VersaSense NV, Kapeldreef 60, B-3001 Leuven, info@versasense.com

Interacting with a MicroPnP network through the RESTful API

Important note: methods described in this guide are supported for network gateway software versions up to 1.0.2.9.

API descriptions for newer gateway software versions are available on: www.versasense.com/docs

Please contact support@versasense.com for the latest gateway software version

1. Connecting and experimenting with the RESTful API

The REST API on the gateway is accessible for client connections. GET requests can typically be issued directly from within your web browser.

Note that the Developer API must be enabled on the Network Manager configuration page.

For example:

- When connected over WiFI: https://gateway.versasense.com:8889/api/v1/devices

Useful tools for testing the API are specific browser extensions such as, for example, Postman (Google Chrome) or RESTer (Firefox). These tools can be used to experiment with PUT-based methods. When experimenting with PUT-methods, please pay attention to the content-types specified for every request.

- Postman (Google Chrome): https://www.getpostman.com/
- RESTer (Firefox): https://addons.mozilla.org/en-US/firefox/addon/rester/

Available methods (up to gateway version 1.0.2.9)

2. Retrieve list of MicroPnP devices

This method returns a list of JSON objects corresponding with the MicroPnP devices currently available in the network.

GET /api/v1/devices

Request parameters: none
Response format: application/json

Example requests and responses:

GET /api/v1/devices

```
HTTP/1.1 200 OK
[
  {
     "uid": "15e4bdcf-5f8b-48b1-b5cc-a8512a8514a9",
     "peripherals": [ ],
     "name": "Network Manager",
     "description": "My Network Manager",
     "location": "Office 02.30",
     "ip_address": "fd34::0017:0d00:0058:b31d",
     "type": "uManager",
     "battery": 100,
     "mac": "00-17-0D-00-00-58-B3-1D",
     "status": "OPERATIONAL"
  },
  {
      uid: "033e01e1-9554-433e-85d1-dc63769e70e0",
      "peripherals" : [
```

```
            {
                  "sampling_rate": "30",
                  "last_updated": "Mon Aug 22 09:29:23 CEST 2016",
                  "id": "1010/9000",
                  "type": "smartbattery"
            },
            {
                  "sampling_rate": "10",
                  "last_updated": "Mon Aug 22 09:29:27 CEST 2016",
                  "id": "3303/5700",
                  "type": "temperature"
            },
            {
                  "sampling_rate": "10",
                  "last_updated": "Mon Aug 22 09:29:28 CEST 2016",
                  "id": "3304/5700",
                  "type": "humidity"
            }
      ],
      "name": "thing1",
      "description": "My description",
      "location": "My location",
      "ip_address": "fd34::0017:0d00:0058:cd55",
      "type": "uDevice",
      "battery": 99.97,
      "mac": "00-17-0D-00-00-58-CD-55",
      "status": "OPERATIONAL"
  }
]
```


3. Retrieve information about a particular MicroPnP device

These methods can be used to retrieve more information of a particular MicroPnP device available in the mesh network.

GET /api/v1/devices/{id}

Request parameters: device uid (required)
Response format: application/json

GET /api/v1/devices/name/{name}

Request parameters: name: device name (required)
Response format: application/json

Example requests and responses:

GET /api/v1/devices/033e01e1-9554-433e-85d1-dc63769e70e0

GET /api/v1/devices/name/thing1

```
HTTP/1.1 200 OK
 {
      "uid": "033e01e1-9554-433e-85d1-dc63769e70e0",
      "peripherals" : [
            {
                  "sampling_rate": "30",
                  "last_updated": "Mon Aug 22 09:29:23 CEST 2016",
                  "id": "1010/9000",
                  "type": "smartbattery"
            },
            {
                  "sampling_rate": "10",
                  "last_updated": "Mon Aug 22 09:29:27 CEST 2016",
                  "id": "3303/5700",
                  "type": "temperature"
            },
            {
                  "sampling_rate": "10",
                  "last_updated": "Mon Aug 22 09:29:28 CEST 2016",
                  "id": "3304/5700",
                  "type": "humidity"
            }
      ],
      "name": "thing1",
      "description": "unknown",
      "location": "unknown",
      "ip_address": "fd34::0017:0d00:0058:cd55",
      "type": "uDevice",
      "battery": 99.97,
      "mac": "00-17-0D-00-00-58-CD-55",
      "status": "OPERATIONAL"
 }
```

4. Rename a MicroPnP device and update information

These methods can be used to update the name, description, and location of a MicroPnP device. The reply confirms the update.

PUT /api/v1/devices/{id}

PUT /api/v1/devices/name/{name}

Request parameters: device name or id (required)
Contents of JSON structure: name (optional), description (optional), location (optional)

Name, description, and location are all optional. Parameters that are omitted in the JSON structure are not taken into account when updating device information.

Response format: application/json

Example requests and responses:

```
PUT /api/v1/devices/033e01e1-9554-433e-85d1-dc63769e70e0

PUT /api/v1/devices/name/thing1
Content-Type: application/json
{
  "name" : "my new name",
  "description": "my new description",
  "location" : "my new location"
}
```

```
HTTP/1.1 200 OK
{
   << updated MicroPnP device JSON structure >>
}
```

5. Retrieve peripherals available on MicroPnP device

This method retrieves more information about all peripherals or one particular peripheral on a device.

GET /api/v1/devices/{id}/peripherals

GET /api/v1/devices/name/{name}/peripherals

Request parameters: device id or name (required)
Response format: application/json

GET /api/v1/devices/{id}/peripherals/{peripheralID}

GET /api/v1/devices/name/{name}/peripherals/{peripheralID}

Request parameters: peripheral id, device id or name (required)
Response format: application/json

Example requests and responses:

```
GET /api/v1/devices/033e01e1-9554-433e-85d1-dc63769e70e0/peripherals

GET /api/v1/devices/name/thing1/peripherals
```


```
[
	{
		"sampling_rate": "30",
		"last_updated": "Mon Aug 22 09:29:23 CEST 2016",
		"id": "1010/9000",
		"type": "smartbattery"
	},
	{
		"sampling_rate": "10",
		"last_updated": "Mon Aug 22 09:29:27 CEST 2016",
		"id": "3303/5700",
		"type": "temperature"
	},
	{
		"sampling_rate": "10",
		"last_updated": "Mon Aug 22 09:29:28 CEST 2016",
		"id": "3304/5700",
		"type": "humidity"
	}
]
```


6. Sample a sensing or actuation peripheral remotely

This method allows to sample a particular sensing or actuation peripheral on a MicroPnP device. Note that this method will issue a separate GET method inside the MicroPnP mesh network.

GET /api/v1/devices/{id}/peripherals/{peripheralID}/sample

GET /api/v1/devices/name/{name}/peripherals/{peripheralID}/sample

Request parameters: name: device name and peripheralID (required)
Response format: application/json

Example requests and responses:

```
GET /api/v1/devices/name/thing1/peripherals/3303/5700/sample
```

```
HTTP/1.1 200 OK
```

```
 {
      "id" : "3303/5700",
      "Last_updated" : "Mon Aug 22 09:29:23 CEST 2016",
      "type" : "temperature",
      "value" : 27.6
}
```

7. Update periodical sampling rate of sensing peripheral

This method updates the sampling rate of a peripheral on a device. The sampling rate supplied should be a positive natural number.

PUT /api/v1/devices/{id}/peripherals/{peripheralID}/rate

PUT /api/v1/devices/name/{name}/peripherals/{peripheralID}/rate

Request parameters: device id or name (required).
Contents of JSON structure: sampling_rate (mandatory, and with positive integer as value)
Response format: application/json

Example requests and responses:

```
PUT
/api/v1/devices/033e01e1-9554-433e-85d1-dc63769e70e0/peripherals/3338
/5850/rate

PUT /api/v1/devices/name/thing1/peripherals/3338/5850/rate

Content-Type: application/json
{ "sampling_rate" : 30 }
```

```
HTTP/1.1 200 OK
{
  << updated device description >>
}
```

8. Control actuator remotely

This method allows to remotely control an actuation peripheral.

PUT /api/v1/devices/{id}/peripherals/{peripheralID}/actuate

PUT /api/v1/devices/name/{name}/peripherals/{peripheralID}/actuate

Request parameters: device id or name, and peripheralID (required)
Contents of JSON structure: contents (mandatory, and with actuator specific value)
Response format:
HTTP/1.1 200 if succeeded, HTTP/1.1 500 in case of error

Example requests and responses:

```
PUT /api/v1/devices/name/thing1/peripherals/3338/5850/actuate

Content-Type: application/json
{"contents" : "1"}
```

```
HTTP/1.1 200 OK
```