



SDXML VT2024

Models and languages for semi-structured data and XML

JSON & JSON Schema

nikos dimitrakas
nikos@dsv.su.se
08-161295

Corresponding reading

Relevant web pages: JSON, JSON Schema

Section 30.3.11 of Database Systems (Connolly, Begg) 6th edition



JSON

- **JavaScript Object Notation**
- **Standard since 2013**
 - ISO since 2017
- **JavaScript (ECMAScript)**
- **Similar to SSD expressions**
- **Case-sensitive**
- **Indentation and formatting are ignored.**
- **Is used mainly within:**
 - Integrations
 - Data transfer to JavaScript applications

JSON - values

- **Object (basically a map)**
 - inside {}
- **Array (comma-separated list of values)**
 - inside []
- **String**
 - inside ""
- **Number**
- **true**
- **false**
- **null**

JSON - example

```
{  
  "title" : "Best Songs",  
  "year" : 2023,  
  "songs" : ["Great Summer", "Tonight", "Everywhere"],  
  "producer" : {"firstname" : "Jim", "lastname" : "Baker"},  
  "available" : true  
}
```

JSON - Object

- **Label-value pairs**
 - Order should not matter
 - Labels should be unique

```
{  
  "title" : "Best Songs",  
  "year" : 2023,  
  "song" : "Great Summer",  
  "song" : "Tonight",  
  "song" : "Everywhere"  
}
```



Should be avoided

JSON - Array

- **Comma-separated list**
 - Order is important
 - Items can be objects, strings, numbers, arrays, true, false, null

```
[  
  {"title" : "Best Songs", "year" : 2023},  
  ["Great Summer", "Tonight", "Everywhere"],  
  "Jim",  
  "Baker",  
  true,  
  3  
]
```

JSON - Escaping

- **Some characters must be escaped**
 - " and \ inside strings
 - Example: `"\"hello\""`

JSON Schema

- **Uses JSON syntax**
- **Defines**
 - Structure
 - Data types
 - Other restrictions
- **Since 2011**
- **A JSON Schema is a JSON object**

JSON Schema

- **Uses JSON syntax**
- **Defines**
 - Structure
 - Data types
 - Other restrictions
- **Since 2011**
- **A JSON Schema is a JSON object**

JSON Schema

- **Schema keywords**
 - **"\$schema"** defines the JSON Schema version
 - » **"https://json-schema.org/draft/2020-12/schema"**
 - **"\$id"** defines a URI for the schema
 - » **"http://dsv.su.se/SDXML/jsonschema/example"**

```
{  
  "$schema" : "https://json-schema.org/draft/2020-12/schema",  
  "$id" : "http://dsv.su.se/SDXML/jsonschema/example"  
}
```

JSON Schema - annotations

- **Schema annotations**

- **"title"**
- **"description"**

```
{  
  "$schema" : "https://json-schema.org/draft/2020-12/schema",  
  "$id" : "http://dsv.su.se/SDXML/jsonschema/example",  
  "title" : "CD",  
  "description" : "A music CD with songs",  
}
```

JSON Schema - type

- **Validation keywords**

- **"type"**
 - » **"object", "array", "string", "number", "integer", "boolean", "null"**
 - » **array of unique of the above values ["string", "boolean"]**
 - means either string or boolean

```
{  
  "$schema" : "https://json-schema.org/draft/2020-12/schema",  
  "$id" : "http://dsv.su.se/SDXML/jsonschema/example",  
  "title" : "CD",  
  "description" : "A music CD with songs",  
  "type" : "object"  
}
```

JSON Schema - enum, const

- **Validation keywords (instead of "type")**

- **"enum"**

- » array of values of any type

- Should be unique
 - Should contain at least one value

- **"const"**

- » single value

"enum" : ["Monday", "Hello", 0, 100, {"n":9, "x" : [true, null]}, 0.5, [1,2,3]]

"const" : "SDXML"

JSON Schema - Objects

- **Validation keywords**

- **"properties"**

- » An object defining the properties and their types

- **"required"**

- » Array of strings (property names)

```
{  
  "$schema" : "https://json-schema.org/draft/2020-12/schema",  
  "$id" : "http://dsv.su.se/SDXML/jsonschema/example",  
  "title" : "CD",  
  "description" : "A music CD with songs",  
  "type" : "object",  
  "properties" : {"title" : {"type" : "string"}, "year" : {"type" : "integer"}},  
  "required" : ["title", "year"]  
}
```

JSON Schema - Objects

- **Validation keywords**

- **"dependentRequired"**
 - » Require other properties if another exists

```
{  
  "type" : "object",  
  "properties" : {"name" : {"type" : "string"},  
                  "height" : {"type" : "integer"},  
                  "width" : {"type" : "integer"}},  
  "required" : ["name"],  
  "dependentRequired" : {"height" : ["width"]},  
}
```

if height exists, width is required to exist, but not the other way.

JSON Schema - Objects

- **Validation keyword**

- **"additionalProperties"**
 - » false, to disallow additional properties, default everything allowed
 - » Allowed type or types

"additionalProperties" : false

"additionalProperties" : { "type": "string" }

"additionalProperties" : { "type": ["string", "number"] }

JSON Schema - Arrays

- **Validation keywords**

- **"items"**
 - » Type of values allowed
- **"maxItems"**
 - » Integer value, default unlimited
- **"minItems"**
 - » Integer value, default 0
- **"uniqueItems"**
 - » true or false, default false

```
{  
  "type" : "array",  
  "items" : {"type" : "integer"},  
  "uniqueItems" : true,  
  "minItems" : 1  
}
```

JSON Schema - Strings

- **Validation keywords**

- **"maxLength"**
 - » Integer value, default unlimited
- **"minLength"**
 - » Integer value, default 0
- **"pattern"**
 - » Regular expression (string)

```
{  
  "type" : "string",  
  "minLength" : 5,  
  "maxLength" : 5  
}
```

JSON Schema - Numbers

- **Validation keywords**

- "maximum", "exclusiveMaximum"
 - » Numeric value, default unlimited
- "minimum", "exclusiveMinimum"
 - » Numeric value, default unlimited
- "multipleOf"
 - » Numeric value greater than 0, default no restriction

```
{  
  "type" : "number",  
  "minimum" : 0,  
  "maximum" : 100,  
  "multipleOf" : 5  
}
```

JSON Schema - combinations

- **Validation keywords**

- "allOf", "anyOf", "oneOf"
 - » Valid according to AND, OR, XOR
- "not"
 - » Invalid

```
"anyOf": [ { "type" : "string", "maxLength" : 5 },  
           { "type" : "number", "minimum" : 0 } ]
```

```
"allOf": [ { "type" : "integer", "maximum" : 5 },  
           { "enum": [1, 3, 5, 7, 9] } ]
```

```
"not": { "type" : "string", "maxLength" : 5 }
```

JSON Schema - References

- Schema keyword
 - "\$ref"
 - » refers to an "\$id"

```
{
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "$id" : "http://dsv.su.se/SDXML/jsonschema/country",
  "type" : "object",
  "properties" : {"name" : {"type" : "string"}, "population" : {"type" : "integer"}},
  "required" : ["name"]
}

{
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "$id" : "http://dsv.su.se/SDXML/jsonschema/example",
  "type" : "object",
  "properties" : {"title" : {"type" : "string"}, "year" : {"type" : "integer"},
    "country" : {"$ref" : "http://dsv.su.se/SDXML/jsonschema/country"}},
  "required" : ["title", "year"]
}
```

Linking JSON to JSON Schema

- No explicit link to the schema in the JSON object
- Can only be done in the application

Validators

- **JSON**

- <https://jsonformatter.curiousconcept.com/>
- Formats and validates (checks syntax)
(equivalent to XML being well-formed)

- **JSON Schema**

- <https://www.jsonschemavalidator.net>
- Validates a JSON value against a JSON Schema
(equivalent to XML being valid)

- **A JSON Schema is a JSON object and can be validated (syntax checked) as such**

- **JSON Schema validation**

- <https://www.json-schema-linter.com>
- Checks a JSON Schema against the meta schema

JSON to/from XML

- **XML representation of JSON**

- Part of the XPath/XQuery 3 standard
- conversion functions `xml-to-json` and `json-to-xml`

```
{
  "title": "Best Songs",
  "year": 2023,
  "songs": [
    "Great Summer",
    "Tonight",
    "Everywhere"
  ],
  "producer": {
    "firstname": "Jim",
    "lastname": "Baker"
  },
  "available": true
}
```

```
<map xmlns="http://www.w3.org/2005/xpath-functions">
  <string key="title">Best Songs</string>
  <number key="year">2023</number>
  <array key="songs">
    <string>Great Summer</string>
    <string>Tonight</string>
    <string>Everywhere</string>
  </array>
  <map key="producer">
    <string key="firstname">Jim</string>
    <string key="lastname">Baker</string>
  </map>
  <boolean key="available">true</boolean>
</map>
```



What to do next

- **Quiz about JSON & JSON Schema (Quiz 3)**