JSON-LD

---

**Example**

```json
{"firstName": "John", "lastName": "Smith", "age": 25, "address": {"streetAddr": "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 are wrapped by square brackets

---

**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](http://json.org)
- Started gaining traction ~2006 and now widely used
- [http://json.org](http://json.org) has more information

---

**The BNF is simple**

```
object  ::=  {
  string  ::=  "\"value\"

array   ::=  [value]
```
Evaluation

- JSON is simpler than XML and more compact
  - No closing tags, but if you compress XML and JSON the difference is not so great
  - XML parsing is hard because of its complexity
- JSON has a better fit for OO systems than XML, but not as extensible
- Preferred for simple data exchange by many
- **MongoDB** is a very popular open-source ‘NoSQL’ database for JSON objects

---

JSON-ld

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

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

---

In the beginning

```
{
  "name": "Manu Sporny",
  "homepage": "http://manu.sporny.org/",
  "image": "http://manu.sporny.org/images/manu.png"
}
```

---

A bit better

```
{
  "http://schema.org/name": "Manu Sporny",
  "http://schema.org/url": {
    "@id": "http://manu.sporny.org/"
  }
  "http://schema.org/image": {
    "@id": "http://manu.sporny.org/images/manu.png"
  }
}
```

- The '@id' keyword means 'This value is an identifier that is an IRI'
Define a context

```json
{ "@context": {
  "@id": "http://schema.org/image",
  "@type": "@id"
},
"name": "Manu Sporny",
"homepage": "http://manu.sporny.org/",
"image": "http://manu.sporny.org/images/manu.png"
}
```

1. This means that 'name' is shorthand for 'http://schema.org/name'
2. This means that 'image' is shorthand for 'http://schema.org/image'
3. This means that a string value associated with 'image' should be interpreted as an identifier that is an IRI
4. This means that 'homepage' is shorthand for 'http://schema.org/url'
5. This means that a string value associated with 'homepage' should be interpreted as an identifier that is an IRI

Reference an external context

```json
{  "@context": "http://json-ld.org/contexts/person.jsonld",
  "name": "Manu Sporny",
  "homepage": "http://manu.sporny.org/",
  "image": "http://manu.sporny.org/images/manu.png"
}
```

Add context inline

```json
{ "@context": {
  "@id": "http://schema.org/image",
  "@type": "@id"
},
"name": "Manu Sporny",
"homepage": "http://manu.sporny.org/",
"image": "http://manu.sporny.org/images/manu.png"
}
```

Making assertions about things

```json
{  "@context": {
    ...
    "Restaurant": "http://schema.org/Restaurant",
    "Brewery": "http://schema.org/Brewery"
  }
  "@id": "http://example.org/places#BrewEats",
  "@type": [ "Restaurant", "Brewery" ],
  ...
}
```
Adding a default vocabulary

```json
{
  "@context": {
    "@vocab": "http://schema.org/
  }
  "@id": "http://example.org/places#BrewEats",
  "@type": "Restaurant",
  "name": "Brew Eats"
...
}
```

Mixing vocabularies

```json
{
  "@context": {
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "foaf": "http://xmlns.com/foaf/0.1/",
    "foaf:homepage": { "@type": "@id" },
    "picture": { "@id": "foaf:depiction", "@type": "@id" }
  },
  "@id": "http://me.markus-lanthaler.com/",
  "@type": "foaf:Person",
  "foaf:name": "Markus Lanthaler",
  "foaf:homepage": "http://www.markus-lanthaler.com/",
  "picture": "http://twitter.com/account/profile_image/markuslanthaler"
}
```

Embedding other objects

```json
{
  ...
  "name": "Manu Sporny",
  "foaf:knows": {
    "@type": "Person",
    "name": "Gregg Kellogg",
  }
  ...
}
```

Google looks for JSON-LD

- Google looks for and uses some JSON-LD markup (e.g., for organizations)
- Put a JSON-LD object in the head of a web page wrapped with script tags:
  ```html
  <script type="application/ld+json">
  {...}
  </script>
  ```
Conclusion

- JSON-LD is a good solution to putting blocks of semantic data on web pages
- It’s aimed at publish linked data, not ontologies, i.e., ABOX not TBOX
- Tools are available for extracting their content as RDF triples
- Search companies are beginning to look for and JSON-LD in web pages that uses vocabularies they understand (i.e., schema.org)
- Look for more of this in the future
http://json-ld.org/playground/