The InfluxDB & HTTP Export plugin PRINTED MANUAL

InfluxDB & HTTP Export plugin

Printed: 3/12/2025

Publisher

AGG Software

Production

© 1999-2025 AGG Software
http://www.aggsoft.com

Table of Contents

1 Introduction

The "InfluxDB HTTP Export" data publishing module for our loggers allows you to write data to an InfluxDB database by using that database's HTTP API.

InfluxDB is a database intended to store time-series data. It was specially developed to handle a lot of read and write requests.

You don't need to know SQL to write data to an InfluxDB database.

The module can also export data using HTTP GET and HTTP POST requests.

The module can create a backup copy of data if a database is offline, or if an attempt to write data to the database has failed. Later, when the connection to the database is reestablished, all data will be restored from the backup copy.

2 System requirements

The following requirements must be met for "InfluxDB & HTTP Export" to be installed:

Operating system: Windows 2000 SP4 and above, including both x86 and x64 workstations and servers. The latest service pack for the corresponding OS is required.

Free disk space: Not less than 5 MB of free disk space is recommended.

Special access requirements: You should log on as a user with Administrator rights in order to install this module.

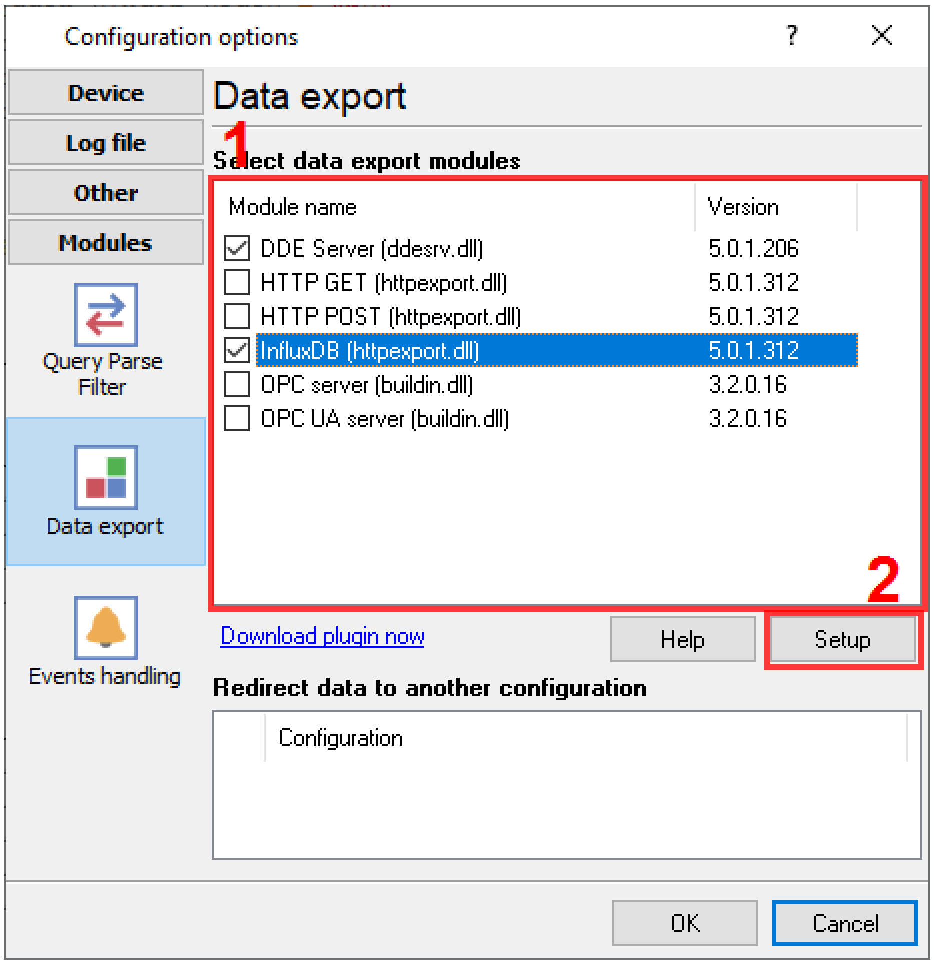
The main application (core) must be installed, for example, Advanced Serial Data Logger.

3 Installing InfluxDB & HTTP Export

1. Close the main application (for example, Advanced Serial Data Logger) if it is running;
2. Copy the program to your hard drive;
3. Run the module installation file with a double click on the file name in Windows Explorer;
4. Follow the instructions of the installation software. Usually, it is enough just to click the "Next" button several times;
5. Start the main application. The name of the module will appear on the "Modules" tab of the "Settings" window if it is successfully installed.

If the module is compatible with the program, its name and version will be displayed in the module list. You can see examples of installed modules on fig.1-2. Some types of modules require additional configuration. To do it, just select a module from the list and click the "Setup" button next to the list. The configuration of the module is described below.

You can see some types of modules on the "Log file" tab. To configure such a module, you should select it from the "File type" list and click the "Advanced" button.



Data export

Select data export modules

Module name	Version
☑ DDE Server (ddesrv.dll)	5.0.1.206
☐ HTTP GET (httpexport.dll)	5.0.1.312
☐ HTTP POST (httpexport.dll)	5.0.1.312
☑ InfluxDB (httpexport.dll)	5.0.1.312
☐ OPC server (buildin.dll)	3.2.0.16
☐ OPC UA server (buildin.dll)	3.2.0.16

Redirect data to another configuration

	Configuration

Fig. 1. Example of installed module

4 Glossary

Main program - it is the main executable of the application, for example, Advanced Serial Data Logger and asdlog.exe. It allows you to create several configurations with different settings and use different plugins.

Plugin - it is the additional plugin module for the main program. The plugin module extends the functionality of the main program.

Parser - it is the plugin module that processes the data flow, singling out data packets from it, and then variables from data packets. These variables are used in data export modules after that.

Core - see "Main program."

5 InfluxDB

The module prompts you to configure the following connection parameters:

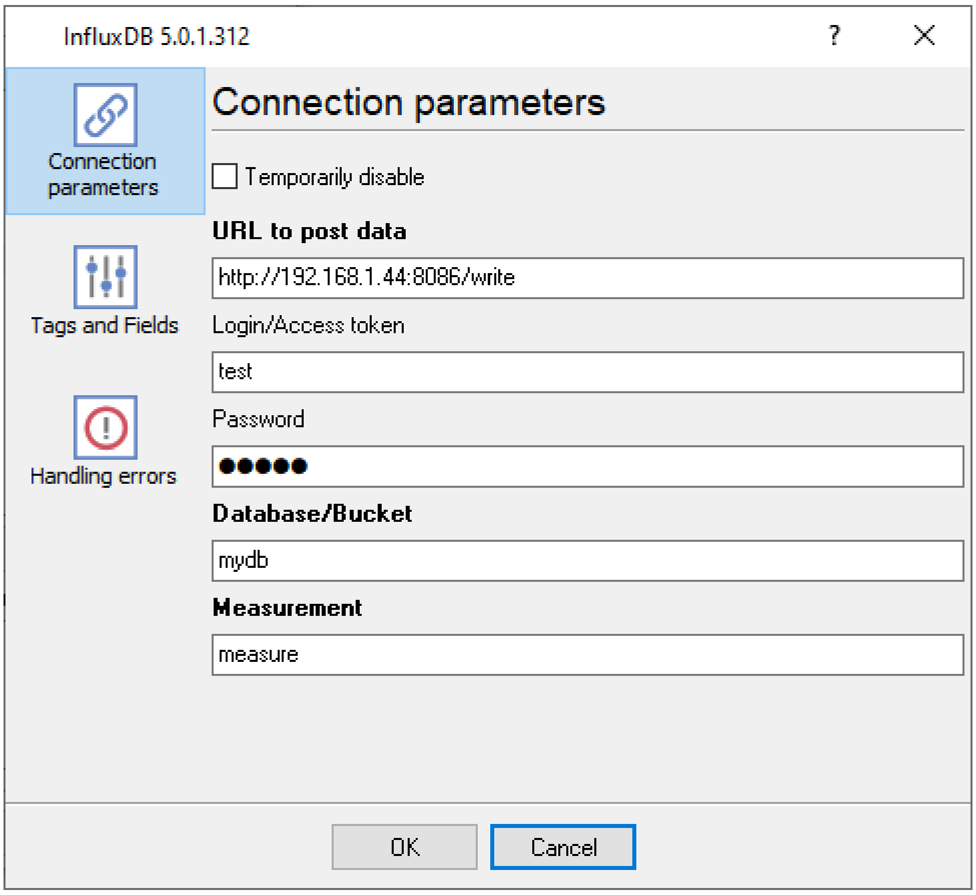
Temporarily disable – This checkbox allows you to temporarily disable a module (for example, to debug or test the program).

URL to post data – Enter a full URL address for posting data. You can see an example of the URL address in Figure **2**. Make sure that the URL address specifies the protocol (http or https) and the port number for accessing the HTTP API.

Login and password – The user's login and password to be used to connect to the database and write data to it. If user authentication is disabled in the database settings when using the HTTP API, leave these fields empty.

Database – The database name. The database must already exist, and the user must have privileges to write to the database.

Measurement – Speaking in terms of the traditional SQL database, it is a table name.



Connection parameters

Temporarily disable

URL to post data

http://192.168.1.44:8086/write

Login/Access token

test

Password

●●●●●

Database/Bucket

mydb

Measurement

measure

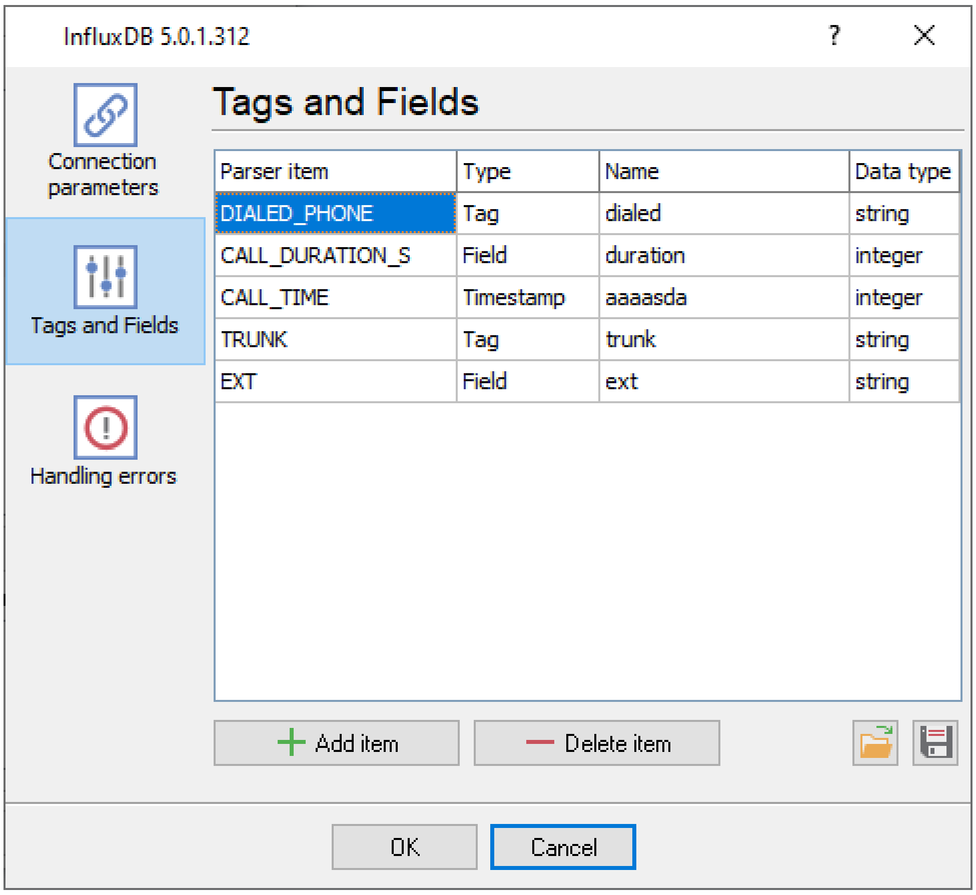
Figure 2: Connection settings

5.1 Tags and values

Speaking in terms of the traditional SQL database, a tag is an indexed table field, and a field is a non-indexed table field. Both tags and fields can store values of certain types.

Tags can store only string values, and fields can store string, integer, float, or boolean values. For more details on data types, please refer to the database manual.

In Figure **3**, you can see a list of tags and fields, which also contains parser variables linked to them.



Parser item	Type	Name	Data type
DIALED_PHONE	Tag	dialed	string
CALL_DURATION_S	Field	duration	integer
CALL_TIME	Timestamp	aaaasda	integer
TRUNK	Tag	trunk	string
EXT	Field	ext	string

Figure 3: Tags and values

Parser variable name – The name of the parser variable whose value should be written to the database. You can select a variable name in the drop-down list, or enter the name manually.

You need to link variables for each tag and field, and also for each timestamp.

Note: Before doing that, you need to configure the parser. If the parser is not working properly, no data will be written to the database. For more details about configuring the parser, please refer to the parser manual or see tutorials on our website.

Note that you cannot write empty values to the database. If a parser variable with the specified name is unavailable when data are being exported, or if its value is empty, it will be ignored.

If values are missing for all specified fields when data are being exported, such a request will be ignored, too.

Type – Defines the parameter type (tag, field, or timestamp). The list can contain an unlimited number of tags and fields. Only one timestamp variable is allowed. If no timestamp variable is specified, a database timestamp will be used when writing data to the database. You don't need to specify the data type and the field name for the timestamp variable.

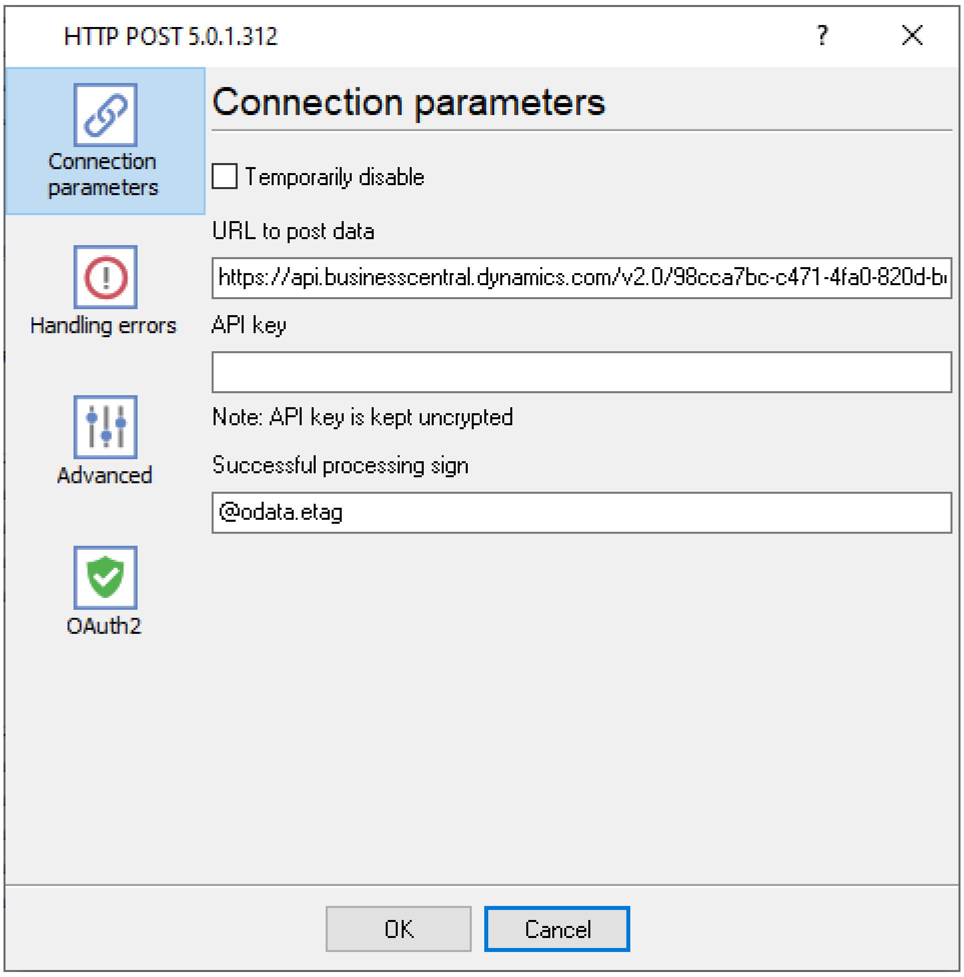
Name – The name of a tag or field in the database to write data to.

Data type – Defines the data type in the database for the specified parameter. Depending on the specified data type, the module will convert data from the parser variable to the necessary data type in the database.

The parameter order in the list doesn't matter. When generating a request to the database, the module will sort tags and fields alphabetically, and the timestamp will always be in the end.

6 HTTP POST

The HTTP Export plugin can also export data using the simple HTTP POST request.



Connection parameters

☐ Temporarily disable

URL to post data: https://api.businesscentral.dynamics.com/v2.0/98cca7bc-c471-4fa0-820d-b

API key:

Note: API key is kept uncrypted

Successful processing sign: @odata.etag

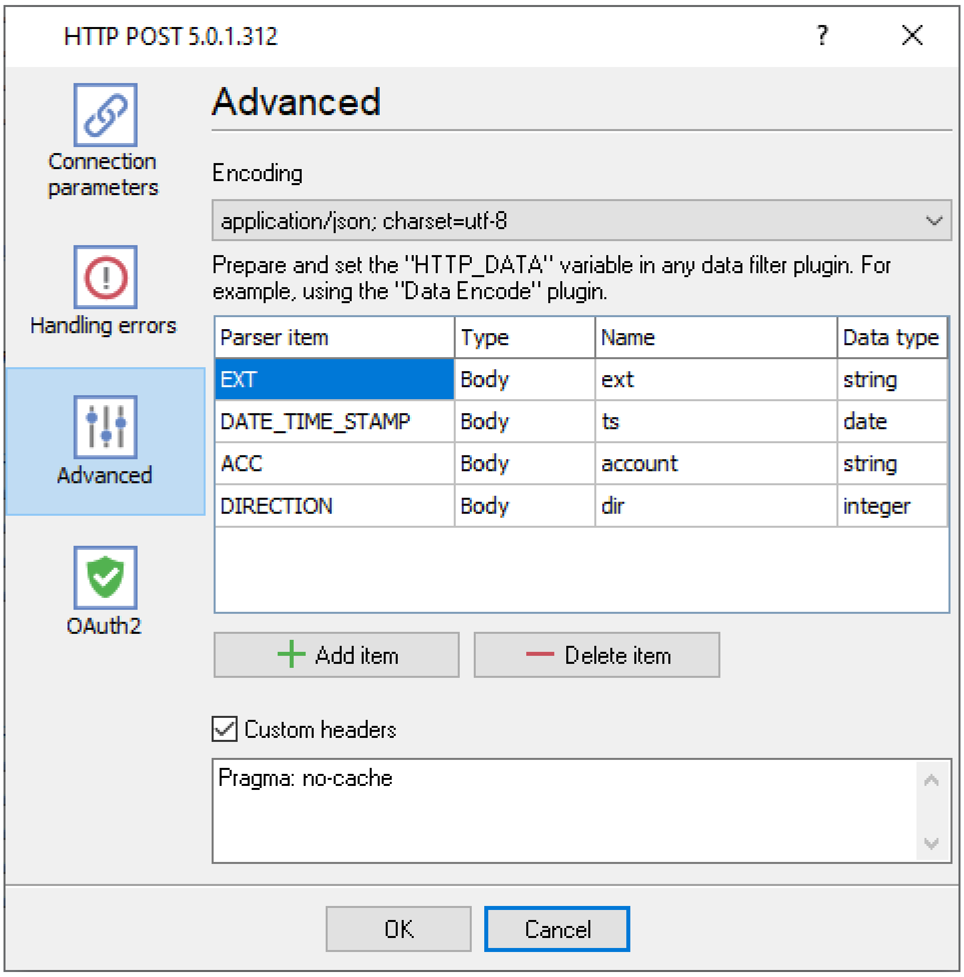
Figure 4: HTTP request basic parameters

URL to post data – Enter a full URL address for posting data. You can see an example of the URL address in Figure **4**. Make sure that the URL address specifies the protocol (http or https) and the port number for accessing the HTTP API. The URL may contain a special placeholder like {PARSER_VALUE}. The plugin replaces this placeholder using the corresponding parser's variable value before each request.

API key (optional) - It allows you to send a special authentication key to the server that is used to check requests validity. The plugin adds the "apikey" value to the request URL. It is applicable for the 'application/x-www-form-urlencoded' encoding only. You may use customer headers for other encoding.

Successful processing sign (optional) - if it is specified, the plugin checks a server response, and mark the request invalid if the server response does not include the specified sign. The plugin tries to repeat unsuccessful requests three times.

Data encoding



Advanced

Encoding: application/json; charset=utf-8

Prepare and set the "HTTP_DATA" variable in any data filter plugin. For example, using the "Data Encode" plugin.

Parser item	Type	Name	Data type
EXT	Body	ext	string
DATE_TIME_STAMP	Body	ts	date
ACC	Body	account	string
DIRECTION	Body	dir	integer

[✓] Custom headers

```
Pragma: no-cache
```

Figure 5: Advanced options

By default, the plugin uses the 'application/x-www-form-urlencoded; charset=utf-8' content type encoding in the HTTP request.

String value - the plugin encodes all string to the UTF-8 encoding.

Decimal numbers - the plugin converts decimal number to a string without any separators (example, 123456789).

Float numbers - the plugin converts float number to a string with the dot delimiter (example, 123.456789).

Datetime - the plugin uses the ISO format YYYY-MM-DD HH:NN:SS (example, 2019-01-01 00:00:01).

Date - YYYY-MM-DD (example, 2019-01-01).

Time - HH:NN:SS (example, 00:00:01).

Boolean values - The plugin encodes the logical "true" to "TRUE", and logical "false" to "FALSE" (without quotes).

The plugin encodes all other data types as strings.

The "Advanced" tab allows you to specify extra options of the HTTP request:

Encoding - you may use a custom encoding type if you send data in JSON or SOAP formats.

application/json
application/soap
texp/plain
application/binary

If you select the custom encoding, you may prepare request data and place it to the "HTTP_DATA" or "HTTP_DATA_RAW" parser variables. For example, you can do it using the "Data encode" and "Expressions" plugins.

Below is the example expressions for the "Expressions" plugin.

HTTP_DATA='{"value": "12345", "value2": 2}'

HTTP_DATA - the plugin additionally encodes this data to UTF-8 and escapes special characters.
HTTP_DATA_RAW - the plugin does not modify this data and sends it as is.

JSON or XML SOAP data encoding

If the HTTP_DATA or HTTP_DATA_RAW variables are not defined, then the plugin attempts to prepare a request body using the list of parameters:

Parser item - source parser item with a value.

Type - you can choose where the parameter will be added, in the request body or in the request URL.

Name - it is the destination name in parameters or in a JSON/XML data. The destination name can contain the "/" path delimiter. For example, for a JSON data, you can specify "object1/object2/value", and the plugin will add the "value" attribute in the "object2" object.

Data type - you should choose the data type of the expected value. The plugin will use the appropriate formatting rules for it.

Please note, if this list is empty, the plugin will add all possible parser variables to the request body.

Custom headers

You may add custom HTTP headers for all your HTTP requests. You should add one header value per line.

Example:

My-Name: My-Value

Please note, you should use semi-color and space as a separator between the header name and value.

7 HTTP GET

The HTTP GET setup is mostly identical to HTTP POST, but with some differences:

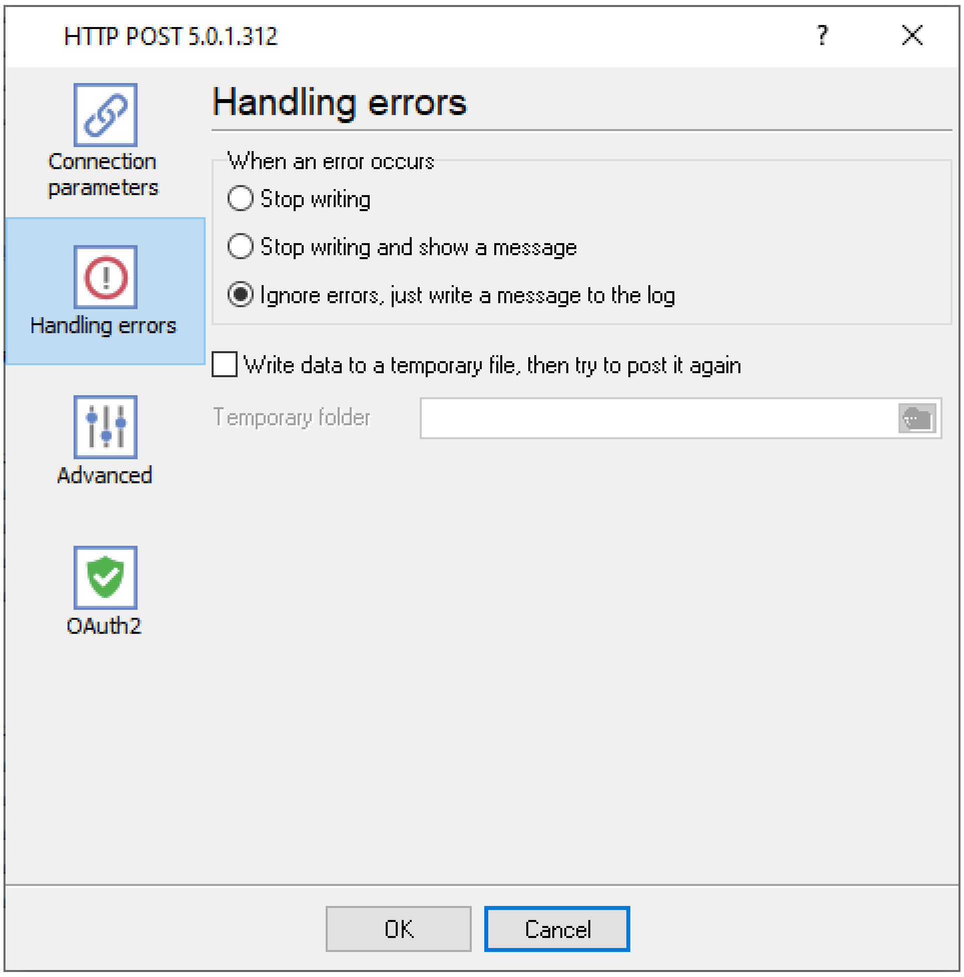
1.The HTTP GET request size is limited to 1024 bytes.

2. If a data packet contains the "HTTP_DATA" variable, the plugin will encode to a URL-compatible string and append the variable value to the request URL. The plugin will ignore all other variables.

3. If a data packet contains the "HTTP_DATA_RAW" variable, the plugin will append that value without any encoding to the request URL. The plugin will ignore all other variables.

8 Error handling

During the module's operation, some errors may occur when the program interacts with the database (for example, no connection to the database, temporary issues with the database, etc.). The "Handling errors" tab allows you to configure the error handling settings (Figure **6**).


HTTP POST 5.0.1.312

Handling errors

When an error occurs

( ) Stop writing

( ) Stop writing and show a message

(●) Ignore errors, just write a message to the log

[ ] Write data to a temporary file, then try to post it again

Temporary folder

Figure 6: Error handling settings

The following options of the module's response to an error are available:

1. **Stop writing** – If an error occurs, the program will generate an error message and set the internal status "**Temporarily disable**"; that is, any further data writing operations will be stopped until the module's settings are changed. Any data received for posting after the error has occurred will NOT be written to a temporary file or to the database. This option may be useful when you are configuring the module's settings.
2. **Stop writing and show a message** – This setting is similar to the previous one. The only difference is that an error message will be displayed.
3. **Ignore errors, just write a message to the log** – If an error occurs, the module will write an error message to the log, but will continue its operation.

In the last two cases, if you want to avoid the loss of data intended for posting, you can save them to a temporary file, which will be created in the **Temporary folder**. The data will be written to a temporary file only if the "**Write data to a temporary file**" option is turned on. If an error occurs when the program tries to post any data, the data will be written to that file.

Later the module will take five attempts to write data from the temporary file to the database. If all the five attempts to write data to the database fail, the data will be deleted from the temporary file (that is, the data will be lost).

Note that if you have not properly configured writing to the database, multiple attempts to restore data from the backup copy will significantly increase the server load and the use of network resources.

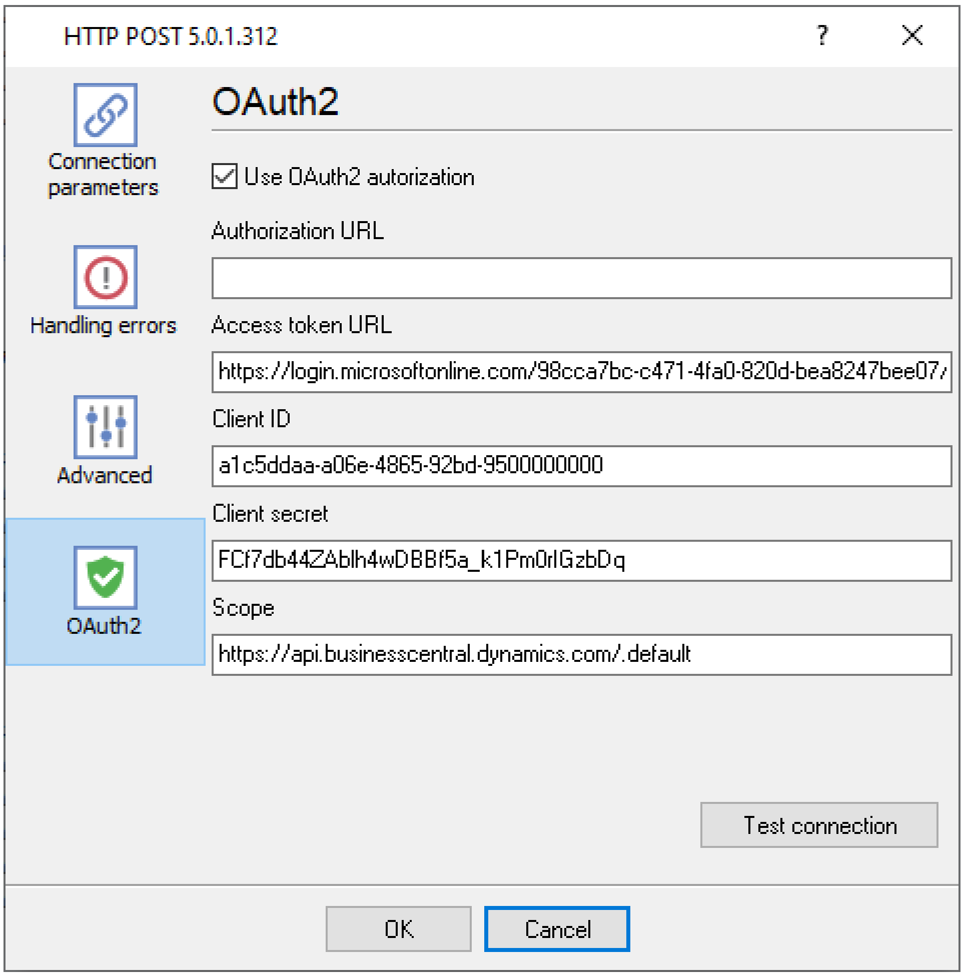
However, you can be assured that if you fix the error, all data from the temporary file will be written to the database.

Note that when specifying the **Temporary folder**, you need to select a folder to which the user who has launched the program has both read and write privileges. Otherwise, the module will be unable to create a temporary file, and the data will be lost. If you leave the "**Temporary folder**" field empty, the program will create a temporary file in the module installation folder.

When restoring data from a temporary file, the module can simultaneously use up to five data streams to transfer data to the server. Keep that in mind when configuring a firewall for the database server.

9 OAuth2

OAuth2 is an open standard protocol for authorization that allows applications to securely access user data from other services without exposing user credentials. It uses access tokens, issued by an authorization server, to grant limited access to resources on behalf of the user.



OAuth2

☑ Use OAuth2 autorization

Authorization URL

Access token URL: https://login.microsoftonline.com/98cca7bc-c471-4fa0-820d-bea8247bee07/

Client ID: a1c5ddaa-a06e-4865-92bd-9500000000

Client secret: FCf7db44ZAblh4wDBBf5a_k1Pm0rlGzbDq

Scope: https://api.businesscentral.dynamics.com/.default

Figure 7: OAuth2 parameters

Use OAuth2 authorization - If this option is enabled, the plugin uses the OAuth2 protocol with the supplied parameter. It retrieves the access token, checks its validity period, and refreshes it if necessary.

The plugin automatically adds the "Authorization: Bearer ...." HTTP header row to each request.

Authorization URL is the URL where the plugin retrieves long-life authorization and refresh tokens.

Access token URL is required to get the short-life access token for the current session. If the authorization token is empty, the program will use the "Client Credentials" grant type with the "**Client ID"** and "**Client secret"** parameters. If the authorization is not empty, the plugin will use the "Authorization Code" grant type if the authorization is not empty.

Scope - it is your API-defined scope of a requested access token. You can use the space delimiter to specify multiple scopes.

Test connection - When you click this button, the plugin will attempt to obtain an access token and report an error if it is impossible.