

# JSON

MPRI 2.26.2: Web Data Management

---

Antoine Amarilli

Friday, December 14th

# JSON

- Meaning: JavaScript Object Notation
- Main goal: Serialization of data

```
{  
  "firstName": "John", "lastName": "Smith",  
  "address": {  
    "streetAddress": "21 2nd Street",  
    "city": "New York",  
    "postalCode": "10021-3100"  
  },  
  "phoneNumbers": [  
    { "type": "home", "number": "212 555-1234" },  
    { "type": "office", "number": "646 555-4567" },  
    { "type": "mobile", "number": "123 456-7890" }  
  ],  
}
```

# JSON syntax

- **Numbers**, including floating numbers, scientific notation
- **Strings**, with delimiters
  - Whitespace out of strings is **ignored**, unlike XML
  - Quotes, backslashes, control characters must be escaped; for Unicode, `\u0042`
- **Array**, ordered list of elements separated by commas
  - For instance: `[0, 1, "a", 2.4]`
- **Object**, dictionary of keys (strings) and values
  - For instance: `{"a": 1, "b": "c"}`
  - The **order** among keys is **irrelevant**
  - Duplicate keys are **discouraged** but **allowed**
- **Boolean values** `true`, `false`; special value `null`

## How to parse JSON with jQuery

```
function request() {  
  $.ajax({  
    url: "data.json",  
    cache: false,  
    success: function (data) {  
      $( "#load" ).html( data.load );  
      $( "#speed" ).html( data.speed );  
    }  
  });  
}
```

## Main differences with XML

- JSON syntax is (mostly) a **subset of JavaScript**
- JSON once deserialized is navigated as an **object** whereas XML is navigated with the DOM
- JSON does not **mix** text and structured data
- JSON syntax is **simpler**: no comments, attribute, namespaces...
- JSON is lighter: compare:

```
{  
  "a": ["a1", "a2"],  
  "b": ["b1", "b2"]  
}
```

```
<r>  
  <a> <v>a1</v> <v>a2</v> </a>  
  <b> <v>b1</v> <v>b2</v> </b>  
</r>
```

- JSON is **less carefully** normalized
- Less **bells and whistles**: little typing, no XSLT, etc.

# JSON ambiguities

- No **version number** for JSON
- Multiple **competing specifications**
- Nicolas Seriot, *Parsing JSON is a Minefield*  
[http://seriot.ch/parsing\\_json.php](http://seriot.ch/parsing_json.php)  
[https://github.com/nst/JSONTestSuite/blob/master/results/pruned\\_results.png](https://github.com/nst/JSONTestSuite/blob/master/results/pruned_results.png)

# JSON extensions

- **JSON Schema**, inspired by XML Schema, IETF (still a draft)
- **Binary serialization formats**: MessagePack, BSON, etc.
- **JSON-LD**, extension to store Linked Web data, used on **schema.org**
- **YAML**: extends JSON with several features, e.g., explicit types, user-defined types, anchors and references
- Some JSON parsers are permissive and allow, e.g., **comments**

# Querying JSON

- **Directly supported** in XPath 3.1 and XQuery 3.1 (March 2017)
- **MongoDB** is a NoSQL database using JSON that has its own query language
- **Many** proposals for a JSON query language: for a survey, see, e.g., Bourhis, Reutter, Suárez, Vrgoč. *JSON: data model, query languages and schema specification*.  
<https://arxiv.org/abs/1701.02221>



# Processing JSON

- In a **programming language**: you can **parse** the JSON then manipulate it as an object
- On the **command line**, use **jq**  
<https://stedolan.github.io/jq/>

```
curl -s 'https://api.github.com/repos/stedolan/jq/commits?per_page=5' |  
jq '[][ {  
    message: .commit.message,  
    name: .commit.committer.name,  
    parents: [.parents[].html_url]  
}]'
```

- Also: **line-delimited JSON** (newline-separated list of JSON objects)

## JSON and XML: Dispassionate assessment

- XML has mostly **failed** on the Web (XHTML, AJAX)
- **Overengineered** ecosystem with many **dusty** technologies: XQuery, XLink, XPointer, XInclude, XSL-FO, RDDL...
- However, XML is still widely used as an **exchange format**
- JSON does not replace all XML uses, e.g., **mixed content**
- Still, if your data is **easy to represent** as JSON and you don't need **fancy tools** (schemas, etc.), use JSON