

{JSON}

The Fat-Free Alternative to XML

JSON as an XML Alternative

- JSON is a light-weight alternative to XML for data-interchange
- JSON = JavaScript Object Notation
 - It's really language independent
 - Most programming languages can easily read it and instantiate objects or some other data structure
- Defined in [RFC 4627](#), IETF, July 2006
 - Current version is [RFC 8259](#), December 2017
- <http://json.org/> has more information

JSON TL;DR

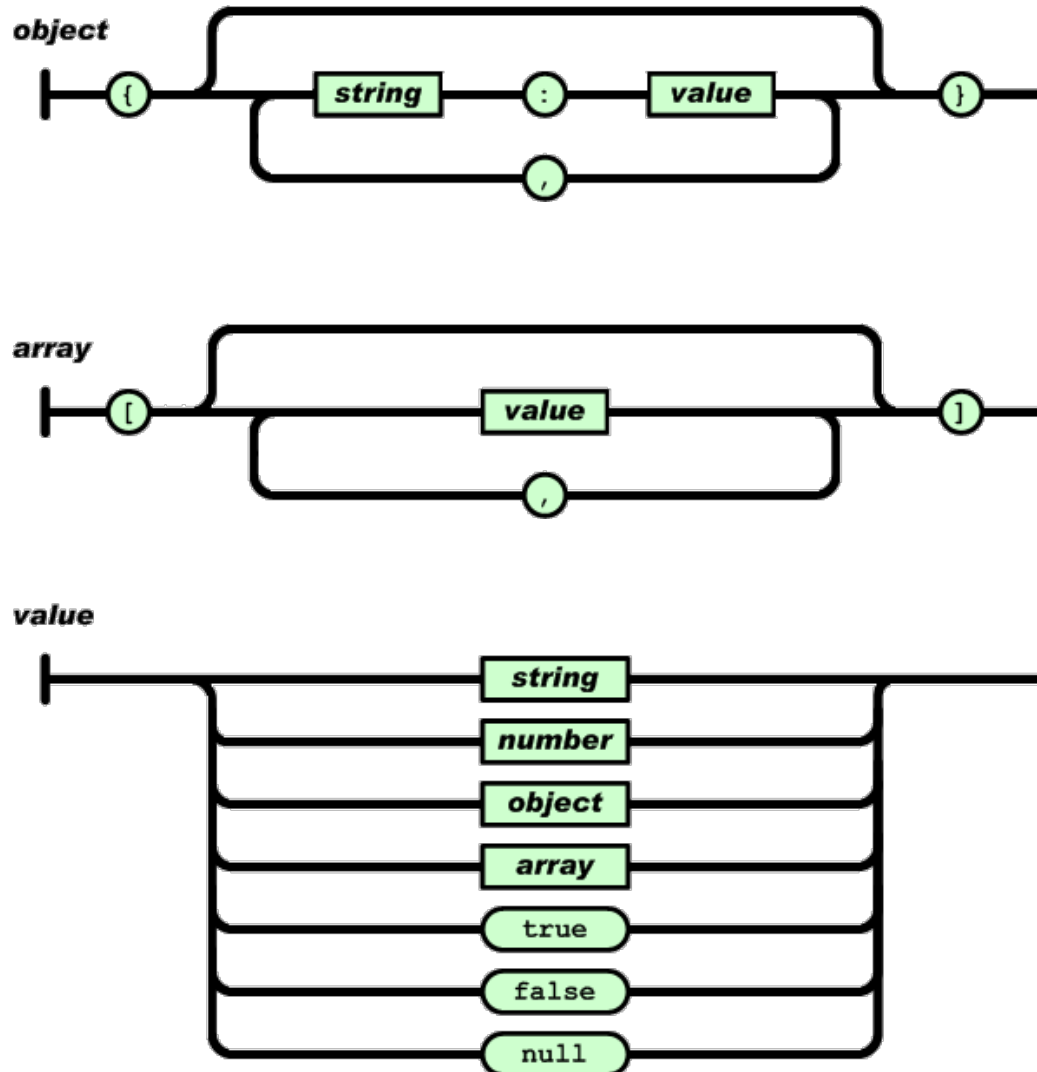
- Lightweight data-interchange format
- Easy for humans to read and write
- Easy for machines to parse and generate
- Not tied tied to JavaScript or Web

Example

```
{
  "firstName": "John",
  "lastName" : "Smith",
  "age"      : 25,
  "address"  :
    {
      "streetAdr" : "21 2nd Street",
      "city"      : "New York",
      "state"     : "NY",
      "zip"       : "10021"},
  "phoneNumber":
    [
      {
        "type" : "home",
        "number": "212-555-1234"},
      {
        "type" : "fax",
        "number" : "646-555-4567"} ]
}
```

- This is a JSON object with five **key-value pairs**
- Objects are wrapped by curly braces
- There are no object IDs
- Keys are strings
- Values are numbers, strings, objects or arrays
- Arrays/lists are wrapped by square brackets

Simple BNF



Evaluation: JSON is ...

- Simpler and more compact than XML
 - No closing tags
 - XML parsing is hard because of its complexity
 - Compressed the two are similar in size
- A better fit for OO systems than XML
- Less extensible than XML
- Widely preferred for simple data exchange

JSON is Simple

- Less syntax, no semantics
- Schemas? We don't need no stinkin schemas!*
- Transforms? Write your own

JSON Schema

- <https://json-schema.org/>
- [IETF draft](#), 3/2018
- Provide annotations
- Specifies
 - Possible properties
 - Required properties
 - Value types
 - Value constraints
 - References

```
{  
  "latitude": 48.858093,  
  "longitude": 2.294694  
}
```

```
{ "id": "http://ex.com/geo-location.schema.json",  
  "$schema": "http://json-schema.org/draft-  
07/schema#",  
  "title": "Longitude and Latitude Values",  
  "description": "A geographical coordinate.",  
  "required": [  
    "latitude",  
    "longitude" ],  
  "type": "object",  
  "properties": {  
    "latitude": {  
      "type": "number",  
      "minimum": -90,  
      "maximum": 90 },  
    "longitude": {  
      "type": "number",  
      "minimum": -180,  
      "maximum": 180 }  
  }  
}
```


JSON-LD

JSON-LD is a W3C recommendation for representing RDF data as JSON objects

```
{ "@context": {  
  "name": "http://xmlns.com/foaf/0.1/name",  
  "homepage": {  
    "@id": "http://xmlns.com/foaf/0.1/workplaceHomepage",  
    "@type": "@id"  
  },  
  "Person": "http://xmlns.com/foaf/0.1/Person"  
},  
"@id": "http://me.markus-lanthaler.com",  
"@type": "Person",  
"name": "Markus Lanthaler",  
"homepage": "http://www.tugraz.at/"  
}
```

Many popular systems use JSON

- [MongoDB](#) is an open-source database for JSON objects
 - Very popular [NoSQL](#) database
 - A NoSQL DB is one that uses a model not based on relational tables
- [Elastic Search](#) is a popular, scalable information retrieval engine that uses JSON as its native representation

Example: JSON in Python

```
>>> import json
>>> x = json.load(open('example.json'))
>>> x
{'lastName': u'Smith', u'age': 25, u'phoneNumber': [{u'type': u'home',
u'number': u'212-555-1234'}, {u'type': u'fax', u'number': u'646-555-4567'}],
u'firstName': u'John', u'address': {u'streetAdr': u'21 2nd Street', u'state':
u'NY', u'zip': u'10021', u'city': u'New York'}}
>>> x['address']['state']
u'NY'
>>> print json.dumps(x, sort_keys=True, separators=(',', ':'), indent=2)
{"address":{"city":"New York",
"state":"NY",
"streetAdr":"21 2nd Street",
"zip":"10021"},
"age":25,
"firstName":"John",
"lastName":"Smith",
"phoneNumber":[
{"number":"212-555-1234",
"type":"home"},
{"number":"646-555-4567",
"type":"fax" } ] }
```

- Python's JSON package reads & writes JSON from/to files & strings
- Maps JSON objects to Python dictionaries
- Maps JSON arrays to Python lists
- Dump (write to file) and dumps (write to string) functions can do simple pretty printing

JSON vs. XML

- JSON: The Fat-Free Alternative to XML

json.org page laying out the case for JSON over XML

- Stop Comparing JSON and XML

Blog post arguing that they're very different things with their own areas of applicability

- XML ↔ JSON

There are many web tools (e.g.: [this one](#)) and software packages (e.g. [xml2dict](#)) that can convert between simple XML and JSON representations

Worse is Better?

- JSON vs. XML as an example of “Worse is Better”
- In 1989 Dick Gabriel headed a company with the best commercial version of Lisp
 - Lisp was considered by programming language experts to be much better than C
 - But C was 10x more popular than Lisp
 - Cf. today: Scheme vs. Python (w.r.t. mutable lists)
- Gabriel explained it as *worse is better*

Software that's limited, but simple to learn/use, and flexible, can be more popular for most users

Google's JSON Style Guide

- Google has style guides for JSON and other languages
- Note: its an XML document with an XSL stylesheet!

Google JSON Style Guide

Revision 0.9

Each style point has a summary for which additional information is available by toggling the accompanying arrow button that looks like this way: . You may toggle all summaries with the big arrow button.

Toggle all summaries

Table of Contents

General Guidelines	Comments Double Quotes Flattened data vs Structured Hierarchy
Property Name Guidelines	Property Name Format Key Names in JSON Maps Reserved Property Names Singular vs Plural Property Names Naming Conflicts
Property Value Guidelines	Property Value Format Empty/Null Property Values Enum Values
Property Value Data Types	Date Property Values Time Duration Property Values Latitude/Longitude Property Values
JSON Structure & Reserved Property Names	
Top-Level Reserved Property Names	apiVersion context id method params data error

HTML

XML source

XSL stylesheet

Motivation for the style guide?

- Google's style guide addresses JSON's shortcomings by specifying conventions, e.g.:
 - JSON doesn't support objects having a unique ID which is needed to support graph structures
 - The style guide has `data.id` as a reserved property for this

```
{"data": {"id": "12345"}}
```
- Later, we'll study the W3C's [JSON-LD](#) as a way to encode semantic web data using JSON
 - JSON-LD specifies conventions needed to represent RDF knowledge graphs in JSON
 - Today's search engines recognize embedded JSON-LD in web pages and use the content

JSON Summary

- JSON is preferred by most people for simple data encoding and exchange
- Most XML documents can be converted to JSON
- Shortcomings:
 - No standard schema mechanism
 - No stylesheets
 - Available datatypes are very simple
 - No semantics, e.g., no type system, cannot say spouse is a symmetric relation, grandparent = parent of parent
- Good example of *worse is better*