MAIT Industries   iNTELLiTROL®

iNTELLiWeb /getdata API
v2.21

21 February, 2019 – Peter Home

1 Introduction

The iNTELLiWeb /getdata API provides a mechanism for third parties to access customer data using standard HTTP 'get' commands. The basic form of the URL is:

http://intelliweb.mait.com.au/getdata

which may be typed into a web browser or embedded into a web page or computer program.

The HTTP request must contain a valid user name and password which may also be embedded into a web page or computer program along with the URL. When accessing this API directly from a browser, the browser will prompt for a user name and password when making a connection for the first time. Refer to Section 6 for more details on User Names and Passwords.

Data are returned as text in CSV format, which is the most efficient method of transferring bulk data of this kind. This data may be viewed in the browser as plain text, handled by a computer program or imported into a spreadsheet for analysis.

Cross Origin Resource Sharing (CORS) has been implemented so data may be either requested from your server or directly from JavaScript code running in the client's browser. Make sure to embed the user name and password into the HTTP header so that there is no authentication prompt.

2 HTTP Status Codes

The following status codes will be returned by the HTTP server:

- 200 (OK) – the request was handled successfully and all requested content has been returned.
- 204 (No Content) – the request was valid but resulted in no data being retrieved. E.g. This may be due to no data being available for the specified module for the specified date range.
- 206 (Partial Content) – data have been returned but have been truncated to the maximum allowable limit of 10,000 lines.
- 400 (Bad Request) – the request contained an error. An invalid module number or improper date/time format will result in this error.
- 401 (Unauthorised) – authentication failure (incorrect user name/password combination).

3 Specifying a Customer's Network

For any data to be returned, a Network ID must be specified. This ID may be obtained from the customer or from MAIT Industries, provided the customer has given permission for that information to be released. Thus, the minimum URL for data retrieval is:

http://intelliweb.mait.com.au/getdata**?network=<id>**

3.1 Trial Network

MAIT Industries provides a trial system to allow interested parties to experiment with the API. To access the trial system use a Network ID of **1001** with the User Name **getdata** and the Password **trial**. The URL would thus be:

http://intelliweb.mait.com.au/getdata?network=1001

If accessing this from a web browser then enter **getdata** and **trial** when prompted by the browser.

This minimal URL returns a list of all Module IDs for which data are available, along with the descriptions as entered by the customer. E.g.

```
Network: 1001 (MAIT Web Trial),Module: List
Module,Description
1,Ararat
2,Ararat EnviroPro
10,Pivot Tracker
21,Edwards Reserve
31,Rose Garden
32,Nature Strip
51,St Andrews WS
52,St Andrews Tank House
53,St Andrews Shed Tank
60,Aquacheck 80cm
```

4 Retrieving Data

Apart from the “List Modules” URL described in the Section 3, all URLs must specify a Module ID. All valid Module IDs are returned by the “List Modules” URL as the first number in each row of the result. In the previous example, valid Module IDs for Network 1001 are 1, 2, 10, 21, 31, 32, etc.

The basic URL specifying a Module ID takes the form:

http://intelliweb.mait.com.au/getdata?network=<id>**&module=<id>**

Continuing with our trial site, a valid URL would be:

http://intelliweb.mait.com.au/getdata?network=1001&module=52

which would return four lines similar to the following:

```
Network: 1001 (MAIT Web Trial),Module: 52 (St Andrews Tank House),Count: 1,Range: Latest record
DateTime,Tank Fill Pump,2,House Tank Level,4,5,6,7,8,9,10,11,12,13,14,15,16,Battery
Units,,,%,,,,,,,,,,,,,,Volts
21/02/2019 16:00,0,,46.696,8.323,8.28,,,,,,,,,,,,10.3
```

4.1 What’s In the Result?

The API will return two or three lines of header information followed by one or more lines of data. Except where the “scope” modifier is specified the /getdata API returns data for all 17 channels available on the specified module. (Refer to Section 7 for an explanation of the “scope” modifier.) All 17 channels are returned even if that channel is not in use. This results in an 18 column report where the first column is the date/time of the sample, followed by the 16 data channels, followed, in turn, by the battery (channel 17).

It can be a bit tricky determining what the results mean by simple visual analysis of the raw CSV text. As the API is generally called from a program capable of handling CSV files, this is not considered to be a problem. If visual analysis is required then the results should be loaded into a spreadsheet or some form of text formatter. This would yield something like:

Network: 1001 (MAIT Web Trial)	Module: 52 (St Andrews Tank House)	Count: 1	Range: Latest record														
DateTime	Tank Fill Pump	2	House Tank Level	4	5	6	7	8	9	10	11	12	13	14	15	16	Battery
Units			%														Volts
21/02/2019 16:00	0		46.696	8.323	8.28												10.3

The four lines are interpreted as follows:

Line 1 contains four fields…

1. The network ID and name. Format is **Network:** <Network ID> **(**<Network Name>**)**
2. The module ID and name. Format is **Module:** <Module ID> **(**<Module Name>**)**
3. The number of records returned.
4. The date range of the included data. Note that, because no date range was specified in the URL, the latest readings have been returned.

Line 2 contains 18 fields. These are the column labels. If the customer has provided labels that identify particular channels then those labels will be shown. Otherwise the channel number will be shown. (The channel number is superfluous as all 17 channels are always present in columns 2 through 18.)

Line 3 contains 18 fields. The first field contains the text "Units" and, if the customer has provided units of measurement for the channels then those will be shown in the remaining 17 columns. Otherwise the fields will be empty.

Line 4 contains 18 fields representing the date/time of the sample and the sampled values for the 17 channels. In the above example, data only exist for channels 1, 3, 4, 5 and 17. Channel 1 is the pump where '1' indicates the pump is ON and '0' indicates it is OFF. Channel 3 is the tank level (the units indicating it is a percentage). Channels 4 and 5 are in use but the customer has not attached a label to those channels nor specified the units of measurement. Channel 17 is the battery reading of 10.3 Volts.

5 Retrieving More Data (Specifying Start and End Dates)

Section 4 showed an example of retrieving the latest reading for all the channels on the specified module. This would be eminently useful for displaying gauges in a client's browser. However, to display historical graphs or analyse trends, significantly more data is required. To achieve this, the **startdate** and **enddate** modifiers may be used.

These modifiers express a date that represents midnight at the start of that day (i.e. hour = 00:00). In a nutshell, the records for **startdate** will be included while the records for **enddate** will not be included. This behaviour can be modified by using the **inclusive** modifier, which will cause all samples on **enddate** to be included in the result. Refer to Section 5.3 for information on the **inclusive** modifier.

5.1 What Happens to Midnight Samples?

Any given data sample actually contains data from the period prior to that sample. E.g. if a logger were sampling rainfall every hour then the 3pm sample would represent rain that fell between 2pm and 3pm. Thus, the midnight sample on 24th March (24/03/2019 00:00) actually contains data for the previous day… the hour commencing 11pm 23rd March.

Applying this logic, the API will not return the 00:00 sample on the start date, as the data do not belong to that day. Similarly, the API will include the 00:00 sample on the end date but no further records for the remainder of the day. It might be helpful to think of **enddate** as being 24:00 on the previous day.

A sample will be included in the result according to the algorithm:

startdate 00:00 < sample date/time <= **enddate 00:00**

Another way of expressing the same algorithm could be:

startdate 00:00 < sample date/time <= (**enddate – 1) 24:00**

There are four possible combinations of start and end dates as follows:

/getdata?network=<id>&module=<id>

Neither is specified; retrieves the latest sample for each channel.

/getdata?network=<id>&module=<id>**&startdate=<date>**

Only **startdate** is specified; retrieves the first sample after midnight at the start of the day specified by **startdate** up to and including the latest sample.

/getdata?network=<id>&module=<id>**&enddate=<date>**

Only **enddate** is specified; retrieves all samples from the very first sample, up to and including midnight at the start of the day specified by **enddate**. I.e. only the midnight sample on **enddate** will be included in the result.

/getdata?network=<id>&module=<id>**&startdate=<date>&enddate=<date>**

Both are specified; retrieves the first sample after midnight at the start of the day specified by **startdate** up to and including the first sample (midnight) on the date specified by **enddate**. I.e. only the midnight sample on **enddate** will be included in the result.

5.2 Date Format

Dates are specified in the format YYYY/MM/DD (e.g. 2019/08/21). However, the date format is fairly flexible and the same date could also be expressed as 2019-8-21, or some similar fashion, if better suited. The primary consideration is the order, which must be year-month-day.

5.3 The **inclusive** modifier

Data for the entire day specified by **enddate** can be included in the result by using the **inclusive** modifier. Consider the following examples.

/getdata?network=1001&module=20&enddate=2019-02-21

The final sample in the result will be 21/02/2019 at 00:00 (i.e. the 21st is excluded)

/getdata?network=1001&module=20&enddate=2019-02-21**&inclusive**

The final sample in the result will be 22/02/2019 at 00:00 (i.e. the 21st is included)

As an example, to retrieve all the samples for a particular day, **startdate** needs to be set to the specific day and **enddate** needs to be set to the following day. Alternatively, set **enddate** to be the same as **startdate** and specify **inclusive**.

Failing to specify **inclusive** in the latter case will result in a 204 (No Content) response.

5.4 The **date** Modifier - Shortcut For a Single Day

As discussed, retrieving data for a single day, say 17th April, 2018 may be achieved by either of the following two URLs:

&startdate=2019/04/17&enddate=2019/04/18

&startdate=2019/04/17&enddate=2019/04/17&inclusive

The **date** modifier provides a shorthand method to achieve the same result:

&date=2019/04/17

5.5 Data Limitation

The result returned to the browser is limited to a maximum of 10,000 lines. If there are more than 10,000 samples between the specified dates then 10,000 lines will be returned and the HTTP result will be set to 206 (Partial Content).

6 User Names and Passwords

A user name and password will need to be entered into the browser (when prompted) or programmatically embedded into the HTTP request. This username must have ‘getdata’ privileges attached to it. Setting up such a user can only be performed by an iNTELLiWeb administrator.

Some customers have personnel that handle their iNTELLiWeb administration. If an administrator has not been appointed, or they are unfamiliar with the process, MAIT Industries can set up ‘getdata’ users as long as permission is obtained from the customer.

For testing purposes, the test network **1001** may be used along with the **getdata/trial** username/password combination.

7 The **scope** Modifier

The standard /getdata API as described thus far will be sufficient for the majority of needs. Data for all 17 channels associated with all modules may be retrieved in this manner.

There are, however, some limitations within the basic API. The first is that all 17 channels are returned, even if no data exist for that channel. The second is that extra information may be available that is not included in the 17 standard channels.

7.1 &scope=existing

The implication of the first limitation is that file sizes will be slightly larger than required. This limitation may be offset by the fact that each channel is in a known column, regardless of how many channels are used. In any case, if the desire is to reduce file size then the modifier **&scope=existing** may be utilised.

Note that only the first five characters of ‘existing’ are required so, exist, exists, existing are all valid values for the scope modifier.

If we use the same example as used in Section 4 with the **scope=existing** modifier

http://intelliweb.mait.com.au/getdata?network=1001&module=52**&scope=exists**

the following result will be returned:

Network: 1001 (MAIT Web Trial)	Module: 52 (St Andrews Tank House)	Count: 1	Range: Latest record		
Channel	1	3	4	5	17
DateTime	Tank Fill Pump	House Tank Level	4	5	Battery
Units		%			Volts
22/02/2019 02:00	0	46.043	8.247	8.222	9.2

As can be seen, the number of columns has been reduced to six and only include data that exist for the specified module within the specified date range (in this case, the latest sample).

Of note is that the channel numbers can no longer be identified by the column in which they reside. To overcome this limitation an extra header line is included which has “Channel” in the first column and the channel numbers in the remaining columns.

7.2 Virtual Channels (Usage Data)

If cloud data is sourced from customers running the MAIT Irrigation Control software then 'virtual' channels, relating to device usage, will be available over and above the 17 standard channels.

In the example above, channel 1 represents the Tank Fill Pump. Typically, the only information available for channel 1 is whether the pump was ON or OFF at the time of the sample.

Virtual channels provide more detailed information, in particular, when a pump or valve was turned on and the duration of operation.

Further, if a customer has installed flow meters and configured the irrigation control software correctly then the volume of water delivered through a particular valve can also be determined.

Usage data are obtained using the identical **scope=usage** or **scope=duration** modifiers as follows:

> /getdata?network=1001&module=52&startdate=2019-02-01**&scope=usage**
>
> /getdata?network=1001&module=52&startdate=2019-02-01**&scope=duration**

As stated, both calls are identical and, in this case, will return all usage data after midnight on 01/02/2019.

Network: 1001 (MAIT Web Trial)	Module: 52 (St Andrews Tank House)	Count: 3	Range: All records after 01/02/2019 00:00
Channel	1 d	1 v	
DateTime	Tank Fill Pump (Dur)	Tank Fill Pump (Vol)	
Units	mins	kl	
02/02/2019 21:00	45	222.8	
05/02/2019 07:28	121	601	
09/02/2019 18:32	31	153	

7.3 Analysing Usage Data

Usage data are generally available for all digital I/O on the specified module. The operating minutes should always be available. The flow volume may be available depending on the customer's specific installation.

In the example above, only one digital I/O channel is in use – the Tank Fill Pump on channel 1. The result contains two columns per channel. In this case, the first column is identified with the channel number **1 d** and labelled **Tank Fill Pump (Dur)**. In a similar fashion, the second column is identified with the channel number **1 v** and labelled **Tank Fill Pump (Vol)**.

Units for the two columns are **minutes** and **kl** (cubic meters) respectively. If volume information is not available, the column will still exist and will contain '0'.

In the above example the pump turned on at 21:00 on 02/02/2019 and ran for 45 minutes. During that time it delivered 222.8 cubic meters. It ran again at 07:28 on 05/02/2019 for 121 minutes delivering 601 cubic meters. And so on.

7.4 How Flow is Determined (Master and Individual Flow Meters)

It is not usual to have a flow meter attached to each irrigation valve. Typically a single flow meter is installed at the pump and, for each flow meter reading, the total flow is apportioned to each valve based on;

- how long it has been in use, and
- the estimated flow rate for that valve.

If three valves are in operation then the total flow as measured by the flow meter should be the sum of all three estimated values. If there is a discrepancy then that discrepancy is shared among all three valves as the total flow is apportioned.

This works well except in cases where one valve has malfunctioned. The malfunctioning valve will cause errors in flow volume for all valves that are open at the same time.

To mitigate this, in rare cases, customers will install individual flow meters at each valve. This allows the irrigation controller to accurately record flow volumes for each valve.

If individual flow meters have been installed then an additional two columns will be available in the CSV data returned by the /getdata API. These columns will be identified as channels **1 id** and **1 iv** and be labelled **<ChannelName> (Indv.Dur)** and **<ChannelName> (Indv.Vol)**.

Network: nnn (Name)	Module: nn (name)	Count: 3	Range: All records after 01/02/2019 00:00	
Channel	1 d	1 v	1 id	1 iv
DateTime	Tank Fill Pump (Dur)	Tank Fill Pump (Vol)	Tank Fill Pump (Indv.Dur)	Tank Fill Pump (Indv.Vol)
Units	mins	kl	mins	kl
02/02/2019 21:00	45	222.8	45	228.7
05/02/2019 07:28	121	601	121	580
09/02/2019 18:32	31	153	31	164.5

Note that the two duration fields should always be identical as they obtain their data from the same source. Conversely, the two volume fields will rarely be identical as they obtain their data from different sources. If an individual volume field is available it should be more accurate than the standard (apportioned) volume field.

7.5 Midnight Readings for Usage Data

It will be very rare if there is an exact midnight reading for usage data. This is due to the fact that usage data are recorded at the time the valve opens or shuts – which will rarely be exactly midnight. This will commonly result in usage data spanning a midnight boundary. For example, if a valve opens at 10pm on 17th Jan and closes at 2am on 18th Jan then the Duration and Volume fields will contain data either side of the midnight boundary.

In this example the entry will be registered at 10pm on 17th Jan with a duration of 240 minutes. It will be included in results returned for 17th Jan. The implications are that, if data are uploaded for 18th Jan then 120 minutes of valve operation will be missing from that data.

It is the user's responsibility to ensure that, if accumulating daily totals, data are retrieved for a few days prior and, if the duration extends into the salient day, the appropriate portion is included in the total.

8 Ongoing Development

While every effort is made to ensure that ongoing development of the API maintains backward compatibility, MAIT Industries does not guarantee that this will always be the case. Users are encouraged to monitor their product from time to time to ensure correct operation.

This document refers to the MAIT Web /getdata Server v2.21.

This document will be updated as new features become available and the latest version can be obtained free of charge by contacting MAIT Industries.