Rapport TP4: Fast prototyping for IoT

Grégoire HEBRAS
Rami KARAOUD

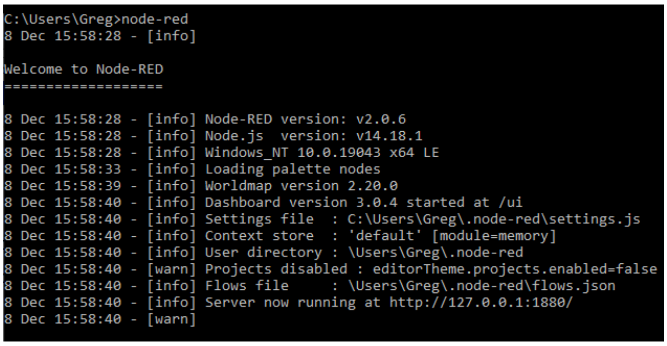
Introduction:

Node-RED is based on Node.js, it is a visual programming tool that allows to deploy high level applications quickly and easily by using a flow based user interface.

The idea of this practical work is to a real life use case where devices are connected to several technologies. We now know how to interact with HTTP REST interface and with MQTT publish/subscribe mechanisms. the aim here is to develop high level application faster using Node-RED

We will use the application developed in previous TPs, such as a OneM2M architecture and built in applications like the lamps GUI (application 41).

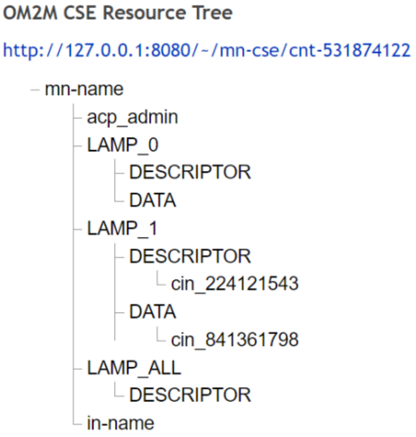
First, after proper downloading and installation, we can launch Node-RED.
We can then access the web interface via the port 1880 of our localhost.
We also launch the lamps application visible on our in-cse and mn-cse.



```
C:\Users\Greg>node-red
8 Dec 15:58:28 - [info]

Welcome to Node-RED
===================

8 Dec 15:58:28 - [info] Node-RED version: v2.0.6
8 Dec 15:58:28 - [info] Node.js  version: v14.18.1
8 Dec 15:58:28 - [info] Windows_NT 10.0.19043 x64 LE
8 Dec 15:58:33 - [info] Loading palette nodes
8 Dec 15:58:39 - [info] Worldmap version 2.20.0
8 Dec 15:58:40 - [info] Dashboard version 3.0.4 started at /ui
8 Dec 15:58:40 - [info] Settings file  : C:\Users\Greg\.node-red\settings.js
8 Dec 15:58:40 - [info] Context store  : 'default' [module=memory]
8 Dec 15:58:40 - [info] User directory : \Users\Greg\.node-red
8 Dec 15:58:40 - [warn] Projects disabled : editorTheme.projects.enabled=false
8 Dec 15:58:40 - [info] Flows file     : \Users\Greg\.node-red\flows.json
8 Dec 15:58:40 - [info] Server now running at http://127.0.0.1:1880/
8 Dec 15:58:40 - [warn]
```



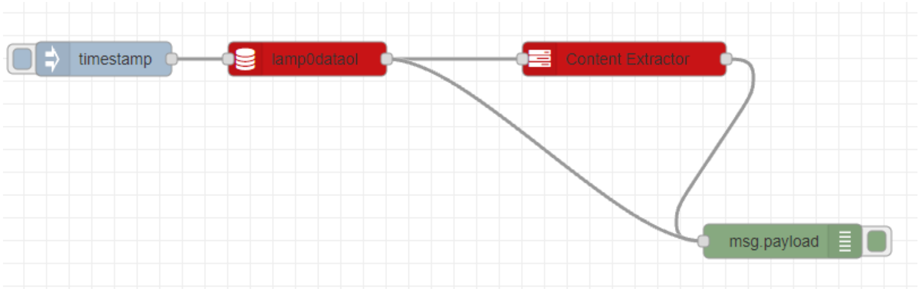
OM2M CSE Resource Tree

http://127.0.0.1:8080/~/mn-cse/cnt-531874122

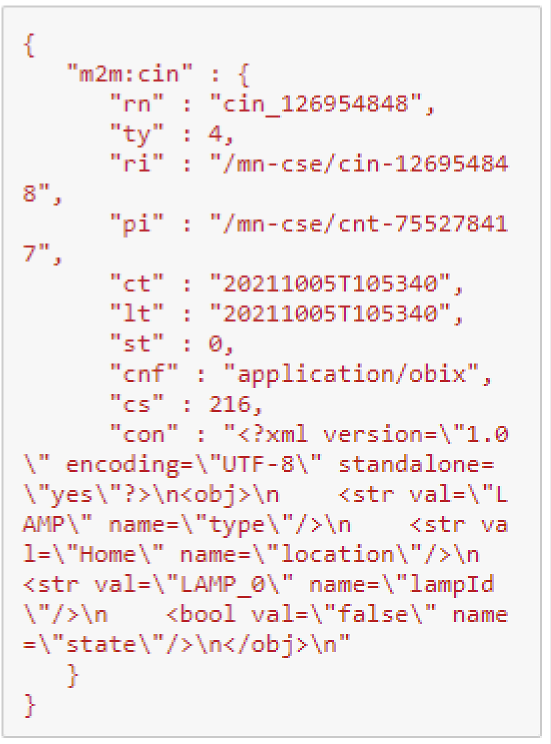
```
mn-name
    acp_admin
    LAMP_0
        DESCRIPTOR
        DATA
    LAMP_1
        DESCRIPTOR
            cin_224121543
        DATA
            cin_841361798
    LAMP_ALL
        DESCRIPTOR
    in-name
```

A. Simple test: get sensor values and display them

We can now use Node-RED to extract data from the content instances in the DATA containers. This can be done using the dedicated LOM2M WIP node pack accessible on the manage palette in the Node-RED web interface. here use the nodes “Named Sensor Data” and “Content Extractor”.

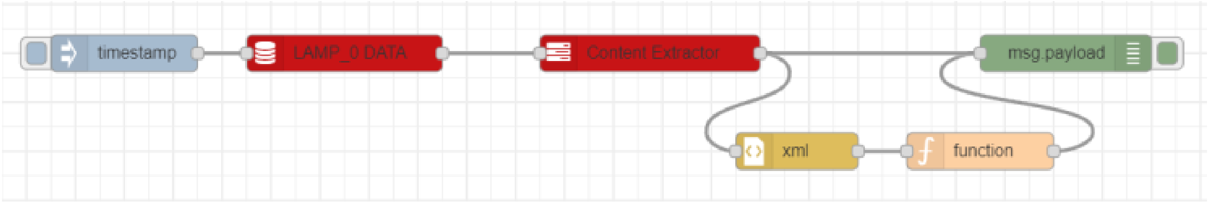


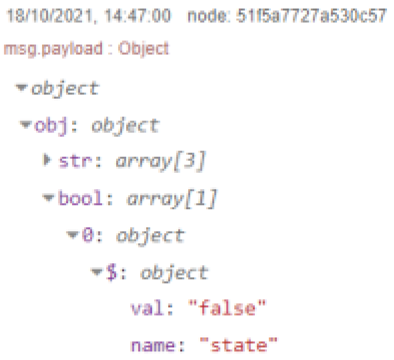
Thanks to the debug node we can display in the console the data extracted:



```
{
   "m2m:cin" : {
      "rn" : "cin_126954848",
      "ty" : 4,
      "ri" : "/mn-cse/cin-12695484
8",
      "pi" : "/mn-cse/cnt-75527841
7",
      "ct" : "20211005T105340",
      "lt" : "20211005T105340",
      "st" : 0,
      "cnf" : "application/obix",
      "cs" : 216,
      "con" : "<?xml version=\"1.0
\" encoding=\"UTF-8\" standalone=
\"yes\"?>\n<obj>\n    <str val=\"L
AMP\" name=\"type\"/>\n    <str va
l=\"Home\" name=\"location\"/>\n
<str val=\"LAMP_0\" name=\"lampId
\"/>\n    <bool val=\"false\" name
=\"state\"/>\n</obj>\n"
   }
}
```

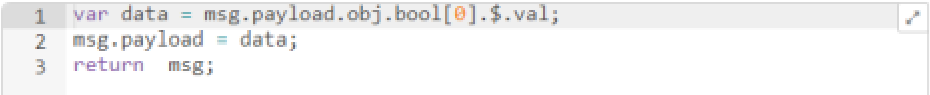
this can be clarified using the xml node that can build the data structure using the xml standard:





```
18/10/2021, 14:47:00   node: 51f5a7727a530c57
msg.payload : Object
▾object
 ▾obj: object
   ▸str: array[3]
   ▾bool: array[1]
     ▾0: object
       ▾$: object
          val: "false"
          name: "state"
```

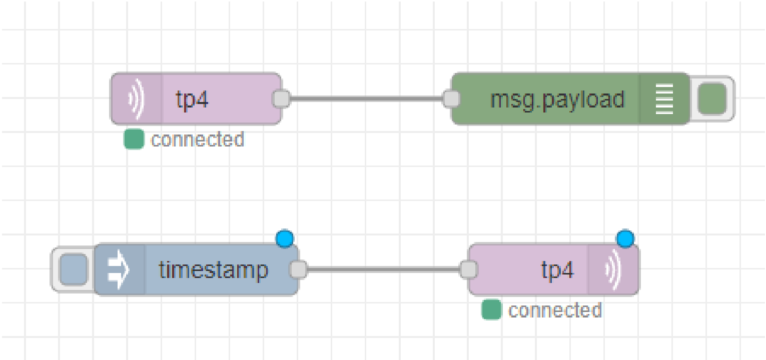
we can even display a single attribute by extracting it from the xml structure via a simple function:

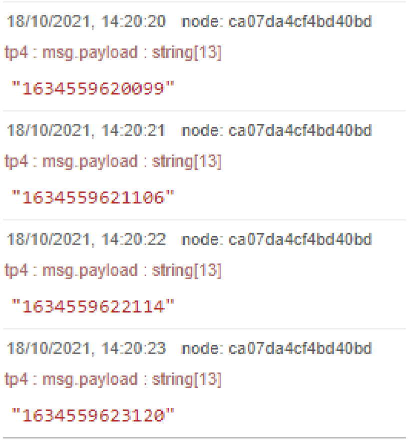


```
var data = msg.payload.obj.bool[0].$.val;
msg.payload = data;
return  msg;
```

In this case, this will return “false”.

This can also be done by a MQTT publish/subscribe mechanism.





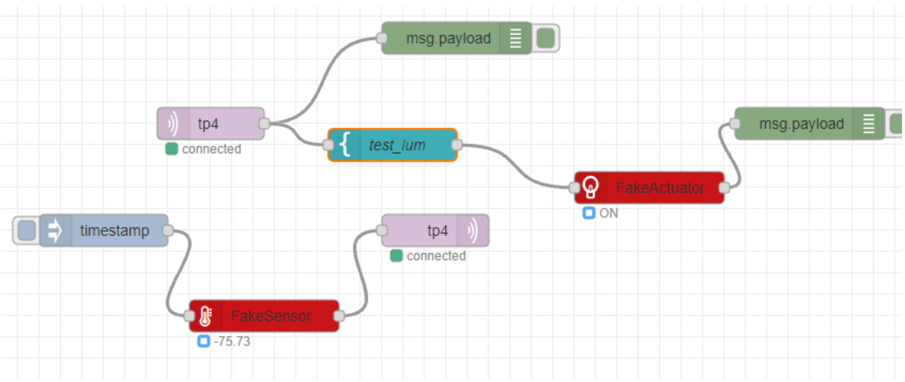
```
18/10/2021, 14:20:20   node: ca07da4cf4bd40bd
tp4 : msg.payload : string[13]
"1634559620099"
18/10/2021, 14:20:21   node: ca07da4cf4bd40bd
tp4 : msg.payload : string[13]
"1634559621106"
18/10/2021, 14:20:22   node: ca07da4cf4bd40bd
tp4 : msg.payload : string[13]
"1634559622114"
18/10/2021, 14:20:23   node: ca07da4cf4bd40bd
tp4 : msg.payload : string[13]
"1634559623120"
```

Using what we learn in the MQTT TP we managed to link the nodes “MQTT in” and “MQTT out” to recreate that mechanism.

This is also possible to do in a real life case, with the ESP board publishing its data to the same topic.
Unfortunately we were unable to make the ESP work properly on this session for unknown reasons. The following applications are done on simulated sensors and actuators nodes rather than actual sensors from the esp its led lamp

B. Sensors and actuators

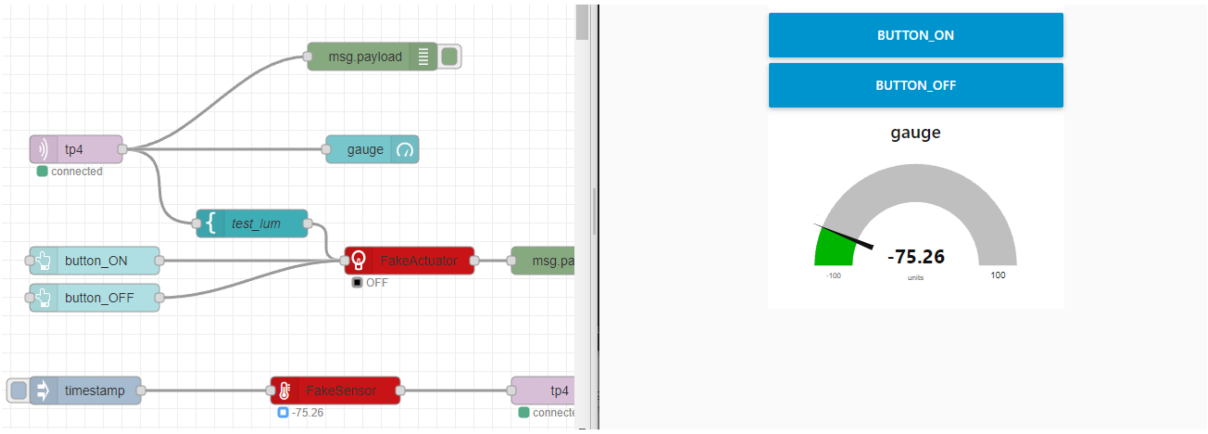
We now perform a simple condition test on a simulated sensor to execute and action, like activate a simulated actuator:



In the exemple above, a fake actuator is activated depending on the value generated by the fake sensor.

C. Dashboard

Finally we added a dashboard interface, with a gauge displaying the value of the simulated sensor, we also added two buttons to switch on and off the actuator (bypassing the simple condition test).



Conclusion:

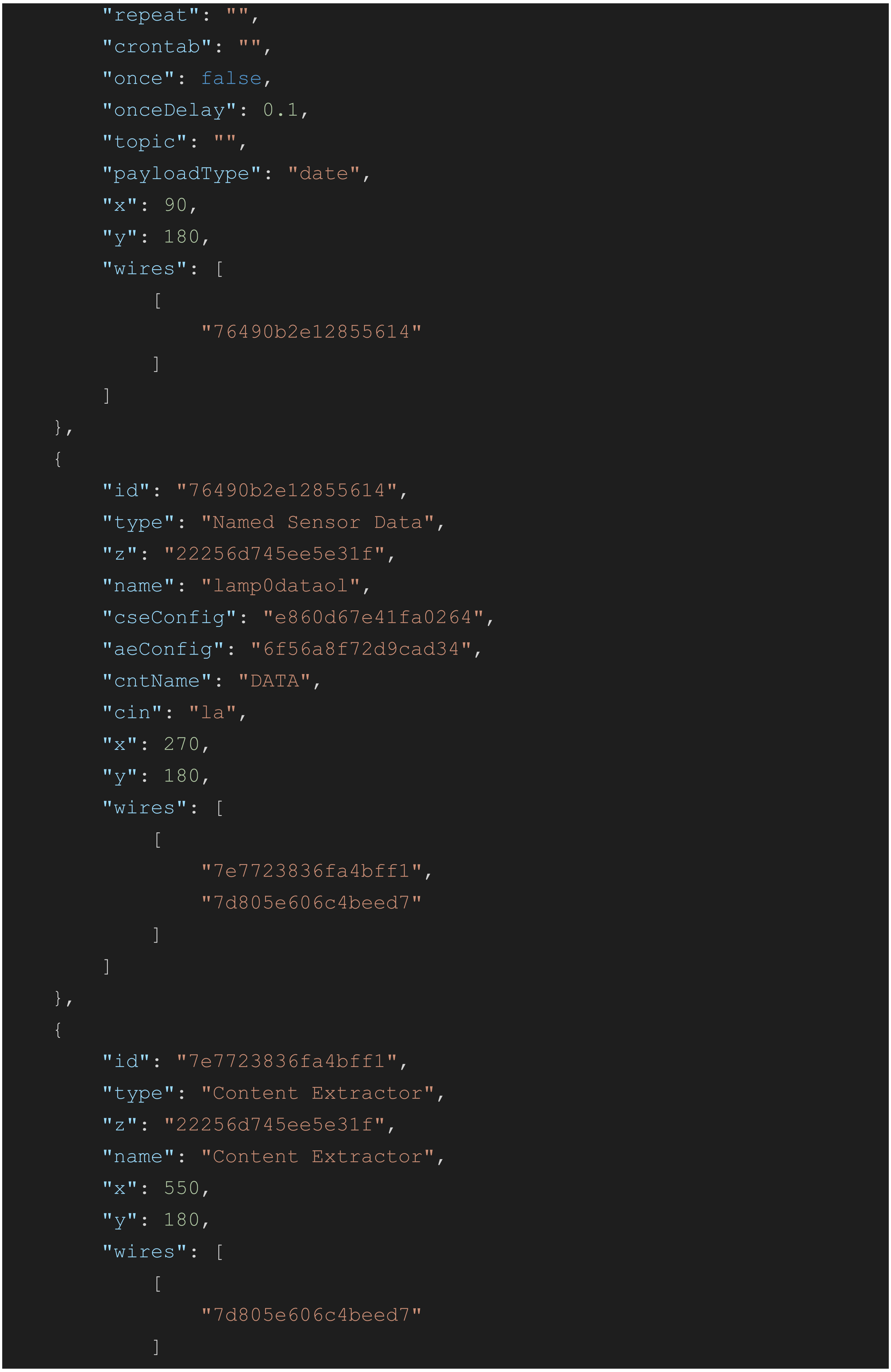
In this practictical work we had a first overview of how to quickly deploy a high level application using Node-RED. We were able to easily interact with heterogeneous devices, and deploy MQTT mechanisms thanks to the wide variety of accessible nodes in Node-RED It was really impressive to see how efficient this tool can be when used properly. It is definitely something that will be useful for future projects.

Unfortunately we lacked the time to implement a real life application use case with the ESP board.

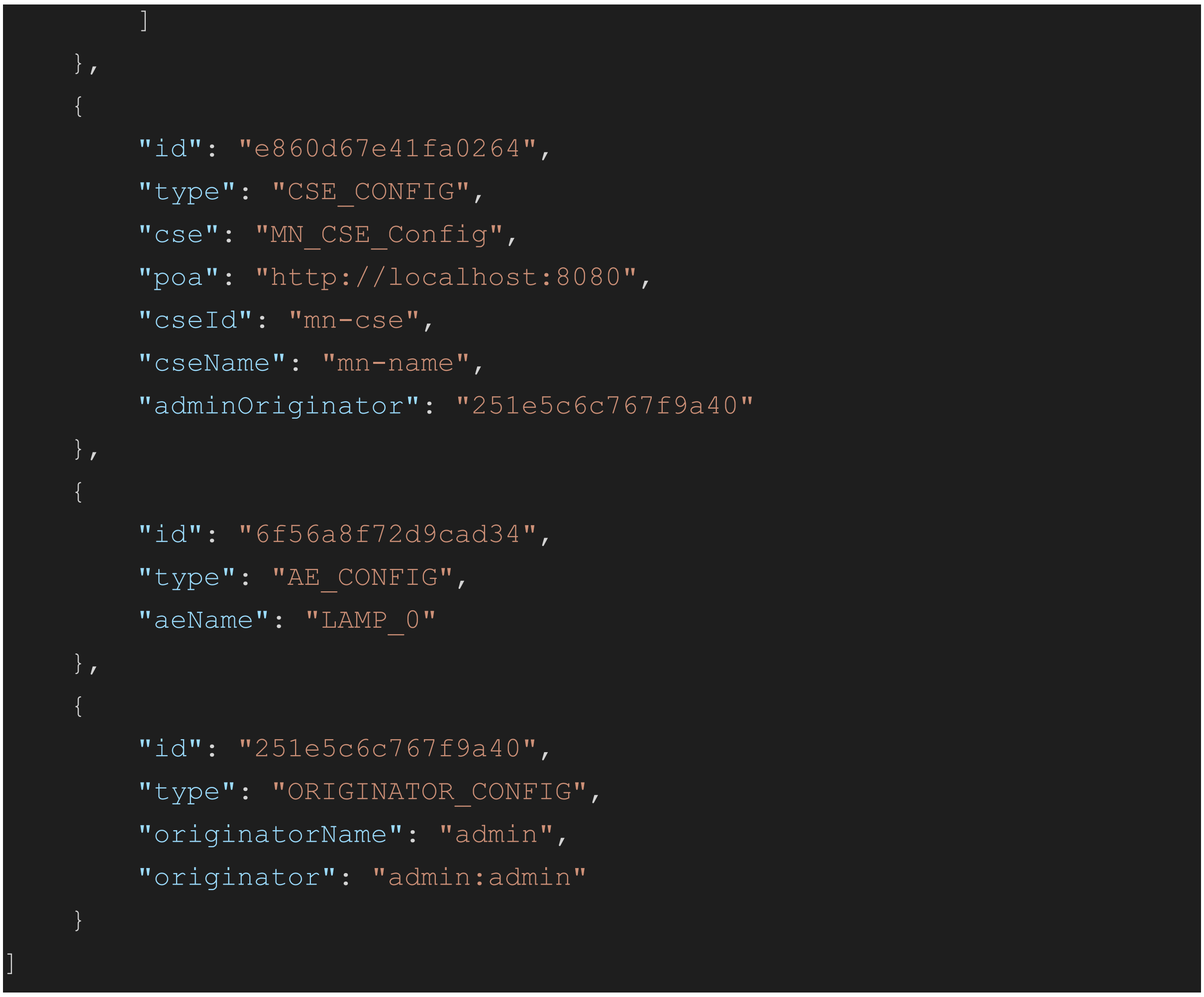
Flows Exports:

1. **content extractor:**

```
[
    {
        "id": "22256d745ee5e31f",
        "type": "tab",
        "label": "om2m_lamp",
        "disabled": false,
        "info": ""
    },
    {
        "id": "7d805e606c4beed7",
        "type": "debug",
        "z": "22256d745ee5e31f",
        "name": "",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "payload",
        "targetType": "msg",
        "statusVal": "",
        "statusType": "auto",
        "x": 690,
        "y": 340,
        "wires": []
    },
    {
        "id": "4296fe61dc605b15",
        "type": "inject",
        "z": "22256d745ee5e31f",
        "name": "",
        "props": [
            {
                "p": "payload"
            },
            {
                "p": "topic",
                "vt": "str"
            }
        ],
```

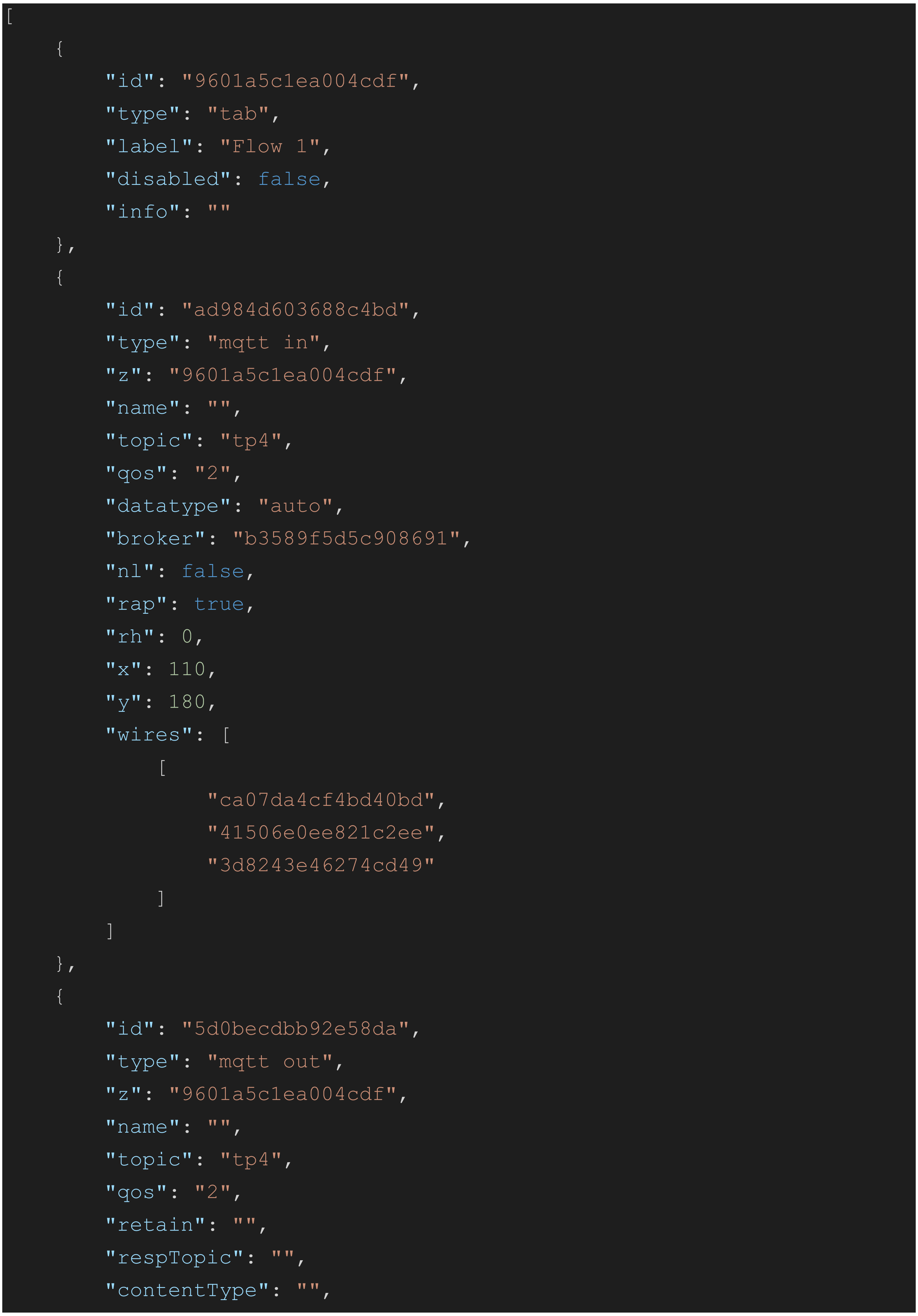


```
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "topic": "",
        "payloadType": "date",
        "x": 90,
        "y": 180,
        "wires": [
            [
                "76490b2e12855614"
            ]
        ]
    },
    {
        "id": "76490b2e12855614",
        "type": "Named Sensor Data",
        "z": "22256d745ee5e31f",
        "name": "lamp0dataol",
        "cseConfig": "e860d67e41fa0264",
        "aeConfig": "6f56a8f72d9cad34",
        "cntName": "DATA",
        "cin": "la",
        "x": 270,
        "y": 180,
        "wires": [
            [
                "7e7723836fa4bff1",
                "7d805e606c4beed7"
            ]
        ]
    },
    {
        "id": "7e7723836fa4bff1",
        "type": "Content Extractor",
        "z": "22256d745ee5e31f",
        "name": "Content Extractor",
        "x": 550,
        "y": 180,
        "wires": [
            [
                "7d805e606c4beed7"
            ]
```

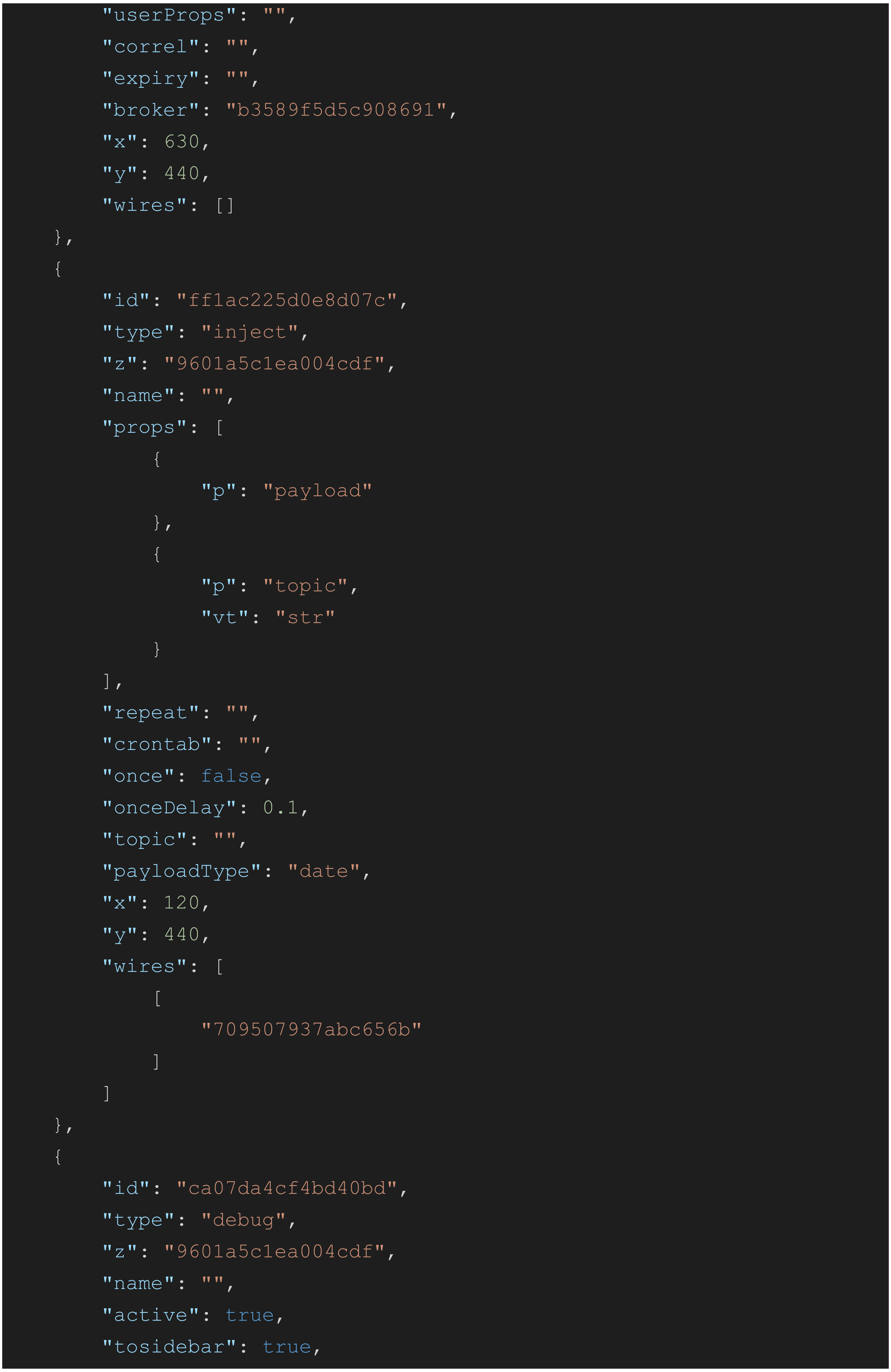


```
        ]
    },
    {
        "id": "e860d67e41fa0264",
        "type": "CSE_CONFIG",
        "cse": "MN_CSE_Config",
        "poa": "http://localhost:8080",
        "cseId": "mn-cse",
        "cseName": "mn-name",
        "adminOriginator": "251e5c6c767f9a40"
    },
    {
        "id": "6f56a8f72d9cad34",
        "type": "AE_CONFIG",
        "aeName": "LAMP_0"
    },
    {
        "id": "251e5c6c767f9a40",
        "type": "ORIGINATOR_CONFIG",
        "originatorName": "admin",
        "originator": "admin:admin"
    }
]
```

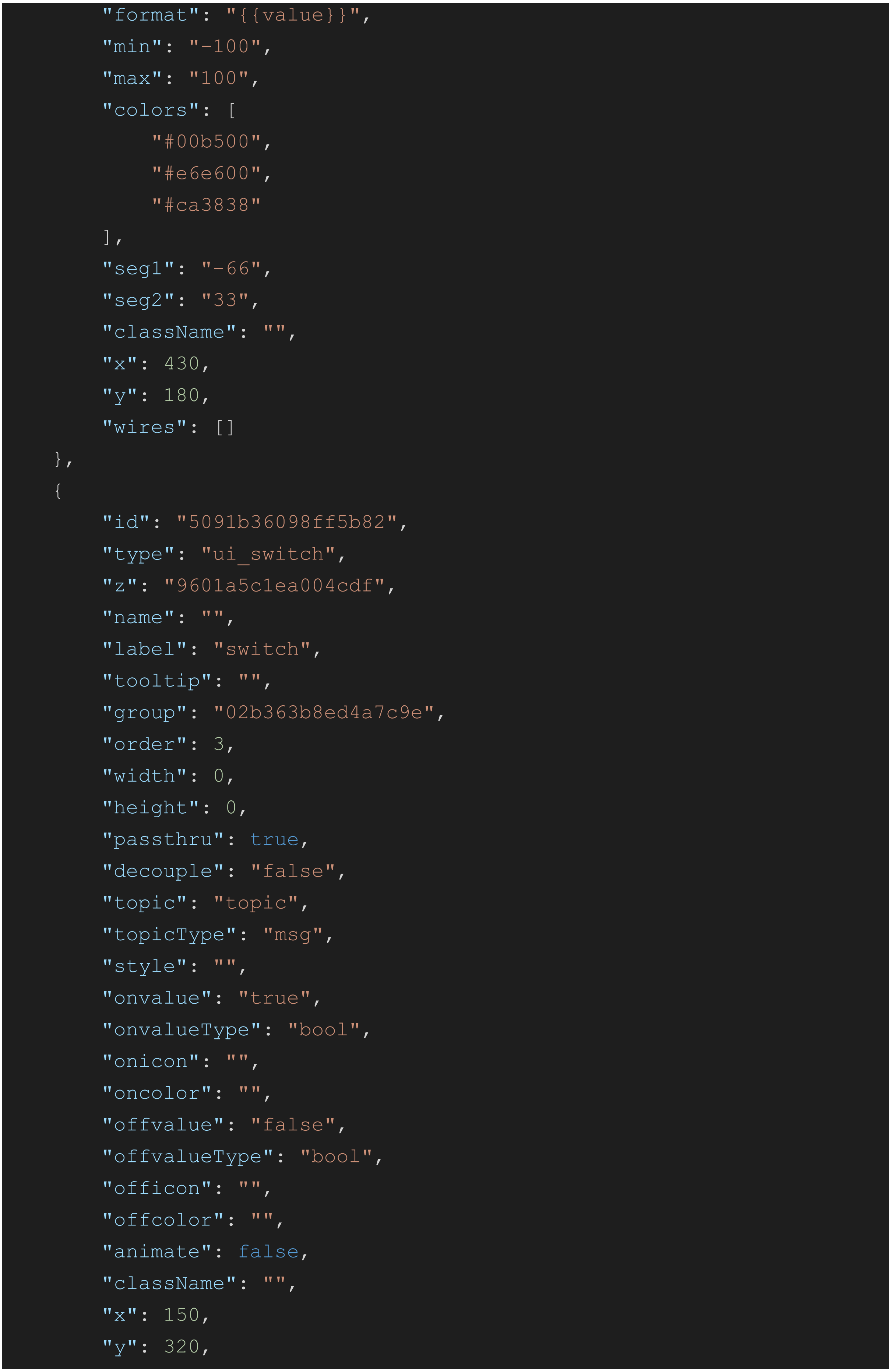
2. mqtt connection:



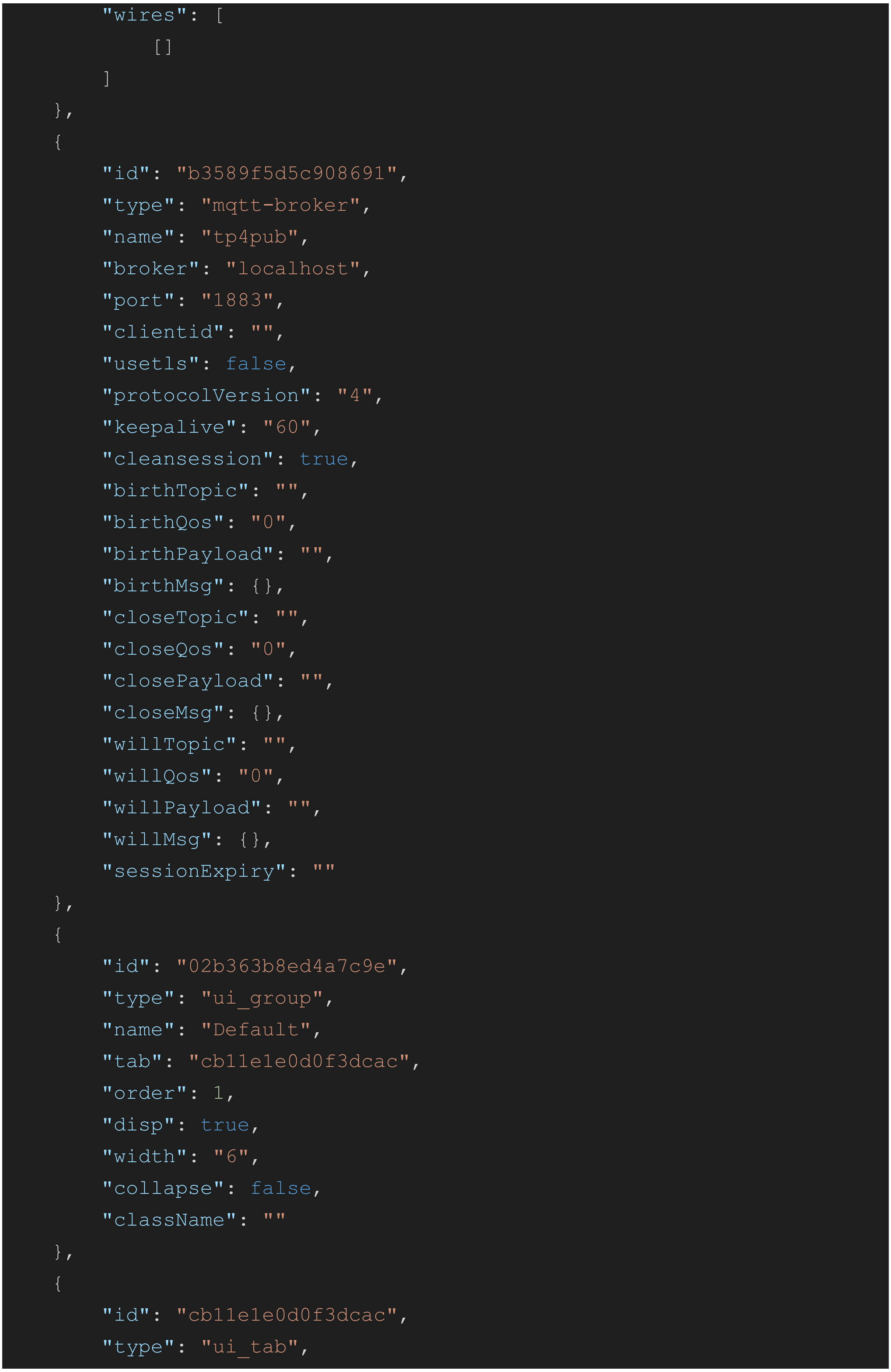
```
[
    {
        "id": "9601a5c1ea004cdf",
        "type": "tab",
        "label": "Flow 1",
        "disabled": false,
        "info": ""
    },
    {
        "id": "ad984d603688c4bd",
        "type": "mqtt in",
        "z": "9601a5c1ea004cdf",
        "name": "",
        "topic": "tp4",
        "qos": "2",
        "datatype": "auto",
        "broker": "b3589f5d5c908691",
        "nl": false,
        "rap": true,
        "rh": 0,
        "x": 110,
        "y": 180,
        "wires": [
            [
                "ca07da4cf4bd40bd",
                "41506e0ee821c2ee",
                "3d8243e46274cd49"
            ]
        ]
    },
    {
        "id": "5d0becdbb92e58da",
        "type": "mqtt out",
        "z": "9601a5c1ea004cdf",
        "name": "",
        "topic": "tp4",
        "qos": "2",
        "retain": "",
        "respTopic": "",
        "contentType": "",
```



```
        "userProps": "",
        "correl": "",
        "expiry": "",
        "broker": "b3589f5d5c908691",
        "x": 630,
        "y": 440,
        "wires": []
    },
    {
        "id": "ff1ac225d0e8d07c",
        "type": "inject",
        "z": "9601a5c1ea004cdf",
        "name": "",
        "props": [
            {
                "p": "payload"
            },
            {
                "p": "topic",
                "vt": "str"
            }
        ],
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "topic": "",
        "payloadType": "date",
        "x": 120,
        "y": 440,
        "wires": [
            [
                "709507937abc656b"
            ]
        ]
    },
    {
        "id": "ca07da4cf4bd40bd",
        "type": "debug",
        "z": "9601a5c1ea004cdf",
        "name": "",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "payload",
        "targetType": "msg",
        "statusVal": "",
        "statusType": "auto",
        "x": 430,
        "y": 80,
        "wires": []
    },
    {
        "id": "709507937abc656b",
        "type": "Fake Sensor",
        "z": "9601a5c1ea004cdf",
        "name": "FakeSensor",
        "minValue": -100,
        "maxValue": 100,
        "x": 390,
        "y": 440,
        "wires": [
            [
                "5d0becdbb92e58da"
            ]
        ]
    },
    {
        "id": "bf56c84337f57ab5",
        "type": "Fake Actuator",
        "z": "9601a5c1ea004cdf",
        "name": "FakeActuator",
        "x": 470,
        "y": 300,
        "wires": [
            [
                "cb1cf4b0d1f66dbe"
            ]
        ],
        "inputLabels": [
            "frommage"
        ]
    },
    {
        "id": "cb1cf4b0d1f66dbe",
        "type": "debug",
        "z": "9601a5c1ea004cdf",
        "name": "",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "false",
        "statusVal": "",
        "statusType": "auto",
        "x": 650,
        "y": 300,
        "wires": []
    },
    {
        "id": "41506e0ee821c2ee",
        "type": "SimpleCondition",
        "z": "9601a5c1ea004cdf",
        "operator": "<",
        "value_c": "0",
        "inputType": "msg",
        "input": "payload",
        "name": "test_lum",
        "x": 300,
        "y": 260,
        "wires": [
            [
                "bf56c84337f57ab5"
            ]
        ]
    },
    {
        "id": "3d8243e46274cd49",
        "type": "ui_gauge",
        "z": "9601a5c1ea004cdf",
        "name": "",
        "group": "02b363b8ed4a7c9e",
        "order": 0,
        "width": 0,
        "height": 0,
        "gtype": "gage",
        "title": "gauge",
        "label": "units",
```



```
        "format": "{{value}}",
        "min": "-100",
        "max": "100",
        "colors": [
            "#00b500",
            "#e6e600",
            "#ca3838"
        ],
        "seg1": "-66",
        "seg2": "33",
        "className": "",
        "x": 430,
        "y": 180,
        "wires": []
    },
    {
        "id": "5091b36098ff5b82",
        "type": "ui_switch",
        "z": "9601a5c1ea004cdf",
        "name": "",
        "label": "switch",
        "tooltip": "",
        "group": "02b363b8ed4a7c9e",
        "order": 3,
        "width": 0,
        "height": 0,
        "passthru": true,
        "decouple": "false",
        "topic": "topic",
        "topicType": "msg",
        "style": "",
        "onvalue": "true",
        "onvalueType": "bool",
        "onicon": "",
        "oncolor": "",
        "offvalue": "false",
        "offvalueType": "bool",
        "officon": "",
        "offcolor": "",
        "animate": false,
        "className": "",
        "x": 150,
        "y": 320,
```



```
        "wires": [
            []
        ]
    },
    {
        "id": "b3589f5d5c908691",
        "type": "mqtt-broker",
        "name": "tp4pub",
        "broker": "localhost",
        "port": "1883",
        "clientid": "",
        "usetls": false,
        "protocolVersion": "4",
        "keepalive": "60",
        "cleansession": true,
        "birthTopic": "",
        "birthQos": "0",
        "birthPayload": "",
        "birthMsg": {},
        "closeTopic": "",
        "closeQos": "0",
        "closePayload": "",
        "closeMsg": {},
        "willTopic": "",
        "willQos": "0",
        "willPayload": "",
        "willMsg": {},
        "sessionExpiry": ""
    },
    {
        "id": "02b363b8ed4a7c9e",
        "type": "ui_group",
        "name": "Default",
        "tab": "cb11e1e0d0f3dcac",
        "order": 1,
        "disp": true,
        "width": "6",
        "collapse": false,
        "className": ""
    },
    {
        "id": "cb11e1e0d0f3dcac",
        "type": "ui_tab",
```

```json
        "name": "Home",
        "icon": "dashboard",
        "disabled": false,
        "hidden": false
    }
]
```