

Web and Apps

3) JSON

Emmanuel Benoist
Spring Term 2021

Last Week

JSON

- Last Week
- Introduction
- Basic JavaScript syntax
- JSON in Node Red
- More complicated example
- The change node
- Conclusion

HTML / CSS

▶ Page HTML contains tags

```
<h1>Hello world</h1>
<div class="menu">
  <div class="menu_item"><a href="page1.html">Page Un</a>\
  →</div>
  <div class="menu_item"><a href="page2.html">Page deux</a>\
  →</div>
</div>
```

▶ HTML is represented by a Tree : The Document Object Model (DOM)

- ▶ Tags are nodes
- ▶ Texts are leaves

▶ Layout is written in CSS

- ▶ Nodes of a given tag : tag
- ▶ nodes of a given class: .class
- ▶ node having a given ID: #id
- ▶ We define properties of the nodes of the DOM
- ▶ Fonts, effects, colors, surroundings.

Introduction

- ▶ **Wikipedia:**
JavaScript Object Notation (JSON) is an open standard file format, and data interchange format, that uses human-readable text to store and transmit data objects consisting of attribute-value pairs and array data types.
- ▶ **Transfer format**
 - ▶ Between an App and a server
 - ▶ Between a Web Client side application (JavaScript in the browser) and a server
 - ▶ Between two servers

JSON syntax

- ▶ **Strings are enclosed in double quotes**
 - ▶ "Hello world"
- ▶ **Lists are represented with square brackets**
 - ▶ ["A", "B", "C", "D"]
- ▶ **Objects are represented with curly brackets**
 - ▶ {}
- ▶ **Objects contain key : value pairs**
 - ▶ {"color": "red", "taste": "sweet", "name": "orange"}
- ▶ **Objects and lists can be mixed**

Example1 JSON

- ▶ **Data for one patient**

```
{  
  "first_name" : "John",  
  "last_name" : "Smith",  
  "weight" : 80,  
  "height" : 1.80,  
  "city" : "Biel",  
  "country" : "Switzerland"  
}
```

Example 2 JSON

► List of temperatures

```
[
  {
    "temperature" : 37.3,
    "date" : "12/02/2020"
  },
  {
    "temperature" : 39.3,
    "date" : "13/02/2020"
  },
  {
    "temperature" : 37.9,
    "date" : "15/02/2020"
  }
]
```

Example 3 JSON

► List of patients

```
[
  { "first_name" : "Johnny", "last_name" : "Smith", "weight" : 81, "height" : 1.80 },
  { "first_name" : "John", "last_name" : "Do", "weight" : 78, "height" : 1.70 },
  { "first_name" : "Edith", "last_name" : "Mayer", "weight" : 50, "height" : 1.60 },
  { "first_name" : "Lili", "last_name" : "Baleine", "weight" : 180, "height" : 1.65 }
]
```

Exercise

► Write a JSON structure for representing the following list of vital signs:

Date	Sign	Value
3/03/2020	temperature	37.0
4/03/2020	temperature	39.0
4/03/2020	pulse	104
5/03/2020	BP	159/107

► Insert this list into the data of Johnny Smith with the key vital_signs.

JavaScript

► Node.js

- Is a framework working server side
- Code is executed inside the server
- Prepares data to be send to the client

► Programming is done in JavaScript

- JavaScript is executed Server-side, inside the server

► Node Red

- A visual framework for programming a Node.js server
- Instructions are represented with nodes
- Nodes can be programmed and configured in JavaScript
- The nodes are executed inside the SERVER

Basic JavaScript syntax

Parsing and Serialization of JSON objects

▶ Parse a JSON string into a JS-Object

- ▶ `JSON.parse()`

```
str = '{"val':12,'foo':'bar'}";  
obj = JSON.parse(str);  
node.log(obj.val); // writes 12
```

▶ Serialize a JavaScript Object in JSON format

- ▶ `JSON.stringify()`

```
var obj= new Object();  
obj.val=34;  
obj.name='John';  
str = JSON.stringify(obj);
```

Debug in Node Red

▶ Use Debug Output

- ▶ Insert it inside a flow of information
- ▶ Prints out the content of "msg.payload" in the *debug message* window

▶ Use log command

- ▶ Inside JavaScript functions
- ▶ `node.log()`
- ▶ Output written inside the console (in our virtual machine).

▶ Example: (node for manipulating an input in a Web page)

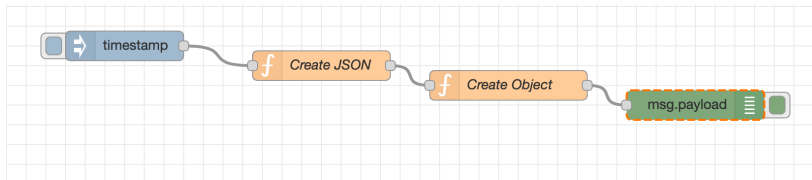
```
node.log("Number=" +msg.req.query.number);  
var val = parseInt(msg.req.query.number);  
msg.req.query.number2=val*val;  
return msg;
```

JSON in Node Red

JSON in Node Red

▶ Example 1

- ▶ We have an injection node (only for having a button)
- ▶ Then a function node that generates the JSON
- ▶ And another function node that consumes the JSON
- ▶ Output is then written out.



Example1 create JSON in a node

▶ Node create string:

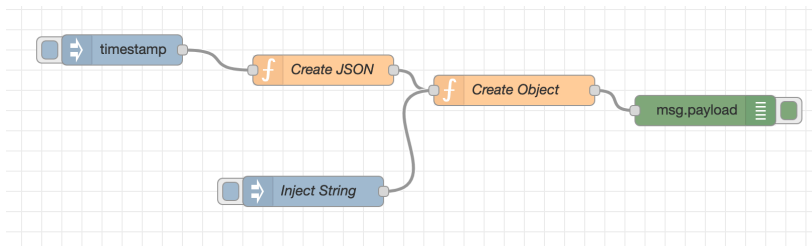
```
var data={"temp":37,"pulse":80};  
node.log(typeof data);  
msg.payload = JSON.stringify(data);  
return msg;
```

▶ Function node creates a JSON object

```
p = JSON.parse(msg.payload);  
node.log(typeof p);  
msg.payload = p.temp;  
return msg;
```

Example 2: Inject JSON

▶ Use an inject Node to inject JSON as a text



Example2

▶ Text injected:

Edit inject node

Delete Cancel Done

⚙️ Properties

✉ Payload

📄 Topic

Inject once after seconds, then

🕒 Repeat

🏷 Name

Note: "interval between times" and "at a specific time" will use cron.
"interval" should be 596 hours or less.
See info box for details.

Example 3

- ▶ We change the type generated by the injection node
 - ▶ From String to JSON
- ▶ We need to change the function node: no need to create an object, payload is already an object (and not a string).

```
//p = JSON.parse(msg.payload);  
p= msg.payload  
node.log(typeof p);  
msg.payload = p.temp;  
return msg;
```

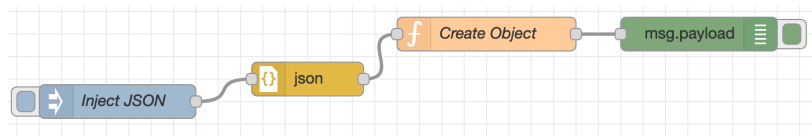
JSON Node

- ▶ Is used to convert a JSON String to a JSON Object
 - ▶ It parses the content of msg.payload
- ▶ Or to convert a JSON Object into a JSON String
 - ▶ It stringifies the msg.payload

Example 4, using a JSON node

- ▶ We add a json node.
 - ▶ We inject a JSON object
 - ▶ JSON node transform the object into a string
 - ▶ In the function we have:

```
p = JSON.parse(msg.payload);  
//p= msg.payload  
node.log(typeof p);  
msg.payload = p.temp;  
return msg;
```



More complicated example

Example 5:

- ▶ Inject JSON Data is more complicated

```
{
  "topic": "test/s1",
  "data": {
    "temp":39,
    "pulse":120}
}
```

- ▶ Function node

```
p=msg.payload;
node.log(typeof p);
msg.payload=p.data.temp;
return msg;
```

Ex6: The change node

- ▶ Inject Node (same data)

```
{
  "topic": "test/s1",
  "data": {
    "temp":39,
    "pulse":120}
}
```

- ▶ The change node

- ▶ We select the expression
- ▶ and write `payload.data.temp`
- ▶ The payload is replaced by the temperature.



The change node

Example 7: emulate a sensor

- ▶ We create different nodes for injecting values and a "Max Min" Node

```
var data = context.get('data') || {};
var data2 = context.get('data2') || {};
var period = 5; // seconds
var d= new Date();
now = d.getTime();
if(data2["count"]===undefined){
  data2["count"]=0;
}
data["sensor_id"]="0001";
data["sensor_type"]="power";
if(data["max"]===undefined){
  data["max"]=msg.payload;
}
if(data["min"]===undefined){
  data["min"]=msg.payload;
}
if(data2["stime"]===undefined)
  data2["stime"]=now;
```

Example 7: emulate a sensor (Cont.)

```
var count=data2.count;
data2.count+=1;
if(msg.payload>data.max)
  data.max=msg.payload;
if(msg.payload<data.min)
  data.min=msg.payload;
if((data2["stime"]+period*1000)<now){
  var out = JSON.stringify(data);
  msg.payload = out;
  data={}; // clear the data
  context.set('data',data);
  data2={};
  context.set('data2',data2);
  return msg;
}
context.set('data',data);
context.set('data2',data2);
return null;
```

Exercise

- ▶ **Import the example 7 into your system**
 - ▶ Access to the example page
 - ▶ Copy the source
 - ▶ Menu Import: paste the source
- ▶ **Modify the function to add an average**

Conclusion

- ▶ **JSON is a format for sending objects and lists**
 - ▶ Is used between all sorts of platforms (not just JavaScript).
- ▶ **One can manipulate JSON using boxes**
- ▶ **Manipulation of JSON is more convenient using code**
 - ▶ Calculating minimum and maximum requires programming
- ▶ **JSON is used to connect to a lot of services (server 2 server)**
 - ▶ One should also provide a JSON interface.

References

- ▶ **Tutorial JSON / Node Red**
<https://www.youtube.com/watch?v=24ZY3CEsiow>
- ▶ **Nasa web site**
<https://api.nasa.gov/>