

---

# XML Schema

Processamento Estruturado de  
Documentos 2003

By jcr

## Motivação

---

- Sintaxe XML
- Suporte para NameSpaces
- data types, abstract data types, references.
- reuse, extension, restriction, ...

# Exemplo: O Poema

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" >
  <xs:complexType name="Tquadra">
    <xs:sequence>
      <xs:complexType name="Tverso" mixed="true">
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element name="nome" type="xs:string"/>
          <xs:element name="lugar" type="xs:string"/>
        </xs:choice>
      </xs:complexType>
      <xs:sequence>
        <xs:element name="verso" type="Tverso"/>
        <xs:element name="verso" type="Tverso"/>
        <xs:element name="verso" type="Tverso"/>
      </xs:sequence>
    </xs:complexType>
  </xs:schema>
... </xs:schema>
```

06-10-2003

jcr - ped2003 - 3

# Tipos Primitivos

- String
- Float
- Decimal
- Date
- ...
  - Demonstrar no XMLSpy

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

06-10-2003

jcr - ped2003 - 4

# Tipos Derivados

---

- Derivam dos tipos primitivos
- integer (decimal), long (integer), ...
- Por restrição
- Por extensão
  - Demonstrar no XML Spy

# Simple and Complex Types

---

- Os tipos de dados em XML Schema ou são “simple” ou “complex”.
- Um tipo “simple” é um dos tipos básicos: string, date, float, double, timeDuration, ...
- Um tipo “simple” restringe o texto que pode aparecer no valor dum atributo ou no conteúdo dum elemento textual.
- Um elemento que tenha atributos ou elementos filho é do tipo “complex”.
- Um tipo “complex” restringe o conteúdo dum elemento relativamente aos atributos e elementos filho que pode ter.

## Exemplo: complex Type

---

```
<xs:complexType name="Tquadra">
  <xs:sequence>
    <xs:element name="verso" type="Tverso"/>
    <xs:element name="verso" type="Tverso"/>
    <xs:element name="verso" type="Tverso"/>
    <xs:element name="verso" type="Tverso"/>
  </xs:sequence>
</xs:complexType>
```

**Tverso** é uma referência a um “complex type” definido algures no schema

**Tquadra** é um complex type com nome (por oposição a um anónimo)

## Scope dos elemento

---

- Um elemento pode ser declarado como **global** ou como **local**.
- Um elemento global é declarado como sendo filho de <schema>.
- Um elemento local é declarado algures na estrutura do Schema.
- Os elementos globais podem ser reutilizados por referência.

## Elementos globais

```
<xs:element name="poema">  
  <xs:complexType>  
    <xs:element name="quadra">  
      <xs:complexType>  
        <xs:sequence>  
          <xs:element ref = "Tverso"/>  
          <xs:element ref = "Tverso"/>  
          <xs:element ref = "Tverso"/>  
          <xs:element ref = "Tverso"/>  
        </xs:sequence>  
      </xs:complexType>  
    </xs:element>  
  </xs:complexType>  
</xs:element>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>  
<quadra>  
  <verso>Olha,<nome>Daisy</nome>: quando eu morrer tu hás-de</verso>  
  <verso>dizer aos meus amigos aí de <nome>Londres</nome>,</verso>  
  <verso>embora não o sintas, que tu escondes</verso>  
  <verso>a grande dor da minha morte. Irás de</verso>  
</quadra>
```

Instância Válida

## Cardinalidade

```
<xs:complexType name="Tquadra">  
  <xs:sequence>  
    <xs:element name="verso" type="Tverso"  
      minOccurs="4" maxOccurs="4"/>  
  </xs:sequence>  
</xs:complexType>
```

A cardinalidade pode ser restringida recorrendo aos atributos: `minOccurs` e `maxOccurs`

# Valores para elementos

---

```
<xs:element name="email" type="xs:string"
            default="jcr@di.uminho.pt"/>
```

Valor por omissão

```
<xs:element name="supervisor" type="xs:string"
            fixed="José Carlos Ramalho"/>
```

Valor fixo

# Atributos

---

- Os atributos declaram-se recorrendo ao elemento “<attribute>”
- Um atributo é sempre do tipo “simple”

```
<xs:attribute name="tipo"
              type="xs:string"
              use="optional"
              default="soneto"/>
```

use: (optional | default | required | fixed )

# Restrição de tipos primitivos

---

```
<xs:simpleType name="dia" base = "xs:integer">  
  <xs:minInclusive value="1"/>  
  <xs:maxInclusive value="31"/>  
</xs:simpleType>
```

Restrição de domínio

```
<xs:simpleType name="telefone" base = "xs:integer">  
  <xs:pattern value="253-\d{6}"/>  
</xs:simpleType>
```

Restrição de formato

# Restrições ou facetas

---

- minInclusive
- maxInclusive
- maxExclusive
- Length
- minLength
- maxLength
- Pattern
- Enumeration
- minExclusive
- Precision
- Scale
- Encoding
- Period
- Duration

## Restrição por enumeração

---

```
<xs:simpleType name="diaSemana" base = "xs:string">
  <xs:enumeration value="Domingo"/>
  <xs:enumeration value="Segunda"/>
  <xs:enumeration value="Terça"/>
  <xs:enumeration value="Quarta"/>
  <xs:enumeration value="Quinta"/>
  <xs:enumeration value="Sexta"/>
  <xs:enumeration value="Sábado"/>
</xs:simpleType>
```

## Elementos vazios

---

```
<xs:element name="ref">
  <xs:complexType>
    <xs:element name="imagem">
      <xs:complexType>
        <xs:attribute name="path" type="xs:string"
          use="required"/>
        <xs:attribute name="formato" type="xs:string"
          use="optional"/>
      </xs:complexType>
    </xs:element>
  </xs:complexType>
</xs:element>
```

# Exercício

---

- Especifique um Schema para relatórios.

**1ª parte do trabalho prático**