command is considered infinite (no expiry).
  - If provided and non-zero, the value must be at least 90 (seconds). Values less than 90 will be rejected with the valid_time_too_short error.
- Commands without actions (no limits or control_* fields):
  - May only be sent when a finite valid_time is explicitly provided (i.e., valid_time is present and more than or equal to 90). This prevents sending empty, non-expiring commands.
- General notes:
  - Prefer sending only the parameters you intend to change.
  - When merge is set to true, new values are merged with existing active commands; otherwise they overwrite them.
  - The latest valid_time among merged commands is used.

5.3.3.2 Required permissions

Device tokens: Control access

Third party tokens: Control

5.3.3.3 Request

5.3.3.3.1 Body parameters

> **Note**
>
> Some parameters are only available after specific software versions. These parameters are marked with a version tag (e.g. [^4.2.0]). This example indicates that the parameter is available from version 4.2.0 onwards. Please ensure that your device is running the required software version to utilize these parameters.

Conflicting setpoints

If conflicting setpoints are provided, the setpoint resulting in the lowest power output will take precedence.

Merging commands

When merge is not used, new commands will overwrite the previous commands. If merge is used, previously set limits will be merged with the new command.
The latest given valid_time will be applied (it will be infinite if the value is 0 or the valid_time variable wasn't given).

POST plant control command body parameters

Var	Description	Mandatc	Format	Unit
id	ID of the token	Yes	String	
p_import_limit	Grid import limit	No	Integer	W
p_export_limit	Grid export limit	No	Integer	W
rel_p_import_limit [^4.2.0]	Relative grid import limit	No	Integer	%
rel_p_export_limit [^4.2.0]	Relative grid export limit	No	Integer	%
control_generation	• **min** minimize generation • **max** maximize generation • **nom** nominal generation	No	String	
control_consumption	• **min** minimize consumption • **max** maximize consumption • **nom** nominal consumption	No	String	
control_pv_limit [^4.2.0]	Control plant PV limit	No	Integer	%
control_battery_setpoint (+charge, -discharge)[^4.2.0]	Control plant battery setpoint	No	Integer	%
control_ev_limit [^4.2.0]	Control plant EV limit	No	Integer	%
min_bess_soc [^5.2.0]	Minimum BESS state of charge	No	Integer	%
max_bess_soc [^5.2.0]	Maximum BESS state of charge	No	Integer	%
des_bess_soc [^5.2.0]	Desired BESS state of charge	No	Integer	%
valid_time	Time in seconds that the given command stays active on the GSE (must be equal to or greater than **90**). Will be infinite if the value is **0** or the variable wasn't given.	No	Integer	sec.
	If true and a current plant control command is active, it will be merged with the given commands			

The control_generation and control_consumption items can be used to control plant generation and consumption independently of the plant configuration.

Plant control actions

Var	Value	Description
control_generation	min	Reduces the power generation to the minimum, resulting is solar power converters to shutdown and wind turbines to stop.
	nom	Allows generation of solar and wind to operate normally.
	max	Allows also the start of any extra generators (if available at plant).
control_consumption	min	Reduces the controlable loads like heatpumps and EV-chargers to minimum consumption.
	nom	Enables normal controlable loads to operate within the plant limits.
	max	Increases the power for controlable loads to maximum. EV-chargers will increase charging power to maximum (within plant limits) and heatpumps will increase or decrease setpoint to increase power consumption.

5.3.3.3.2 Example request

https://api.hub.embion.nl/v1/plantcontrol

```
{
  "id": "119mt001pj51d",
  "p_export_limit": 20000,
  "p_import_limit": 50000,
  "control_generation": "max",
  "control_consumption": "nom",
  "valid_time": 200
}
```

5.3.3.4 Response

The response body contains info about whether the command was successfully sent.

5.3.3.4.1 Response body parameters

var	descripion	format	optional
success	Whether the command was sent (true = sent)	boolean	No
value	Optional description message	string	Yes

The value field shows up if the command couldn't be sent or when an existing command was overwritten. The field can have any of the following values:

Plant control values

Var	Description
unsupported	The plant control feature is not supported on the device
disabled	The plant control feature is actively disabled by the device
valid_time_too_short	The **valid_time** field must be equal to or greater than **90**, if it isn't this error is shown
offline	The device is offline (no data or status received in the last 2 minutes)
overwritten	The previous command will be overwritten
merged	The previous command is merged with the provided one
Hub command is currently active	A plant control command from the HUB interface (such as an asset status check) is currently active. Wait for the HUB command to complete before sending a new command via the API.

5.3.3.4.2 Example response body

Plant control command successfully sent

```
{
  "success": true
}
```

Plant control command couldn't be sent (plant is offline)

```
{
   "success": false,
   "value": "offline"
}
```


Plant control command couldn't be sent (HUB command is active)


```
{
   "success": false,
   "value": "Hub command is currently active"
}
```

5.3.4 GET plant control schedule

Method: GET | URL: https://api.hub.embion.nl/v1/plantcontrol/schedule

This endpoint returns all plant control commands that were scheduled for the given device.

5.3.4.1 Required permissions

Device tokens: (all allowed)

Third party tokens: Monitoring

5.3.4.2 Request

5.3.4.2.1 Query parameters

GET plant control schedule request query parameters

Var	Description	Mandatory
id	ID of the token (generated in the HUB)	Yes
dtstart	Start date and time of the schedule (ISO 8601 format)	No
dtend	End date and time of the schedule (ISO 8601 format)	No
source	Source of the schedule (e.g. public_api, energy_pricing)	No
command_name	Name of plant control command action (e.g. control_generation, p_export_limit)	No

5.3.4.2.2 Example request

https://api.hub.embion.nl/v1/plantcontrol/schedule?id=[id]

5.3.4.3 Response

5.3.4.3.1 Response body parameters

The response body contains the plant control schedule in JSON format. The body will contain the following parameters:

var	description	format
id	ID of the command	string
deviceid	ID of the device	string
command_name	Command action name (see the request body of POST Plant control)	string
command_value	Value associated with the command action (see the request body of POST Plant control)	string
dtstart	Start date and time of the command (ISO 8601 format)	string
dtend	End date and time of the command (ISO 8601 format)	string
source	Source of the command (e.g. public_api, energy_pricing)	string

5.3.4.3.2 Example response body


```
[
  {
    "id": "1r0ac125uj51d",
    "deviceid": "1quxh001ob51d",
    "command_name": "p_import_limit",
    "command_value": "1000",
    "dtstart": "2025-09-04T06:00:00.000Z",
    "dtend": "2025-09-04T06:15:00.000Z",
    "source": "energy_pricing"
  },
  {
    "id": "1r0ac126uj50d",
    "deviceid": "1quxh001ob51d",
    "command_name": "p_import_limit",
    "command_value": "1000",
    "dtstart": "2025-09-04T06:15:00.000Z",
    "dtend": "2025-09-04T06:30:00.000Z",
    "source": "energy_pricing"
  },
  {
```

```
    "id": "1r0ac127uj51d",
    "deviceid": "1quxh001ob51d",
    "command_name": "p_import_limit",
    "command_value": "1000",
    "dtstart": "2025-09-04T06:30:00.000Z",
    "dtend": "2025-09-04T06:45:00.000Z",
    "source": "energy_pricing"
  },
  {
    "id": "1r0ac128uj50d",
    "deviceid": "1quxh001ob51d",
    "command_name": "p_import_limit",
    "command_value": "1000",
    "dtstart": "2025-09-04T06:45:00.000Z",
    "dtend": "2025-09-04T07:00:00.000Z",
    "source": "energy_pricing"
  }
]
```

5.3.5 POST plant control schedule

Method: POST | URL: https://api.hub.embion.nl/v1/plantcontrol/schedule

This endpoint allows you to create new plant control schedule items for a device, so that you can schedule when specific plant control commands should be executed.

5.3.5.1 Limits

There is a maximum limit of 500 schedule items per command action (control_generation, p_import_limit, etc...) per device. If you attempt to add more than this limit, the API will return a response with status code 200 OK and the following body:

```
{
  "success": false,
  "value": "Schedule item limit exceeded for p_import_limit: would have 501 items (limit: 500)"
}
```

Schedule items with the same command_name that have overlapping time periods (i.e., their dtstart and dtend values overlap) will be considered colliding and the API will reject the request with a response body like this:

```
{
  "success": false,
  "value": "Plant control commands collide with existing commands"
}
```

5.3.5.2 Required permissions

Device tokens: Control access

Third party tokens: Control

5.3.5.3 Request

5.3.5.3.1 Body parameters

The request body must be a JSON array containing one or more command objects. Each command object should include the following parameters:

var	description	format	mandatory
dtstart	Start date and time of the command (ISO 8601 format)	string	Yes
dtend	End date and time of the command (ISO 8601 format) (command stays active indefinitely if not given)	string	No
command_name	Name of the command action to execute	string	Yes
command_value	Value associated with the command action	string or integer	Yes

5.3.5.3.2 Example request body

```
[
  {
    "dtstart": "2025-08-25T17:00:00Z",
    "dtend": null,
    "command_name": "control_generation",
    "command_value": "max"
  },
  {
    "dtstart": "2025-08-26T17:00:00Z",
    "dtend": "2025-08-28T17:00:00Z",
    "command_name": "control_consumption",
    "command_value": "max"
  },
  {
    "dtstart": "2025-08-26T11:00:00Z",
    "dtend": "2025-08-28T16:00:00Z",
    "command_name": "p_export_limit",
    "command_value": 0
  }
]
```

5.3.5.4 Response

5.3.5.4.1 Response body parameters

The response body contains the following parameters:

var	description	format
success	Whether the schedule items were successfully stored (true = stored)	boolean
value	Optional description message	string

5.3.5.4.2 Example response body

Plant control schedule successfully sent:

```
{
  "success": true,
  "value": "Plant control schedule items inserted successfully"
}
```

Plant control schedule couldn't be sent (due to collisions with existing commands):

```
{
  "success": false,
  "value": "Plant control commands collide with existing commands"
}
```

Plant control schedule couldn't be sent (due to exceeding schedule item limit):

```
{
  "success": false,
  "value": "Schedule item limit exceeded for control_generation: would have 501 items (limit: 500)"
}
```

5.3.6 DELETE plant control schedule

Method: DELETE | URL: https://api.hub.embion.nl/v1/plantcontrol/schedule

This endpoint allows you to delete existing plant control schedule items for a device. Only schedule items that were created via the public API can be deleted using this endpoint. Schedule items generated by other sources (e.g. energy pricing) cannot be deleted.

It is either possible to delete all schedule items for a device, or to delete all schedule items with a specific command action name. This can be done by providing the respective command_name query parameter.

5.3.6.1 Required permissions

Device tokens: Control access

Third party tokens: Control

5.3.6.2 Request

5.3.6.2.1 Query parameters

DELETE plant control schedule request query parameters

Var	Description	Mandatory
id	ID of the token (generated in the HUB)	Yes
command_name	Name of the command action (deletes all schedule items with this command action name if given)	No

5.3.6.2.2 Example request

https://api.hub.embion.nl/v1/plantcontrol/schedule?id=[id]

5.3.6.3 Response

5.3.6.3.1 Response body parameters

The response body contains the following parameters:

var	description	format
success	Whether the schedule item was successfully deleted (true = deleted)	boolean
value	Optional description message	string

5.3.6.3.2 Example response body

```
{
  "success": true,
  "value": "All scheduled plant control commands removed successfully"
}
```

5.3.7 DELETE plant control schedule item by ID

Method: DELETE | URL: https://api.hub.embion.nl/v1/plantcontrol/schedule/<schedule_item_id>

This endpoint allows you to delete a specific plant control schedule item by its ID. Only schedule items that were created via the public API can be deleted using this endpoint. Schedule items generated by other sources (e.g. energy pricing) cannot be deleted.

5.3.7.1 Required permissions

Device tokens: Control access

Third party tokens: Control

5.3.7.2 Request

5.3.7.2.1 URL parameters

DELETE plant control schedule request URL parameters

Var	Description	Mandatory
schedule_item_id	ID of the plant control schedule item to be deleted	Yes

5.3.7.2.2 Query parameters

DELETE plant control schedule by ID request query parameters

Var	Description	Mandatory
id	ID of the token (generated in the HUB)	Yes

5.3.7.2.3 Example request

https://api.hub.embion.nl/v1/plantcontrol/schedule/<schedule_item_id>?id=[id]

5.3.7.3 Response

5.3.7.3.1 Response body parameters

The response body contains the following parameters:

var	description	format
success	Whether the schedule item was successfully deleted (true = deleted)	boolean
value	Optional description message	string

5.3.7.3.2 Example response body

```
{
   "success": true,
   "value": "Plant control schedule item removed successfully"
}
```

5.4 Third party information

5.4.1 GET connected devices

Method: GET | URL: https://api.hub.embion.nl/v1/logged-in-third-party/devices

This endpoint returns all devices that are connected to the logged in third party. It is therefore only accessible with a third party token. The response contains a combination of the device serial number and an ID to use in API calls related to that device.

5.4.1.1 Required permissions

Third party tokens: Monitoring

5.4.1.2 Request

5.4.1.2.1 Query parameters

Query parameters

Var	Description	Mandatory
ID	ID of the third party token	Yes

5.4.1.2.2 Example request

https://api.hub.embion.nl/v1/logged-in-third-party/devices?id=[id]

5.4.1.3 Response

5.4.1.3.1 Response body parameters

The response body is a JSON array of objects, each representing a connected device.

var	description	format
id	ID of the device (for use in other API calls)	string
serial_number	Serial number of the device	string

5.4.1.3.2 Example response body


```
[
  {
    "id": "1q6dz002pp50d",
    "serial_number": "05002516010026"
  },
  {
    "id": "1qayl002cf50d",
    "serial_number": "05002516010029"
  },
  {
    "id": "1qbu3002bs50d",
    "serial_number": "01002309010028"
  }
]
```

5.4.2 GET connected device by its serial number

Method: GET | **URL:** https://api.hub.embion.nl/v1/logged-in-third-party/devices/[serial number]

This endpoint returns information of a device that corresponds to the given serial number if it is connected to the logged in third party. This endpoint is only accessible with a third party token. The response contains a combination of the device serial number and an ID to use in API calls related to that device.

5.4.2.1 Required permissions

Third party tokens: Monitoring

5.4.2.2 Request

5.4.2.2.1 URL parameters

URL parameters

Var	Description	Mandatory
serial_number	Serial number of the device	Yes

5.4.2.2.2 Query parameters

Query parameters

Var	Description	Mandatory
ID	ID of the third party token	Yes

5.4.2.2.3 Example request

https://api.hub.embion.nl/v1/logged-in-third-party/devices/<serial number>?id=[id]

5.4.2.3 Response

5.4.2.3.1 Response body parameters

var	description	format
id	ID of the device (for use in other API calls)	string
serial_number	Serial number of the device	string

5.4.2.3.2 Example response body


```
{
   "id": "1qayl002cf50d",
   "sn": "05002516010029"
}
```

All products described in this document are owned by **Embion B.V.**

Address

Embion B.V.
Kalundborg 10
5026 SE, Tilburg

Contact

www.embion.eu
info@embion.nl