



oneM2M – Zigbee Interworking

HyeonSeo Son (hyeonseo0128@keti.re.kr)

Korea Electronics Technology Institute

2020.10.08

The project "International Digital Cooperation - ICT Standardisation" is funded by the European Union



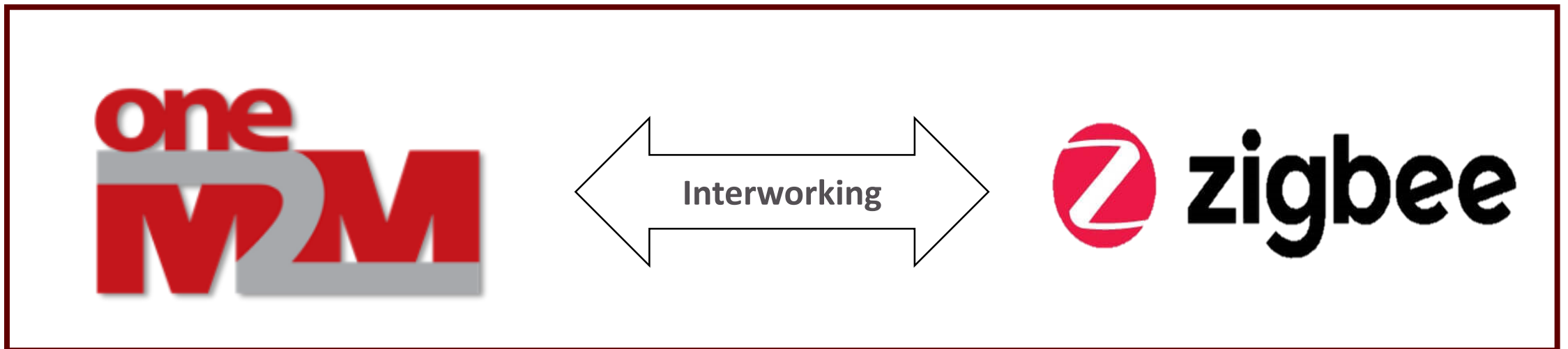
- Zigbee protocol is used a lot for home devices, so providing Zigbee interworking can be beneficial to have smart home services later
- oneM2M defines home appliances information model (i.e. TS-0023) and this Zigbee interworking used that standard models
- TS-0023 adopts Smart Device Template (SDT) to represent home appliances and their mapping to <flexContainer> resources. This tutorial demonstrates how to use those specifications for Zigbee interworking

- Scope
- What is Zigbee?
 - Zigbee Protocol
 - Zigbee Characteristics
- oneM2M-Zigbee Interworking overview
 - Smart Device Template(SDT)
 - IPE Architecture
 - System Deployment
- Demo Implementation
 - Zigbee gateway installation
 - Register Zigbee sensor
 - Get gateway API key

Scope



- oneM2M-Zigbee interworking process based on oneM2M standard
- Zigbee data model is mapped to oneM2M resources
- How to apply Home Application using Smart Device Template(SDT) from TS-0023
- Practice IPE using Zigbee gateway and sensor



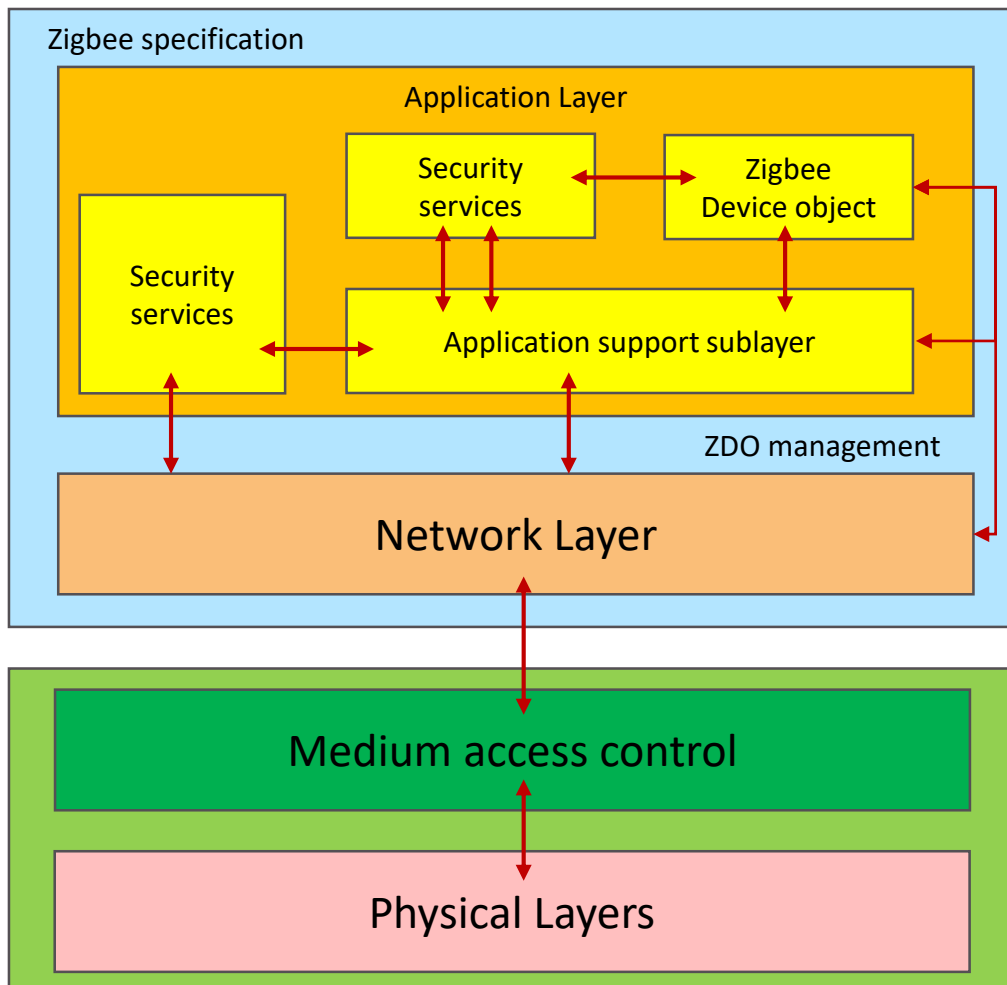
Prerequisites

- H/W
 - Raspberry pi3B+, Conbee 2(Gateway module), Zigbee sensors(Xiaomi aquara)
- S/W
 - Node.js
 - Postman
 - cmdr(Terminal tool)
 - Zigbee gateway software(deCONZ)
 - IPE source
 - Github(https://github.com/loTKETI/zigbee_IPE.git)



What is Zigbee?

Zigbee Protocol



- Zigbee is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used to create personal area networks with small, low-power digital radios, such as for home automation, medical device data collection, and other low-power low-bandwidth needs, designed for small scale projects which need wireless connection
- The specification includes four additional key components: network layer, application layer, *Zigbee Device Objects* (ZDOs) and manufacturer-defined application objects
- ZDOs are responsible for some tasks, including keeping track of device roles, managing requests to join a network, as well as device discovery and security.

source: <https://en.wikipedia.org/>

Zigbee Characteristics

- Standard wireless network established for low latency and energy consumption
- Supports simple protocol stacks to provide network reliability
- Zigbee is simpler and cheaper than other communication technologies such as Bluetooth and Wi-Fi



Zigbee Device



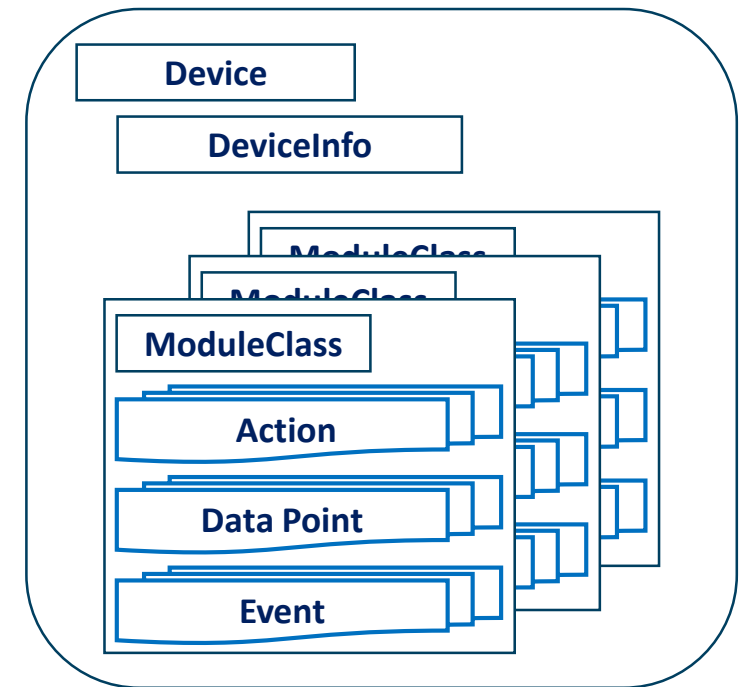
Zigbee Gateway



oneM2M-Zigbee Interworking overview

Smart Device Template (SDT)

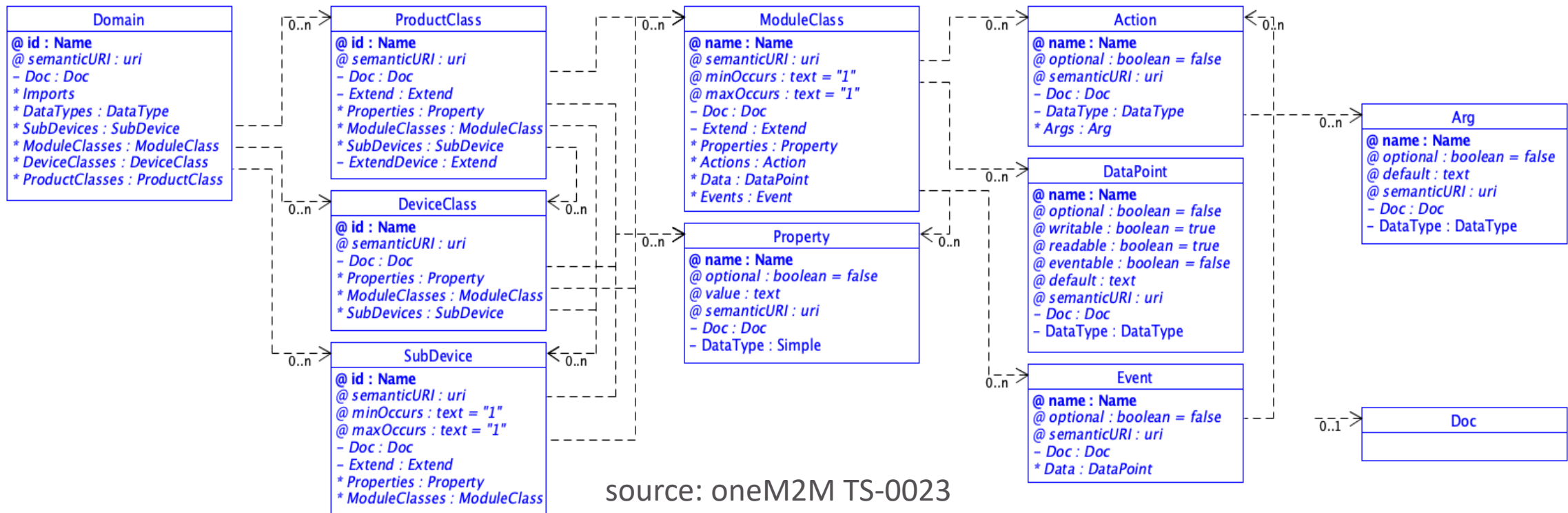
- SDT template is made to integrate smart home devices from home gateway, and it defines DeviceInfo and ModuleClass
- The ModuleClass has action, datapoint and event data
- SDT Goals
 - Modularize device functions and types
 - Keep it simple to device development
 - The SDT supports the use of a set of templates for generic devices or appliances which form the basis of APIs used by application developers.



source: <http://www.onem2m.org>

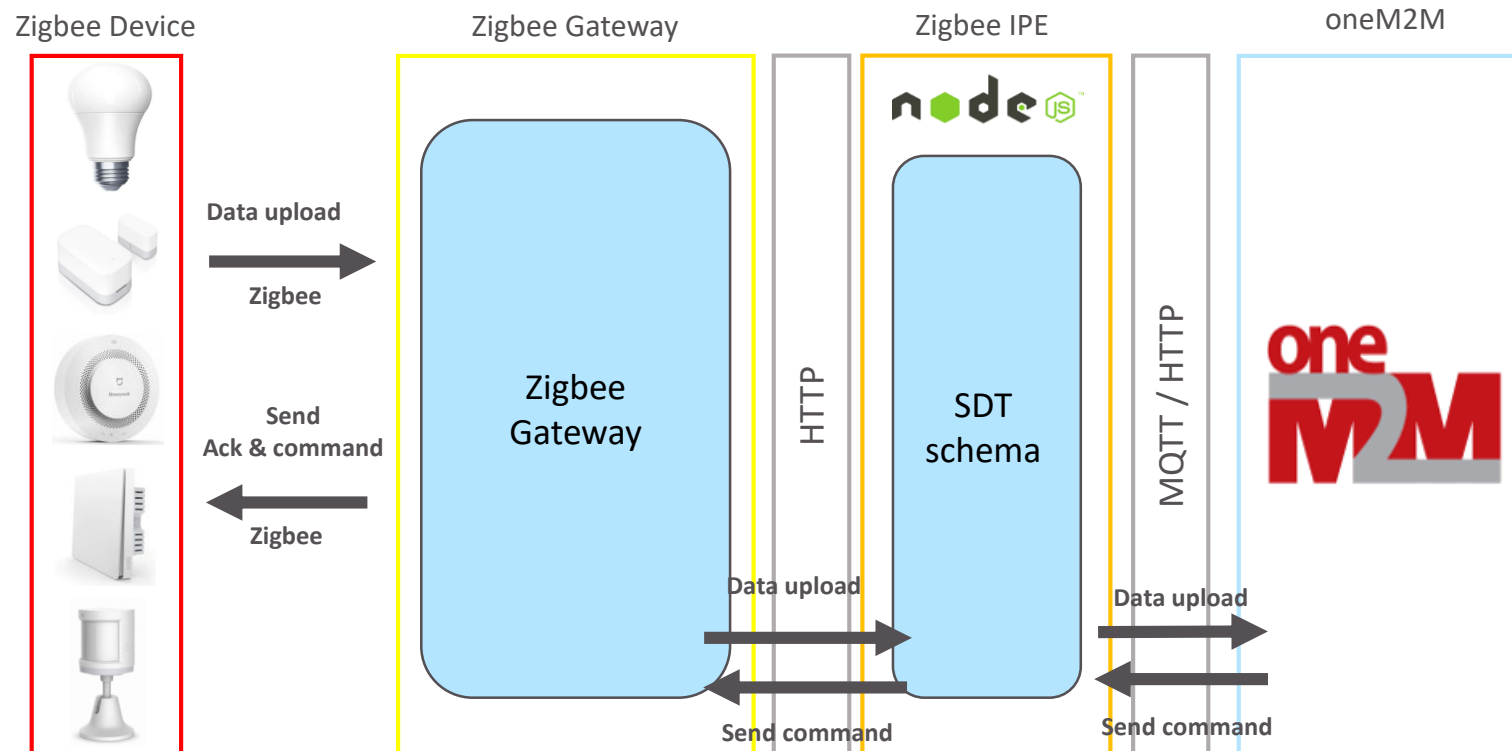
Smart Device Template (SDT)

- SDT hierarchy
 - Devices have module classes, and they are mapped to <flexContainer> resource
 - Each ModuleClass can have action, data point properties and event



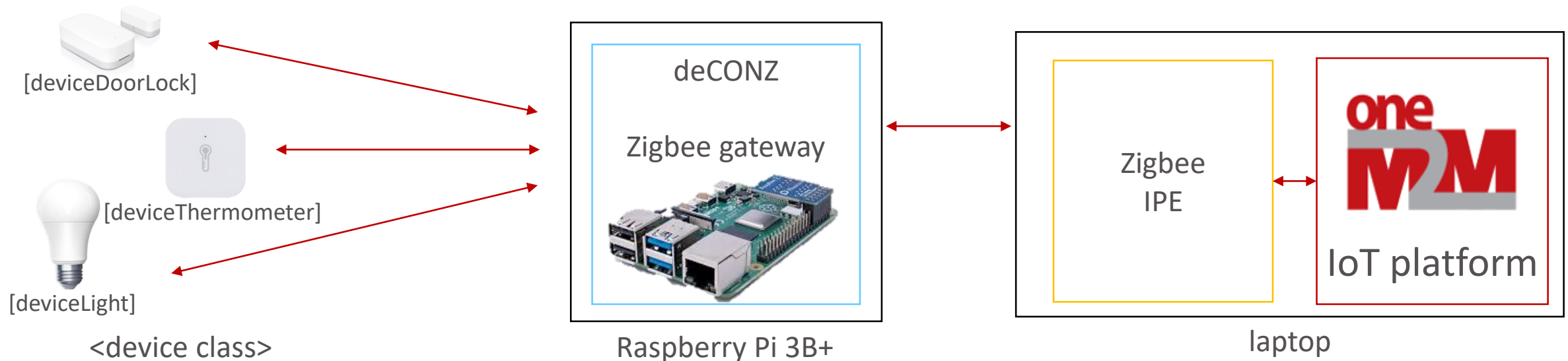
Architecture for IPE

- Interworking Proxy Entity(IPE) with others
 - IPE is a special AE that allows seamlessly interaction between oneM2M system and other systems

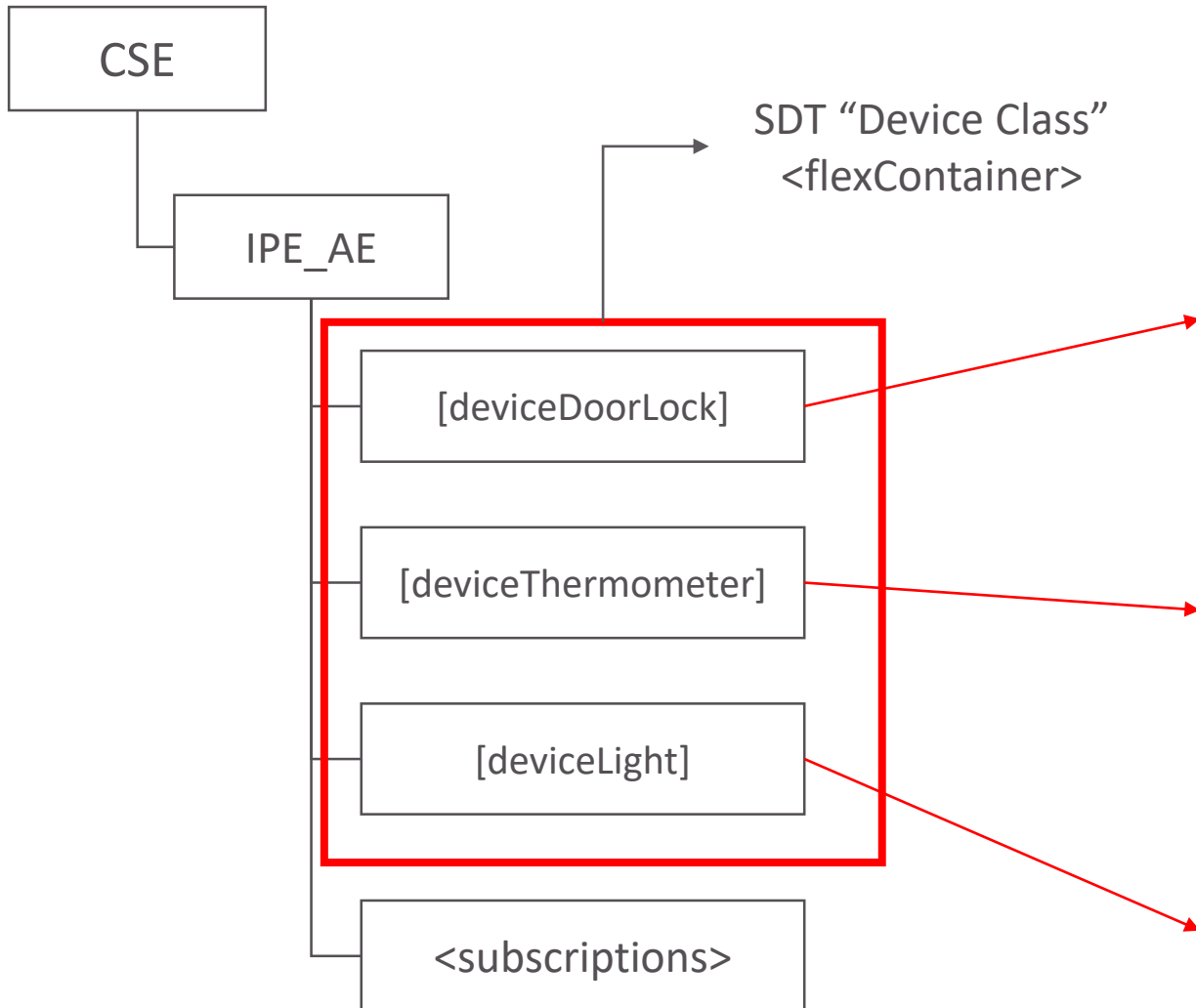


System Deployment

- TS-0023 specifies 'device class' for interworking home devices to oneM2M platform
 - Look at the process of interworking deviceDoorLock, deviceThermometer, and deviceLight to the oneM2M platform.
 - To interworking Zigbee device, deCONZ is used as a gateway software
 - Therefore, only devices provided by deCONZ can be used.



System Deployment



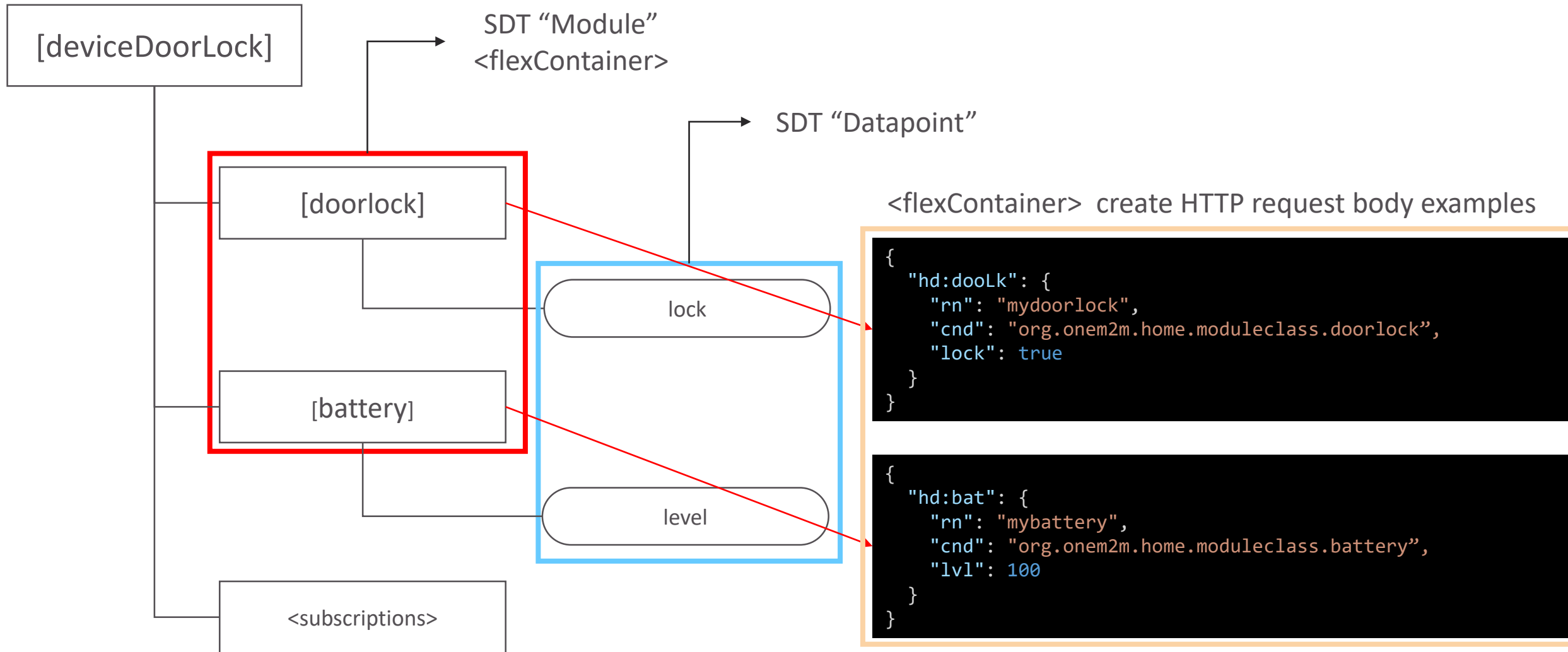
<flexContainer> create HTTP request body examples

```
{
  "m2m:fcnt": {
    "rn": "DoorLock",
    "cnd": "org.onem2m.home.device.deviceDoorLock"
  }
}
```

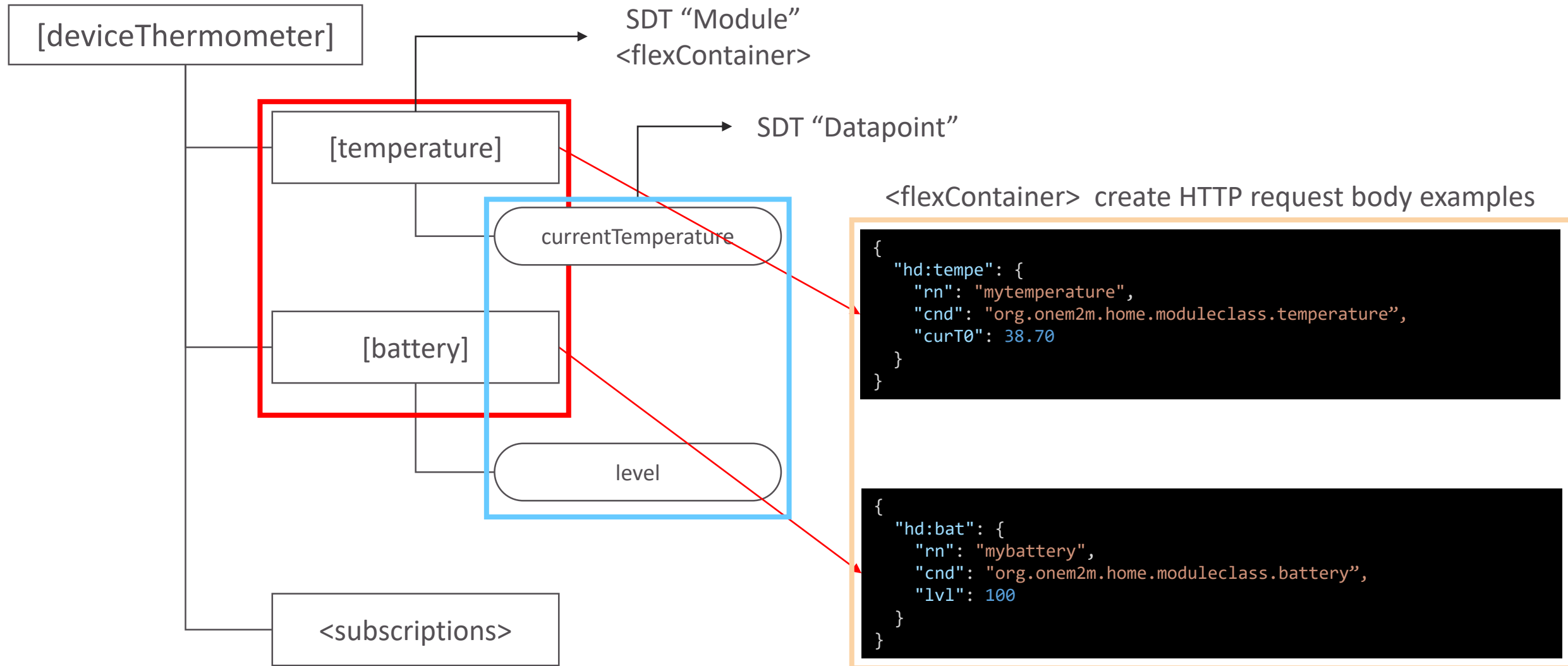
```
{
  "m2m:fcnt": {
    "rn": "Thermometer",
    "cnd": "org.onem2m.home.device.deviceThermometer"
  }
}
```

```
{
  "m2m:fcnt": {
    "rn": "Light",
    "cnd": "org.onem2m.home.device.deviceLight"
  }
}
```

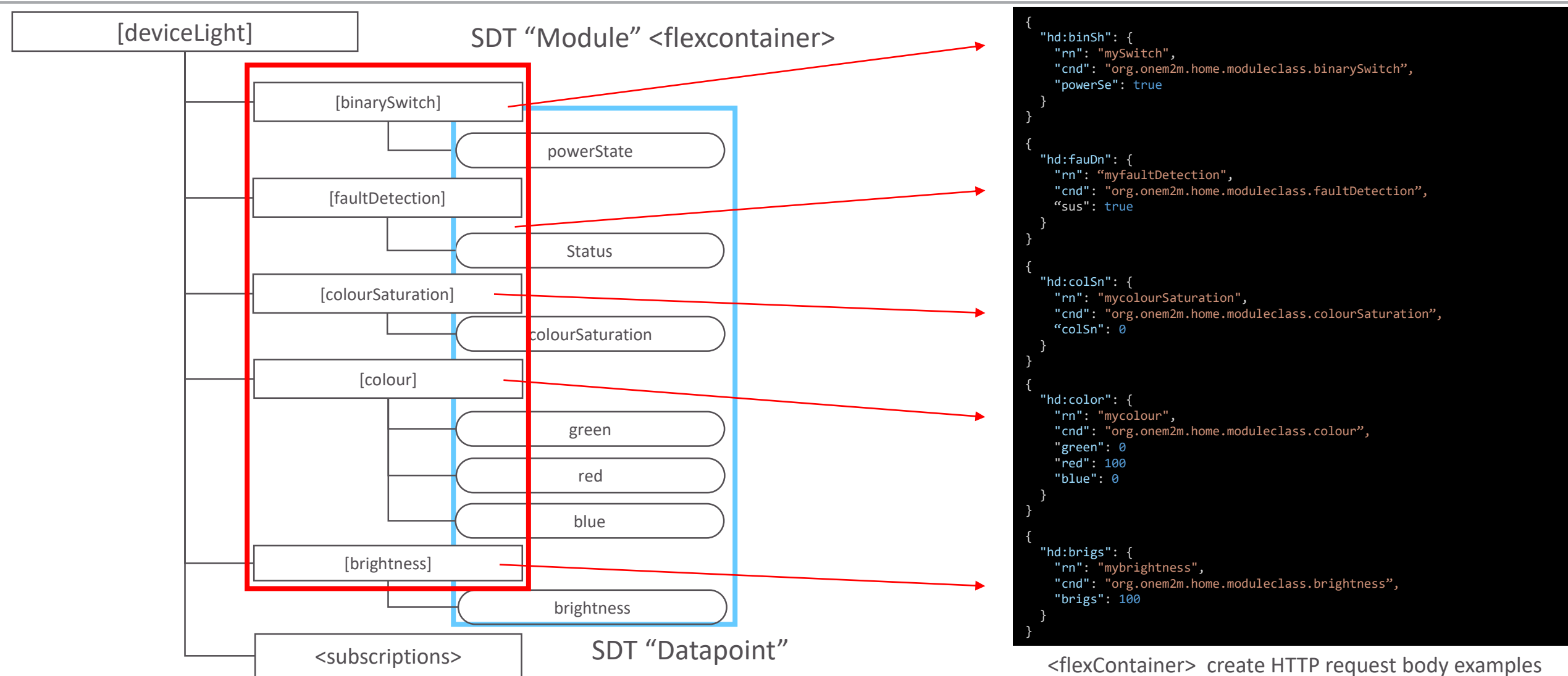
System Deployment



System Deployment

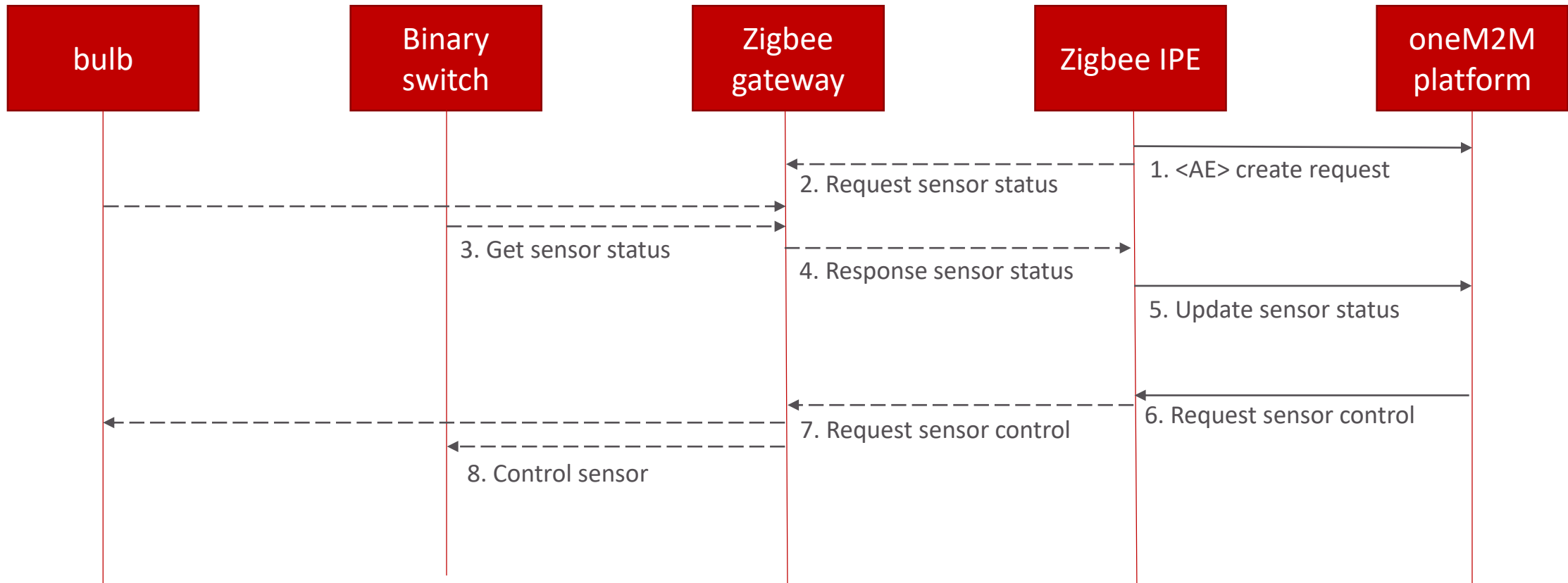


System Deployment



Interworking scenarios

- Simple scenario

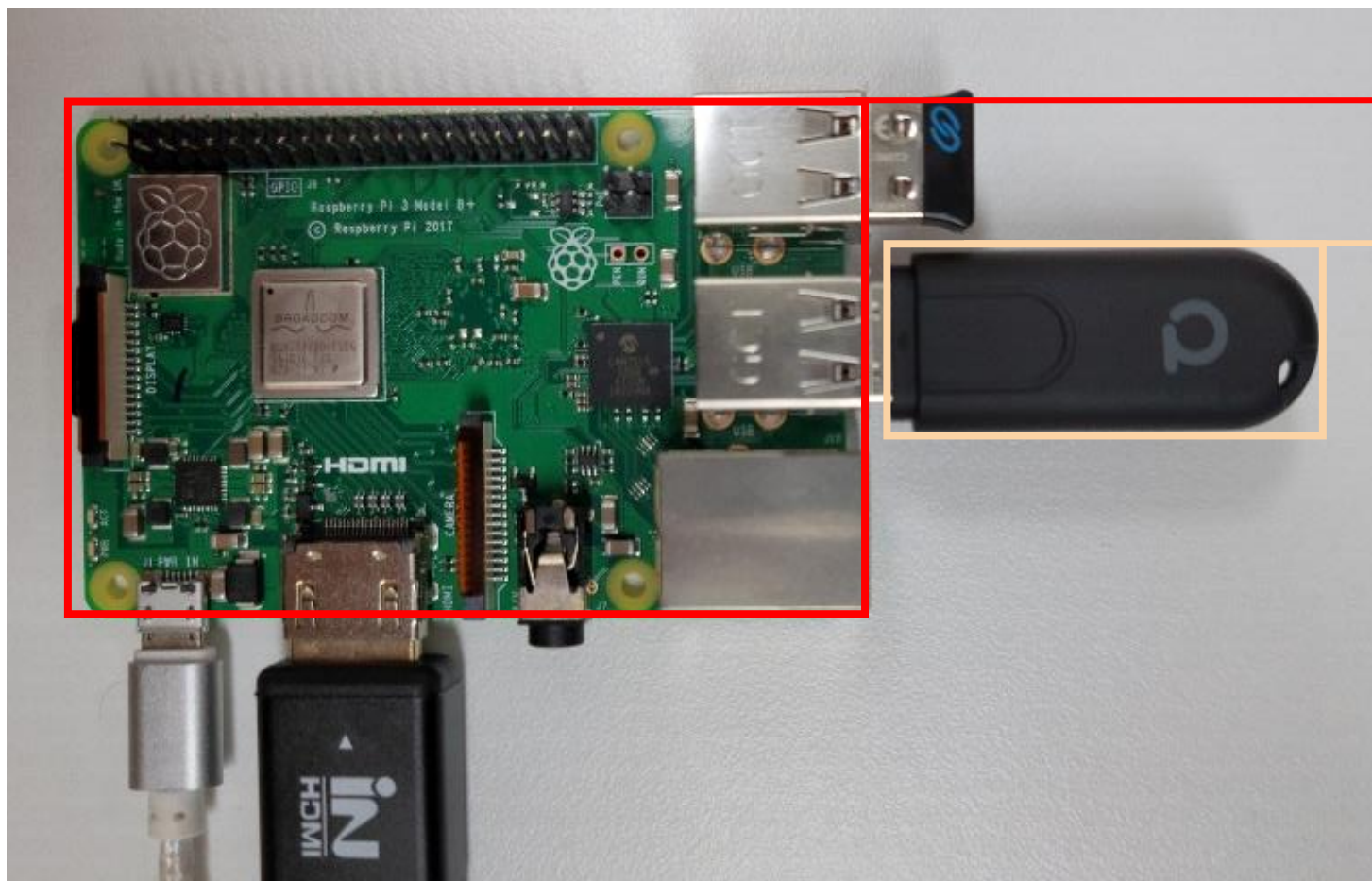




Demo Implementation

Zigbee gateway installation

- Gateway configuration



Raspberry pi 3B+

Conbee 2

- This is gateway module for Zigbee
- USB type module

Zigbee gateway installation

- Set user USB access rights
 - Process for adding {\$USER} account to dialout group
 - > sudo gpasswd -a \$USER dialout

```
pi@raspberrypi:~ $ sudo gpasswd -a $USER dialout
Adding user pi to group dialout
```

- Import Phoscon public key
 - Add a key to use when authenticating the Phoscon package
 - > wget -O - http://phoscon.de/apt/deconz.pub.key | sudo apt-key add -

```
pi@raspberrypi:~ $ wget -O - http://phoscon.de/apt/deconz.pub.key | sudo apt-key add -
--2020-09-15 21:28:11-- http://phoscon.de/apt/deconz.pub.key
Resolving phoscon.de (phoscon.de)... 144.76.96.194
Connecting to phoscon.de (phoscon.de)|144.76.96.194|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1692 (1.7K) [application/octet-stream]
Saving to: 'STDOUT'

-                               100%[=====] 1.65K --.-KB/s in 0s

2020-09-15 21:28:12 (42.2 MB/s) - written to stdout [1692/1692]

OK
```

Zigbee gateway installation

- Configure the APT repository for deCONZ
 - Configure apt repository to install deCONZ
 - > sudo sh -c "echo 'deb http://phoscon.de/apt/deconz \\$(lsb_release -cs) main' > \ /etc/apt/sources.list.d/deconz.list"

```
pi@raspberrypi:~ $ sudo sh -c "echo 'deb http://phoscon.de/apt/deconz $(lsb_release -cs) main' > /etc/apt/sources.list.d/deconz.list"
```

- Update APT package list
 - > sudo apt update

```
pi@raspberrypi:~ $ sudo apt update
Hit:1 http://archive.raspberrypi.org/debian buster InRelease
Hit:2 http://raspbian.raspberrypi.org/raspbian buster InRelease
Get:3 http://phoscon.de/apt/deconz buster InRelease [2,857 B]
Get:4 http://phoscon.de/apt/deconz buster/main armhf Packages [719 B]
Fetched 3,576 B in 3s (1,179 B/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
```

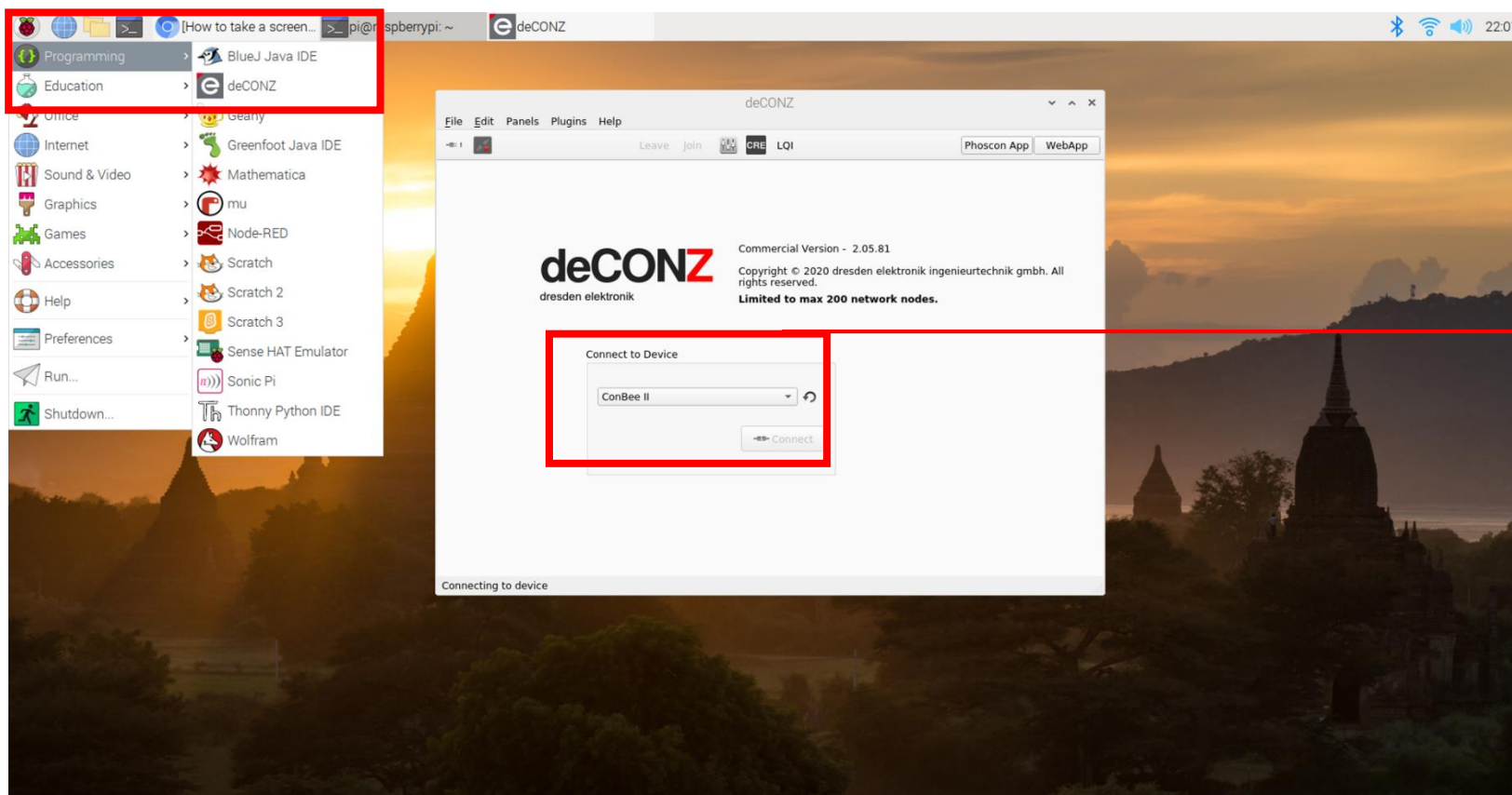
Zigbee gateway installation

- Install deCONZ
 - > sudo apt install deconz

```
pi@raspberrypi:~ $ sudo apt install deconz
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 libssl1.0.2 rpi-eeeprom-images uuid-dev
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libqt5websockets5 sqlite3
Suggested packages:
  sqlite3-doc
The following NEW packages will be installed:
  deconz libqt5websockets5 sqlite3
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 6,782 kB of archives.
After this operation, 36.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```


Zigbee gateway installation

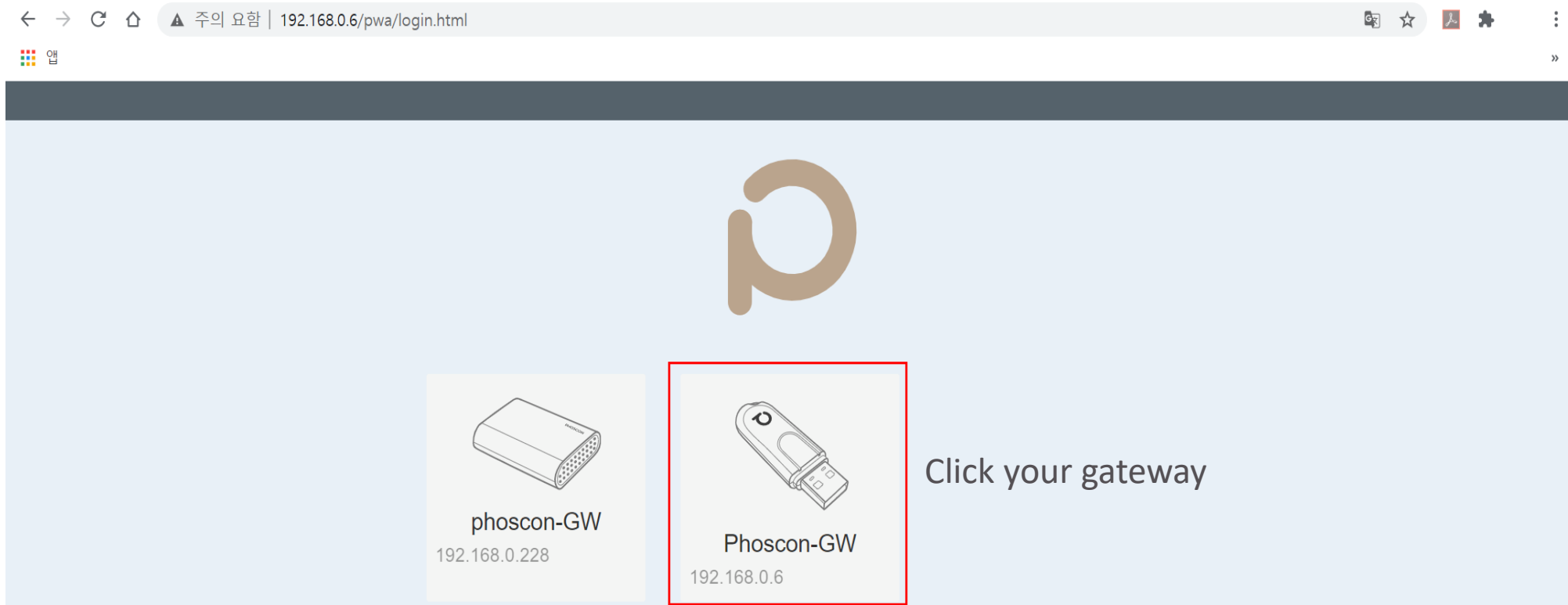
- After the installation deCONZ can be started via the application menu.
 - Menu > Programming > deCONZ



After a few seconds,
Connect button gets
activated

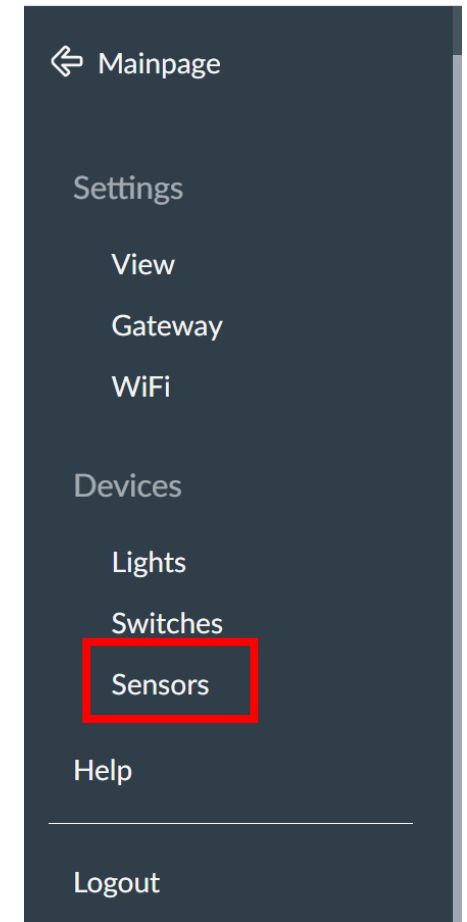
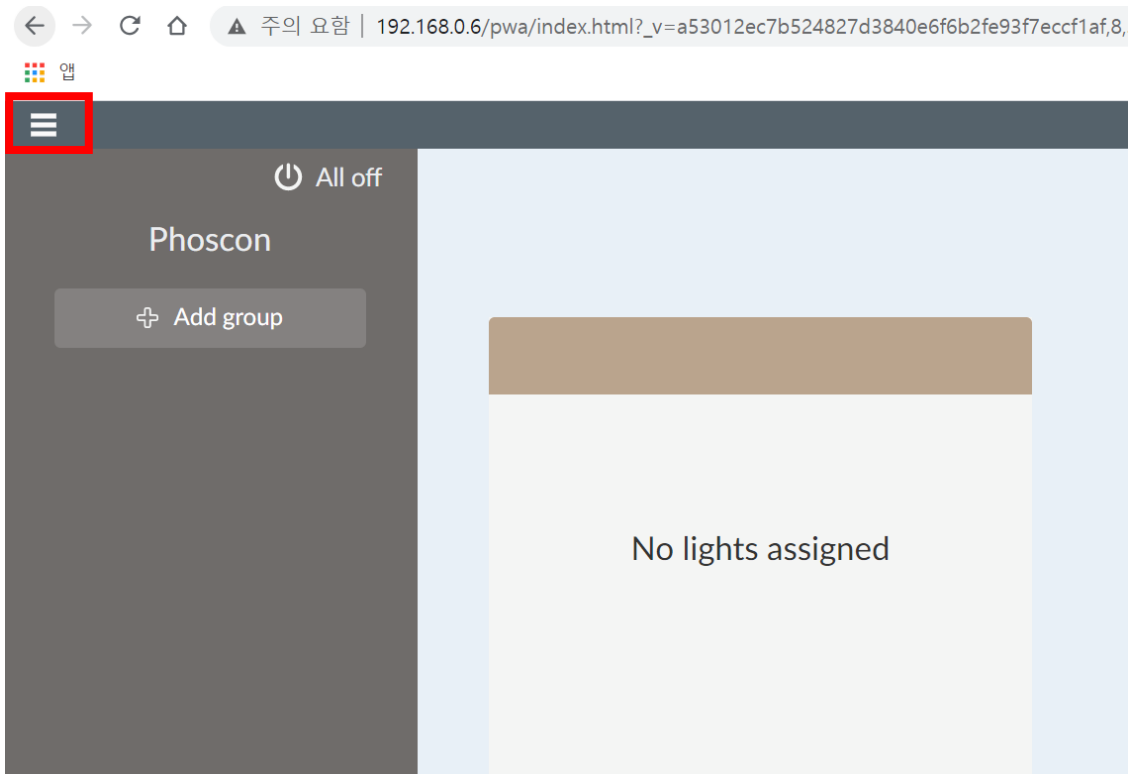
Register Zigbee sensor

- If you access to gateway address, you can check the connected gateway list
 - [http://\[your gateway IP\]/pwa/login.html](http://[your gateway IP]/pwa/login.html)



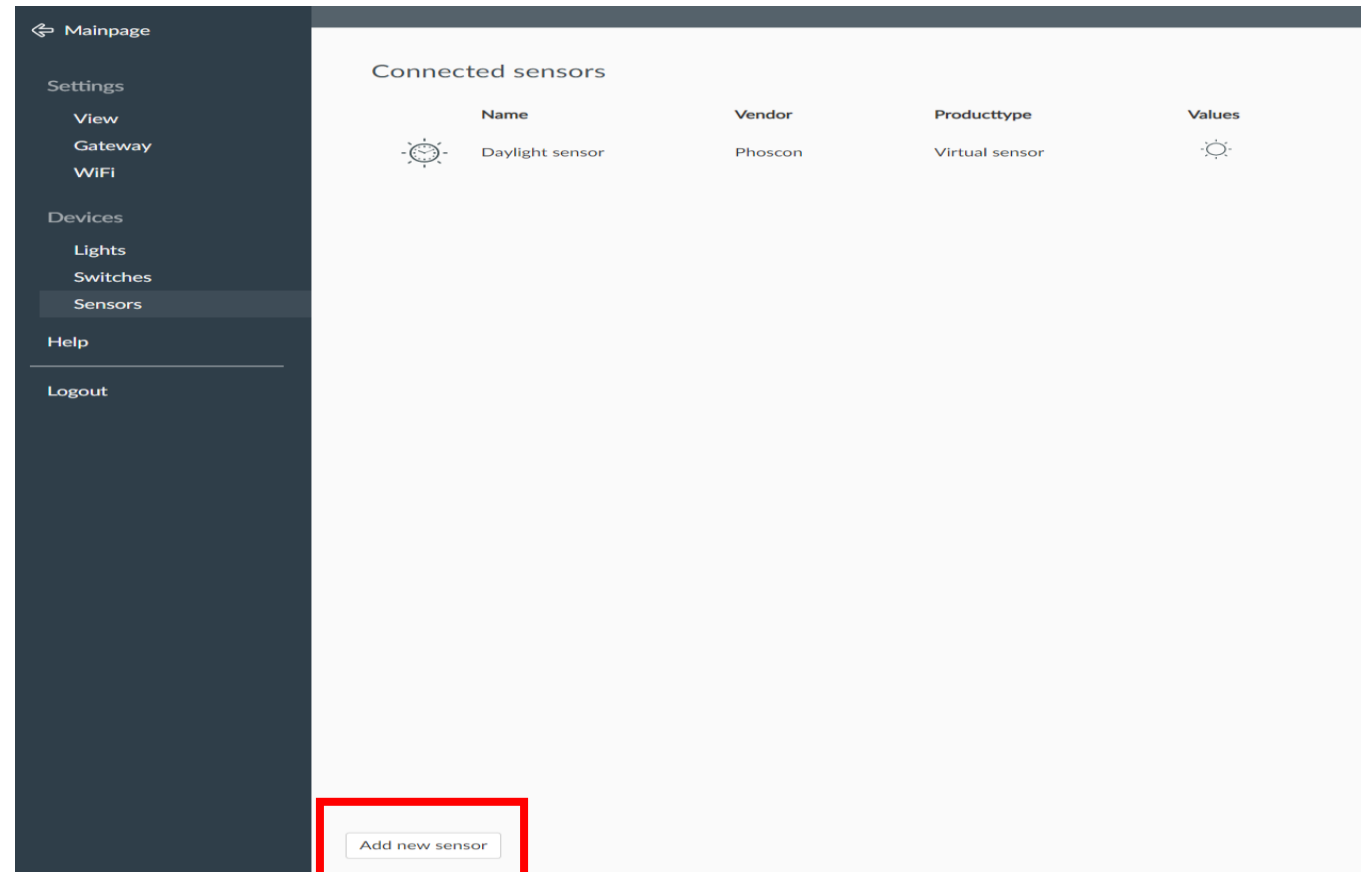
Register Zigbee sensor

- Click the `≡` button on the top of left, then click sensors in the Devices tab



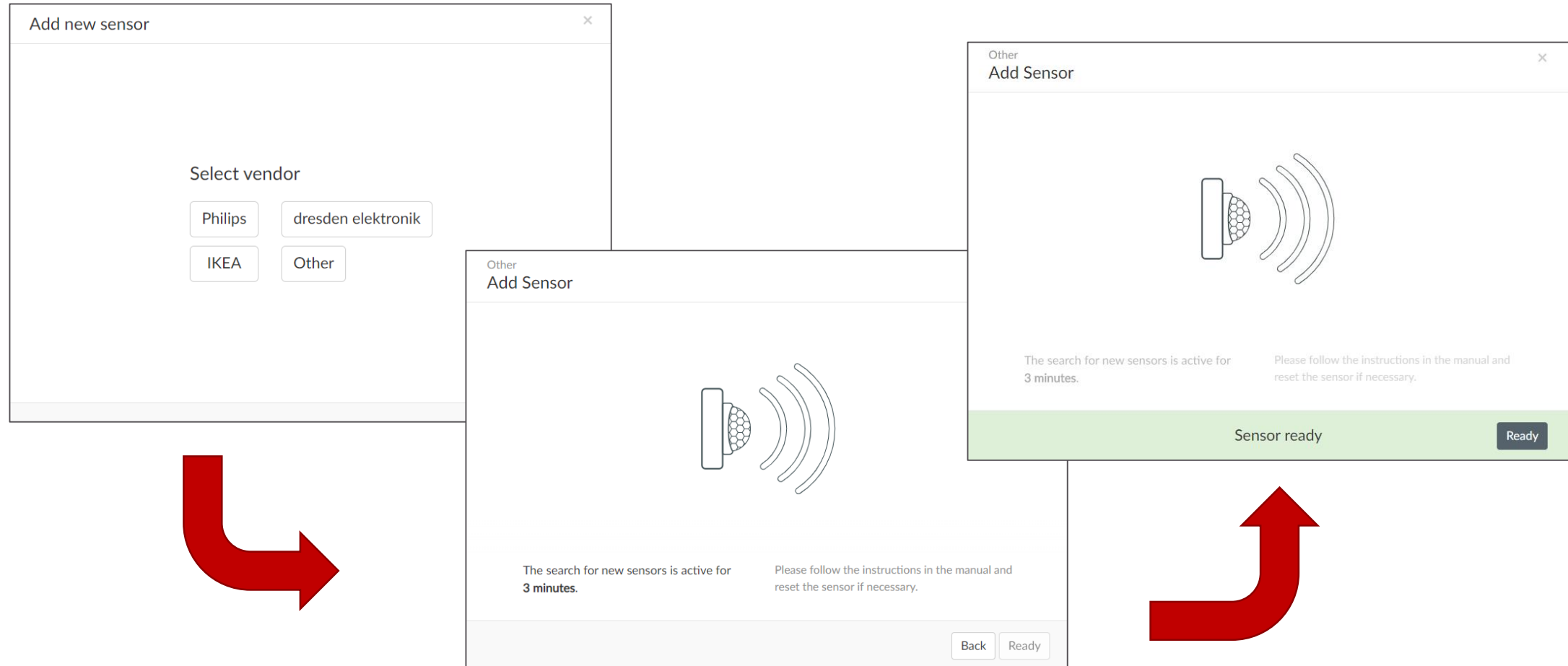
Register Zigbee sensors

- Click the `Add new sensor` button to connect the sensor



Register Zigbee sensors

- After checking the type of sensor, press the reset button of the sensor



Register Zigbee sensors

- You can show the connected sensor

[Mainpage](#)




[Settings](#)
[View](#)
[Gateway](#)
[WiFi](#)

[Devices](#)
[Lights](#)
[Switches](#)
[Sensors](#)

[Help](#)

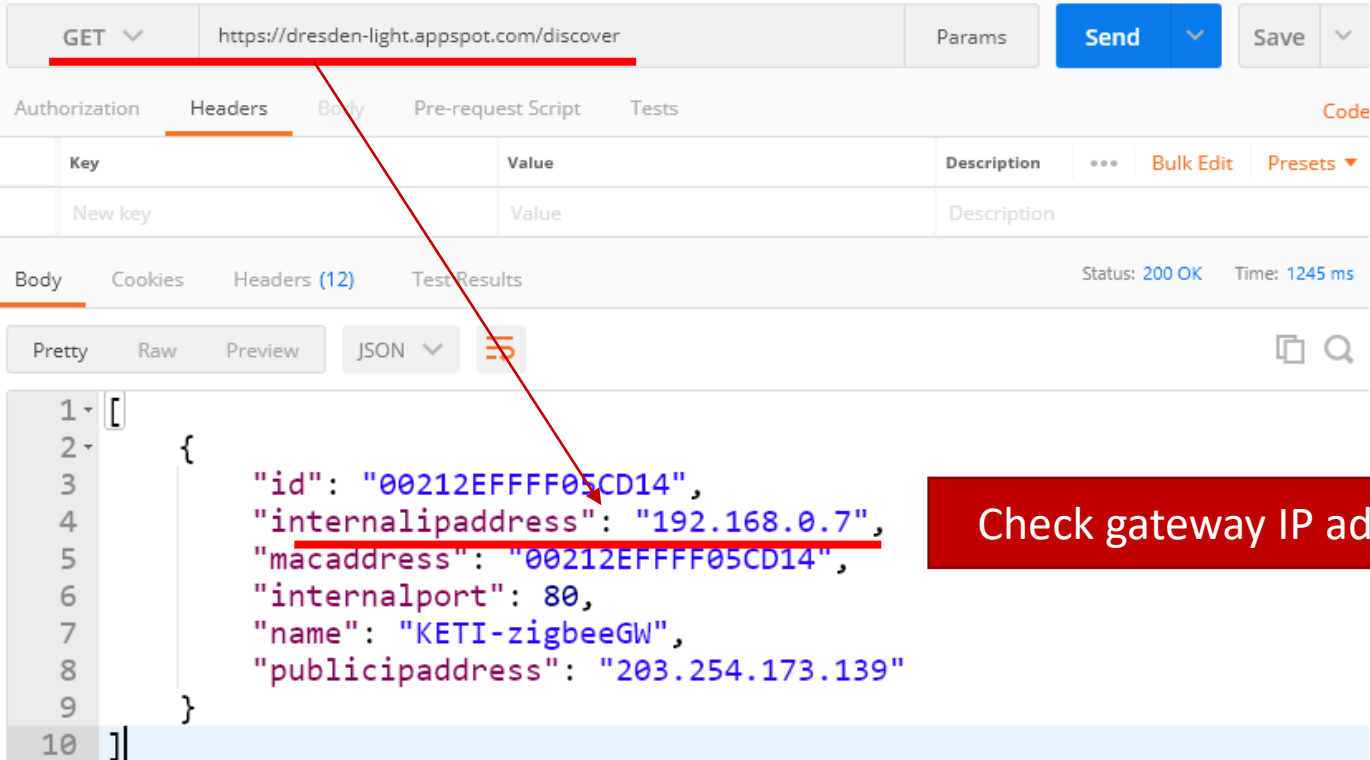
[Logout](#)

Connected sensors

	Name	Vendor	Producttype	Values
	Daylight sensor	Phoscon	Virtual sensor	
	Multi Sensor	Xiaomi	Multi sensor	29 °C, 61.7 %, 994 hPa

Get gateway API key

- deCONZ REST API is used for this Zigbee IPE to correspond data/control
 - From the G/W API, you can see all Zigbee gateways in your network
 - GET <https://dresden-light.appspot.com/discover>



GET <https://dresden-light.appspot.com/discover> Params Send Save

Authorization Headers Body Pre-request Script Tests Code

Key	Value	Description
New key	Value	Description

Body Cookies Headers (12) Test Results Status: 200 OK Time: 1245 ms

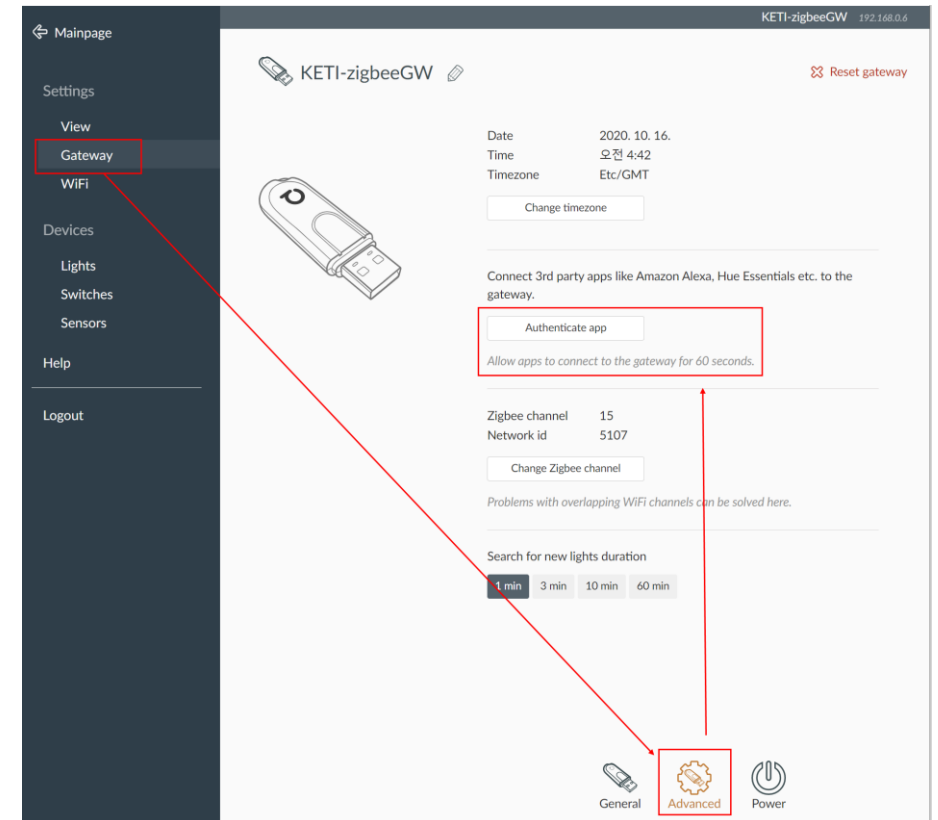
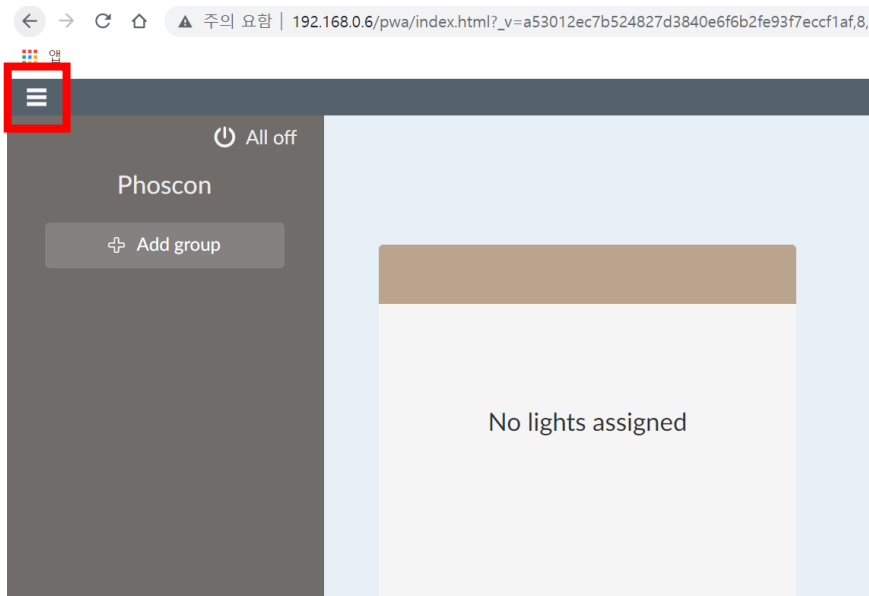
Pretty Raw Preview JSON

```
1 [
2   {
3     "id": "00212EFFFF05CD14",
4     "internalipaddress": "192.168.0.7",
5     "macaddress": "00212EFFFF05CD14",
6     "internalport": 80,
7     "name": "KETI-zigbeeGW",
8     "publicipaddress": "203.254.173.139"
9   }
10 ]
```

Check gateway IP address

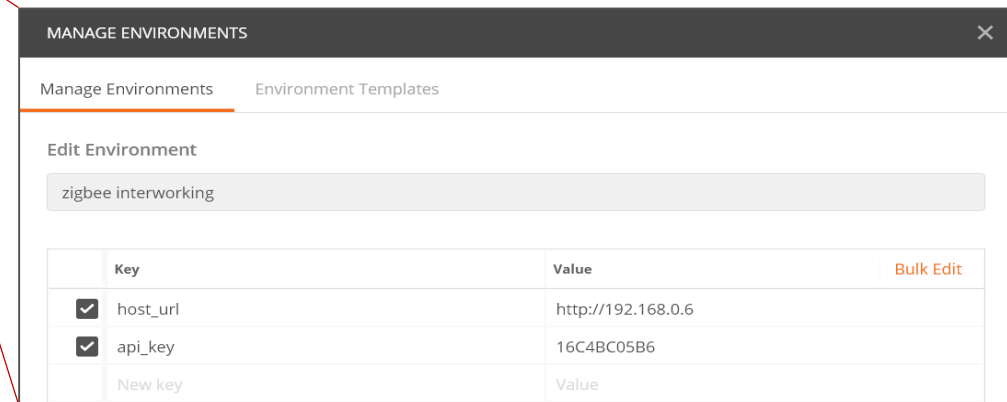
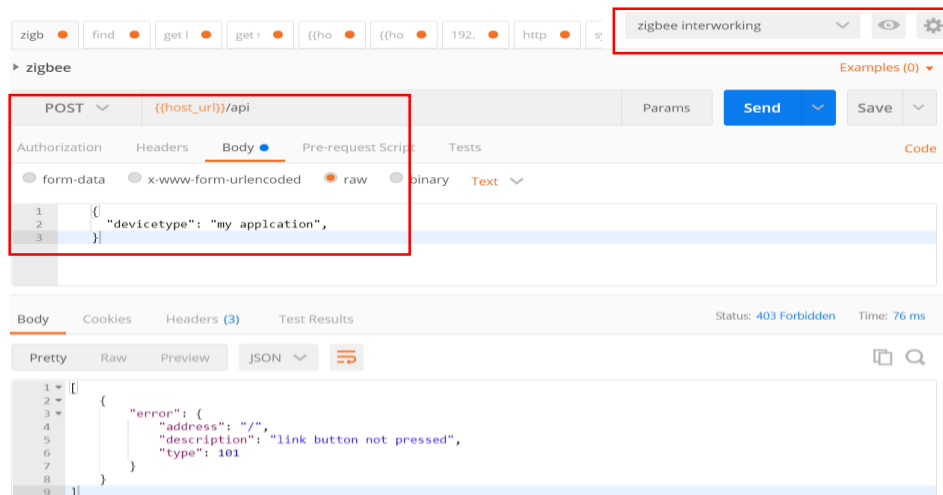
Get gateway API key

- Click the `≡` button
 - When push the button, the gateway unlock lasts 60 seconds to allow 3rd party app to register themselves to the gateway



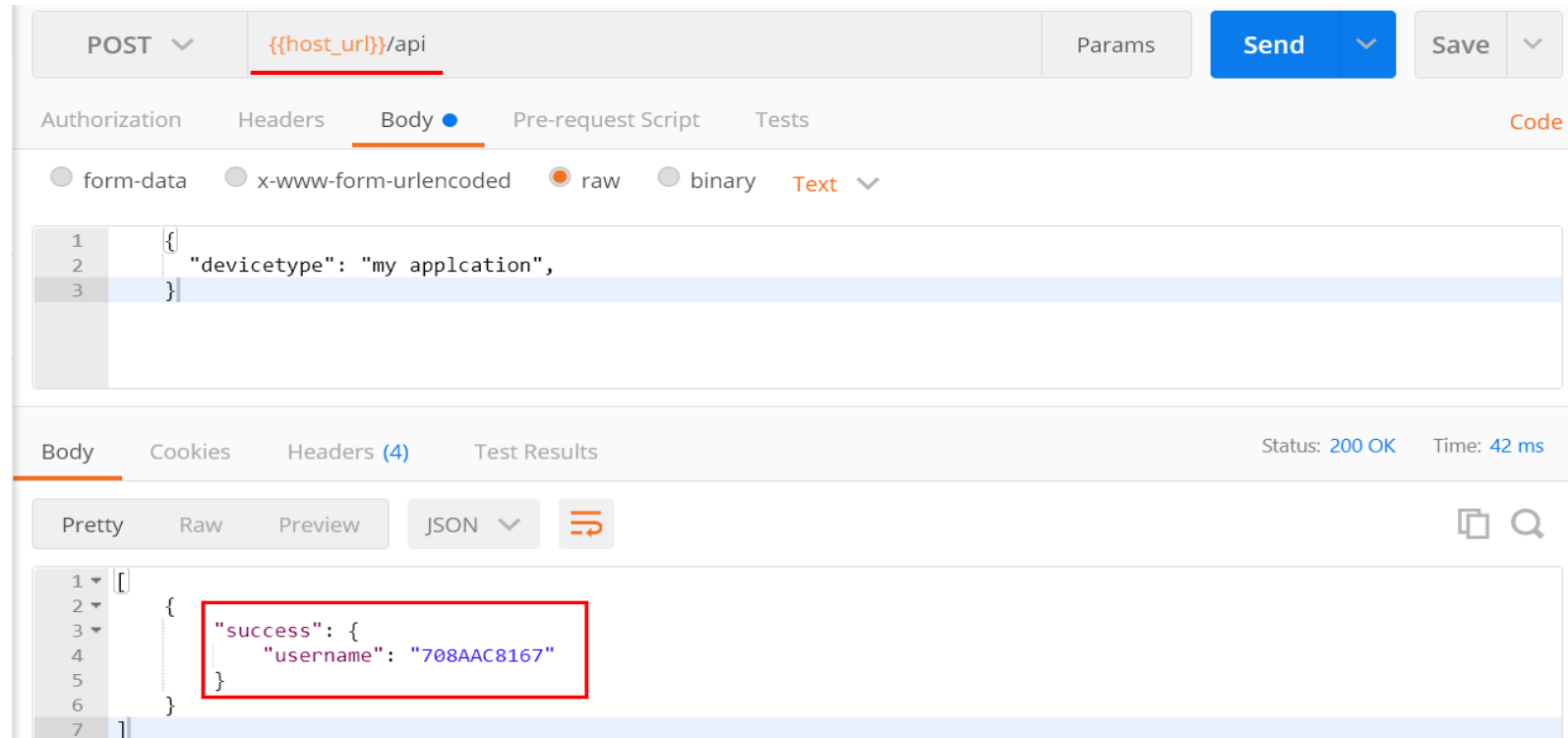
Get gateway API key

- To control the gateway, you need to get the gateway API key
 - POST http://[your gateway IP]/api
 - Body
 - {
 “devicetype”: “my application”,
 }



Get gateway API key

- You can get the gateway API key



Get sensor lists

- A numeric sensor ID is generated in the order of sensor registrations
 - GET `http://{Gateway_host}/api/{api_key}/sensors`

```
GET {{host_url}}/api/{{api_key}}/sensors
Send Save

Type: No Auth

Body: Cookies Headers (4) Test Results Status: 200 OK Time: 147 ms
Pretty Raw Preview JSON Save Response

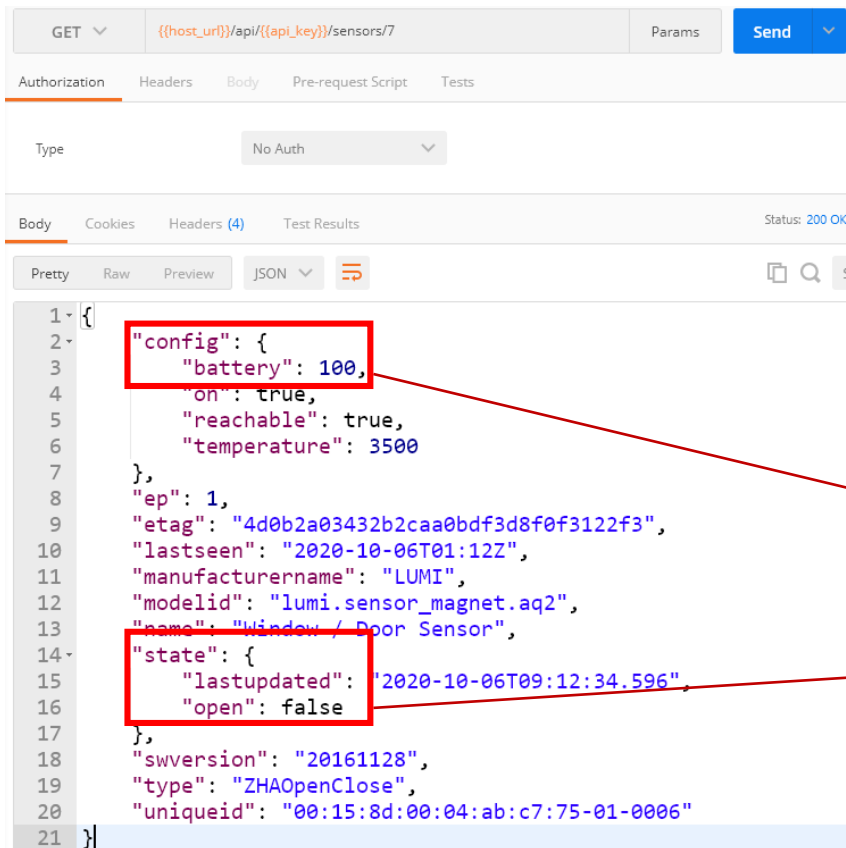
1- {
2-   "2": {
3-     "config": {
4-       "battery": 100,
5-       "offset": 0,
6-       "on": true,
7-       "reachable": true
8-     },
9-     "ep": 1,
10-    "etag": "4d0b2a03432b2caa0bdf3d8f0f3122f3",
11-    "lastseen": "2020-10-06T01:12Z",
12-    "manufacturername": "LUMI",
13-    "modelid": "lumi.weather",
14-    "name": "Multi Sensor",
15-    "state": {
16-      "lastupdated": "2020-10-06T09:52:42.268",
17-      "temperature": 2877
18-    },
19-    "swversion": "20191205",
20-    "type": "ZHATemperature",
21-    "uniqueid": "00:15:8d:00:04:aa:c2:67-01-0402"
22-  },
23-   "3": {
24-     "config": {
25-       "battery": 100,
26-       "offset": 0,
27-       "on": true,
28-       "reachable": true
29-     },
30-     "ep": 1,
31-     "etag": "4d0b2a03432b2caa0bdf3d8f0f3122f3",
32-     "lastseen": "2020-10-06T01:12Z",
```

Sensor ID

Sensor type and value

Get sensor status

- You can check the sensor (e.g. deviceDoorLock) status
 - GET `http://{Gateway_host}/api/{api_key}/sensors/{sensorID}`



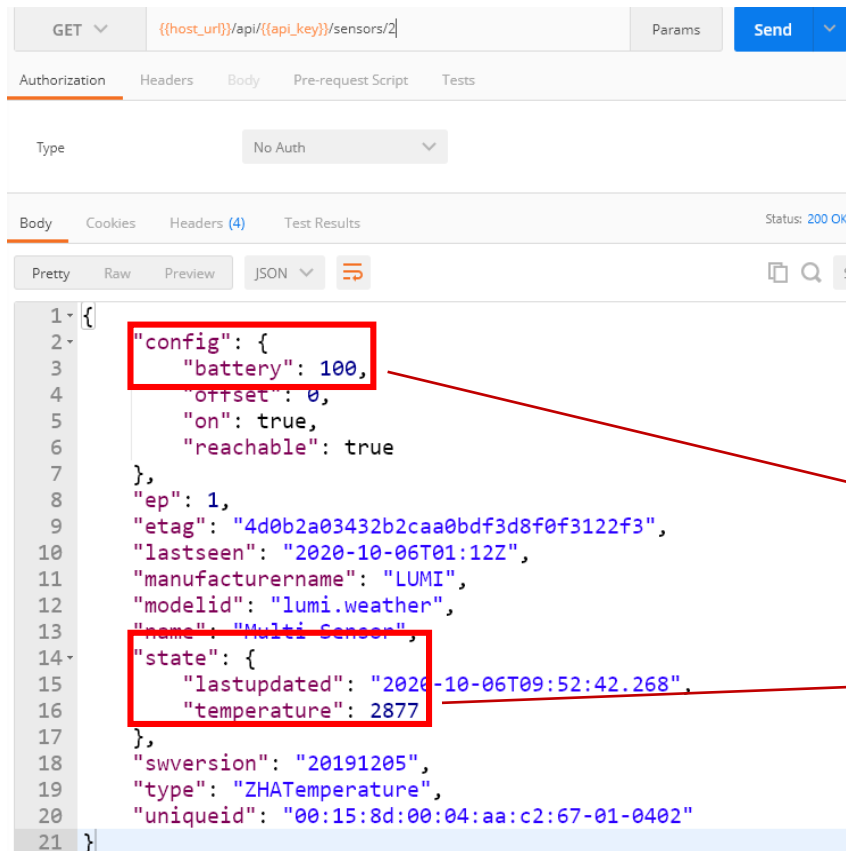
```
1 {
2   "config": {
3     "battery": 100,
4     "on": true,
5     "reachable": true,
6     "temperature": 3500
7   },
8   "ep": 1,
9   "etag": "4d0b2a03432b2caa0bdf3d8f0f3122f3",
10  "lastseen": "2020-10-06T01:12Z",
11  "manufacturername": "LUMI",
12  "modelid": "lumi.sensor_magnet.aq2",
13  "name": "Window / Door Sensor",
14  "state": {
15    "lastupdated": "2020-10-06T09:12:34.596",
16    "open": false
17  },
18  "swversion": "20161128",
19  "type": "ZHAOpenClose",
20  "uniqueid": "00:15:8d:00:04:ab:c7:75-01-0006"
21 }
```

Data point mapping

Property of Zigbee sensor data	Attribute name of the SDT module class
config.battery	level of [battery]
state.open	lock of [doorlock]

Get sensor status

- You can check the sensor (e.g. deviceThermometer) status
 - GET `http://{Gateway_host}/api/{api_key}/sensors/{sensorID}`



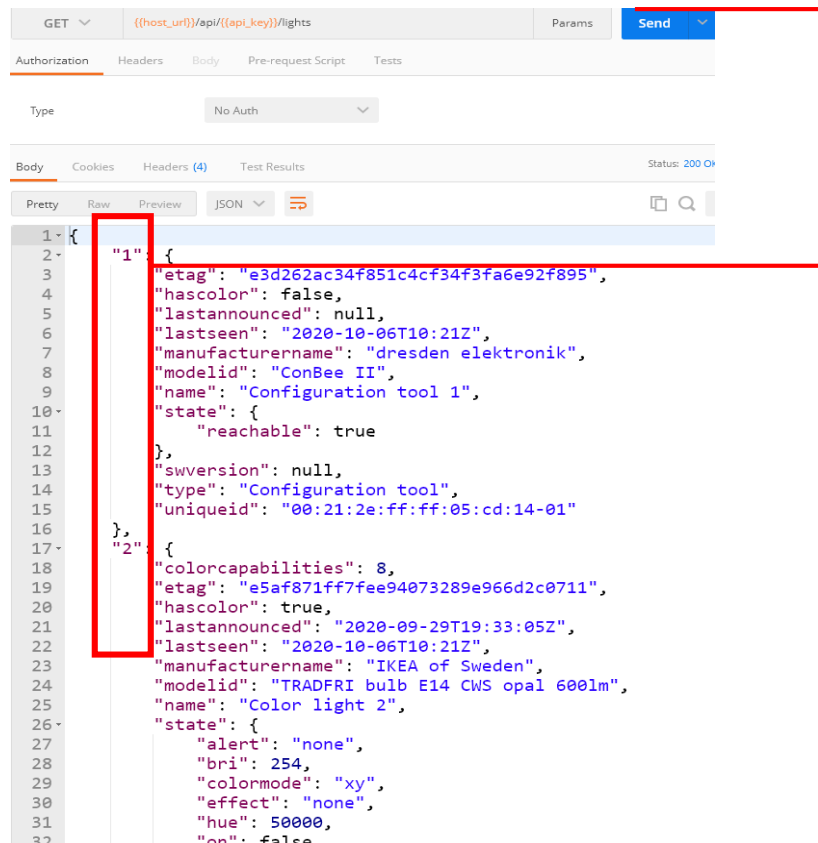
```
1- {
2-   "config": {
3-     "battery": 100,
4-     "offset": 0,
5-     "on": true,
6-     "reachable": true
7-   },
8-   "ep": 1,
9-   "etag": "4d0b2a03432b2caa0bdf3d8f0f3122f3",
10-  "lastseen": "2020-10-06T01:12Z",
11-  "manufacturername": "LUMI",
12-  "modelid": "lumi.weather",
13-  "name": "Multi Sensor",
14-  "state": {
15-    "lastupdated": "2020-10-06T09:52:42.268",
16-    "temperature": 2877
17-  },
18-  "swversion": "20191205",
19-  "type": "ZHATemperature",
20-  "uniqueid": "00:15:8d:00:04:aa:c2:67-01-0402"
21- }
```

Data point mapping

Property of Zigbee sensor data	Attribute name of the SDT module class
config.battery	level of [battery]
state.temperature	currentTemperature of [temperature]

Get lights list

- You can check the lights list
 - GET `http://{Gateway_host}/api/{api_key}/lights`



```
GET {{host_url}}/api/{{api_key}}/lights
Send

Authorization: No Auth

Body
Cookies
Headers (4)
Test Results
Status: 200 OK

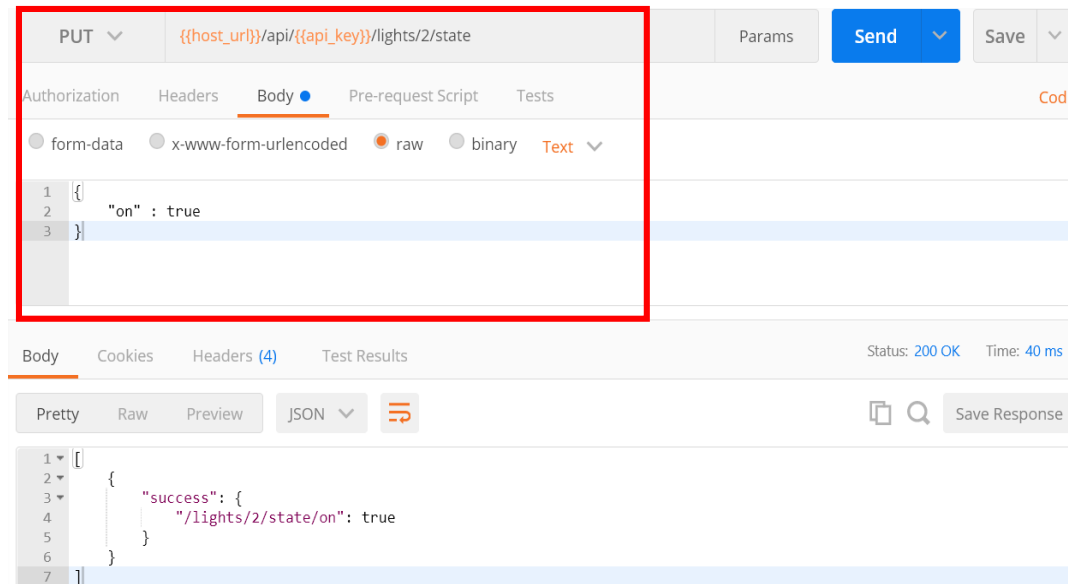
Pretty
Raw
Preview
JSON
1 {
2   "1": {
3     "etag": "e3d262ac34f851c4cf34f3fa6e92f895",
4     "hascolor": false,
5     "lastannounced": null,
6     "lastseen": "2020-10-06T10:21Z",
7     "manufacturername": "dresden elektronik",
8     "modelid": "ConBee II",
9     "name": "Configuration tool 1",
10    "state": {
11      "reachable": true
12    },
13    "swversion": null,
14    "type": "Configuration tool",
15    "uniqueid": "00:21:2e:ff:ff:05:cd:14-01"
16  },
17  "2": {
18    "colorcapabilities": 8,
19    "etag": "e5af871ff7fee94073289e966d2c0711",
20    "hascolor": true,
21    "lastannounced": "2020-09-29T19:33:05Z",
22    "lastseen": "2020-10-06T10:21Z",
23    "manufacturername": "IKEA of Sweden",
24    "modelid": "TRADFRI bulb E14 CWS opal 600lm",
25    "name": "Color light 2",
26    "state": {
27      "alert": "none",
28      "bri": 254,
29      "colormode": "xy",
30      "effect": "none",
31      "hue": 50000,
32      "on": false
33    }
34  }
35 }
```

Zigbee bulb ID

Light control

- You can check the lights status
 - URL : `http://{Gateway_host}/api/{api_key}/lights/{light_ID}/state`
 - Body

```
{  
  "on": true // on = true, off = false  
}
```



Get light status

- You can get the lights status
 - URL : `http://{Gateway_host}/api/{api_key}/lights/{light_ID}`

```
GET {{host_url}}/api/{{api_key}}/lights/2
Authorization
Headers
Body
Pre-request Script
Tests
Type
No Auth
Status: 200 OK
Pretty
Raw
Preview
JSON
1- {
2-   "colorcapabilities": 8,
3-   "etag": "5fb7c073e76230e16c74d2c06ef4757e",
4-   "hascolor": true,
5-   "lastannounced": "2020-09-29T19:33:05Z",
6-   "lastseen": "2020-10-06T10:24Z",
7-   "manufacturername": "IKEA of Sweden",
8-   "modelid": "TRADFRI bulb E14 CWS opal 600lm",
9-   "name": "Color light 2",
10-  "state": {
11-    "alert": "none",
12-    "bri": 254,
13-    "colormode": "xy",
14-    "effect": "none",
15-    "hue": 50000,
16-    "on": false,
17-    "reachable": true,
18-    "sat": 254,
19-    "xy": [
20-      0.458,
21-      0.41
22-    ]
23-  },
24-  "swversion": "1.3.002",
25-  "type": "Color light",
26-  "uniqueid": "84:2e:14:ff:fe:17:af:e3-01"
27- }
```

Data point mapping

Property of Zigbee sensor data	Attribute name of the SDT module class
status.bri	brightness of [brightness]
status.on	powerState of [binarySwitch]
status.reachable	status of [faultDetection]
status.sat	colourSaturation of [colourSaturation]
status.xy	green, red, blue of [colour]

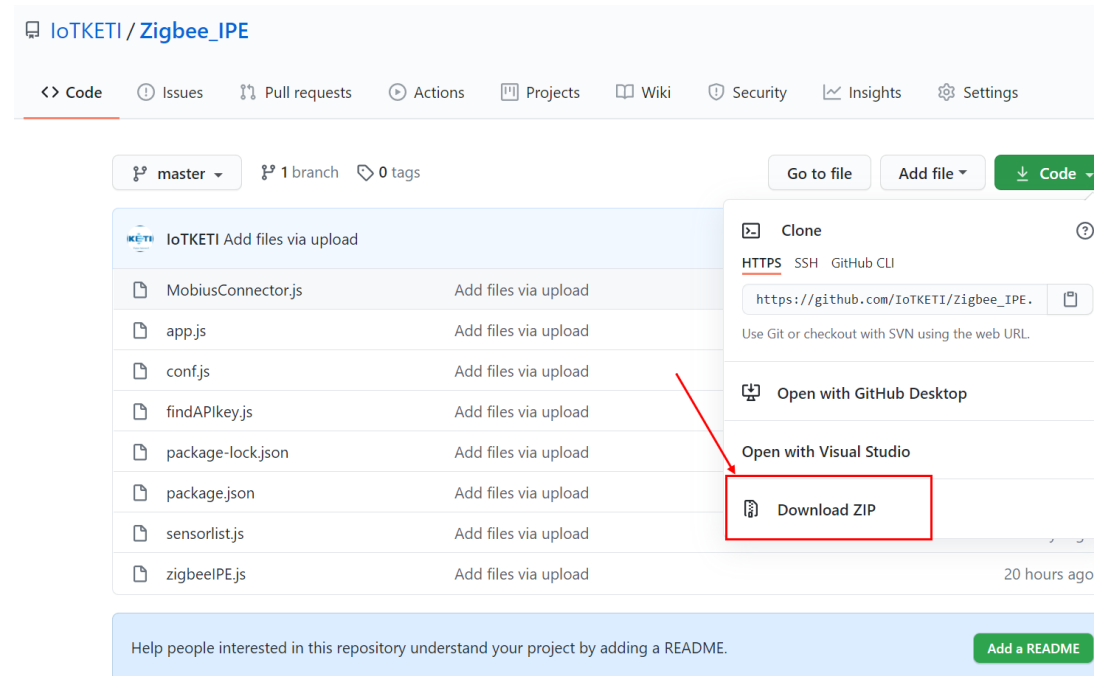
IPE working procedure



1. Download Zigbee IPE source code
2. Modify IPE configuration
 - oneM2M platform (e.g. Mobius) and Zigbee G/W
3. Apply the sensor IDs to the IPE source code
4. npm module installation and run IPE

Running Zigbee IPE

- Download Zigbee IPE source code
 - You can Download Zip
 - Alternatively, you can use the git clone command
 - > git clone https://github.com/IoTKETI/Zigbee_IPE.git



Running Zigbee IPE



- Modify IPE configuration
- In conf.js file
 - Modify the configuration per your oneM2M platform
 - Apply api_key to zigbee.api_key

```
JS conf.js > ...
1  let conf = {};
2  let cse = {};
3  let ae = {};
4  let zigbee = {};
5  let api_key = {};
6
7  //cse config
8  cse.host = "127.0.0.1";
9  cse.port = "7579"
10 cse.name = "Mobius"
11 cse.id = "/Mobius2"
12 cse.mqttport = "1883";
13
14 //ae config
15 ae.name = "zigbee_smarthome";
16 ae.id = "S" + ae.name;
17 ae.parent = "/" + cse.name;
18 ae.appid = "zigbee"
19
20 zigbee.host = "192.168.0.6"
21 zigbee.api_key = "708AAC8167"
22
23 conf.cse = cse;
24 conf.ae = ae;
25 conf.zigbee = zigbee;
26 conf.api_key = api_key;
27
28 module.exports = conf;
```

< conf.js >

Running Zigbee IPE

- Check sensor list
 - > node sensorlist.js

```
C:\Users\sejong\Documents\Visual Studio 2019\ZigbeeIPE (zigbee_ipe@1.0.0)
λ node sensorlist.js
{
  '2': {
    config: { battery: 100, offset: 0, on: true, reachable: true },
    ep: 1,
    etag: 'dd70a98a41182627f6d959ec73d10334',
    lastseen: '2020-09-25T09:14Z',
    manufacturername: 'LUMI',
    modelid: 'lumi.weather',
    name: 'Multi Sensor',
    state: { lastupdated: '2020-09-25T09:14:25.043', temperature: 2805 },
    swversion: '20191205',
    type: 'ZHATemperature',
    uniqueid: '00:15:8d:00:04:aa:c2:67-01-0402'
  },
  '3': {
    config: { battery: 100, offset: 0, on: true, reachable: true },
    ep: 1,
    etag: 'dd70a98a41182627f6d959ec73d10334',
    lastseen: '2020-09-25T09:14Z',
    manufacturername: 'LUMI',
    modelid: 'lumi.weather',
    name: 'Multi Sensor',
    state: { humidity: 4163, lastupdated: '2020-09-25T09:14:25.047' },
    swversion: '20191205',
    type: 'ZHAHumidity',
    uniqueid: '00:15:8d:00:04:aa:c2:67-01-0405'
  },
  '4': {
    config: { battery: 100, on: true, reachable: true },
    ep: 1,
    etag: 'dd70a98a41182627f6d959ec73d10334',
    lastseen: '2020-09-25T09:14Z',
    manufacturername: 'LUMI',
    modelid: 'lumi.weather',
    name: 'Multi Sensor',
    state: { lastupdated: '2020-09-25T09:14:25.051', pressure: 999 },
    swversion: '20191205',
    type: 'ZHAPressure',
    uniqueid: '00:15:8d:00:04:aa:c2:67-01-0403'
  },
}
```

Running Zigbee IPE

- Apply my sensor list to the code
 - In the app.js and zigbeeIPE.js

```
function typetonum(sen){  
  switch (sen){  
    case 'temperature':  
      return 2;  
    case 'humidity':  
      return 3;  
    case 'pressure':  
      return 4;  
    case 'lux':  
      return 5;  
    case 'presence':  
      return 6;  
    case 'open':  
      return 7;  
    case 'buttonevent':  
      return 8;  
  }  
}
```

< app.js >

```
function numtotype(sensorty) {  
  switch (sensorty) {  
    case 2:  
      return 'temperature';  
    case 3:  
      return 'humidity';  
    case 4:  
      return 'pressure';  
    case 5:  
      return 'lux';  
    case 6:  
      return 'presence';  
    case 7:  
      return 'open';  
    case 8:  
      return 'buttonevent';  
  }  
}
```

< zigbeeIPE.js >

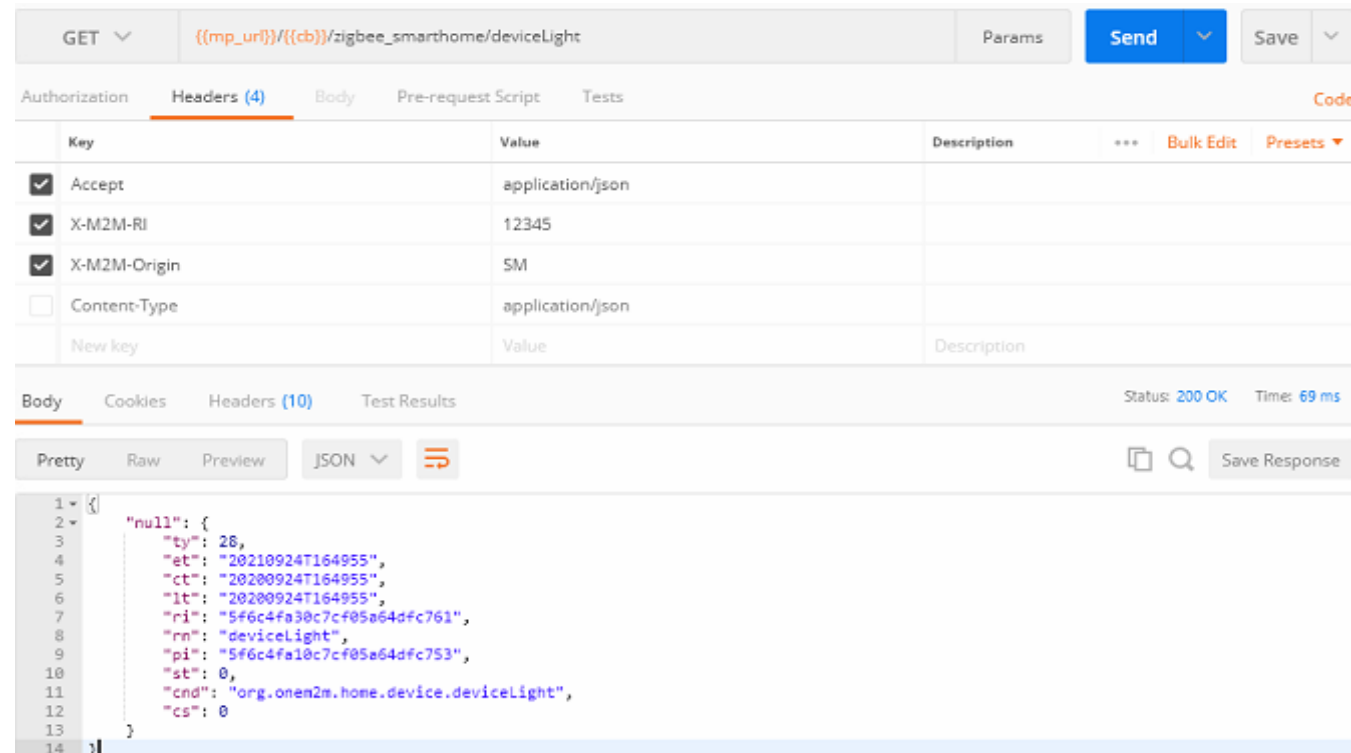
Running Zigbee IPE



- Clone or copy the IPE repository to local machine
- Move to the folder and then install npm modules
 - > npm install
- Run IPE source code
 - > node app.js
 - If device resources are already created in the oneM2M platform, the IPE uploads event data (i.e. flexContainer update) to the oneM2M platform
 - If some resources are deleted, the IPE creates new flexContainer resources in the oneM2M platform

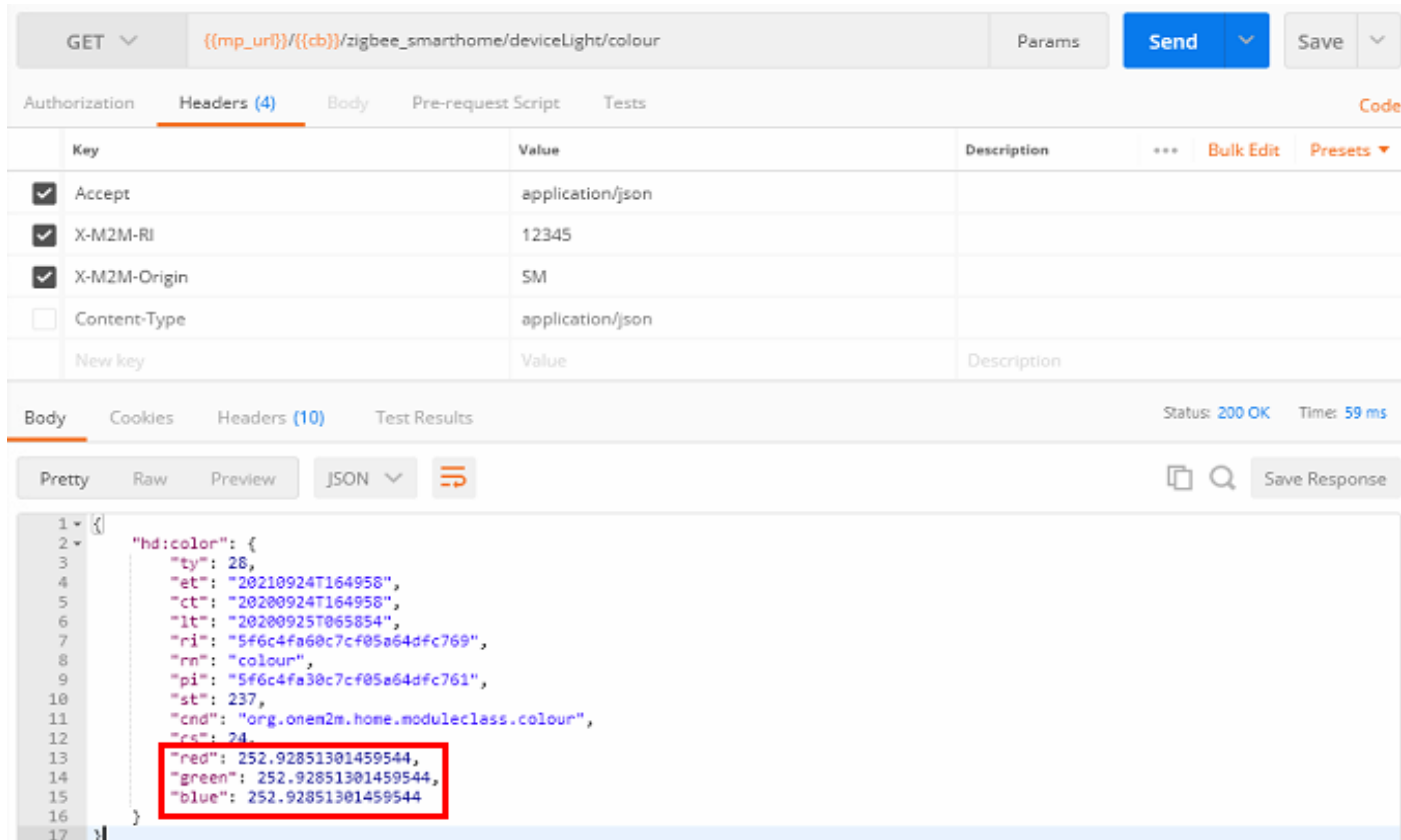
Running Zigbee IPE

- Get deviceLight (device class) information
 - IPE create AE (zigbee_smarthome) in the oneM2M platform
 - You can check your device resource by postman



Running Zigbee IPE

- Get colour (module class) info
 - colour was create under deviceLight



GET `{{mp_url}}/{{cb}}/zigbee_smarthome/deviceLight/colour` Params Send Save

Authorization Headers (4) Body Pre-request Script Tests Code

Key	Value	Description
<input checked="" type="checkbox"/> Accept	application/json	
<input checked="" type="checkbox"/> X-M2M-RI	12345	
<input checked="" type="checkbox"/> X-M2M-Origin	SM	
<input type="checkbox"/> Content-Type	application/json	
New key	Value	Description

Body Cookies Headers (10) Test Results Status: 200 OK Time: 59 ms

Pretty Raw Preview JSON Save Response

```
1 {
2   "hd:color": {
3     "ty": 28,
4     "et": "20210924T164958",
5     "ct": "20200924T164958",
6     "lt": "20200925T065854",
7     "ri": "5f6c4fa60c7cf05a64dfc769",
8     "rn": "colour",
9     "pi": "5f6c4fa30c7cf05a64dfc761",
10    "st": 237,
11    "cnd": "org.onem2m.home.moduleclass.colour",
12    "re": 24,
13    "red": 252.92851301459544,
14    "green": 252.92851301459544,
15    "blue": 252.92851301459544
16  }
17 }
```

Running Zigbee IPE

- New sensor data sync
 - When the sensor data changes, that is updated on the oneM2M platform

```
create cin: PUT -> http://127.0.0.1:7599/wdc_base/zigbee_smarthome/deviceDoorLock/battery
create cin: 200 <- {"hd:bat":{"ty":28,"et":"20210924T164954","ct":"20200924T164954","lt":"20200925T065853","ri":"5f6c4fa20c7cf05a64dfc759","rn":"battery","pi":"5f6c4fa10c7cf05a64dfc755","st":239,"cnd":"org.onem2m.home.moduleclass.battery","cs":8,"lvl":100}}
{
  code: 200,
  body: '{"hd:bat":{"ty":28,"et":"20210924T164954","ct":"20200924T164954","lt":"20200925T065853","ri":"5f6c4fa20c7cf05a64dfc759","rn":"battery","pi":"5f6c4fa10c7cf05a64dfc755","st":239,"cnd":"org.onem2m.home.moduleclass.battery","cs":8,"lvl":100}}'
}
create cin: PUT -> http://127.0.0.1:7599/wdc_base/zigbee_smarthome/deviceThermometer/temperature
create cin: 200 <- {"hd:tempe":{"ty":28,"et":"20210924T164955","ct":"20200924T164955","lt":"20200925T065853","ri":"5f6c4fa30c7cf05a64dfc75d","rn":"temperature","pi":"5f6c4fa20c7cf05a64dfc75b","st":246,"cnd":"org.onem2m.home.moduleclass.temperature","cs":8,"curT0":2549}}
{
  code: 200,
  body: '{"hd:tempe":{"ty":28,"et":"20210924T164955","ct":"20200924T164955","lt":"20200925T065853","ri":"5f6c4fa30c7cf05a64dfc75d","rn":"temperature","pi":"5f6c4fa20c7cf05a64dfc75b","st":246,"cnd":"org.onem2m.home.moduleclass.temperature","cs":8,"curT0":2549}}'
}
create cin: PUT -> http://127.0.0.1:7599/wdc_base/zigbee_smarthome/deviceThermometer/battery
create cin: 200 <- {"hd:bat":{"ty":28,"et":"20210924T164955","ct":"20200924T164955","lt":"20200925T065853","ri":"5f6c4fa30c7cf05a64dfc75f","rn":"battery","pi":"5f6c4fa20c7cf05a64dfc75b","st":239,"cnd":"org.onem2m.home.moduleclass.battery","cs":8,"lvl":100}}
{
  code: 200,
  body: '{"hd:bat":{"ty":28,"et":"20210924T164955","ct":"20200924T164955","lt":"20200925T065853","ri":"5f6c4fa30c7cf05a64dfc75f","rn":"battery","pi":"5f6c4fa20c7cf05a64dfc75b","st":239,"cnd":"org.onem2m.home.moduleclass.battery","cs":8,"lvl":100}}'
}
create cin: PUT -> http://127.0.0.1:7599/wdc_base/zigbee_smarthome/deviceLight/faultDetection
create cin: 200 <- {"hd:fauDn":{"ty":28,"et":"20210924T164957","ct":"20200924T164957","lt":"20200925T065854","ri":"5f6c4fa50c7cf05a64dfc765","rn":"faultDetection","pi":"5f6c4fa30c7cf05a64dfc761","st":239,"cnd":"org.onem2m.home.moduleclass.faultDetection","cs":4,"sus":true}}
{
  code: 200,
  body: '{"hd:fauDn":{"ty":28,"et":"20210924T164957","ct":"20200924T164957","lt":"20200925T065854","ri":"5f6c4fa50c7cf05a64dfc765","rn":"faultDetection","pi":"5f6c4fa30c7cf05a64dfc761","st":239,"cnd":"org.onem2m.home.moduleclass.faultDetection","cs":4,"sus":true}}'
}
```