JSON

JavaScript Object Notation (JSON)

- A lightweight format for storing and transferring data
- Is a text-based format that is valid JavaScript code
- It is "self-describing" and easy to read and understand
- While it uses JavaScript syntax, it is supported by nearly every programming language
- Supports strings, numbers, objects, arrays, booleans and null values.

```
{ "name":"John", "age":30, "car":null }
```

JavaScript Object Notation (JSON)

- JSON objects are surrounded by curly braces {}.
- JSON objects are written in key/value pairs.
- Keys must be strings, and values must be a valid JSON data type (string, number, object, array, boolean or null).
- Keys and values are separated by a colon.
- Each key/value pair is separated by a comma.

```
{ "name":"John", "age":30, "car":null }
```

JSON Data Types

- **Strings** - Strings must be double quotes. `{"name":"Bob"}`
- **Numbers** - Numbers must be integer or floating point numbers.

  ```
  5, 5.6
  ```

- **Objects** - Values can be JSON objects.

  ```
  {"employee":{ "name":"John", "age":30, "city":"New York" }}
  ```

- **Arrays** - Arrays must be an order list of any values.

  ```
  {"employees":[ "John", "Anna", "Peter" ]}
  ```

- **Boolean** - Must be true or false value. `{"sale":true}`
- **Null** - Values can also be null. `{"middlename":null}`

JSON Objects

An object is an unordered set of name/value pairs.

An object begins with { (left brace) and ends with } (right brace).

Each name is followed by : (colon) and the name/value pairs are separated by , (comma).

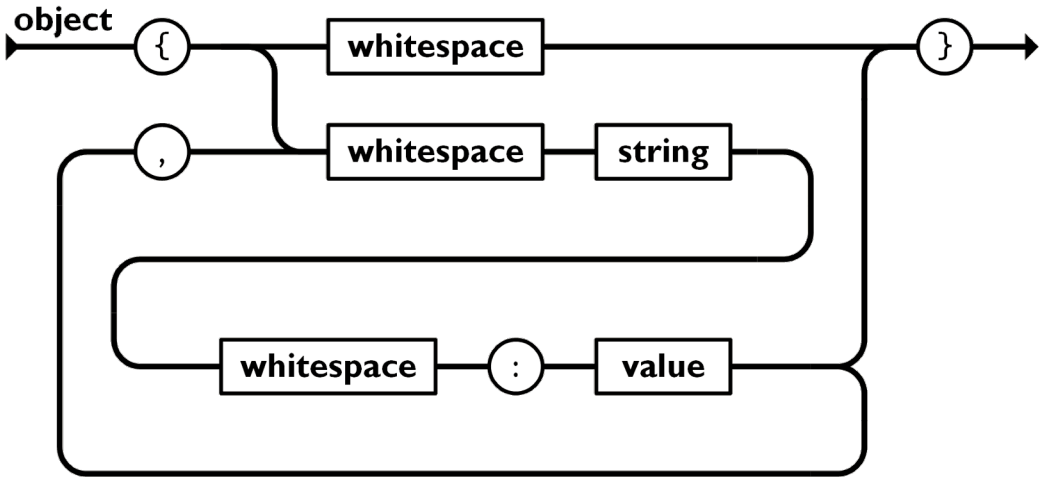

Image from json.org

JSON Arrays

An array is an ordered collection of values.

An array begins with [(left bracket) and ends with] (right bracket). Values are separated by , (comma).

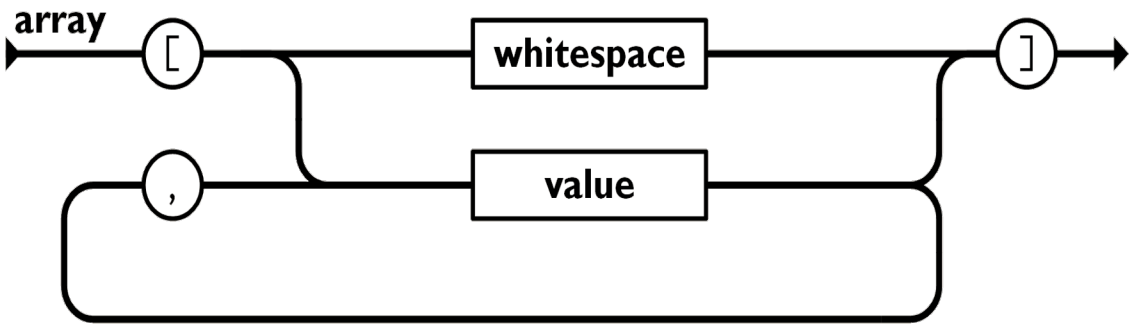

Image from json.org

JSON values

A value can be a string in double quotes, or a number, or true or false or null, or an object or an array.

These structures can be nested.

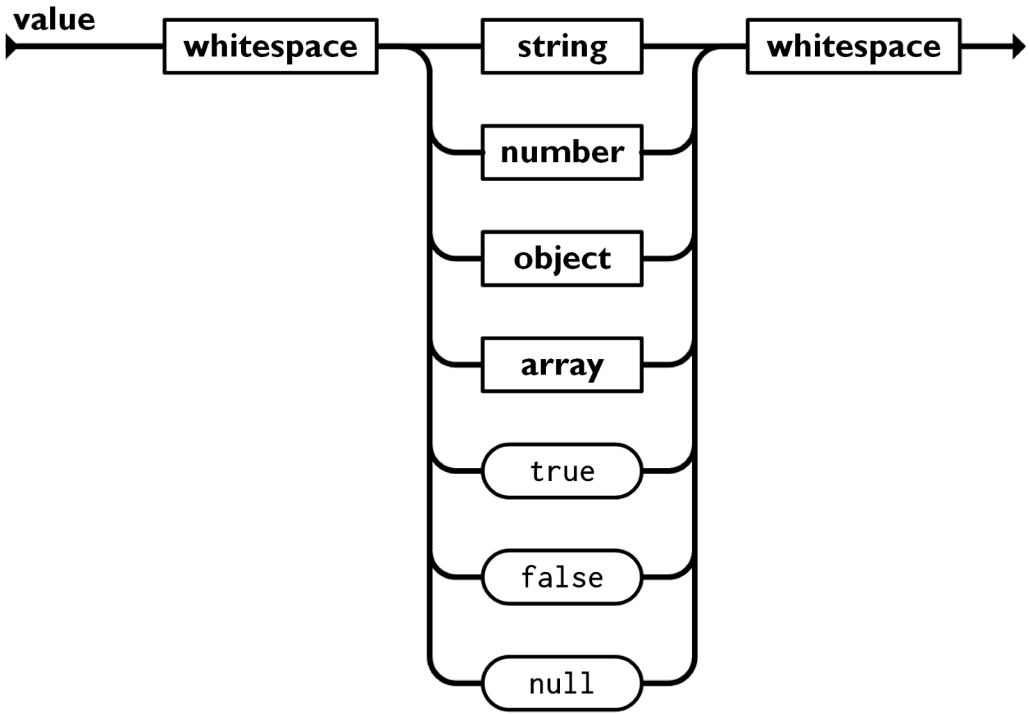

Image from json.org

JSON Strings, Characters and Numbers

- A string is a sequence of zero or more Unicode characters, wrapped in double quotes, using backslash escapes.
- A character is represented as a single character string. A string is very much like a C or Java string.
- A number is one or more positive or negative digits and decimals.

You can validate if a string is valid JSON using online JSON validators, such as JSONLint or JSON Formatter.

eXtensible Markup Language (XML)

- Markup language like HTML
- Designed to store and transport data
- Designed to be self-descriptive

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

JSON vs XML

JSON is Unlike XML because:

- JSON is shorter
- JSON is quicker to read and write
- JSON can use arrays
- XML has to be parsed with an XML parser. JSON can be parsed by a standard JavaScript function.

JSON vs XML

JSON is like XML because:

- 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 AJAX

JSON vs XML

```json
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
    "menuitem": [
      {"value": "New", "onclick": "CreateNewDoc()"},
      {"value": "Open", "onclick": "OpenDoc()"},
      {"value": "Close", "onclick": "CloseDoc()"}
    ]
  }
}}
```

```xml
<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateNewDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Close" onclick="CloseDoc()" />
  </popup>
</menu>
```

Why JSON is better than XML

- XML is hard to parse, JSON is parsed into a ready-to-use JavaScript object.
- When using JS with XML:
  - Fetch an XML document
  - Use the XML DOM to loop through the document
  - Extract values and store in variables
  - Process the data
- When using JS with JSON:
  - Fetch a JSON string
  - JSON.Parse the JSON string into a JavaScript object
  - Process the data