

# DMWeb API for DataMailbox

Talk2M DataMailbox SDK – 21.1.0

## REFERENCE GUIDE

RG-0005-00 2.2 en-US ENGLISH

---

# Important User Information

## Disclaimer

The information in this document is for informational purposes only. Please inform HMS Networks of any inaccuracies or omissions found in this document. HMS Networks disclaims any responsibility or liability for any errors that may appear in this document.

HMS Networks reserves the right to modify its products in line with its policy of continuous product development. The information in this document shall therefore not be construed as a commitment on the part of HMS Networks and is subject to change without notice. HMS Networks makes no commitment to update or keep current the information in this document.

The data, examples and illustrations found in this document are included for illustrative purposes and are only intended to help improve understanding of the functionality and handling of the product. In view of the wide range of possible applications of the product, and because of the many variables and requirements associated with any particular implementation, HMS Networks cannot assume responsibility or liability for actual use based on the data, examples or illustrations included in this document nor for any damages incurred during installation of the product. Those responsible for the use of the product must acquire sufficient knowledge in order to ensure that the product is used correctly in their specific application and that the application meets all performance and safety requirements including any applicable laws, regulations, codes and standards. Further, HMS Networks will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features or functional side effects found outside the documented scope of the product. The effects caused by any direct or indirect use of such aspects of the product are undefined and may include e.g. compatibility issues and stability issues.

---

# Table of Contents

Page

<b>1</b>	<b>Preface .....</b>	<b>3</b>
1.1	About this document .....	3
1.2	Document history .....	3
1.3	Related documents .....	3
1.4	Trademark information .....	3
<b>2</b>	<b>Introduction.....</b>	<b>4</b>
2.1	Prerequisites .....	4
2.2	Talk2M DataMailbox SDK .....	4
<b>3</b>	<b>Ewon Configuration.....</b>	<b>6</b>
3.1	Data Synchronizing Parameters .....	6
3.2	Synchronization by Script .....	7
3.3	Synchronization Status.....	7
<b>4</b>	<b>DMWeb API .....</b>	<b>8</b>
4.1	Request Structure .....	8
4.2	Response Structure .....	8
4.3	Authentication.....	8
4.4	Quota .....	9
4.5	Date format .....	10
4.6	Data Services.....	14
4.7	Talk2M Token Management.....	28

**This page intentionally left blank**

# 1 Preface

## 1.1 About this document

This document explains how to use the DMWeb API to generate RESTful API requests to the Talk2M DataMailbox and to easily retrieve data stored in the Talk2M DataMailbox.

For additional related documentation and file downloads, please visit [developer.ewon.biz](https://developer.ewon.biz).

## 1.2 Document history

Version	Date	Description
1.0	2018-05-03	First release
1.1	2018-08-29	Added: GET or POST content.
1.2	2016-08-31	Changed: Version adjustment to match SDK version
1.3	2018-02-02	Added: getstatus sub-chapter Added: Quality tag parameter of getData Changed: syncdata typo
1.4	2018-11-09	Changed: <a href="#">Date format, p. 10</a>
1.5	2019-01-21	Changed: Talk2M SDK version
1.6	2019-04-03	Changed: Examples in <a href="#">&lt;syncdata&gt;, p. 15</a>
1.7	2019-07-31	Added: “eworlds” parameter in <a href="#">&lt;syncdata&gt;, p. 15</a>
1.8	2019-08-20	Changed: Encode URL parameters in <a href="#">Authentication, p. 8</a> .
1.9	2019-11-07	Changed: <a href="#">&lt;syncdata&gt;, p. 15</a>
2.0	2020-09-04	Added: Integration of Talk2M Token in <a href="#">Authentication, p. 8</a> Changed: GET method is no longer supported in <a href="#">Authentication, p. 8</a>
2.1	2020-10-28	Changed: <a href="#">Backward Compatibility, p. 9</a>
2.2	2020-02-18	Added: <a href="#">Quota, p. 9</a> Changed: <a href="#">&lt;getdata&gt;, p. 21</a>

## 1.3 Related documents

Document	Author	Document ID
General Reference Guide	HMS	RG-0001-00

## 1.4 Trademark information

Ewon® is a registered trademark of HMS Industrial Networks SA. All other trademarks mentioned in this document are the property of their respective holders.

## 2 Introduction

The DMWeb API is a RESTful web service combined with the Talk2M service “DataMailbox” which allows easy retrieval of Ewon gateway historical data.

Application developers can easily write code to retrieve historical data of multiple Ewon gateway using the DataMailbox without the need of learning a whole new environment.

The Ewon gateway pushes its historical data to the DataMailbox running on Talk2M servers. This historical data is temporarily stored and is then available using the DMWeb protocol.

If the DataMailbox acts as a temporary storage, there is a limitation in the capacity that can be stored: a point can be stored for 10 days maximum, beginning when it was sent to the DataMailbox.

This HTTP based protocol allows third party applications to retrieve data from the DataMailbox in an easy way.

If you wish to learn more about the pricing of the DataMailbox, please refer to the [Talk2M Pricing](#) web page.

### 2.1 Prerequisites

To use correctly the DMWeb protocol, you need prerequisites such as:

- **Talk2M Developer ID**  
You also need a Talk2M Developer ID to send requests to the Talk2M servers. This Talk2M Developer ID can be requested by sending a web form on [Ewon Developer](#) website.
- **Talk2M Token**  
You need a Talk2M Token — a secured ID generated by eCatcher — to send requests to the Talk2M servers targeting an Ewon gateway or a group of Ewon gateways.

The Talk2M Developer ID and the Talk2M Token must be combined to received data from the Talk2M APIs:

- The Talk2M Developer ID tells the Talk2M servers that you are allowed to use the Talk2M APIs.
- The Talk2M Token tells the Talk2M servers which Talk2M account you are allowed to access, and more specifically which Ewon gateway(s) or pool(s) of Ewon gateway(s) you can interact with.

### 2.2 Talk2M DataMailbox SDK

This document is part of the Talk2M DataMailbox SDK which contains everything you need to get the DMWeb API running in your project.

The DataMailbox is explained in the present document. The DataMailbox folder from the Talk2M DataMailbox SDK should contain:

- **This reference guide: RG-0005-00**

- **Viewer**

“Talk2M DataMailbox Viewer” is a software that offers the possibility to see how the DMWeb API works, especially with the Talk2M token.

The software helps you by proposing which request should be sent to the DataMailbox and shows its result immediately.

It is useful to check if the request will succeed or not, which information you want to retrieve... before going in to production

To use the Viewer software, simply double-click on the “DMBoxViewer.exe” file.

- **DataMailbox Samples**

This folder contains applications with their source codes. Applications that are:

- My Little Historian

Sample C# console application that retrieves the contents of the DataMailbox using the DMWeb “syncdata” mechanism, outputs the contents in text files (one sub-directory per Ewon gateway, one text file per tag) and deletes the contents of the DataMailbox.

This sample program shows how to download data using compressed HTTPS in C#, turns it into a dynamic .NET object, browses through its contents and uses the DMWeb transaction mechanism.

Requirement: Visual Studio 2010+ / .NET 4.0+ / C#

- DMBoxViewer

The source code of theTalk2M Datamailbox Viewer, which is explained here above, can be found in this folder.

It demonstrates the creation of a DMWeb URL, including the various options of each API call and the download of the DataMailbox contents.

Requirement: Visual Studio 2010+ / .NET 4.0+ / C#

## 3 Ewon Configuration

The Ewon gateway shown as example in this section has already been configured with tags that are retrieved from a PLC. If you need help with the configuration of your Ewon gateway to retrieve data from a third-party device and store this data as tags, please refer to the [Ewon support website](#).

### 3.1 Data Synchronizing Parameters

To enable the historical data of your Ewon gateway in the DataMailbox, go to: **Setup > System > Main > Net services > Data Management**.

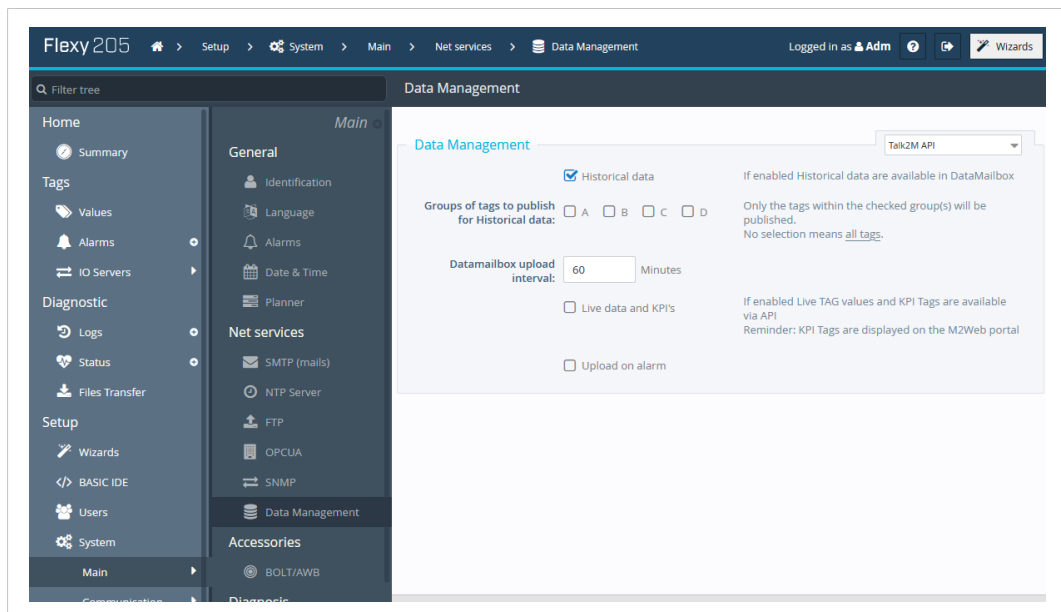


Fig. 1 Data Management web configuration

#### 3.1.1 Talk2M API

As the Data Management is set to “Talk2M API”, check the “Historical data” box. This will tell your Ewon gateway to send its historical data to the DataMailbox.

Several options can be modified:

<b>Groups of tags to publish</b>	This option allows the synchronization of the historical data only for a few tags instead of all Ewon tags.  When a tag group is selected, the Ewon gateway exports the historical data values only for the tags belonging to the selected group.  If no group is selected, than all tags are sent.
<b>Upload interval</b>	Specifies the synchronization interval. For example “45”, to synchronize every 45 minutes.  Default value: 60.
<b>Upload on alarm</b>	When selected, the data synchronization will also be triggered when one of the tags of the selected groups triggers an alarm.

The group of tags filter applies only to the historical stored values of the Ewon gateway. The filter is not applied on the real-time and alarm values. Therefore, the real-time values and alarm info of all tags will still be synchronized.



## 3.2 Synchronization by Script

Beside the automatic synchronization described here above, it is also possible to trigger the data synchronization using BASIC script.

The BASIC function “DMSync” triggers the Ewon to synchronize its data.

To trigger a single synchronization, you can proceed as follows:

- open the Ewon Basic IDE section,
- make sure the console frame is opened,
- Type “DMSync” in the console frame and hit “Enter”.

A data synchronization is then immediately initiated.

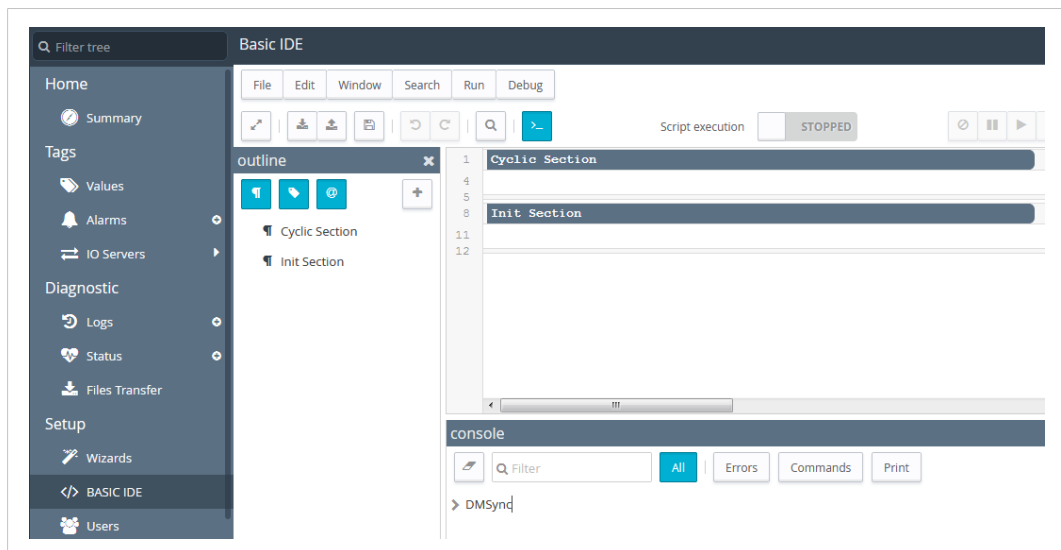


Fig. 2

## 3.3 Synchronization Status

To check if the synchronization succeeded, open the diagnostic window of the Ewon gateway by going to: **Diagnostic > Logs > Event Logs**.

Make sure the level of the logs is set *Trace*.

The event log should contain 2 messages:

- one for the synchronization start.
- one for the synchronization end.

If so, the Ewon has transferred the data to the DataMailbox. You can now have a look on the server side to visualize the received data.

## 4 DMWeb API

The DMWeb API is used to retrieve the historical data of all your Ewons from the DataMailbox. Using this API, you will be able:

- read the data stored in the DataMailbox,
- delete the data from the DataMailbox.

### 4.1 Request Structure

The hostname of the DMWeb API is:

```
https://data.talk2m.com/
```

### 4.2 Response Structure

The API response is formatted in [json](#). It always contains the following value:

<b>success</b>	A boolean value. It is set to <i>true</i> when the API was treated without any issue. It is set to <i>false</i> when an error occurred during the treatment of the response.
<b>code</b>	An integer The error code displayed when the request didn't succeed. This field is shown only when "success" returns <i>false</i> .
<b>message</b>	A string The description explaining the issue that occurred. This field is shown only when "success" returns <i>false</i> .

### 4.3 Authentication

You must have the right credentials to be able to use the Talk2M APIs,

The credentials have to be defined in the content of the HTTP POST query.

The following parameters must be provided:

- **<t2mdevid>**
- **<t2mtoken>**

The **<t2mdevid>** is a Talk2M Developer ID specific to the DataMailbox. To request one, fill in the web form that can be found on [Ewon Developers](#) website.

The **<t2mtoken>** is an ID generated by the administrator of a Talk2M account. This Talk2M token gives access to an Ewon gateway or to a pool of Ewon gateways. You can read more about the **<t2mtoken>** in the [Talk2M Token, p. 9](#).



Those parameters must be encoded as UTF-8 strings. The way to "percent encode" a UTF-8 character is to encode each bytes of its UTF-8 sequence (e.g.: "é" becomes "%C3%A9", "ø" becomes "%E2%8D%B5", "#" becomes %23).

### 4.3.1 Talk2M Token

API tokens are kind of a user login but limited to API usage.

The Talk2M account administrator creates a token for each application or 3rd party platform that they plan to use.

The API Token allows access to the Ewon gateways of one selected pool.

The list of tokens is created and maintained in a specific section of the *Account* page in the eCatcher software. To learn more on how to manage the Talk2M token in eCatcher, please refer to [Talk2M Token Management, p. 28](#)

A token is a string of maximum 50 characters (ASCII numbers and letters, uppercase and lowercase).

You can read more about the Talk2M token in the [Talk2M Token Management, p. 28](#)

### 4.3.2 Backward Compatibility

As former versions of the Talk2M DataMailbox SDK were using other parameters to authenticate, we maintain a backward compatibility, which still involves the Talk2M token, so you can continue to use your existing application by specifying the Talk2M token inside a username and password combination.

The following parameters must be provided:

- **<t2maccount>**
- **<t2mdevid>**
- **<t2musername>** which must be set to: `t2m-api-token`
- **<t2mpassword>** which must be filled with your Talk2M token

However, if in the past we could use the HTTP GET request, the credentials must now be in an HTTP POST request. GET HTTP requests will receive an HTTP 405 – Method Not Allowed error.



Those parameters, especially the password parameter, must be encoded as UTF-8 strings. The way to “percent encode” a UTF-8 character is to encode each bytes of its UTF-8 sequence (e.g.: “é” becomes “%C3%A9”, “ø” becomes “%E2%8D%B5”, “#” becomes %23).

## 4.4 Quota

There is a data quota when using the DataMailbox.

Depending on the type of your Talk2M account, the maximum number of records that an Ewon gateway can send per month to the DataMailbox is as follows:

- For Talk2M Free+ account: 5 millions datapoints.
- For Talk2M PRO account: 10 millions datapoints.

If a Talk2M account exceeds this limit, it will be capped. An Ewon gateway that belongs to a capped Talk2M account will no longer be able to send its records to the DataMailbox.

The response of an API request to the DataMailbox from a capped Talk2M account has the following header `X-Talk2M-Message` with the following message `Your datamailbox quota exceeded, uploads are blocked.`

## 4.5 Date format

The format of the dates used by the DataMailbox follows the standard [ISO 8601](#).

The JSON feed sent by the DataMailbox API responses shows the dates under the following format:

```
2015-02-24T03:03:42Z
```

For the dates used as parameters sent to the DataMailbox, they may have the following format :

```
2015-02-24T03:03:42Z  
  
2015-02-24T03:03:42  
  
2015-02-24T03:03  
// meaning 2015-02-24T03:03:00Z  
  
2015-02-24  
// meaning 2015-02-24T00:00:00Z
```

The dates used in the URL must be encoded in a special format. The colon “:” character needs to be encoded as “%3A”:

```
2015-02-24T03%3A03%3A42Z
```

#### 4.5.1 Ewon Timestamps Logged in UTC

Before firmware 13.2, the Ewon gateway is always logging data in local time.

As of firmware 13.2, the Ewon gateway has the option to record data using UTC timestamps.

To switch to UTC timestamps, enable the “Record data in UTC” option which is represented by a checkbox in the configuration of your Ewon gateway date & time.

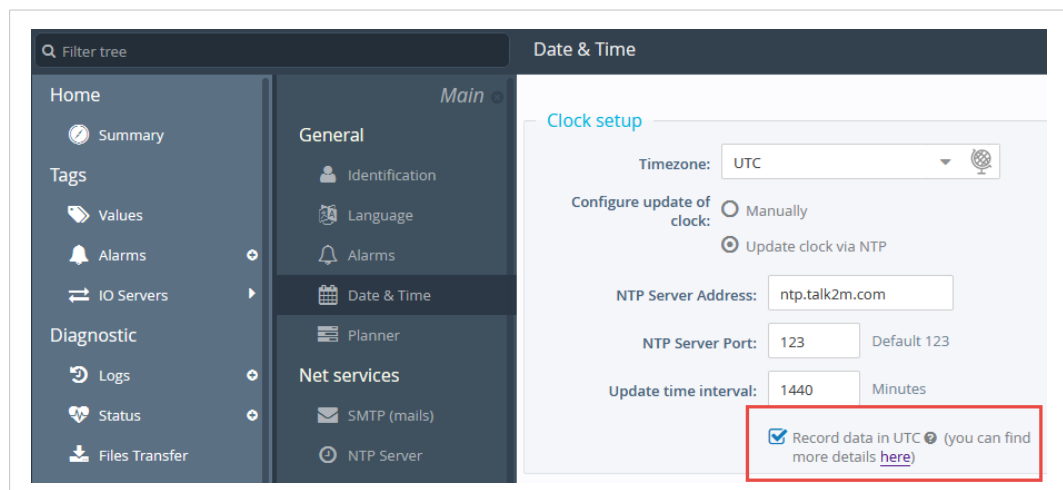


Fig. 3 Record data using UTC timestamps

If the “Record data in UTC” option is checked, the DMWeb API will add an extra parameter in its JSON feed. This extra parameter tells the applications using the DMWeb API that the data is recorded in UTC.

This extra parameter is called **timeZone** and returns the UTC + timezone of your Ewon gateway.

If your Ewon gateway does not record in UTC, as the **<Record data in UTC>** option is unchecked, the **timeZone** parameter will not appear in the DMWeb JSON feed.



The value of the **timeZone** parameter is based on TZ time zone. For more information, check [https://www.wikiwand.com/en/List\\_of\\_tz\\_database\\_time\\_zones](https://www.wikiwand.com/en/List_of_tz_database_time_zones).

This parameter comes with conditions:

- The time zone is maximum 40 characters long.
- The format of the dates does not change. It remains the same regardless how the data is logged in the Ewon gateway.

#### Request concerned by the **timeZone** parameter

The following DMWeb API requests will have the **timeZone** parameter if the “Record data in UTC” option – from the Ewon gateway date & time configuration – is enabled.

### <getewons>

This request returns the **timeZone** field (if available) for each Ewon gateway listed in the response feed.

#### Example 1: **timeZone** inside the <getewons> response

```
{
  "success":true,
  "ewons":[
    {
      "id":3003,
      "name":"ExampleName",
      "timeZone":"Europe/Brussels",
      "lastSynchroDate":"2018-10-30T16:02:01Z"
    }
  ]
}
```

### <getewon>

This request returns the **timeZone** field (if available) of a specific Ewon gateway listed in the response feed.

#### Example 2: **timeZone** inside the <getewon> response

```
{
  "success":true,
  "id":3003,
  "name":"ExampleName",
  "tags":[
    {
      "id":3223,
      "name":"TagName",
      "dataType":"Float",
      "description":"Description for tag",
      "alarmHint":"",
      "value":19.0,
      "quality":"good",
      "ewonTagId":3
    }
  ],
  "timeZone":"Europe/Brussels",
  "lastSynchroDate":"2018-10-30T16:02:01Z"
}
```

### <getdata>

This request returns the **timeZone** field (if available) of a specific Ewon gateway listed in the response feed.

#### Example 3: **timeZone** inside the <getdata> response

```
{
  "success":true,
  "ewons":[
    {
      "id":3003,
      "name":"ExampleName",
      "tags":[
        {
          "id":3223,
          "name":"TagName",
```

```

        "dataType":"Float",
        "description":"Description for tag",
        "alarmHint":"",
        "value":19.0,
        "quality":"good",
        "ewonTagId":3,
        "history":[
            {
                "date":"2014-04-16T16:34:16Z",
                "value":"4.0"
            },
            {
                "date":"2014-04-17T16:34:26Z",
                "value":5.0
            }
        ]
    },
    "timeZone":"Europe/Brussels",
    "lastSynchroDate":"2018-10-30T16:02:01Z"
}
]
}

```

### <syncdata>

This request returns the **timeZone** field (if available) of a specific Ewon gateway listed in the response feed.

#### Example 4: **timeZone** inside the <syncdata> response

```

{
  "success":true,
  "ewons":[
    {
      "id":3003,
      "name":"ExampleName",
      "tags":[
        {
          "id":3223,
          "name":"TagName",
          "dataType":"Float",
          "description":"Description for tag",
          "alarmHint":"",
          "value":19.0,
          "quality":"good",
          "ewonTagId":3,
          "history":[
            {
              "date":"2014-04-16T16:34:16Z",
              "value":"4.0"
            },
            {
              "date":"2014-04-17T16:34:26Z",
              "value":5.0
            }
          ]
        }
      ]
    },
    "timeZone":"Europe/Brussels",
    "lastSynchroDate":"2018-10-30T16:02:01Z"
  }
]
}

```

```
}
```

## 4.6 Data Services

### 4.6.1 <getewons>

#### Request

```
https://data.talk2m.com/getewons
```

The following keys must be sent as part of the body request (set as x-www-form-urlencoded):

- t2mdevid
- t2mtoken

#### Response

The <getewons> service returns the list of Ewon gateways sending data to be stored in the DataMailbox.

The result contains the following information for each Ewon gateway:

- its name and id,
- its number of tags,
- the date of its last data upload to the DataMailbox.

#### Example 5: <getewons> request

The screenshot shows a REST client interface with the following details:

- Request:** POST https://data.talk2m.com/getewons
- Environment:** No Environment
- Parameters:** t2mdevid (2ac95...), t2mtoken (9oDVB...)
- Body:** JSON response showing a list of Ewon gateways.

```
1 {
2   "success": true,
3   "ewons": [
4     {
5       "id": 638409,
6       "name": "Machine 15",
7       "lastSynchroDate": "2020-09-01T13:45:36Z"
8     },
9     {
10      "id": 457115,
11      "name": "Machine 12",
12      "lastSynchroDate": "2020-09-01T13:45:03Z"
13    },
14    {
15      "id": 67,
16      "name": "Machine 11",
17      "lastSynchroDate": "2020-09-01T13:45:13Z"
18    },
19    {
20      "id": 371359,
21      "name": "Smart Building 001"
22    }
23  ]
24 }
```



## 4.6.2 <syncdata>

### Request

```
https://data.talk2m.com/syncdata
```



The **<getdata>** and **<syncdata>** are different and should not be interchanged.

**<getdata>** is used as a “one-shot” request to retrieve filtered data based on specific criteria. It is not destined to grab historical data with the same timestamp or enormous data involving the use of the **moreData** filter.

**<syncdata>** is used to retrieve all the data. This service is destined to grab the whole set of data regardless the amount.

The **<syncdata>** service retrieves all data of a Talk2M account incrementally. Therefore, only new data is returned on each API request.

The following keys must be sent as part of the body request (set as x-www-form-urlencoded):

- t2mdevid
- t2mtoken

You can filter the request with the following parameters:

<b>lastTransactionId</b>	The ID of the last set of data sent by the DataMailbox. By referencing the <b>lastTransactionId</b> , the DataMailbox will send a set of data more recent than the data linked to this transaction ID.
<b>createTransaction</b>	The indication to the server that a new transaction ID should be created for this request.
<b>ewonIds</b>	A comma separated list of Ewon gateway IDs. If <b>ewonIds</b> is used, DataMailbox sends values history of the targeted Ewon gateways. If not used, DataMailbox sends the values history of all Ewon gateways.

As a **<syncdata>** request should be used to retrieve only incremental data, the normal process of the **<syncdata>** requests is:

- When the data is retrieved for the first time, the user specifies only the **createTransaction** filter.  
The DataMailbox returns all the data of all Ewon gateways – with historical data – of the account along a transaction ID.
- For the next calls to the API, the client specifies both **createTransaction** and **lastTransactionId** filters. The **lastTransactionId** is the ID of the transaction that was returned by the latest **<syncdata>** request.  
The system returns all the historical data that has been received by the DataMailbox since the last transaction and a new transaction ID.
- **ewonIds** is an additional filter on the returned result.

You must be cautious when using the combination of **lastTransactionId**, **createTransaction** and **ewonIds**.

**lastTransactionId** is first used to determine what set of data — newer than this transaction ID and from all the Ewon gateways — must be returned from the DataMailbox, then **ewonIds** filters this set of data to send data only from the desired Ewon gateways.

If a first request is called with **lastTransactionId**, **createTransaction** and **ewonIds**, the following request — implying a new **lastTransactionId** — does not contain values history from the previous **lastTransactionId** of the Ewon gateways that were not in the **ewonIds** from previous request.

## Response

The result of the `<syncdata>` request is the same as the `<getdata>`, [p. 21](#) response. However, the `<syncdata>` result can contain an additional value:

### **transactionId**

The transaction ID of the `<syncdata>` response.

This value is returned only when the `createTransaction` filter is used and should be mentioned as `lastTransactionId` in the following `<syncdata>` request to retrieve only new data.

Similar to the `<getdata>` command, a flag `moreDataAvailable` can be found in the response of a `<syncdata>` and indicates that only a part of the full set of data has been transferred. In this case, you must perform another `<syncdata>` call with the latest `transactionId` to retrieve the additional data. Repeat the process until the flag `moreDataAvailable` is no longer in the response.

### Example 6: `<syncdata>` request

The screenshot shows a REST client interface with the following details:

- Request:** POST `https://data.talk2m.com/syncdata`
- Parameters:**

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> t2mdevid	2ac952	
<input checked="" type="checkbox"/> t2mtoken	9oDVB	
Key	Value	Description
- Response:** Status: 200 OK, Time: 2.40s, Size: 188.54 KB
- Response Body (JSON):**

```

1  {
2    "success": true,
3    "moreDataAvailable": true,
4    "ewons": [
5      {
6        "id": 53,
7        "name": "Machine 07",
8        "tags": [
9          {
10         "id": 389586,
11         "name": "Counter1",
12         "dataType": "Int",
13         "description": "Counter1 on ABLogix",
14         "alarmHint": "",
15         "value": 319,
16         "quality": "good",
17         "ewonTagId": 2,
18         "history": [
19           {
20             "date": "2020-08-21T01:08:42Z",
21             "dataType": "Int",
22             "value": 2883
23           },
24           {
25             "date": "2020-08-21T01:08:44Z",
26             "dataType": "Int",
27             "value": 2217
28           }
29         ]
30       }
31     ]
32   }

```

### Example 7: <syncdata> request with createTransaction

The screenshot displays a REST client interface for a POST request to `https://data.talk2m.com/syncdata`. The request is configured with the following parameters:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> t2mdevId	2ac95...	
<input checked="" type="checkbox"/> t2mtoken	9oDV...	
<input checked="" type="checkbox"/> createTransaction	true	
Key	Value	Description

The response status is 200 OK, with a time of 2.29s and a size of 188.57 KB. The response body is shown in JSON format:

```

1  {
2    "success": true,
3    "transactionId": "2273350",
4    "moreDataAvailable": true,
5    "ewons": [
6      {
7        "id": 638409,
8        "name": "Machine 15",
9        "tags": [
10         {
11           "id": 988417,
12           "name": "ConveyorSpeed",
13           "dataType": "Float",
14           "description": "Rotation/min",
15           "alarmHint": "",
16           "value": 0.0,
17           "quality": "good",
18           "ewonTagId": 2,
19           "history": [
20             {
21               "date": "2020-08-20T23:57:27Z",
22               "dataType": "Float",
23               "value": 0.0
24             },
25             {
26               "date": "2020-08-20T23:57:37Z"

```

**Example 8: <syncdata> request with createTransaction and ewonlds**

The screenshot shows a REST client interface for a POST request to `https://data.talk2m.com/syncdata`. The request body is `x-www-form-urlencoded` and contains the following parameters:

KEY	VALUE	DESCRIPTION
t2mdevid	2ac952d0-9e8c-4a8e-a9b3-ae18b4b144f6	
t2mtoken	9oDVBWDOOr92hwoHXwYj2e6HKurLIU8KtW0rG0...	
createTransaction	true	
ewonlds	638409,53	

The response status is `200 OK` with a time of `349ms` and a size of `188.13 KB`. The response body is a JSON object:

```

1  {
2    "success": true,
3    "transactionId": "2273350",
4    "moreDataAvailable": true,
5    "ewons": [
6      {
7        "id": 53,
8        "name": "Machine 07",
9        "tags": [
10       {
11         "id": 2451691,
12         "name": "ConveyorSpeed",
13         "dataType": "Float",
14         "description": "",
15         "alarmHint": "",
16         "value": 5.6307692527771,
17         "quality": "good",
18         "ewonTagId": 77,
19         "history": [...
500     ]
501   },
502   {
503     "id": 988525,
504     "name": "Gaz_Consumption",
505     "description": "Gaz_Consumption"
506   }
507 ]
508 }

```

**Example 9: <syncdata> request with createTransaction and lastTransactionId**

POST https://data.talk2m.com/sync... + ... No Environment

Untitled Request Comments (0)

POST https://data.talk2m.com/syncdata Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Code

none form-data x-www-form-urlencoded raw binary GraphQL BETA

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> t2mdevId	2ac952c...			
<input checked="" type="checkbox"/> t2mtoken	9oDVBV...			
<input checked="" type="checkbox"/> createTransaction	true			X
<input checked="" type="checkbox"/> lastTransactionId	2274997			
Key	Value	Description		

Body Cookies Headers (10) Test Results Status: 200 OK Time: 489ms Size: 183.95 KB Save Response

Pretty Raw Preview Visualize BETA JSON

```

1  {
2    "success": true,
3    "transactionId": "2274998",
4    "moreDataAvailable": true,
5    "ewons": [
6      {
7        "id": 53,
8        "name": "Machine 07",
9        "tags": [
10         {
11           "id": 389586,
12           "name": "Counter1",
13           "dataType": "Int",
14           "description": "Counter1 on ABLogix",
15           "alarmHint": "",
16           "value": 2808,
17           "quality": "good",
18           "ewonTagId": 2,
19           "history": [
20             {
21               "date": "2020-08-22T01:28:35Z",
22               "dataType": "Int",
23               "value": 573
24             },
25             {
26               "date": "2020-08-22T01:28:37Z",

```

### 4.6.3 <getewon>

#### Request

`https://data.talk2m.com/getewon`

The following keys must be sent as part of the body request (set as x-www-form-urlencoded):

- t2mdevid
- t2mtoken

The Ewon gateway can be identified using one of the following filter:

**id** ID of the Ewon gateway as returned by the <getewons> API request.

**name** Name of the Ewon gateway as returned by the <getewons> API request.

#### Response

The <getewon> service returns the configuration of the targeted Ewon gateway as seen by the DataMailbox.

#### Example 10: <getewon> request by id

The screenshot shows a REST client interface with the following details:

- Request Method:** POST
- URL:** https://data.talk2m.com/getewon
- Environment:** No Environment
- Body Type:** x-www-form-urlencoded
- Parameters:**

KEY	VALUE	DESCRIPTION
t2mdevid	2ac952d0-9e8c-4a8e-a9b3-ae18b4b144f6	
t2mtoken	9oDVBWDOr92hwaoHXwYj2e6HKurLIU8KtW0rGO...	
id	638409	
- Response:**

```

1  {
2    "success": true,
3    "id": 638409,
4    "name": "Machine 15",
5    "tags": [
6      {
7        "id": 988437,
8        "name": "StartProcess",
9        "dataType": "Float",
10       "description": "",
11       "alarmHint": "",
12       "value": 0.0,
13       "quality": "good",
14       "ewonTagId": 12
15     },
16     {
17       "id": 988423,
18       "name": "PLC_EmergencyButton",
19       "dataType": "Bool",

```

**Example 11: <getewon> request by name**

The screenshot shows a REST client interface for a POST request to `https://data.talk2m.com/getewon`. The request body is set to `x-www-form-urlencoded` and contains the following parameters:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> t2mdevid	2ac952	
<input checked="" type="checkbox"/> t2mtoken	9oDVBW	
<input checked="" type="checkbox"/> name	Machine 15	
Key	Value	Description

The response status is 200 OK, with a time of 407ms and a size of 1.18 KB. The response body is shown in JSON format:

```

1  {
2    "success": true,
3    "id": 638409,
4    "name": "Machine 15",
5    "tags": [
6      {
7        "id": 988453,
8        "name": "Solar_Energy",
9        "dataType": "Float",
10       "description": "kWh",
11       "alarmHint": "",
12       "value": 1.2300000190734863,
13       "quality": "good",
14       "ewonTagId": 20
15     },
16     {
17       "id": 988437,
18       "name": "StartProcess",
19       "dataType": "Float"
20     }
21   ]
22 }

```

**4.6.4 <getdata>****Request**

`https://data.talk2m.com/getdata`



The **<getdata>** and **<syncdata>** are different and should not be interchanged.

**<getdata>** is used as a “one-shot” request to retrieve filtered data based on specific criteria. It is not destined to grab historical data with the same timestamp or enormous data involving the use of the **moreData** filter.

**<syncdata>** is used to retrieve all the data. This service is destined to grab the whole set of data regardless the amount.

The following keys must be sent as part of the body request (set as `x-www-form-urlencoded`):

- `t2mdevid`
- `t2mtoken`

You can filter the **<getdata>** request to retrieve only part of the mailbox contents. The filters consist of a series of string parameters and are all optional:

<b>ewonId</b>	ID of the singleEwon gateway for which data from DataMailbox is requested.
<b>tagId</b>	ID of the single tag for which data from DataMailbox is requested.
<b>from</b>	Timestamp after which data should be returned. No data older than this time stamp will be sent.

<b>to</b>	Timestamp before which data should be returned. No data newer than this time stamp will be sent.
<b>fullConfig</b>	<p>By default, <code>&lt;getdata&gt;</code> returns configuration information only for Ewon gateways / tags that contain historical data.</p> <p>If the request contains <b>fullConfig</b> as parameter, all tags / Ewon gateways will appear in the data set, even if they do not contain historical data.</p> <p><b>fullConfig</b> doesn't accept any value. It is used as is.</p>
<b>limit</b>	<p>The maximum amount of historical data returned.</p> <p>The historical data is the historical tag values but also the historical alarms. If you set the <b>limit</b> to 4, the response will consist in 4 historical tag values and 4 historical alarms (if available) for each tag of each Ewon gateway allowed by the Talk2M token.</p> <p>If the size of the historical data saved in the DataMailbox exceeds this limit, only the oldest historical data will be returned and the result contains a <b>moreDataAvailable</b> value indicating that more data is available on the server.</p> <p>If <b>limit</b> is not used or is too high, the DataMailbox uses a limit pre-defined in the system (server-side).</p>

The parameters can be combined. For example: **ewonId + from** returns the historical data of an Ewon gateway since a given date

### Response

The `<getdata>` service returns the content of the DataMailbox: configuration, tag history and alarm history.

The result of the `<getdata>` API call contains, in a general scope:

- a flag **moreDataAvailable** set to *true* if some historical data satisfying the criteria is available on the DataMailbox but could not be returned. See the **limit** parameter;
- a list of Ewon gateways.

The result also contains the following information for each Ewon gateway listed:

- its ID;
- its configuration: ID, name, data type and description;
- its alarm hint;
- its list of tags.

The result also contains the following information for each tag of each Ewon gateway listed:

- its ID as registered in the DataMailbox;
- its configuration: name, data type and description;

The data type can be one of the following values:

- **Bool**
- **Float**
- **Int**
- **UInt**
- **String**
- **Unknown**

However, it is not recommended to base your code on the value of the data type parameter. Instead, you should check the json type of the `<value>` parameter.



- its alarm description (hint) and its alarm state;

The type of an alarm can be one of:

- for analog tags: **LOWLOW, LOW, HIGH, HIGHHIGH, NONE**.
- for boolean tags: **LEVEL, NONE**.

The state of an alarm can be one of:

- **ALM**
- **RTN**
- **ACK**
- **END**

The type is set to **NONE** if the state is set on **RTN** or **END**.

- its last known value;
- the quality of the instant value;

The quality can be one of the following values:

- **good**
- **bad**
- **uncertain**
- **initialGood**
- **unknown**

If there is no value for the quality, it assumed to be as **good**

- the history of tag values: date, value and quality;
- the history of tag alarms: date, status and type.

The quality of a tag can receive 3 possible values: good, bad and uncertain. It is always shown in the feed.

The quality of an historical tag value will only be displayed if quality is different than good.

**Example 12: <getdata> request by eworld**

The screenshot shows a REST client interface with the following details:

- Request Method:** POST
- URL:** https://data.talk2m.com/getdata
- Environment:** No Environment
- Request Body Type:** x-www-form-urlencoded
- Parameters:**

KEY	VALUE	DESCRIPTION
t2mdevid	2ac952d...	
t2mtoken	9oDVBWD...	
eworld	638409	
Key	Value	Description
- Response Status:** 200 OK, Time: 2.20s, Size: 262.25 KB
- Response Body (JSON):**

```

1 {
2   "success": true,
3   "moreDataAvailable": true,
4   "ewons": [
5     {
6       "id": 638409,
7       "name": "Machine 15",
8       "tags": [
9         {
10          "id": 988429,
11          "name": "PLC_MachineSpeed",
12          "dataType": "UInt",
13          "description": "",
14          "alarmHint": "Machine Speed too fast!!",
15          "value": 0,
16          "quality": "good",
17          "alarmState": {
18            "dateStatus": "2020-09-01T12:54:48Z",
19            "dateStart": "2020-09-01T12:54:48Z",
20            "type": "Low",
21            "status": "ALM"
22          },
23          "ewonTagId": 8,
24          "alarmHistory": [
25            {
26              "date": "2020-09-01T04:00:13Z"

```

**Example 13: <getdata> request by tagId with from and to**

The screenshot shows a REST client interface for a POST request to `https://data.talk2m.com/getdata`. The request body is set to `x-www-form-urlencoded`. The parameters are:

KEY	VALUE	DESCRIPTION
t2mdevId	2ac952	
t2mtoken	9cDVBV	
tagId	988429	
from	2020-08-20	ISO 8601 Format
to	2020-08-30	ISO 8601 Format

The response status is 200 OK, with a time of 54.09s and a size of 2.1 KB. The response body is JSON:

```

1  {
2    "success": true,
3    "ewons": [
4      {
5        "id": 638409,
6        "name": "Machine 15",
7        "tags": [
8          {
9            "id": 988429,
10           "name": "PLC_MachineSpeed",
11           "dataType": "UInt",
12           "description": "",
13           "alarmHint": "Machine Speed too fast!!",
14           "value": 0,
15           "quality": "good",
16           "alarmState": {
17             "dateStatus": "2020-09-01T12:54:48Z",
18             "dateStart": "2020-09-01T12:54:48Z",
19             "type": "Low",
20             "status": "ALM"
21           }
22         },
23       "ewonTagId": 8,

```

**4.6.5 <getstatus>****Request URL**

```
https://data.talk2m.com/getstatus
```

The following keys must be sent as part of the body request (set as `x-www-form-urlencoded`):

- t2mdevId
- t2mtoken

**Response**

The `<getstatus>` service returns the storage consumption of the account and of each Ewon gateway.

The result contains the following information:

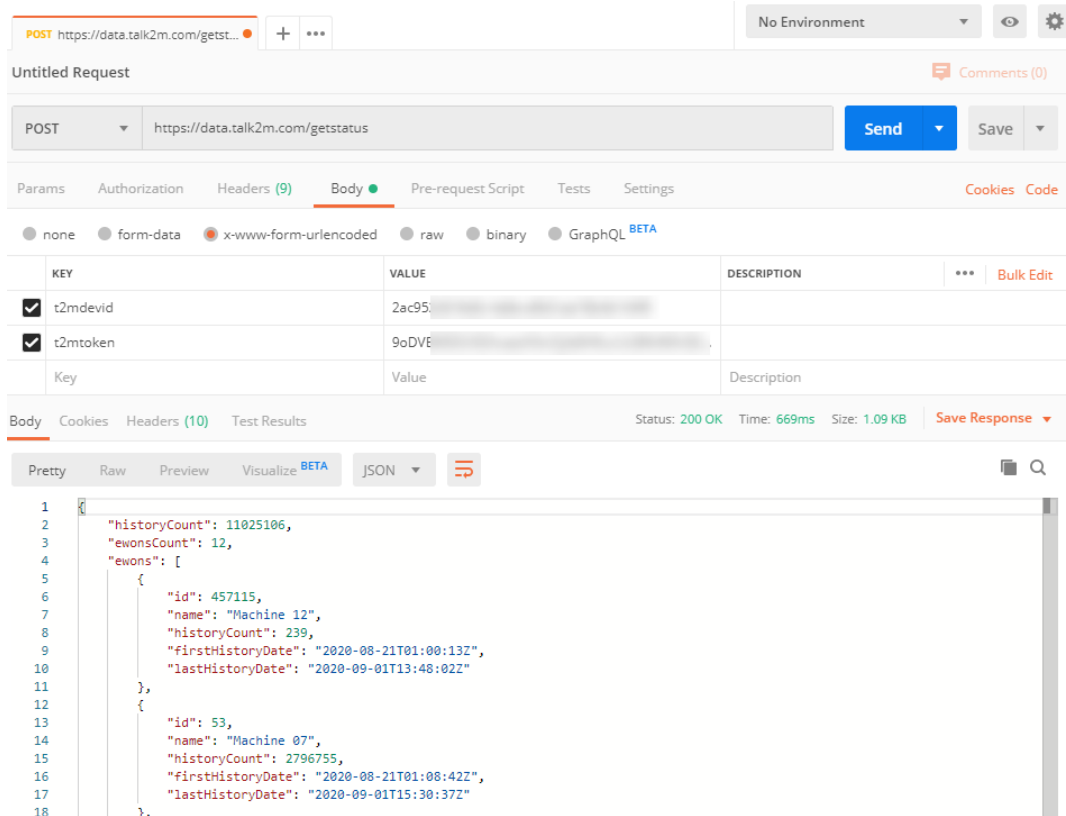
- number of history points currently stored in the DataMailbox for this account,
- number of Ewon gateways sending data to the DataMailbox.

The result also contains the following information for each Ewon gateway listed:

- its id,
- its name,
- its number of history points currently stored in the DataMailbox,

- its date of the first history point currently saved in the DataMailbox,
- its date of the last history point currently saved in the DataMailbox.

#### Example 14: <getstatus> request



The screenshot shows a REST client interface for a POST request to `https://data.talk2m.com/getstatus`. The request body is set to `x-www-form-urlencoded` and contains two parameters:

KEY	VALUE	DESCRIPTION
t2mdevid	2ac95...	
t2mtoken	9oDVE...	

The response is JSON, showing a list of two machines with their respective history counts and dates:

```

1 {
2   "historyCount": 11025106,
3   "ewonsCount": 12,
4   "ewons": [
5     {
6       "id": 457115,
7       "name": "Machine 12",
8       "historyCount": 239,
9       "firstHistoryDate": "2020-08-21T01:00:13Z",
10      "lastHistoryDate": "2020-09-01T13:48:02Z"
11     },
12     {
13       "id": 53,
14       "name": "Machine 07",
15       "historyCount": 2796755,
16       "firstHistoryDate": "2020-08-21T01:08:42Z",
17       "lastHistoryDate": "2020-09-01T15:30:37Z"
18     }
19   ]
20 }

```

### 4.6.6 <delete>

#### Request

```
https://data.talk2m.com/delete
```

As the DataMailbox is a temporary storage, the data must be deleted at one point. It is common practice once the data has been retrieved and treated by your application to delete that data from the DataMailbox.

Deleting data can be done either automatically or manually.

As the DataMailbox keeps the data for 10 days maximum, it erases automatically any data with a timestamps longer than those 10 days.

Or you can do it manually by sending the delete request.

Delete only deletes historical data. It preserves configuration data including indicators that allow Ewon gateways to send only their new content.

The following keys must be sent as part of the body request (set as `x-www-form-urlencoded`):

- `t2mdevid`
- `t2mtoken`

You can filter the request with the following parameters:

<b>all</b>	Empty the data mailbox from historical data. This parameter is used as standalone. Value is not needed.
<b>transactionId</b>	Delete all historical data prior to this transaction ID.
<b>ewonId</b>	Delete all historical data of the Ewon gateway.
<b>to</b>	Delete all history data up to the given timestamp.

## Response

The result of the **<delete>** request is a message with a success status. Check [Response Structure, p. 8](#).

### Example 15: **<delete>** request with **transactionId**

The screenshot shows a REST client interface for a POST request to `https://data.talk2m.com/delete`. The request body is set to `x-www-form-urlencoded` and contains the following parameters:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> t2mdevid	2ac95...	
<input checked="" type="checkbox"/> t2mtoken	9oDVB...	
<input checked="" type="checkbox"/> transactionId	2274998	
Key	Value	Description

The response status is `200 OK` with a time of `4.58s` and a size of `401 B`. The response body is a JSON object:

```

1 {
2   "success": true
3 }

```

## 4.6.7 **<clean>**

### Request

```
https://data.talk2m.com/clean
```

The **<clean>** request is stronger than the **<delete>** request. Using clean, even the configuration is deleted. Upon next data upload to the DataMailbox, your Ewon gateway will send its whole content again.

The following keys must be sent as part of the body request (set as `x-www-form-urlencoded`):

- t2mdevid
- t2mtoken

You can filter the request with the following parameters:

**all** Empty the DataMailbox and its metadata such as Ewon gateways and tags config, DMBin related info.

This parameter is used as standalone. Value is not needed.

**eworld** Delete all historical and configuration data related to a specific Ewon gateway. Also deletes the Ewon gateway metadata.

## Response

The result of the **<clean>** request is a message with a success status. Check [Response Structure, p. 8](#).

### Example 16: **<clean>** request with **eworld**

The screenshot shows a REST client interface for a POST request to `https://data.talk2m.com/clean`. The request body is `x-www-form-urlencoded` and contains the following parameters:

KEY	VALUE	DESCRIPTION
t2mdevId	2ac952c...	
t2mtoken	9oDVBW...	
eworld	457115	
Key	Value	Description

The response status is `200 OK` with a time of `9:19s` and a size of `401 B`. The response body is a JSON object:

```

1 {
2   "success": true
3 }

```

## 4.7 Talk2M Token Management

To be able to manage the API tokens of your Talk2M account, you must use the eCatcher software running version 6.6 or higher.

### 4.7.1 Overview

The tokens are managed inside the advanced settings of the *Account* configuration page of eCatcher.

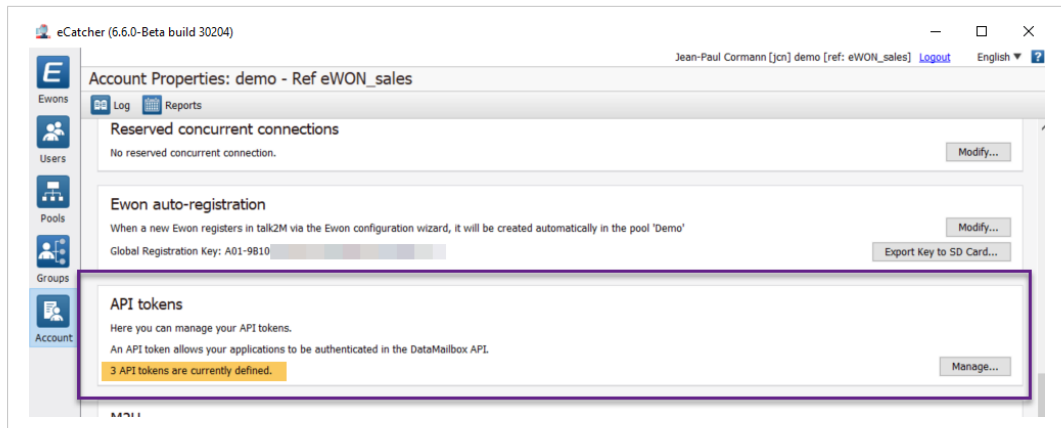


Fig. 4 Account tab to manage the Talk2M token

If there are already API tokens configured for your account, then the number of defined tokens is indicated in this tab.

To open the API Token configuration page, click on the **Manage...** button.

A modal dialog window opens containing the existing Talk2M token and the following possible actions: **New token**, **Modify**, **Delete** and **Renew**.

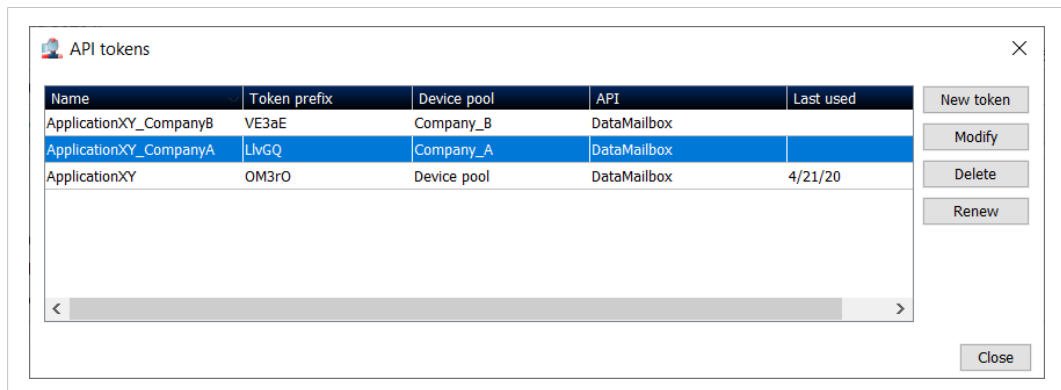


Fig. 5 List of the existing Talk2M token

Column	Description
<b>Name</b>	The name you specified for the token
<b>Token prefix</b>	The prefix of the token. For security reasons, the token itself can only be retrieved during the creation process of the token. The display of the prefix eases the identification of the token later on.
<b>Device Pool</b>	The Device pool linked to the token. The token will give access to the data of all Ewon gateways contained inside this pool.
<b>API</b>	The type of API. Currently fixed to <i>DataMailbox</i> .
<b>Last used</b>	The date — the day — when the API was last used. For implementation reasons, the time of modification is not displayed, only the date. This is more accurate and avoids the update of the database for each API call.



*Talk2M Free+ accounts do not have pools, therefore, for Free+ accounts, eCatcher doesn't display the **Device pool** column.*

#### 4.7.2 Add

To add a new Talk2M token in your Talk2M account, proceed as follows:

1. Click on **New token** in the API token configuration page.

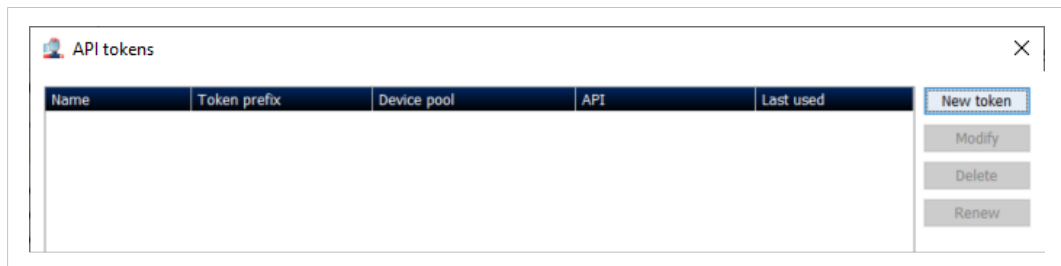


Fig. 6 Create a new Talk2M token.

2. Provide the different token information:
  - **Name**: specify a name to identify your token. The name could for example contain information of the application or the partner who will be using the token.
  - **Device pool**: the API token will give access to all Ewon gateways contained inside the device pool selected here.
  - **DataMailbox access**: check the *read only* option to create a token which cannot delete data out of the DataMailbox. *Delete, reset,...* are actions that are not allowed.



*Talk2M Free+ accounts do not have pools, therefore, for Free+ accounts, eCatcher doesn't display the the **Device pool** column.*

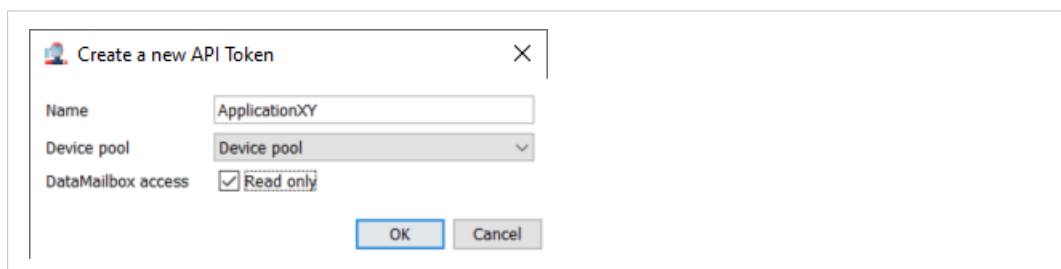


Fig. 7 Fill the information for new Talk2M token.

3. Click on **OK** to create the new Talk2M token.
  - The next window will show the new created token.



Fig. 8 New Talk2M token created

4. Use the **Copy to clipboard** button to retrieve the token and integrate it in your application.



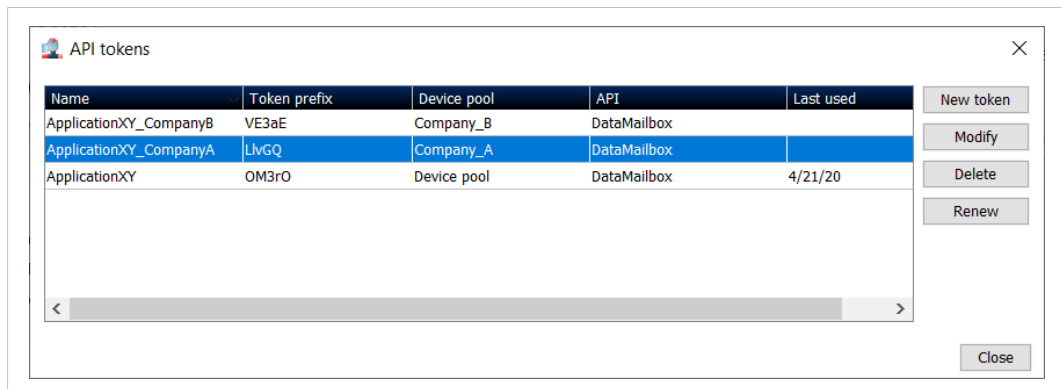
As mention on the window, this is the only time you will have access to the API token. Once the window is closed, you will no longer be able to display it. If you forgot the token you'll need to renew it.

### 4.7.3 Remove

To remove a Talk2M token in your Talk2M account, proceed as follows:

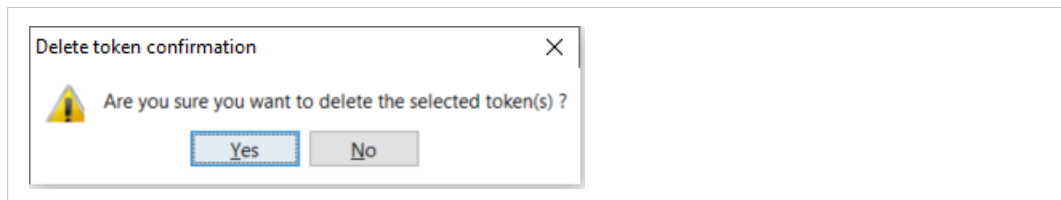


1. Select **the API token** you want to remove from the API token list.



**Fig. 9** Delete a Talk2M token

2. Click on **Yes** in the *Delete token confirmation* popup to remove definitively the Talk2M token.



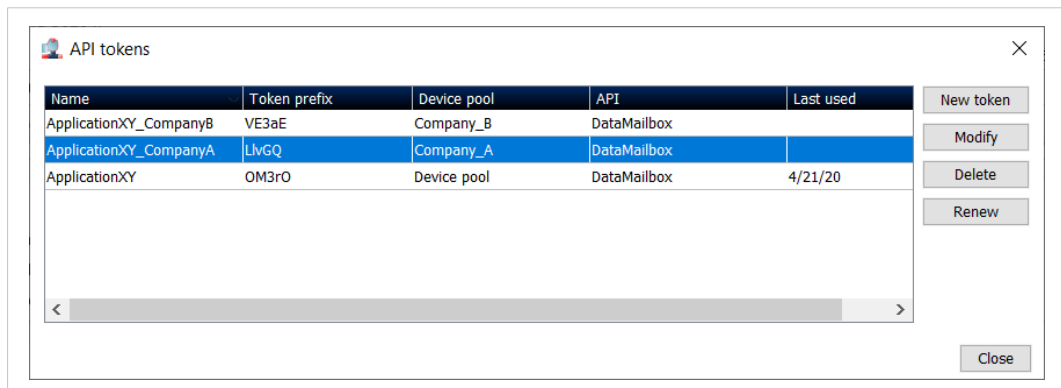
**Fig. 10** Confirm the Talk2M token deletion

#### 4.7.4 Renew

The *Renew* button combines the *Delete* action with the *New token* one: the information of the token remains the same, but a new token is generated.

To renew a Talk2M token in your Talk2M account, proceed as follows:

1. Select **the API token** you want to renew Inside the API token list
2. Click on the **Renew** button.



**Fig. 11** Renew a Talk2M token

3. Confirm that you want to renew the Talk2M token.

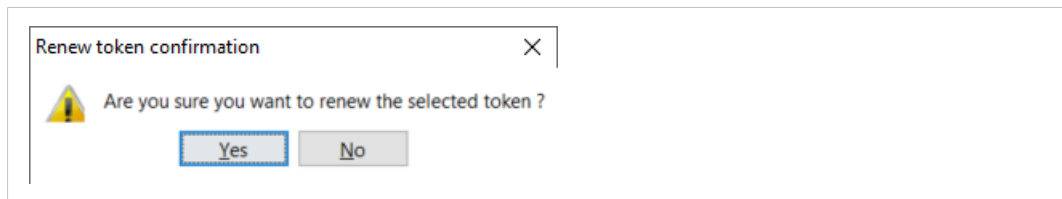


Fig. 12 Confirm the renewal of a Talk2M token

→ The next window will show the new created token.

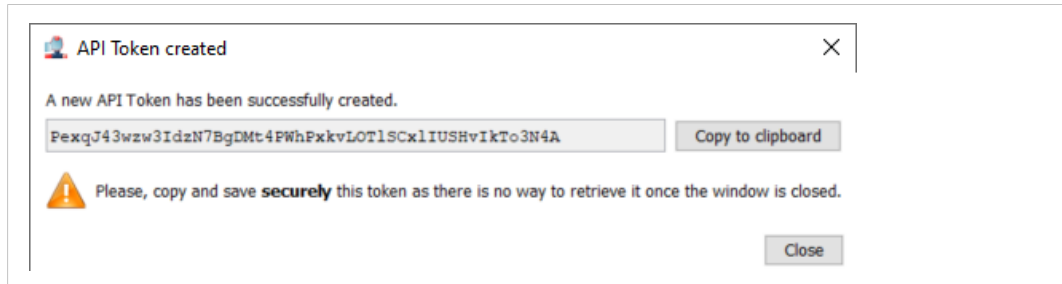
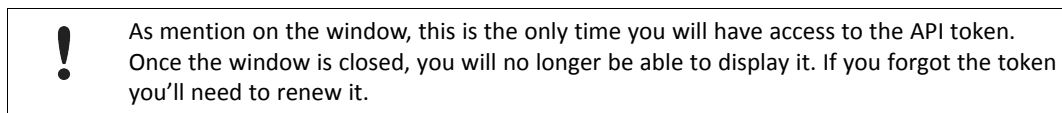


Fig. 13 New Talk2M token created

4. Use the **Copy to clipboard** button to retrieve the token and integrate it in your application.



## 4.7.5 Modify

To modify a Talk2M token in your Talk2M account, proceed as follows:

1. Select **the API token** you want to change in the API token list.
2. Click on **Modify**.

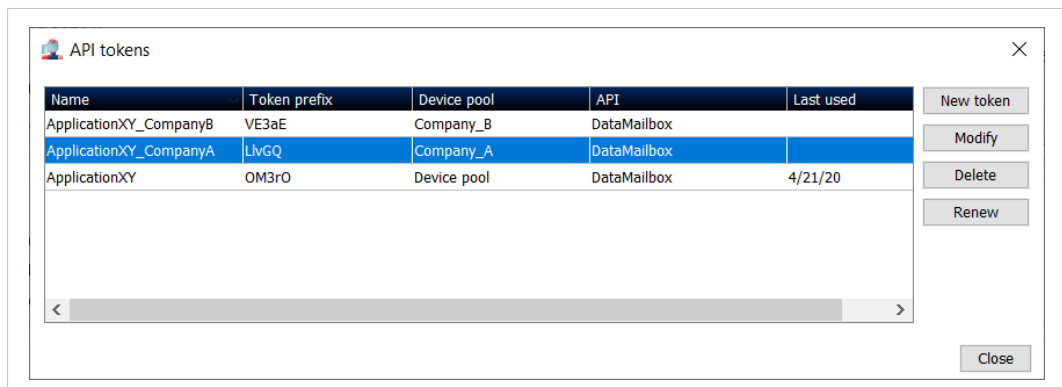
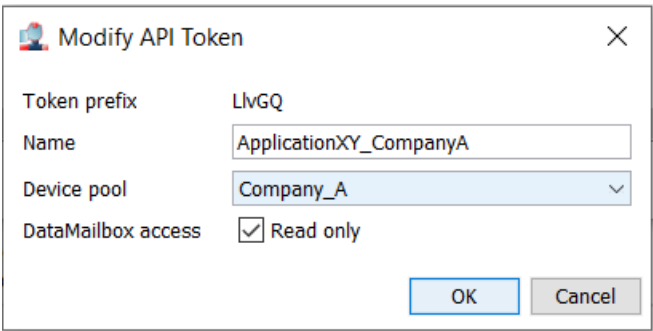


Fig. 14 Modify a Talk2M Token

### 3. Change the needed fields.

You can change the **Name** and the **Device pool** associated to the existing API token. You can also change the **DataMailbox access type**.



The screenshot shows a 'Modify API Token' dialog box with the following fields and values:

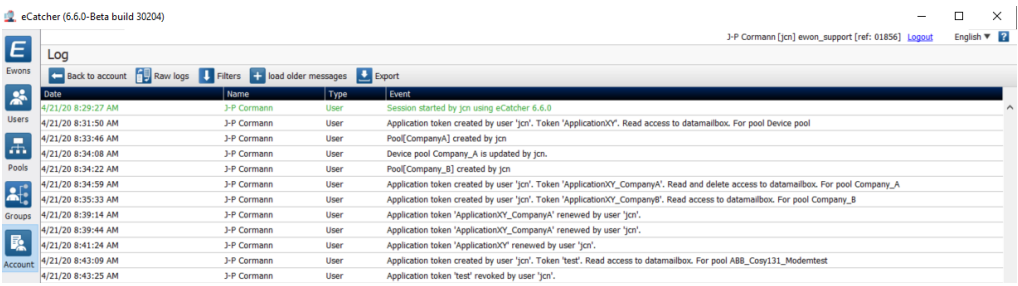
- Token prefix: LlvGQ
- Name: ApplicationXY\_CompanyA
- Device pool: Company\_A
- DataMailbox access:  Read only

Buttons: OK, Cancel

Fig. 15 Modification of the information linked to a Talk2M token

## 4.7.6 Changelog

All token operations (creation, deletion, modification,..) are logged in the account logs, including the user that performed the operation.



The screenshot shows the 'Log' interface with the following data:

Date	Name	Type	Event
4/21/20 8:29:27 AM	J-P Cormann	User	Session started by jcn using eCatcher 6.6.0
4/21/20 8:31:50 AM	J-P Cormann	User	Application token created by user 'jcn'. Token 'ApplicationXY'. Read access to datamailbox. For pool Device pool
4/21/20 8:33:46 AM	J-P Cormann	User	Pool[CompanyA] created by jcn
4/21/20 8:34:08 AM	J-P Cormann	User	Device pool Company_A is updated by jcn.
4/21/20 8:34:22 AM	J-P Cormann	User	Pool[Company_B] created by jcn
4/21/20 8:34:59 AM	J-P Cormann	User	Application token created by user 'jcn'. Token 'ApplicationXY_CompanyA'. Read and delete access to datamailbox. For pool Company_A
4/21/20 8:35:33 AM	J-P Cormann	User	Application token created by user 'jcn'. Token 'ApplicationXY_CompanyB'. Read access to datamailbox. For pool Company_B
4/21/20 8:39:14 AM	J-P Cormann	User	Application token 'ApplicationXY_CompanyA' renewed by user 'jcn'.
4/21/20 8:39:44 AM	J-P Cormann	User	Application token 'ApplicationXY_CompanyA' renewed by user 'jcn'.
4/21/20 8:41:24 AM	J-P Cormann	User	Application token 'ApplicationXY' renewed by user 'jcn'.
4/21/20 8:43:09 AM	J-P Cormann	User	Application token created by user 'jcn'. Token 'test'. Read access to datamailbox. For pool AB8_Cosy131_Modemtest
4/21/20 8:43:25 AM	J-P Cormann	User	Application token 'test' revoked by user 'jcn'.

Fig. 16 Chngelog

