MODELOS DE DADOS
JAVA SCRIPT OBJECT NOTATION

(JSON)

dwdm
Desenvolvimento para a Web e Dispositivos Móveis

2018/2019
FILIPE SÁ FILIPE.SA@ESTGV.IPV.PT
CARLOS CUNHA

JSON - INTRODUÇÃO

- JSON: Java Script Object Notation
  - Modelo para armazenamento e transmissão de informações em formato texto
  - Capacidade de estruturar dados
  - Estruturação dos dados mais compacta que a conseguida pelo modelo XML
  - Ideal para transmitir grandes volumes de dados

JSON - SINTAXE

- Define a forma de representar um valor
- É derivada da forma utilizada pelo JavaScript para representar informações
  - Um par nome/valor deve ser representado por:
    - Nome entre aspas duplas
    - Seguido de dois pontos
    - Seguido do valor
  - Exemplo: como representar o ano 2016?
    - "ano":2016

JSON – TIPOS DE DADOS

- Os valores representados podem possuir os seguintes tipos de dados:
  - numérico (inteiro ou real, positivo/negativo)
    - "altura": 1.77
    - "idade": 32
    - "temperatura": -3
  - booleano
    - "aprovado": true
  - string
    - "site": "www.di.estgv.ipv.pt"

JSON – TIPOS DE DADOS COMPLEXOS

- A partir dos tipos de dados simples é possível construir tipos complexos: array e objecto
  - Array
    - Delimitado por []
    - Cada elemento é separado por ,
    - Exemplo de um array de strings:

      ["AI3", "IRSC", "RC1", "BD"]
    - Exemplo de uma matriz de inteiros:

```
[
    [2, 5],
    [21, -5],
    [232, 435]
]
```

JSON – TIPOS DE DADOS COMPLEXOS

- Exemplo: um OBJECTO pode representar dados sobre um filme

```
{
    "filme":
    {
        "titulo": “Era uma vez na América",
         "resumo": “filme de gangsters”,
        "ano": 1984,
        "genero": ["aventura", "drama", "acçao"]
    }
}
```

JSON: TIPOS DE DADOS COMPLEXOS

- Exemplo: um array de OBJECTOS pode representar dados sobre vários filmes:

```
{
    "filmes":
    [
        {
        "titulo": “Era uma vez na América",
        "resumo": “filme de gangsters”,
        "ano": 1984,
        "genero": ["aventura", "drama", "acçao"]
        },
        {
        "titulo": "Duelo no Texas",
        "resumo": "velho oeste”,
        "ano": 1963,
        "genero": ["western"]
        }
    ]

}
```

OBJ 1

OBJ 2

JSON: VALORES NULOS

- Os valores nulos são representados pela palavra `null`

```
"site": null
```

JSON E XML: SEMELHANÇAS

- Formato texto
- Auto-descritos
- Capazes de representar informação complexa
  - Objetos compostos / elementos agregadores
  - Relações de hierarquia
  - Atributos multivalor
  - Arrays
- Independentes da linguagem, podem ser usados em qualquer linguagem de programação, através de APIs específicos

JSON E XML: DIFERENÇAS

- JSON não é uma linguagem de marcação
- XML possui um conjunto de tecnologias associadas – Xpath, Xquery, XSLT; o JSON não possui
- O JSON representa a informação de forma mais compacta -> mais vocacionado para transmissão/troca de dados
- XML mais vocacionado para armazenamento.

XML E JSON: REPRESENTAÇÃO DE DADOS

ELEMENTOS = OBJECTOS

- **XML**

```
<nome>Filipe Sá</nome>
```

- **JSON**

```
{
   "nome": “Filipe Sá"
}
```

- **XML**

```
<pessoa>
  <nome>Filipe Sá</nome>
  <idade>39</idade>
</pessoa>
```

- **JSON**

```
{
  "pessoa":{
     "nome": “Filipe Sá",
     "idade": 39
   }
}
```

XML E JSON: REPRESENTAÇÃO DE DADOS

ELEMENTOS = OBJECTOS / ARRAYS

- **XML**

```
<lista>
        <item>1</item>
        <item>2</item>
        <item>3</item>
        <item>4</item>
        <item>5</item>
</lista>
```

- **JSON**

```
{
   "lista": {
      "item": [1,2,3,4,5]
   }
}
```

XML E JSON: REPRESENTAÇÃO DE DADOS

ELEMENTOS = OBJECTOS / ARRAYS

- XML

```
<pessoas>
    <pessoa>
        <nome>Filipe</nome>
        <idade>38</idade>
    </pessoa>
    <pessoa>
        <nome>Carlos</nome>
        <idade>35</idade>
    </pessoa>
</pessoas>
```

- **JSON**

```
{
  "pessoas": {
     "pessoa":[
       {
          "nome":“Filipe",
          "idade":38
       },
       {
          "nome":“Carlos"",
          "idade":35
       }
     ]
   }
}
```

XML E JSON: REPRESENTAÇÃO DE DADOS

ELEMENTOS E ATRIBUTOS

- XML

```
<nome nac="pt">Filipe</nome>
```

- **JSON**

```
{
   "nome":
        {
        "nac":"pt",
        “nome": “Filipe"
        }
}
```

- **XML**

```
<pessoa tipo="singular">
  <nome>Filipe</nome>
  <idade>38</idade>
</pessoa>
```

- **JSON**

```
{
  "pessoa":{
        "tipo": "singular",
        "nome": “Filipe"
        "idade": 38
    }
}
```

XML E JSON: REPRESENTAÇÃO DE DADOS

ELEMENTOS E ATRIBUTOS

- **XML**

```
<pessoas ano = "2014">
  <pessoa id="001">
     <nome tipo="apelido">Sá</nome>
     <idade>38</idade>
  </pessoa>
  <pessoa id="002">
    <nome tipo="proprio">Carlos</nome>
    <idade>35</idade>
  </pessoa>
</pessoas>
```

- **JSON**

```
{
  "pessoas":{
    "ano":"2014",
    "pessoa”:[
       {
          "id":"001",
          "nome":{
             “tipo":"apelido",
             “ultimo":“Sá"
           },
           "idade":38
       },
       {
         “id”:"002",
         "nome”:{
            "tipo":"proprio",
            “primeiro":“Carlos"
          },
          "idade":35
       }   ] } }
```

dwdm
Desenvolvimento para a Web e Dispositivos Móveis

XML E JSON: EXERCÍCIO

REPRESENTE O FICHEIRO XML EM NOTAÇÃO JSON

- XML

```
<pais>
        <nome>Portugal</nome>
        <pop>10</pop>
        <capital>Lisboa</capital>
        <simbolo>PT</simbolo>
</pais>
```

- **JSON**

```
{
  "pais":{
    "nome": "Portugal",
    "pop": 10,
    "capital": "Lisboa",
    "simbolo": "PT"
    }
}
```

XML E JSON: EXERCÍCIO

REPRESENTE O FICHEIRO XML EM NOTAÇÃO JSON

- **XML**

```
<paises>
    <pais>
        <nome>Portugal</nome>
        <pop>10</pop>
        <capital>Lisboa</capital>
        <simbolo>PT</simbolo>
    </pais>
   <pais>
        <nome>Espanha</nome>
        <pop>35</pop>
        <capital>Madrid</capital>
        <simbolo>ES</simbolo>
    </pais>
</paises>
```

- **JSON**

```
{
 "paises":{
    "pais":[
      {
        "nome": "Portugal",
        "pop": 10,
        "capital": "Lisboa",
        "simbolo": "PT"
      },
      {
        "nome": "Espanha",
        "pop": 35,
        "capital": "Madrid",
        "simbolo": "ES"
      }
    ]
  }
}
```

XML E JSON: EXERCÍCIO

REPRESENTE O FICHEIRO XML EM NOTAÇÃO JSON

- XML

```
<paises>
    <pais id="p001">
        <nome>Portugal</nome>
        <pop uni="milhoes">10</pop>
        <capital>Lisboa</capital>
        <simbolo>PT</simbolo>
    </pais>
   <pais id="p002">
        <nome>Espanha</nome>
        <pop uni="milhoes">35</pop>
        <capital>Madrid</capital>
        <simbolo>ES</simbolo>
   </pais>
</paises>
```

- **JSON**

```
{
 "paises":{
    "pais":[
      {
        "id": "p001",
        "nome": "Portugal",
        "pop": {
            "uni": "milhões”,
            "text": "10”
          }
        "capital": "Lisboa",
        "simbolo": "PT"
      },
      {
        ...
      }
    ]
  }
}
```

XML E JSON: EXERCÍCIO

REPRESENTE O FICHEIRO JSON EM XML

- **JSON**

```
{"catalogo”: {
"livro”: [
      {
        "categoria": "Culinária",
        "titulo”:{
            "lang":"pt",
            "text":"Receitas"
        },
        "autores":{
            "autor":
            ["Maria Silva","Joana Mota"]
        },
        "ano”:"2005",
        "preco": {
            "moeda": "euro",
            "desc":"10%",
            “valor":"30.00"
        }
      },
      {...}
      }
   ]
  }
}
```

- XML

```
<catalogo>
      <livro categoria="Culinária">
         <titulo lang=“pt”>Receitas</titulo>
                  <autores>
                              <autor>Maria Silva</autor>
                              <autor>Joana Mota</autor>
                  </autores>

                  <ano>2015</ano>

                  <preco moeda="euro" desc="10%">30.00</preco>

               </livro>
      <livro categoria="…">
         <titulo lang=“pt”>…</titulo>
                  <autores>
                              <autor>…</autor>
                              <autor>…</autor>
                  </autores>

                  <ano>2015</ano>

                  <preco moeda="..” desc="... ">…</preco>

               </livro>

</catalogo>
```

dwdm
Desenvolvimento para a Web e Dispositivos Móveis

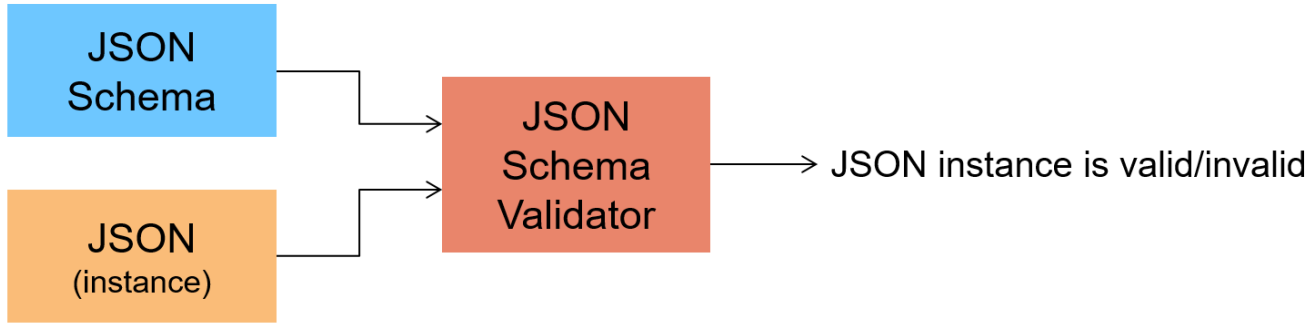
JSON SCHEMA

- Tecnologia para descrever e validar a estrutura dos documentos JSON
- Fornece uma descrição da estrutura sobre um qualquer documento JSON
- Documentos JSON que forem especificados com um SCHEMA JSON podem ser estruturalmente validados confrontado o seu SCHEMA
- Similar aos esquemas XML

JSON SCHEMA

Existem duas especificações no que respeita ao JSON-Schema

- http://json-schema.org/latest/json-schema-core.html
- http://json-schema.org/latest/json-schema-validation.html



XML SCHEMA - LIVRO

```
<xs:element name="Book">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Title" type="xs:string" />
      <xs:element name="Authors">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Author" type="xs:string" maxOccurs="5"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Date" type="xs:gYear" />
      <xs:element name="Publisher" minOccurs="0">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="Springer" />
            <xs:enumeration value="MIT Press" />
            <xs:enumeration value="Harvard Press" />
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

JSON SCHEMA - LIVRO

```
{
    "$schema": http://json-schema.org/draft-04/schema",
    "type": "object",
    "properties": {
      "Book": {
        "type": "object",
        "properties": {
          "Title": {"type": "string"},
          "Authors": {"type": "array", "minItems": 1, "maxItems": 5, "items": { "type": "string" }},
          "Date": {"type": "string", "pattern": "^[0-9]{4}$"},
          "Publisher": {"type": "string", "enum": ["Springer", "MIT Press", "Harvard Press"]}
        },
        "required": ["Title", "Authors", "Date"],
        "additionalProperties": false
      }
    },
    "required": ["Book"],
    "additionalProperties": false
}
```

JSON SCHEMA - LIVRO

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "type": "object",
  "properties": {
    "Book": {
      "type": "object",
      "properties": {
        "Title": {"type": "string"},
        "Authors": {"type": "array", "minItems": 1, "maxItems": 5, "items": { "type": "string" }},
        "Date": {"type": "string", "pattern": "^[0-9]{4}$"},
        "Publisher": {"type": "string", "enum": ["Springer", "MIT Press", "Harvard Press"]}
      },
      "required": ["Title", "Authors", "Date"],
      "additionalProperties": false
    }
  },
  "required": ["Book"],
  "additionalProperties": false
}
```

```
<xs:element name="Authors">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Author" type="xs:string" maxOccurs="5"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```
<xs:element name="Title" type="xs:string" />
```

```
<xs:element name="Date" type="xs:gYear" />
```

```
<xs:element name="Publisher" minOccurs="0">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Springer" />
      <xs:enumeration value="MIT Press" />
      <xs:enumeration value="Harvard Press" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

dwdm
Desenvolvimento para a Web e Dispositivos Móveis