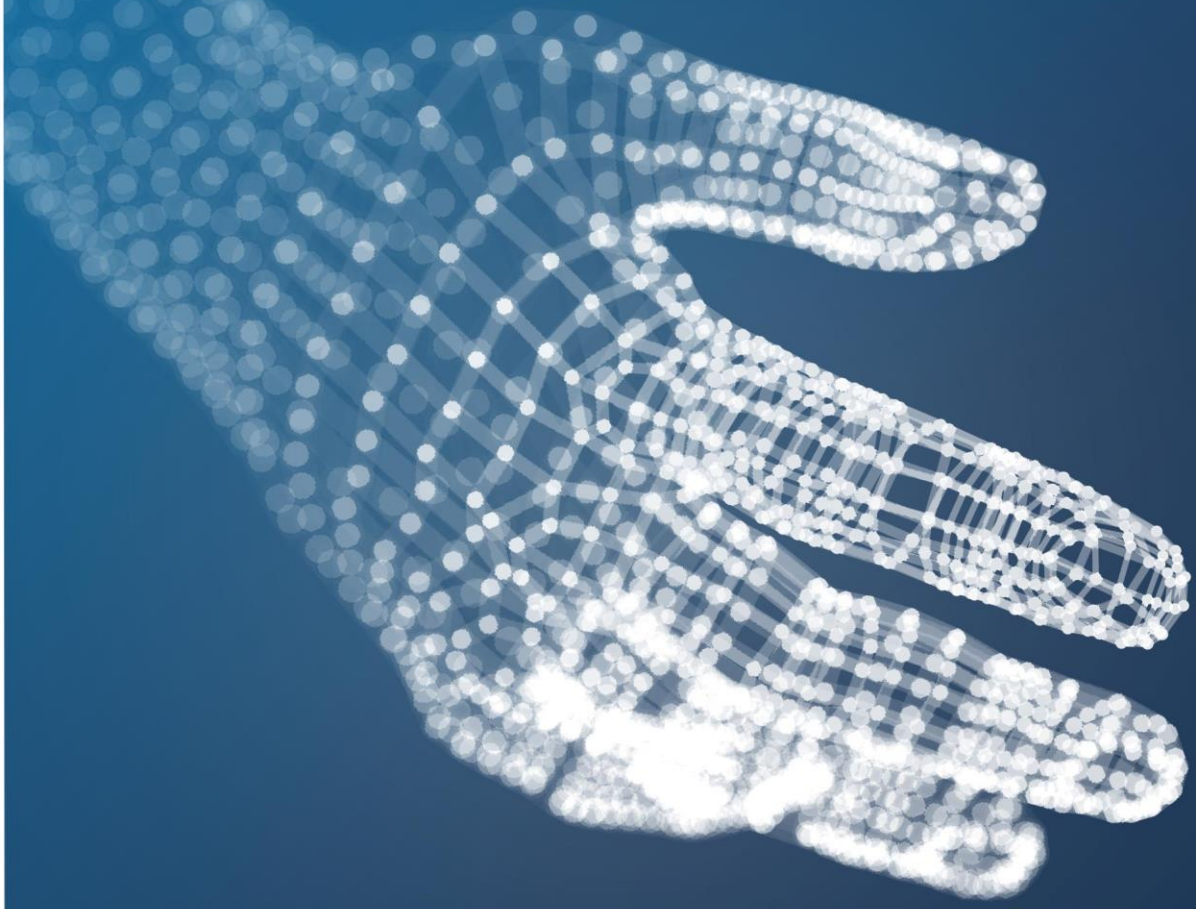
N3uron

Industrial IoT connectivity solutions



REFERENCE MANUAL

REST API

SERVER

www.n3uron.com

Index

Introduction

REST is a software architectural style used for exposing resources. REST calls can be used to retrieve data/information or send control commands to the system.

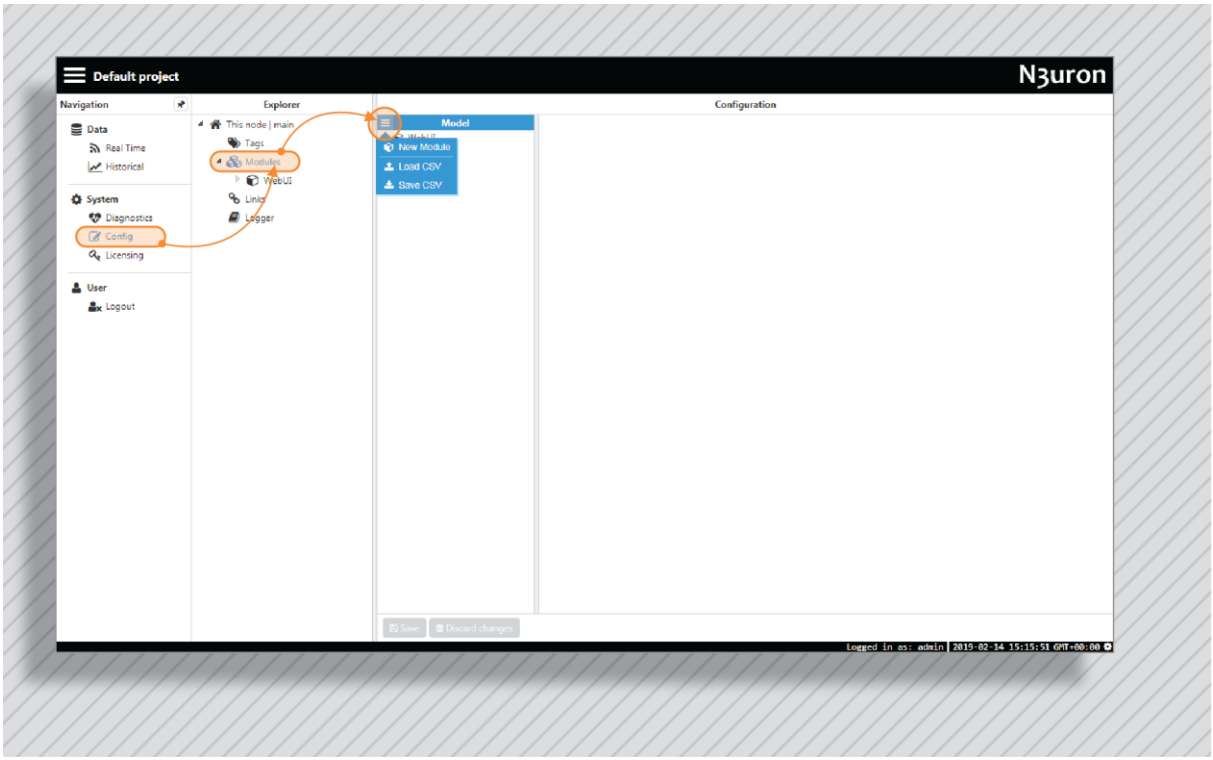
The REST server uses port 3003 by default. It can be disabled or initiated in another port if configured this way during the initial setup.

In order to create a REST request, a URL is built containing the IP address, service port, and all other information the server needs in order to complete the desired task. Data is returned in JSON format.

Creating Module Instances

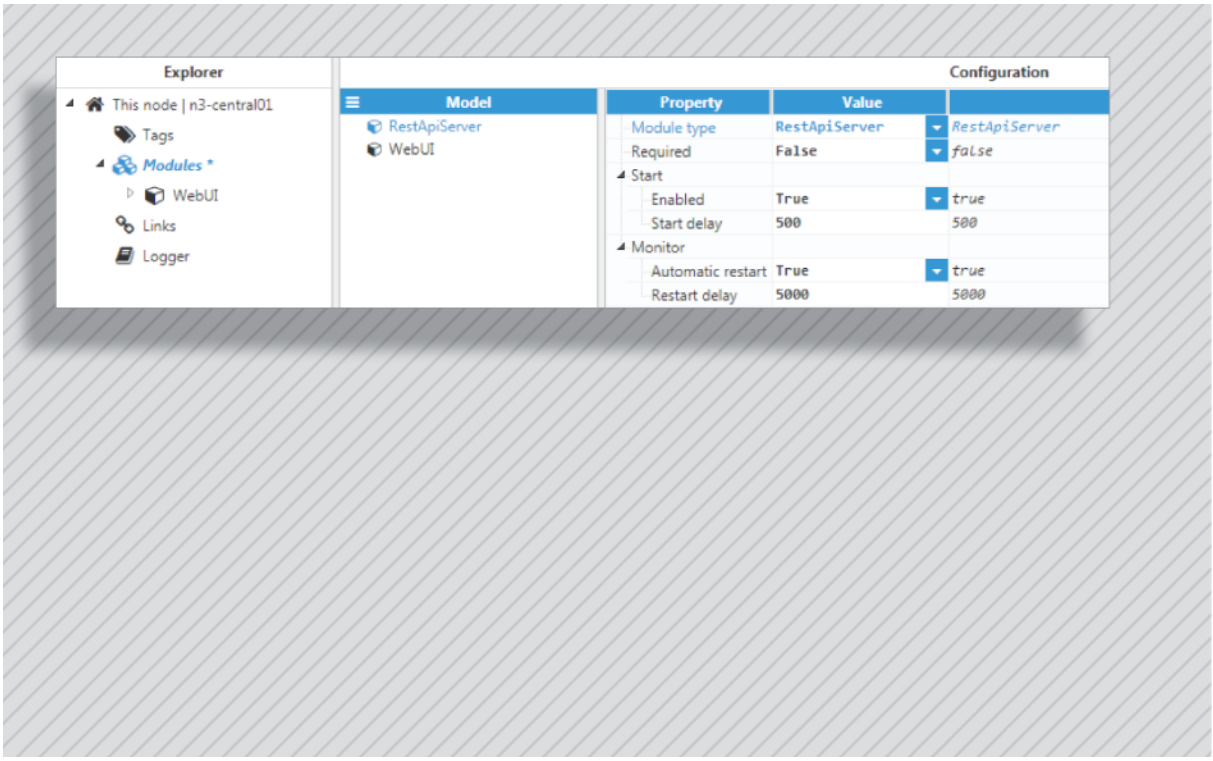
The first step when using **RestApiServer** after **N3uron** installation is to instantiate a RestApiServer module:

- Open **N3uron** and navigate to the “Config” menu.
- Click on “Modules”, then create a new module. This instance can be given any name (except names with special characters like ‘.’, ‘/’, etc.), although users are recommended to name instances in a similar way to the name of the module being instantiated for easy identification. In this example, it has been named **RestApiServer**.



Creating new module instances

By setting the module **type** to **RestApiServer**, the created instance will automatically become a **RestApiServer** instance. Once saved, **RestApiServer** should appear in bold in the module list because there are unsaved changes.



Setting the instance type

Each instance can also be configured with the following options:

- **Required:** When set to enabled and when this module is receiving data from other **N3uron** nodes, all links will be paused whilst the module is offline to avoid data loss. If disabled, this module will have no effect on links when offline.
- **Start:** This section controls how the module behaves when the **N3uron** service is started (which also includes service restarts).
    - **Enabled:** If true, the module will automatically start when the **N3uron** service starts. Otherwise, the module must be started manually.
    - **Start delay:** When automatic start is enabled, this setting is used to control how much delay there should be between starting the **N3uron** service and starting the module. This value is displayed in milliseconds.
- **Monitor:** This section is used to monitor the status of each module, as well as to enable automatic restart if the module goes offline.
    - **Automatic restart:** If true, whenever the module goes offline (except when manually stopped by the user) the module will automatically restart.
    - **Restart delay:** Determines the delay before restarting the module after going offline.

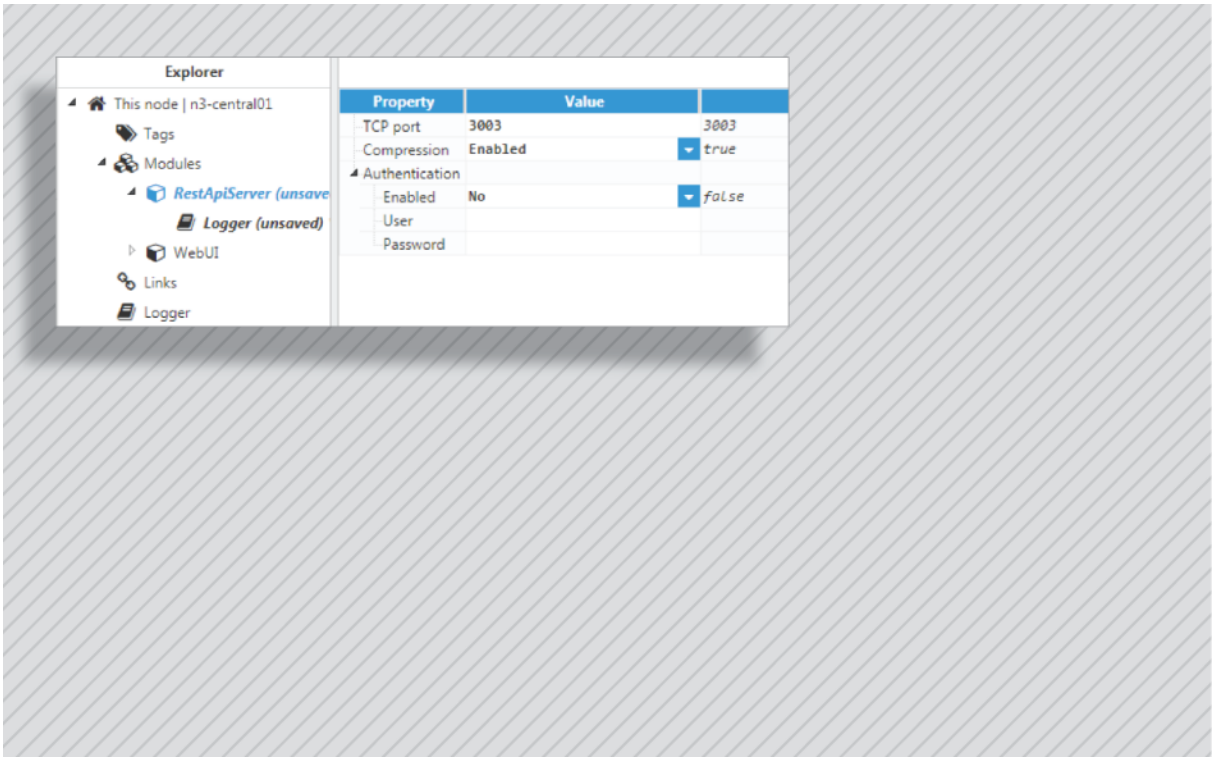
In addition to configuring this instance, each module has a Logger which needs to be configured separately. The default settings will be sufficient for this, but users will need to actively open the Logger configuration settings and save the default values to fully apply the settings.

Configuration

The RestApiServer configuration menu includes the following parameters:

- **TCP port:** TCP port for connecting to the REST API. The default setting is 3003. Valid range is 1 to 65535.
- **Compression:** When compression is enabled, the HTTP client is able to specify supported methods by setting the Accepted-Encoding header accordingly. Supported methods are GZIP (preferred) and DEFLATE.

- **Authentication:** enables basic HTTP authentication with a user and password. The password is encrypted for security reasons.



RestApiClient module configuration

Important: The TCP port configured for the RestApiServer must not be used by any other application within the same machine to avoid any conflicts.

N3uron REST API Calls

N3uron REST API calls use the GET method and can be categorised into the following groups:

- **Modules**
- **Links**
- **Licenses**
- **Tags**

Module calls

N3uron REST API module calls provide information about the status and monitoring of each module. In order to execute the desired command, module calls should start with http://<IP>:<PORT>/modules, followed by '?' and the parameters (key-values). The character '&' should be used to separate each parameter.

<IP> must be replaced by the server's IP address and <PORT> must be replaced by the port assigned to the REST API server (3003 by default).

Module commands:

- **GET**: Retrieves information about the module.
- **STOP**: Stops a module
- **START**: Starts a module
- **RESTART**: Restarts a module

GET command

Retrieves information about the modules:

```
http://<IP>:<port>/modules?cmd=get
```

STOP command

Stops a module using the name of the module as a parameter:

```
http://<IP>:<port>/modules?cmd=stop&name=WebUI
```

START command

Starts a module using the name of the module as a parameter:

```
http://<IP>:<port>/modules?cmd=start&name=WebUI
```

RESTART command

Restarts a module using the name of the module as a parameter:

```
http://<IP>:<port>/modules?cmd=restart&name=WebUI
```

Link calls

N3uron REST API link calls provide information about the status of the links in the target node. In order to execute the desired command, link calls should start with http://<IP>:<PORT>/links, followed by '?' and the parameters (key-values), using '&' to separate each parameter.
<IP> must be replaced by the server's IP address and <PORT> must be replaced by the port assigned to the REST API server (3003 by default).

Links commands:

- **GET**

GET command

Retrieves information about the links

```
http://<IP>:<port>/links?cmd=get
```

License calls

Provides information about the status of licenses in the target node.

License commands:

- **GET**

GET command

Retrieves information about the licenses

```
http://<IP>:<port>/licenses?cmd=get
```

Tag calls

N3uron REST API tag calls provide information about the status and monitoring of tags.

Tags commands:

- **Browse**
- **Details**
- **Read**
- **Write**
- **History**

Command parameters:

- **Path**
- **Value** (for write commands only)

Browse command

Retrieves a list of tags available in the node:

```
http://<IP>:<port>/tags?cmd=browse&path=/
```

The Path parameter can either be the entire tag hierarchy (“/”) or a specific branch. Browse commands do not require the use of wildcards to browse sub-groups contained within the specified path.

Details command

Retrieves a list of the tags provided with the parameter path, including current value, quality, timestamp and configuration. “*” can be added as a wildcard to filter the path of the required tags.

```
http://<IP>:<port>/tags?cmd=details&path=/*
```

Read command

Retrieves the current value, quality and timestamp of all tags provided with the parameter path. “*” can be used as a wildcard to filter the path of the required tags.

```
http://<IP>:<port>/tags?cmd=read&path=/*
```

Write command

Writes a value to the destination tag. Write commands must be executed tag by tag. The tag must be configured as read/write.

```
http://<IP>:<port>/tags?cmd=write&path=/Group01/SetPoint01&value=25.3
```

History command

The Historian module provides a set of calls to retrieve data stored over a period of time. Data is retrieved tag by tag and is returned in arrays of [time, value], expressing time in Epoch format (including milliseconds).

Every history call contains the following parameters:

- **Path**: Path of the tag.
- **Start**: Start date in ISO format: yyyy-mm-ddThh:mmZ. 'T' must be written after year-month-day and 'Z' after the hour.
- **End**: End date. Same format as above.
- **Options**: Information about the selected mode contained in an object.

The options parameter defines data retrieval:

- **Aggregated:** requires an aggregation interval (in milliseconds) and supports one of the following aggregated methods:
    - **Avg:** shows the average value for each interval.
    - **Min:** minimum value for each interval.
    - **Max:** maximum value for each interval.
    - **First:** first value for each interval.
    - **Last:** last value for each interval.
- **Raw:** returns values as stored in the database.
- **Delta:** requires a dead-band and returns values displaying changes that are bigger than the dead-band threshold.
- **Filter:** requires a dead-band and returns changes bigger than the dead-band threshold. The difference from delta is that this filter returns the value both before and after the change.

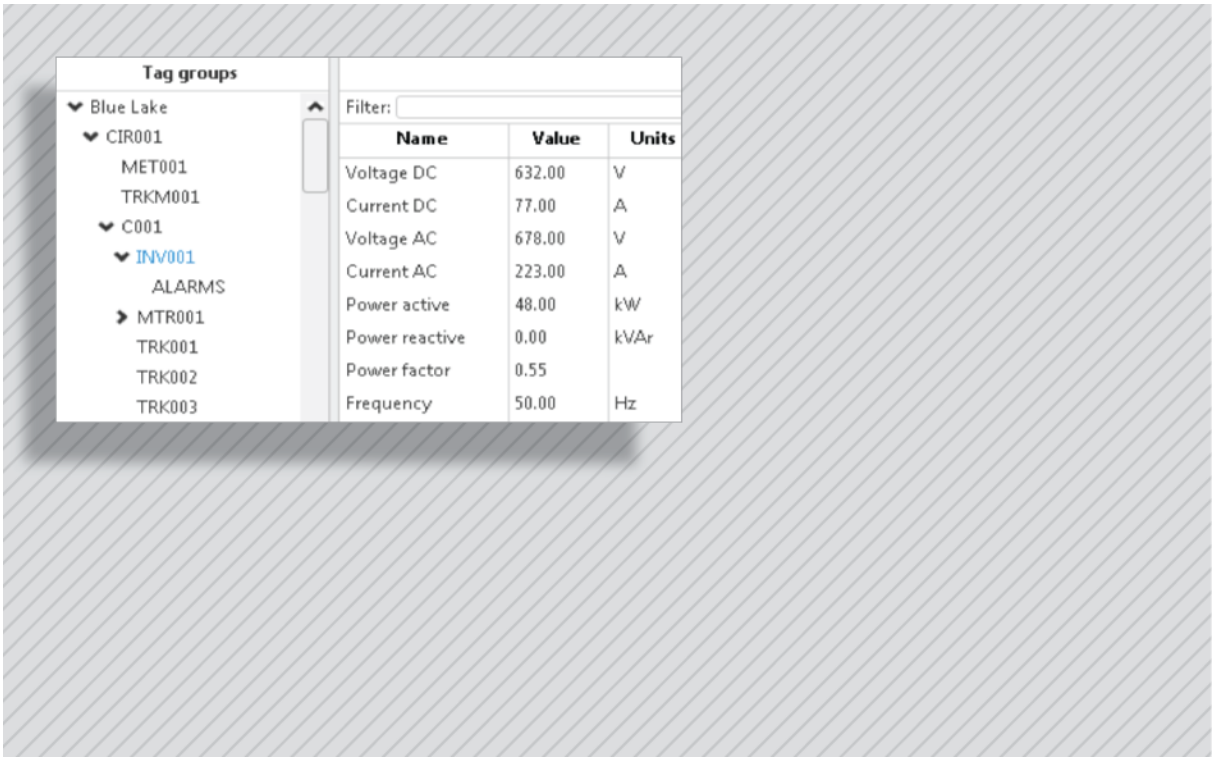
See below example of a request to read values with an average of 5 minutes from Historian:

```
http://<IP>:<port>/tags?cmd=history&path=/Group01/Temperature&start=2016-12-07T00:00Z&end=2016-12-08T00:00Z&options={"mode":"aggregated","method":"avg","interval":300000}
```

Examples

Reading values using the RestApiServer

For sites with the following tag groups (WebUI View):



Tag hierarchy

Browse command

Browse request from the local machine:

```
http://127.0.0.1:3003/tags?cmd=browse&path=/
```

Response (only initial elements shown):

```
{
  "groups": {
    "Blue Lake": {
      "groups": {
        "CIR001": {
          "groups": {
            "MET001": {
              "tags": [
                "Irradiation 30",
```

```
        "Irradiance 30",
        "Temperature BOM",
        "Temperature ambient",
        "Wind speed"
      ]
    },
    "TRKM001": {
      "tags": [
        "Azimuth",
        "Elevation",
        "G1 Tilt",
        "G2 Zenith"
      ]
    },
    "C001": {
      "groups": {
        "INV001": {
          "tags": [
            "Voltage DC",
            "Current DC",
            "Voltage AC",
            "Current AC",
            "Power active",
            "Power reactive",
            "Power factor",
            "Frequency"
          ],
          "groups": {
            "ALARMS": {
              "tags": [
                "Grid failure",
                "Frequency failure",
                "Blown fuse",
               ...
              ]
            }
          }
        }
      }
    }
```

```
                }
              }
            }
          }
        }
      }
```

Details command

Requesting the details of a specific tag:

```
http://127.0.0.1:3003/tags?cmd=details&path=/Blue
Lake/CIR001/C001/INV001/Current DC
```

Response:

```
{
  "fullPath": "/Blue Lake/CIR001/C001/INV001/Current DC",
  "remote": false,
  "status": {
    "value": 1,
    "ts": "2016-11-07T14:48:09.257Z"
  },
  "data": {
    "raw": 88,
    "value": 88,
    "quality": 192,
    "ts": "2016-12-08T12:48:16.660Z"
  },
  "config": {
    "type": "number",
    "default": 0,
    "deadband": "0.1u",
    "description": "DC current",
    "engUnits": "A",
    "clientAccess": "R",
    "simulation": {
      "enabled": false
    },
    "format": "%.2f",
    "security": {},
```

```
        "extensions": {
          "scaling": {
            "enabled": true,
            "raw": [
              0,
              100
            ],
            "eu": [
              0,
              100
            ],
            "clamp": [
              false,
              false
            ]
          },
          "source": {
            "enabled": true,
            "module": "ModbusClient",
            "config": {
              "device": "Channel01/Device01",
              "address": "401002",
              "type": "Int16",
              "rate": 600000
            }
          },
          "history": {
            "enabled": true,
            "module": "Historian",
            "config": {
              "mode": "change",
              "deadband": "0.01u",
              "interpolation": "linear",
              "defaultMethod": "avg",
              "rate": [
                600000,
                0
              ]
            }
          },
```

```
        "alarms": {},
        "events": {}
      }
    }
  }
```

Read command

Reading all tag values in the N3uron node

```
http://127.0.0.1:3003/tags?cmd=read&path=/*
```

Reading a specific tag

```
http://127.0.0.1:3003/tags?cmd=read&path=/Blue
Lake/CIR001/C001/INV001/Current DC
```

Response for a specific tag:

```
{
  "status": {
    "value": 1,
    "ts": "2016-11-07T14:48:09.257Z"
  },
  "data": {
    "raw": 88,
    "value": 88,
    "quality": 192,
    "ts": "2016-12-08T12:48:16.660Z"
  }
}
```

Write command

Writing a specific value in a Read/Write tag:

```
http://127.0.0.1:3003/tags?cmd=write&path=/Blue
Lake/CIR001/MET01/Pyr_offset&value=25.3
```

Response:

```
OK
```

History calls

Request to retrieve values with an average of 5 minutes:

```
http://127.0.0.1:3003/tags?cmd=history&path=/Blue
Lake/CIR001/C001/INV001/Current DC&start=2016-12-
07T00:00Z&end=2016-12-08T00:00Z&options={"mode":"aggregated",
"method":"avg","interval":300000}
```

Response:

```
{
  "request": {
    "tag": "/Blue Lake/CIR001/C001/INV001/Current DC",
    "start": 1481068800000,
    "end": 1481155200000,
    "options": {
      "mode": "aggregated",
      "method": "avg",
      "interval": 300000
    }
  },
  "data": [
    [
      1481069100000,
      -6
    ],
    [
      1481069400000,
      -6
    ],
```

```
        [
          1481069700000,
          -6
        ]
         ...
         ...
         ...
      ]
    }
```

Request to retrieve all raw values in the database for a specific tag:

```
http://127.0.0.1:3003/tags?cmd=history&path=/Blue
Lake/CIR001/C001/INV001/Current DC&start=2016-12-07T00:00Z&end=2016-
12-08T00:00Z&options={"mode":"raw"}
```

Request to retrieve values from Historian using Delta:

```
http://127.0.0.1:3003/tags?cmd=history&path=/Blue
Lake/CIR001/C001/INV001/Current DC&start=2016-12-
07T00:00Z&end=2016-12-08T00:00Z&options={"mode":"delta",
"deadband":"0.1"}
```

Request to retrieve values from Historian using Filter:

```
http://127.0.0.1:3003/tags?cmd=history&path=/Blue
Lake/CIR001/C001/INV001/Current DC&start=2016-12-
07T00:00Z&end=2016-12-08T00:00Z&options={"mode":"filter",
"deadband":"0.1"}
```

N3uron

Industrial IoT connectivity solutions

REFERENCE MANUAL

REST API

SERVER