JSON: introduction

Mario G.C.A. Cimino

a) JSON Example

```
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```

Editor online to indent JSON:
https://jsoneditoronline.org

b) XML Example:

```
<employees>
  <employee>
    <firstName>John</firstName> <lastName>Doe</lastName>
  </employee>
  <employee>
    <firstName>Anna</firstName> <lastName>Smith</lastName>
  </employee>
  <employee>
    <firstName>Peter</firstName> <lastName>Jones</lastName>
  </employee>
</employees>
```

c) XML Example (compact):

```
<employees>
  <employee firstName="John" lastName="Doe" />
  <employee firstName="Anna" lastName="Smith" />
  <employee firstName="Peter" lastName="Jones" />
</employees>
```

JSON, JSON-Schema: introduction

JSON (JavaScript Object Notation) is a format (alternative to XML) to structure objects, in the form of pair attribute-value. W.r.t. to XML it is more compact, with simple parsing rules.

JSON for data exchange and store. E.g. "MongoDB", a NoSQL db to archive documents in a semistructured form.

JSON schema (https://json-schema.org) for validation of JSON documents.

a) JSON Types

number (e.g. "3.14"), string (e.g. "Antonio"), boolean ("true" or "false")

b) JSON Object (made by eterogeneous properties)

"player": *{ "email": "mario.rossi@gmail.com",* "scores": *"340" }*

"objectname": { "propertyname": "propertyvalue", "propertyname": "propertyvalue", … }

c) JSON Array (made by elements with same structure)

*"*cars":*["Ford", "BMW", "Fiat"]*

"arrayname": ["first element", "second element", "third element"]

JSON, JSON-Schema: introduzione

identifier of the json schema person

version of the json-schema vocabulary to express the schema

Person schema:{

```
 "$id": "https://example.com/person.schema.json",
 "$schema": "http://json-schema.org/draft-07/schema#",
 "title": "Person",
 "type": "object",
 "properties": {
  "firstName": {
    "type": "string",
    "description": "The person's first name."
  },
  "lastName": {
    "type": "string",
    "description": "The person's last name."
  },
  "age": {
    "description": "Age in years which must be equal to or greater than zero.",
    "type": "integer",
    "minimum": 0
  }
 }
}
```

Person instance:{

```
 "firstName": "John",
 "lastName": "Doe",
 "age": 21
}
```

JSON, JSON-Schema: introduction

geographical-location schema:

```
{
  "$id": "https://example.com/geographical-location.schema.json",
  "$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
    }
  }
}
```

geographical-location instance:

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

JSON, JSON-Schema: introduction

fruit-and-vegetables schema:

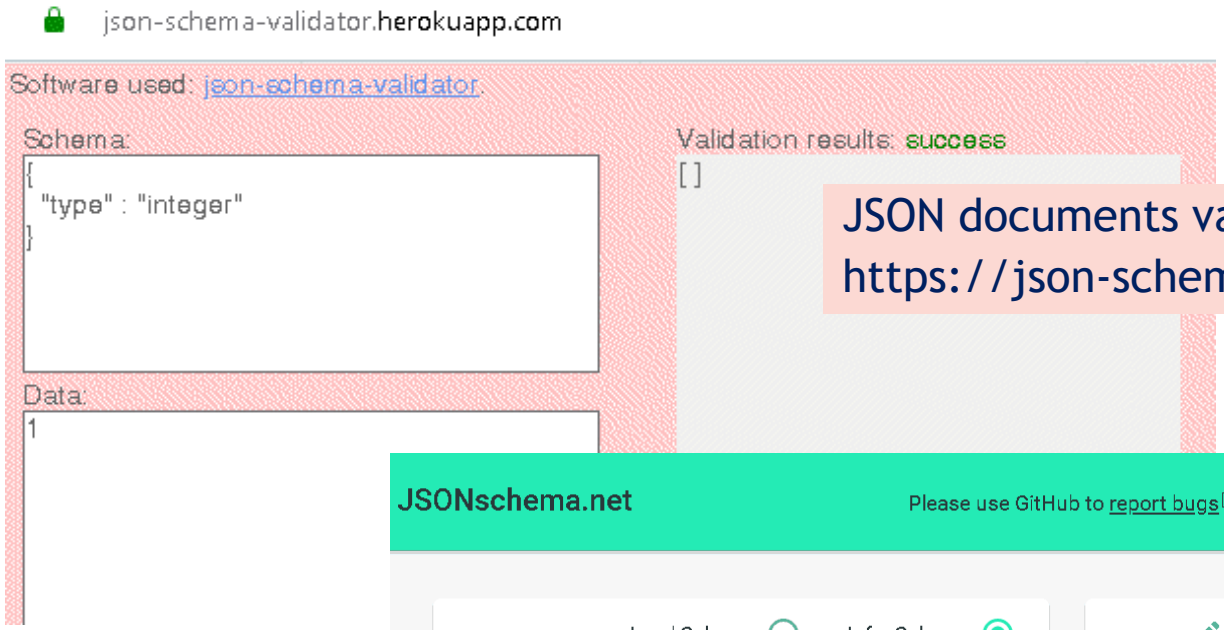

```
{
 "$id": "https://example.com/vegetables.schema.json",
 "$schema": "http://json-schema.org/draft-07/schema#",
 "description": "Fruits and vegetables of interest",
 "type": "object",
 "properties": {
  "fruits": {
   "type": "array",
   "items": {
    "type": "string"
   }
  },
  "vegetables": {
   "type": "array",
   "items": { "$ref": "#/definitions/veggie" }
  }
 },
 "definitions": {
  "veggie": {
   "type": "object",
   "required": [ "veggieName", "veggieLike" ],
   "properties": {
    "veggieName": {
     "type": "string",
     "description": "The name of the vegetable."
    },
    "veggieLike": {
     "type": "boolean",
     "description": "Do I like this vegetable?"
    }
   }
  }
 }
}
```

fruit-and-vegetables instance:{

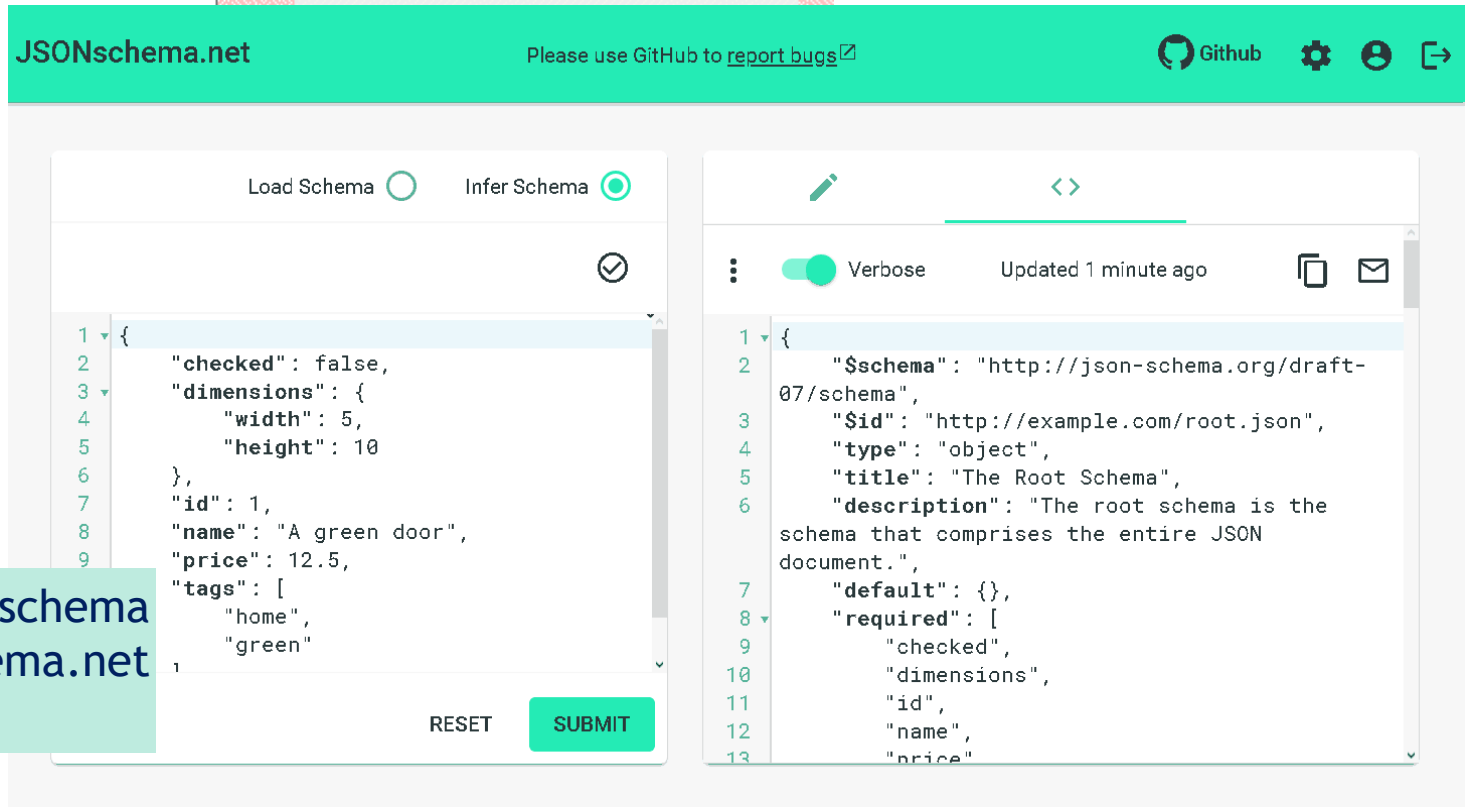
```
 "fruits": [ "apple", "orange", "pear" ],
 "vegetables": [
  {
   "veggieName": "potato",
   "veggieLike": true
  },
  {
   "veggieName": "broccoli",
   "veggieLike": false
  }
 ]
}
```

reference to a sub-schema

JSON e JSON-Schema: introduction



JSON documents validation:
https://json-schema-validator.herokuapp.com



Infer a-JSON-schema
https://jsonschema.net