# Table of Contents

# REST API Documentation

- The interface has to be a HTTP request with a JSON object (Encoding: UTF-8) in the request body.

Example:

```
POST /api/json HTTP/1.1
Content-Type: application/json; charset=utf-8
Host: localhost:8181
Content-Length: 79

{
   "tk": "1:QqjkQAtk4hyWRnbTt1+dZdGFCg3QE+nS",
   "browse": { "na": "" }
}
```

## Access

Access to the REST API can be granted using following URL:

```
http(s)://<ip/domain>:<port>/api/json
```

e.g.

```
http://localhost:8181/api/json
```

See Configuration of the Port

## Request & Response

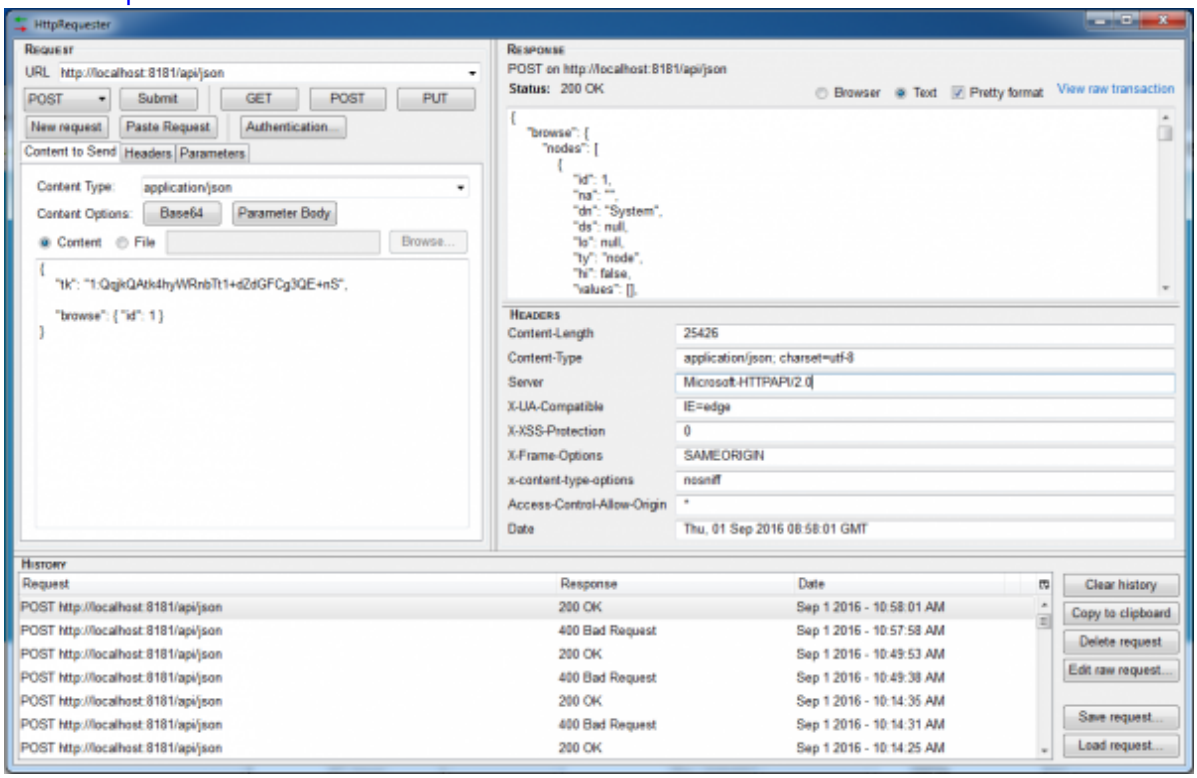
### Specifications of the Request

| | |
|---|---|
| **Method** | POST |
| **Content type** | `application/json` |
| **Encoding** | UTF-8 |

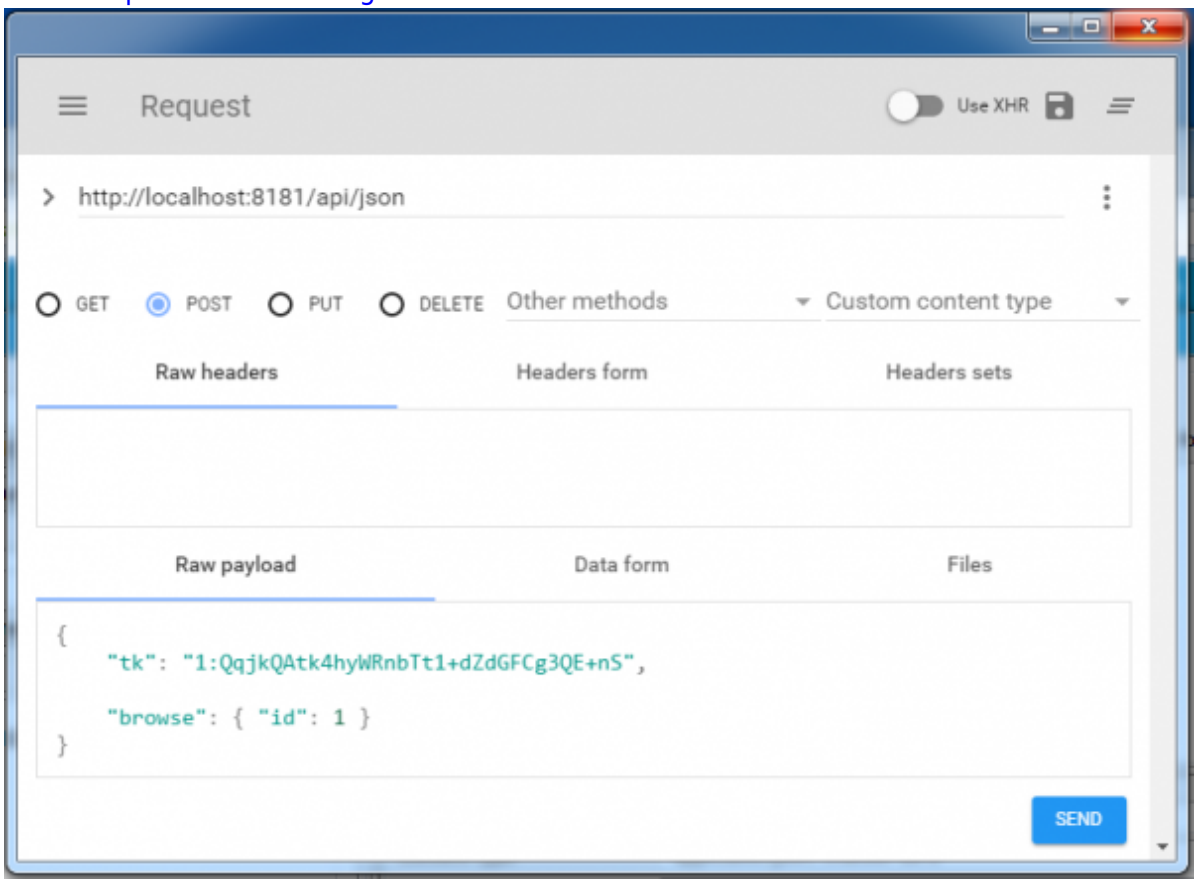
| **Content** | JSON object |
|---|---|

### Tools for Example Request

- HTTP Request Script
- HTTP Request Tool for Firefox




- HTTP Request Tool for Google Chrome



### General Request

| | Mandatory | Type | Purpose |
|---|---|---|---|
| `tk` | **yes** (if `username` and `password` are not specified) | string | The **Token** of the Node which is to be authenticated. From this Node, all children will be relatively addressed.<br>For how to find the Token see Node Access.<br><br>e.g. the token for Node `System/Plugins` is set, then `"na":"OPC UA Server"` is the relative path for Node `/System/Plugins/OPC UA Server`. |
| `username` | **yes** (if `tk` is not specified) | string | The email address or telephone number of the user to authenticate. |
| `password` | **yes** (if `tk` is not specified) | string | The password of the user. |
| `get` | no | object | Read Nodes |
| `browse` | no | object | Read Nodes with all children (recursive) |
| `set` | no | object | Write Nodes |
| `create` | no | object | Create Nodes |
| `update` | no | object | Update Nodes |
| `delete` | no | object | Delete Nodes |

**Example Request (using a Token):**

```
{
   "tk": "961:oseIqCm8WOOnPEE5g0tt1TZz2wbL/qUq",
   "get": {
      "na": "Temperature"
   }
}
```

**Example Request (using username-password):**

```
{
   "username": "demo@user.org",
   "password": "demo",
   "get": {
      "na": "/Demo-Nodes/Temperature"
   }
}
```

**Request frame:**

```
{
   "tk": /* Token */,
   "username": /* Username (if 'tk' not specified) */,
   "password": /* Password (if 'tk' not specified) */,

   "get":    { /* "Get" elements */ },
   "browse": { /* "Browse" elements */ },
   "set":    { /* "Set" elements */ },
```

```
    "create": { /* "Create" elements */ },
    "update": { /* "Update" elements */ },
    "delete": { /* "Delete" elements */ }
}
```

### General Response

| | Mandatory | Type | Purpose |
|---|---|---|---|
| get | no | object | Read & Browse response |
| browse | no | object | Read & Browse response |
| set | no | object | Write response |
| create | no | object | Create response |
| update | no | object | Update response |
| delete | no | object | Delete response |
| res | **yes** | object | Result of the query |

Possible res results:

| | Mandatory | Type | Purpose |
|---|---|---|---|
| value | **yes** | number | Result codes:<br>>= 0: OK - Everything is good!<br>< 0: An error has occured! |
| reason | no | string | Reason for the error |

**Example Response:**

```
{
   "get": {
      "nodes": [
         {
            "id": 963,
            "na": "Temperature",
            "dn": "Temperature (°C)",
            "ds": "",
            "lo": null,
            "ty": "double",
            "hi": false,
            "min": -20,
            "max": 90,
            "unit", "°C",
            "values": [
               {
                  "va": 2,
                  "ts": 1472738754519,
                  "st": 0
               }
            ]
         }
      ],
```

```
      "res": {
         "value": 0
      }
   }
}
```

**Response Frame:**

```
{
   "get":    { /* "Get" elements */ },
   "browse": { /* "Browse" elements */ },
   "set":    { /* "Set" elements */ },
   "create": { /* "Create" elements */ },
   "update": { /* "Update" elements */ },
   "delete": { /* "Delete" elements */ },
   "res":    { /* "Result" elements */ },
}
```

# Read & Browse

### Request

- The “Get” function makes it possible to read one or more (historical) values. “Browse” is like “Get” but allows only to get the actual value of the Node, therefore it will recursively return all child Nodes including their actual value.

| | Mandatory | Type | Purpose |
|---|---|---|---|
| `id` | **yes** (if `na` is not specified) | number | Local Identifier (“Id”) of the Node. |
| `na` | **yes** (if `id` is not specified) | string | Name of the Node.<br>When using a token for authentication, this is the relative path from the authenticated Node to the destination Node (if empty, the authenticated Node itself is the destination Node). Otherwise (for username-password authentication), it is the absolute path to the destination Node. |
| `count` (if `ttl` is not present) | no | number | The amount of history values (max. 1000). Default is 1, which means the actual value (instead of history values) is returned. |
| `from` (if `ttl` is not present) | no | number | If specified, the start timestamp from which to retrieve history values. |
| `to` (if `ttl` is not present) | no | number | If specified, the end timestamp up to which to retrieve history values. |

| | Mandatory | Type | Purpose |
|---|---|---|---|
| `ttl` (if `count` is not present) | no | number or `null` | Time To Live. If this parameter is not present, the function directly returns the current ("cached") value of the Node. If the parameter is present, a **synchronous read** is done on the Node in order to read from the underlying device. A value of `null` means to use the default maximum value age of the Node. Otherwise, the value is the age in milliseconds that the current Node value can have to be returned directly. If the value of the Node is older, it is read synchronously from the device.<br>To force a synchronous read regardless of the node's Max Value Age, specify a value of `0`. |

**Example Read Request (using a Token):**

```
{
   "tk": "961:oseIqCm8WOOnPEE5g0tt1TZz2wbL/qUq", /* Authentication for
"/Nodes/Demo-Nodes/" */
   "get": [{
      "na":    "Temperature", /* Reads Node: "/Nodes/Demo-
Nodes/Temperatur" */
   },
   {
      "id":    964,          /* Reads Node: "/Nodes/Demo-
Nodes/Pressure" */
      "count": 5             /* Reads the last 5 historical values */
   }]
}
```

**Example Read Request (using username-password):**

```
{
   "username": "demo@user.org",
   "password": "demo",

   "get": [{
      "na":    "/Nodes/Demo-Nodes/Temperature", /* must specify the
absolute Node Path here */
   },
   {
      "id":    964,          /* Reads Node: "/Nodes/Demo-
Nodes/Pressure" */
      "count": 5             /* Reads the last 5 historical values */
   }]
}
```

### Response

- For each requested "Get" or "Browse" object the Server will respond with a Node (in

array) which has the following properties:

| | Mandatory | Type | Purpose | | | |
|---|---|---|---|---|---|---|
| res | **yes** | object | Result of the query | | | |
| id | **yes** | number | Identifier | | | |
| na | **yes** | string | Name | | | |
| dn | **yes** | string | Display name | | | |
| ds | **yes** | string | Description | | | |
| lo | **yes** | string | Location | | | |
| ty | **yes** | string | Type | | | |
| hi | **yes** | boolean | History Options | | | |
| min | **yes** | number | Minimum value | | | |
| max | **yes** | number | Maximum value | | | |
| values | **yes** | object array | | | | |
| nodes | no (only browse) | object array | Contains all child Nodes (recursive). For each child the properties are the same as in this table. | | | |

Possible res results:

| | Mandatory | Type | Purpose |
|---|---|---|---|
| value | **yes** | number | Result Codes:<br>>= 0: OK - Everything is good!<br>< 0: An error has occured! |
| reason | no | string | Reason for the error |

Possible value results:

| | Mandatory | Type | Purpose |
|---|---|---|---|
| va | **yes** | any | Value |
| ts | **yes** | number | Timestamp |
| ct | **yes** | number | Category |
| st | **yes** | number | Status |
| sttext | no | string | Status text (if set) |

**Example Read Response:**


```
{
   "get": {
      "nodes": [
         {
            "id": 963,
            "na": "Temperature",
            "dn": "Temperature (°C)",
            "ds": "",
            "lo": null,
            "ty": "double",
            "hi": false,
            "min": -20,
            "max": 90,
            "values": [
```

```
                {
                    "va": 78,
                    "ts": 1472740618590,
                    "st": 0
                }
            ]
        },
        {
            "id": 964,
            "na": "Pressure",
            "dn": "Pressure",
            "ds": "",
            "lo": "",
            "ty": "double",
            "hi": true,
            "min": 10,
            "max": 110,
            "values": [
                {
                    "va": 86,
                    "ts": 1472740619105,
                    "st": 0
                },
                {
                    "va": 84,
                    "ts": 1472740617887,
                    "st": 0
                },
                {
                    "va": 98,
                    "ts": 1472740616967,
                    "st": 0
                },
                {
                    "va": 32,
                    "ts": 1472740615136,
                    "st": 0
                },
                {
                    "va": 31,
                    "ts": 1472740612718,
                    "st": 0
                }
            ]
        }
    ],
    "res": {
        "value": 0
    }
```

```
    }
}
```

# Write

### Request

- With the “Set” function it is possible to write values to a specified Node.
- If the timestamp is not set, it is important to send the requests for the same Node sequentially. So before sending another request it is important to wait for the successful response. Otherwise it could happen, that the values are sorted in another sequence as intended. Different Nodes of course can be written parallelly.

| | Mandatory | Type | Purpose |
|---|---|---|---|
| id | **yes** (if na is not specified) | number | Identifier |
| na | **yes** (if id is not specified) | string | Name |
| va | **yes** | any | new value |
| ts | No | number | Timestamp |
| st | No | number | Status |

**Example Write Request (using a Token):**

```
{
    "tk": "961:oseIqCm8WOOnPEE5g0tt1TZz2wbL/qUq", /* Authentication
for "/Nodes/Demo-Nodes/" */
    "set": [
        {
            "na": "Temperature",  /* Write Value 21 to Node:
"/Nodes/Demo-Nodes/Temperatur" */
            "va": 21
        },
        {
            "id": 964,            /* Write Value 84 with the
Timestamp 02.09.16 07:09   */
            "va": 84,             /* and the Status 0
*/
            "ts": 1472800198510,  /* to Node: "/Nodes/Demo-
Nodes/Pressure"              */
            "st": 0
        }
    ]
}
```

**Example Write Request (using username-password):**

```
{
    "username": "demo@user.org",
    "password": "demo",

    "set": [
        {
            "na": "/Nodes/Demo-Nodes/Temperature",  /* Write Value 21
to Node (specified by absolute Node Path) */
            "va": 21
        },
        {
            "id": 964,            /* Write Value 84 with the Timestamp
02.09.16 07:09   */
            "va": 84,             /* and the Status 0
*/
            "ts": 1472800198510,  /* to Node: "/Nodes/Demo-
Nodes/Pressure"              */
            "st": 0
        }
    ]
}
```

### Response

- For the “Set” function the Node array only has response objects for failed requests objects.

| | **Mandatory** | **Type** | **Purpose** |
|---|---|---|---|
| res | **yes** | object | Result of the query |
| nodes | **yes** | object array | Contains all child Nodes (recursive). For each child the properties are the same as in this table. |

Possible res results:

| | **Mandatory** | **Type** | **Purpose** |
|---|---|---|---|
| value | **yes** | number | Result codes:<br>>= 0: OK - Everything is good!<br>< 0: An error has occured! |
| reason | no | string | Reason for the error |

**Example Write Response:**

```
{
   "set": {
      "nodes": [],
      "res": {
         "value": 0
      }
   }
```

```
}
```

**Example Write Response (Error Case):**

```
{
   "set": {
      "nodes": [
         {
            "id": 964,
            "va": 84,
            "ts": 4728001980, /* 24.02.1970 17:20 */
            "st": 0,
            "res": {
               "value": -1,
               "reason": "values[0].Timestamp is lower than 01.01.2000
00:00:00 +00:00"
            }
         }
      ],
      "res": {
         "value": -1,
         "reason": "At least one error occured when processing the
nodes: values[0].Timestamp is lower than 01.01.2000 00:00:00 +00:00"
      }
   }
}
```

# Create

### Request

| | Mandatory | Type | Purpose |
|---|---|---|---|
| pid | **yes** (if pna is not specified) | number | Parent Identifier<br>New Node will be created as a Child of this Node. |
| pna | **yes** | string | Parent Name<br>New Node will be created as a Child of this Node.<br><br>e.g if pna="Demo-Nodes", the new Node will be created in this folder. The new Node will be "/Nodes/Demo-Nodes/Name" |
| na | **yes** (if pid is not specified) | string | Name |
| dn | no | string | Display name |
| ds | no | string | Description |
| lo | no | string | Location |
| path | no | string | Path (e.g. for device variables) |

| | Mandatory | Type | Purpose |
|---|---|---|---|
| ty | **yes** | string | Type |
| hi | no | number | Hysteresis |
| min | no | number | Minimum value |
| max | no | number | Maximum value |

**Example Create Request:**

```
{
   "tk": "938:843uqQeSaLAg+7TALdOhGDkLOHFZevZw", /* Authentication for "/Nodes" */
   "create": [{
      "pna": "Demo-Nodes",              /* Creates Node "/Nodes/Demo-Nodes/New-Node-1" */
      "na": "New-Node-1",               /* with Type string */
      "ty": "string"
   },
   {
      "pid": 961,                       /* Creates Node "Nodes/Demo-Nodes/New-Node-2"  */
      "na": "New-Node-2",               /* with specified Displayname (dn),          */
      "dn": "Display New-Node-2",       /* Description (ds) and Type double          */
      "ds": "This is the new Node 2",
      "ty": "double"
   }]
}
```

### Response

| | Mandatory | Type | Purpose |
|---|---|---|---|
| res | **yes** | object | Result of the query |
| nodes | **yes** | object array | Contains all child Nodes (recursive). For each child the properties are the same as in this table. |

Possible res results:

| | Mandatory | Type | Purpose |
|---|---|---|---|
| value | **yes** | number | Result Codes:<br>>= 0: OK - Everything is good!<br>< 0: An error has occured! |
| reason | no | string | Reason for the error |

**Example Create Response:**

```
{
   "tk": "938:843uqQeSaLAg+7TALdOhGDkLOHFZevZw",
   "create": [{
```

```
      "pna": "Demo-Nodes",
      "na": "New-Node-1",
      "ty": "string"
   },
   {
      "pid": 961,
      "na": "New-Node-2",
      "dn": "Display New-Node-2",
      "ds": "This is the new Node 2",
      "ty": "double"
   }]
}
```

**Exampe Create Response (Error Case):**

```
{
   "create": {
      "nodes": [
         {
            "pna": "Demo-Nodes",
            "na": "Existing-Node",
            "ty": "string",
            "res": {
               "value": -1,
               "reason": "An object with the same name does already
exist. Please choose another name."
            }
         },
         {
            "pid": 961,
            "na": "New-Node-2",
            "dn": "Display New-Node-2",
            "ds": "This is the new Node 2",
            "ty": "qwertz",
            "res": {
               "value": -1,
               "reason": "Could not find the Node Type \"qwertz\"."
            }
         }
      ],
      "res": {
         "value": -1,
         "reason": "At least one error occured when processing the
nodes: An object with the same name does already exist. Please choose
another name."
      }
   }
}
```

# Update

- Modify all properties of a Node.
- This feature is not implemented yet but will be comming soon.

# Delete

- Remove a Node.
- This feature is not implemented yet but will be comming soon.