

# Gate HTTP - getting information about air quality from Sensor Community

**Note!** The following instruction is dedicated for the second generation of HTTP Gate module (FW: 1.1.11-2218B)!

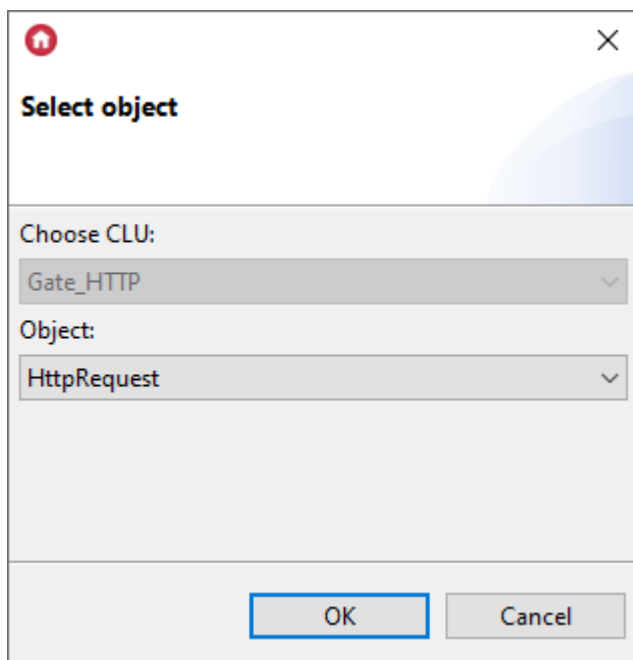
If we want to use information about air quality in the system, we may use an external service for this purpose e.g.: <https://sensor.community>.

According to the example on the website: <https://github.com/opendata-stuttgart/meta/wiki/EN-APIs>, the API request might look like this: `https:// data.sensor.community /airrohr/v1/sensor/apiID/`

where: **apiID** it is a sensor marking visible after selecting it on the [map](#).

## 1. Reading values from sensors

- Create the `HttpRequest` virtual object:



- Set the following parameters in the `HttpRequest` virtual object:

**Object properties**

Name:  Type:

Id:

Control  Events  Embedded features

Feature name	Current value	Initial value	Unit	Range
Host	-	<input type="text" value="https://data.sensor.community"/>	string	
Path	-	<input type="text" value="/airrohr/v1/sensor/61982/"/>	string	
QueryStringParams	-	<input type="text" value="\z"/>	string	
Method	-	<input type="text" value="GET"/>	string	
Timeout	-	<input type="text" value="10"/>	s	[1-255]
RequestType	-	<input type="text" value="Text"/>	-	0,1,2,3,4,5
ResponseType	-	<input type="text" value="JSON"/>	-	0,1,2,3,4,5
RequestHeaders	-	<input type="text" value="\z"/>	string	
RequestBody	-	<input type="text" value="\z"/>	string	
ResponseBody	-	<input type="text" value="\z"/>	string	
StatusCode	-		-	

Auto refresh

where:

```
Host: https://data.sensor.community
Path: /airrohr/v1/sensor/61982/
```

- Create another `HttpRequest` virtual object and set its parameters as follows:

**Object properties**

Name:  Type:

Id:

Control  Events  Embedded features

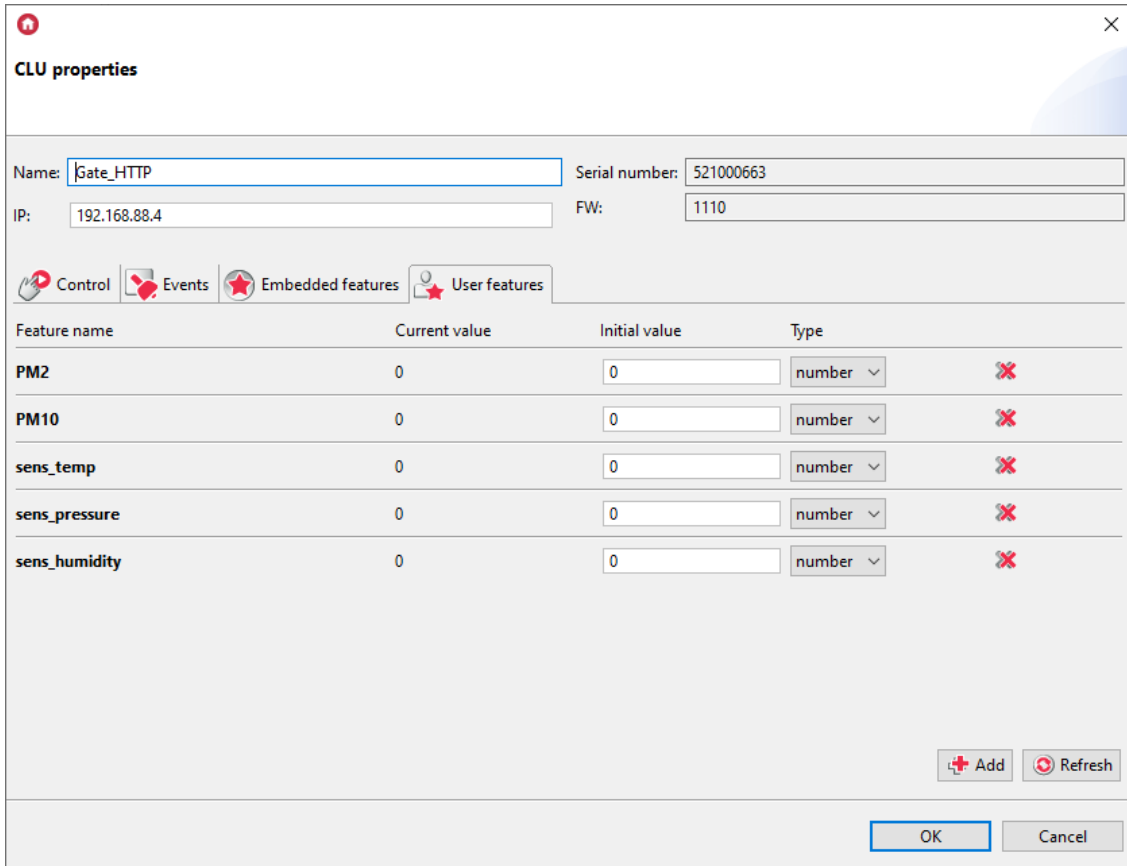
Feature name	Current value	Initial value	Unit	Range
Host	-	<input type="text" value="https://data.sensor.community"/>	string	
Path	-	<input type="text" value="/airrohr/v1/sensor/61983/"/>	string	
QueryStringParams	-	<input type="text" value="\z"/>	string	
Method	-	<input type="text" value="GET"/>	string	
Timeout	-	<input type="text" value="10"/>	s	[1-255]
RequestType	-	<input type="text" value="Text"/>	-	0,1,2,3,4,5
ResponseType	-	<input type="text" value="JSON"/>	-	0,1,2,3,4,5
RequestHeaders	-	<input type="text" value="\z"/>	string	
RequestBody	-	<input type="text" value="\z"/>	string	
ResponseBody	-	<input type="text" value="\z"/>	string	
StatusCode	-		-	

Auto refresh

where:

```
Host: https://data.sensor.community  
Path: /airrohr/v1/sensor/61983/
```

- In the next step, create user features of the *number* type:



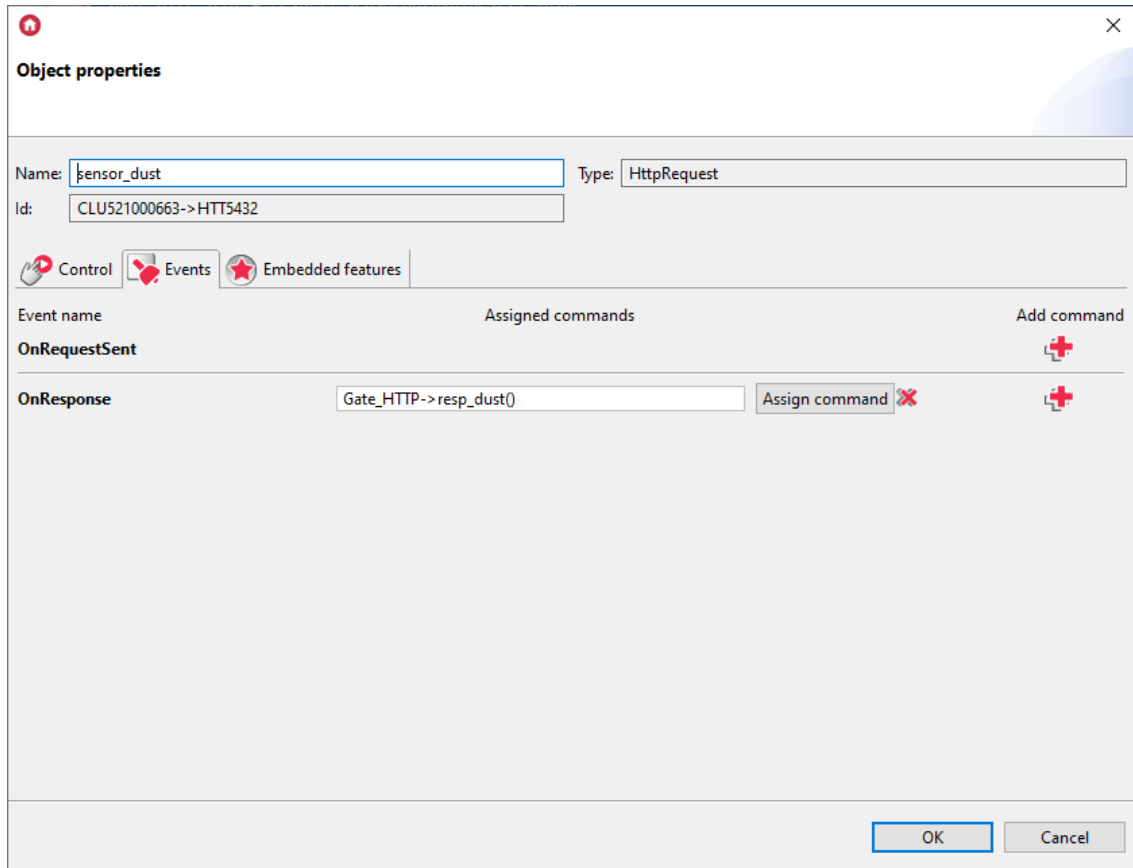
The screenshot shows a dialog box titled "CLU properties" with a close button (X) in the top right corner. The dialog contains several input fields: "Name:" with the value "Gate\_HTTP", "Serial number:" with "521000663", "IP:" with "192.168.88.4", and "FW:" with "1110". Below these fields are four tabs: "Control", "Events", "Embedded features", and "User features", with "User features" being the active tab. The "User features" tab displays a table with the following columns: "Feature name", "Current value", "Initial value", "Type", and a red "X" icon. The table lists five features: "PM2", "PM10", "sens\_temp", "sens\_pressure", and "sens\_humidity", each with a current value of 0 and an initial value of 0, and a type of "number". At the bottom right of the dialog are "Add" and "Refresh" buttons, and at the very bottom are "OK" and "Cancel" buttons.

Feature name	Current value	Initial value	Type	
PM2	0	0	number	X
PM10	0	0	number	X
sens_temp	0	0	number	X
sens_pressure	0	0	number	X
sens_humidity	0	0	number	X

- Then prepare a script called *resp\_dust*:

```
if(Gate_HTTP->sensor_dust->StatusCode==200) then  
  
    local resp = Gate_HTTP->sensor_dust->ResponseBody  
    Gate_HTTP->PM10 = resp[1].sensordatavalues[1].value  
    Gate_HTTP->PM2 = resp[1].sensordatavalues[2].value  
  
end
```

- Assign the script to the **OnResponse** event in the `HttpRequest` virtual object named `sensor_dust`:



- Then prepare a script called `resp_air`:

```
if(Gate_HTTP->sensor_air->StatusCode==200) then

    local resp = Gate_HTTP->sensor_air->ResponseBody
    Gate_HTTP->sens_temp = resp[1].sensordatavalues[1].value
    Gate_HTTP->sens_pressure = resp[1].sensordatavalues[2].value/100
    Gate_HTTP->sens_humidity = resp[1].sensordatavalues[3].value

end
```

- Assign the script to the **OnResponse** event in the `HttpRequest` virtual object named `sensor_air`.

The screenshot shows the 'Object properties' dialog box for a virtual object named 'sensor\_air'. The object type is 'HttpRequest' and its ID is 'CLU521000663->HTT1790'. The 'Events' tab is selected, showing a table of events. The 'OnResponse' event is assigned the command 'Gate\_HTTP->resp\_air()'. There are 'Add command' and 'Assign command' buttons for each event.

Event name	Assigned commands	Add command
OnRequestSent		
OnResponse	Gate_HTTP->resp_air()	

- Send the configuration to the CLU.
- After sending the configuration, in both objects call the **SendRequest** method.
- After calling the script, the **StatusCode** feature in both objects should get the value of `200`.
- User feature values should get the appropriate values:

The screenshot shows the 'CLU properties' dialog box for a CLU named 'Gate\_HTTP'. The serial number is '521000663' and the IP is '192.168.88.4'. The 'User features' tab is selected, showing a table of user features with their current and initial values and types.

Feature name	Current value	Initial value	Type	
PM2	3.10	0	number	
PM10	4.70	0	number	
sens_temp	23.29	0	number	
sens_pressure	991.46	0	number	
sens_humidity	41.57	0	number	

- For comparison - responses to requests sent via the browser:

```
▼ 0:
  sampling_rate:      null
  ▼ sensordatavalues:
    ▼ 0:
      value:          "4.70"
      value_type:     "P1"
      id:             24475488983
    ▼ 1:
      value:          "3.10"
      value_type:     "P2"
      id:             24475489007
  timestamp:         "2022-06-23 11:05:20"
  ▼ location:
    altitude:        "208.9"
    indoor:          0
    exact_location:  1
    country:         "PL"
    id:              51978
    longitude:       "19.96903990550"
    latitude:        "50.07818090000"
  id:                11016208223
  ▼ sensor:
    ▼ sensor_type:
      manufacturer:  "Nova Fitness"
      name:          "SDS011"
      id:            14
    pin:            "1"
    id:             61982
```

```

▼ 0:
  sampling_rate:      null
  ▼ sensordatavalues:
    ▼ 0:
      value:          "23.29"
      value_type:     "temperature"
      id:             24475490299
    ▼ 1:
      value:          "99145.56"
      value_type:     "pressure"
      id:             24475490368
    ▼ 2:
      value:          "41.57"
      value_type:     "humidity"
      id:             24475490397
    ▼ 3:
      value:          101555.43
      value_type:     "pressure_at_sealevel"
  timestamp:         "2022-06-23 11:05:27"
  ▼ location:
    altitude:        "208.9"
    indoor:          0
    exact_location:  1
    country:         "PL"
    id:              51978
    longitude:       "19.96903990550"
    latitude:        "50.07818090000"
  id:                11016208774
  ▼ sensor:
    ▼ sensor_type:
      manufacturer:  "Bosch"
      name:          "BME280"
      id:            17
      pin:           "11"
      id:            61983

```

- Uzyskane dane można wyświetlić w aplikacji mobilnej, na Smart Panelu lub wykorzystać do tworzenia logiki w systemie.