Web Development

JSON - JavaScript Object Notation

JSON - JavaScript Object Notation

JavaScript Object Notation (JSON) is a standard text-based format for representing structured data based on JavaScript object syntax. It is commonly used for transmitting data in web applications.

- JSON is a syntax for storing and exchanging data
- It is light-weight
- It is language independent
- Easy to read and write
- Text based, human readable data exchange format

JSON - JavaScript Object Notation

JSON exists as a string — useful when you want to transmit data across a network. It needs to be converted to a native JavaScript object when you want to access the data.

This is not a big issue — JavaScript provides a global JSON object that has methods available for converting between the two.

JSON Structure

```json
{
  "squadName": "Super hero squad",
  "homeTown": "Metro City",
  "formed": 2016,
  "secretBase": "Super tower",
  "members": [
    {
      "name": "Molecule Man",
      "age": 29,
      "secretIdentity": "Dan Jukes",
      "powers": [
        "Radiation resistance",
        "Radiation blast"
      ]
    },
    {
      "name": "Madame Uppercut",
      "age": 39,
      "secretIdentity": "Jane Wilson",
      "powers": [
        "Million tonne punch",
        "Superhuman reflexes"
      ]
    }
  ]
}
```

JSON - Reading data

If we loaded this object into a JavaScript program, parsed in a variable called superHeroes for example, we could then access the data inside it using the same dot/bracket notation.
For example:

```
1 superHeroes.homeTown
2 superHeroes['formed']
3
```

JSON - Reading data

To access data further down the hierarchy, you simply have to chain the required property names and array indexes together. For example, to access the second superpower of the first hero listed in the members list:

```
1 superHeroes['members'][0]['powers'][1]
2
```

JSON - Reading data

1. First we have the variable name — superHeroes.
2. Inside that we want to access the members property, so we use ["members"].
3. Members contains an array populated by objects. We want to access the first object inside the array, so we use [0].
4. Inside this object, we want to access the powers property, so we use ["powers"].
5. Inside the powers property is an array containing the selected hero's superpowers. We want the second one, so we use [1].

JSON as Array

JSON text basically looks like a JavaScript object, and this is mostly right.

The reason we said "mostly right" is that an array is also valid JSON.

```
[
  {
    "name": "Molecule Man",
    "age": 29,
    "secretIdentity": "Dan Jukes",
    "powers": [
      "Radiation resistance",
      "Turning tiny",
      "Radiation blast"
    ]
  },
  {
    "name": "Madame Uppercut",
    "age": 39,
    "secretIdentity": "Jane Wilson",
    "powers": [
      "Million tonne punch",
      "Damage resistance",
      "Superhuman reflexes"
    ]
  }
]
```

JSON - More notes

- JSON is purely a data format — it contains **only properties**, no methods.
- JSON **requires double quotes** to be used around strings and property names. Single quotes are not valid.
- Even a single misplaced comma or colon can cause a JSON file to go wrong, and not work. You should be careful to validate any data you are attempting to use (although computer-generated JSON is less likely to include errors, as long as the generator program is working correctly). You can validate JSON using an application like JSONLint.
- JSON can actually take the form of any data type that is valid for inclusion inside JSON, not just arrays or objects. So for example, a single string or number would be a valid JSON object.
- Unlike in JavaScript code in which object properties may be unquoted, in JSON, only quoted strings may be used as properties.

JSON vs. XML

Common things:

- Both JSON and XML are "self describing" (human readable)
- Both JSON and XML are hierarchical (values within values)
- Both JSON and XML can be parsed and used by lots of programming languages
- Both JSON and XML can be fetched with an XMLHttpRequest

JSON vs. XML

Different things:

- JSON doesn't use end tag
- JSON is shorter
- JSON is quicker to read and write
- JSON can use arrays

The **biggest** difference is:
XML has to be parsed with an XML parser. JSON can be parsed by a standard JavaScript function.

```json
{
  "squadName": "Super hero squad",
  "homeTown": "Metro City",
  "formed": 2016,
  "secretBase": "Super tower",
  "members": [
    {
      "name": "Molecule Man",
      "age": 29,
      "secretIdentity": "Dan Jukes",
      "powers": [
        "Radiation resistance",
        "Radiation blast"
      ]
    },
    {
      "name": "Madame Uppercut",
      "age": 39,
      "secretIdentity": "Jane Wilson",
      "powers": [
        "Million tonne punch",
        "Superhuman reflexes"
      ]
    }
  ]
}
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<squad>
   <formed>2016</formed>
   <homeTown>Metro City</homeTown>
   <secretBase>Super tower</secretBase>
   <squadName>Super hero squad</squadName>
   <members>
      <superHero>
         <age>29</age>
         <name>Molecule Man</name>
         <secretIdentity>Dan Jukes</secretIdentity>
         <powers>
            <power>Radiation resistance</power>
            <power>Radiation blast</power>
         </powers>
      </superHero>
      <superHero>
         <age>39</age>
         <name>Madame Uppercut</name>
         <secretIdentity>Jane Wilson</secretIdentity>
         <powers>
            <power>Million tonne punch</power>
            <power>Superhuman reflexes</power>
         </powers>
      </superHero>
   </members>
```

JSON vs. XML

Why JSON is Better Than XML?

- XML is much more difficult to parse than JSON.
- JSON is parsed into a ready-to-use JavaScript object.

JSON Tutorial

First: download the files

Second: hands-on!

References and tools:

- MDN
- Beginners Book
- W3Schools
- Free Formatter - Convert XML to JSON and vice-versa
- JSONLint