SCC0565 – Sistemas Interativos Web

Prof.: Dr. Rudinei Goularte
(rudinei@icmc.usp.br)

Introdução a XML Schema.

Introdução

- DTDs: Problemas??
  - Sintaxe diferente de XML – dificulta implementação de parsers e processamento da estrutura por aplicações
  - DTD especifica vocabulário fechado e não estensível – dificulta reuso
  - DTD NÃO permite definir tipos de dados
  - DTD possui poucos tipos pré definidos
  - DTD NÃO permite especificar quantidades
  - A única vantagem do DTD é a simplicidade

Introdução

- XML Schema: O que é?
  - É um *Standard* W3C (Outubro/2004);
  - É uma alternativa para o DTD baseada em XML;
  - Descreve a estrutura de um documento XML;
  - A linguagem XML Schema é também referenciada como XML Schema Definition (**XSD**).

Introdução

- XML Schema: O que é?
  - É extensível para futuras adições;
  - É mais rico e mais útil que os DTD's;
  - É escrito em XML;
  - Suporta tipos de dados;
  - Suporta *namespaces*.

Introdução

- Um XML Schema define:
  - elementos que podem aparecer em um documento;
  - atributos que podem aparecer em um documento;
  - quais elementos são elementos filhos;
  - o ordem dos elementos filhos;

(continua...)

Introdução

(continuação...)

- o número de elementos filhos;
- se um elemento é vazio ou pode incluir algum texto;
- os tipos de dados para elementos e atributos;
- valores default e fixos para elementos e atributos.

Namespaces

Namespaces

- Remove a ambigüidade por meio da associação de uma URI com cada aplicação XML e da anexação de um prefixo a cada elemento para indicar a qual aplicação ele pertence

Exemplo

- Prefixos para "title"
  - html:title
  - book:title
  - person:title

Namespaces

Outro exemplo

- Scalable Vector Graphics (SVG)
  - Documentos SVG podem ser embutidos em documentos HTML ou XHTML para incluir gráficos nas páginas Web

- Elementos SVG
  - title, metadata, defs, path, text, rect, circle, ellipse, line, polyline, polygon, use, image, svg g, view switch, a, altGlyphDef, script, style, symbol, marker, clipPath, mask, linearGradient, radialGradient, pattern, filter, cursor, font, animate, set, animateMotion, animateColor, animateTransform, color-profile, font-face

Namespaces

Solução

- Namespaces
- Elementos SVG
  - http://www.w3.org/2000/svg
- Elementos XHTML
  - http://www.w3.org/1999/xhtml
- XML Schema
  - http://www.w3.org/2001/XMLSchema

Atenção!

- Não há garantias que o documento na URI do namespace descreve a sintaxe usada; ou que o documento existe

"404 Not Found"

- Se existe uma URI para uma aplicação XML, então essa URI é uma boa escolha para a definição do namespace

Declarando um *Schema*

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.example.com/teste"
        xmlns:teste="http://www.example.com/teste"
        xmlns="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified"
        attributeFormDefault="unqualified">

        ...
</schema>
```

- Necessidade de um elemento raíz: <esquemaTeste>

Elemento Simples

- É um elemento XML que pode conter somente texto; não pode conter outros elementos ou atributos.
- O texto pode ser de vários tipos diferentes, incluindo tipos customizados.
- Sintaxe da definição ("declaração"):

```
<element name="title" type="string"/>
```

Elemento Simples - exemplo

Elementos XML:

```
<lastname>Smith</lastname>
<age>34</age>
<dateborn>1970-05-26</dateborn>
```

Definição em XML Schema:

```
<element name="lastname" type="string"/>
<element name="age" type="integer"/>
<element name="dateborn" type="date"/>
```

Elemento Simples – valores default e fixed

```
<element name="color" type="string"
default="red"/>

<element name="color" type="string"
fixed="red"/>

<color>red</color>
```

Tipos Pré-definidos

- Principais tipos pré-definidos. Outros podem ser obtidos no Primer (ver bibliografia)

simpleType	Exemplo/descrição
string	
byte	-1 ... 126
unsignedByte	0 ... 126
integer	-126789 ... 126789
int	-1 ... 126789675
long	-1...2678967543233
short	-1, 12678
decimal	-1.23, 0, 123.4, 1000.00

simpleType	Exemplo/descrição
float	Ponto flutuante 32-bits
double	Ponto flutuante 624-bits
boolean	true, false, 1, 0
time	13:20:00.000
date	1999-05-31
ID	Usado em atributos como nos DTDs

Tipos simples

- Novos simple types podem ser derivados a partir dos já existentes.
  - Na maioria das vezes, apenas restringe-se os valores dos simple types pré-definidos.
    - Use-se o elemento simpleType para definir o nome do novo simple type.
    - Usa-se o elemento restriction para indicar o tipo existente (base) e para identificar as "facetas" que contém o conjunto de valores.

Declarando um novo tipo simples

- Exemplo:
  - Criar um novo tipo de inteiro chamado myInteger, com valores entre 10000 e 99999 (inclusive).
    - Pode-se restringir a base do tipo inteiro (restriction base).
    - Pode-se aplicar as facetas "minInclusive" e "maxInclusive".

```
<element name="NovoNumero" type="teste:MyInteger"/>
```

```
<simpleType name="myInteger">
      <restriction base="integer">
            <minInclusive value="10000"/>
            <maxInclusive value="99999"/>
      </restriction>
</simpleType>
```

Declarando simpleType

- No link

http://www.w3.org/TR/xmlschema-0/#SimpleTypeFacets

há duas tabelas com todos os simple types e as facetas aplicáveis a cada um deles.

- As tabelas apresentam um total de 12 facetas, mas as mais importantes são minInclusive, maxInclusive (usadas no exemplo anterior), minExclusive, maxExclusive, length, enumeration e pattern.

Restrições

- Restrições são usadas para controlar valores aceitáveis para elementos XML ou atributos.

```
<element name="age">
  <simpleType>
    <restriction base="integer">
      <minInclusive value="0"/>
      <maxInclusive value="100"/>
    </restriction>
  </simpleType>
</element>
```

Restrições

- Sobre um conjunto de valores - enumeration:

```
<element name="car">
<simpleType>
  <restriction base="string">
    <enumeration value="Audi"/>
    <enumeration value="Ferrari"/>
    <enumeration value="BMW"/>
  </restriction>
</simpleType>
</element>
```

Restrições

- O mesmo exemplo poderia ser:

```
<element name="car" type="teste:carType"/>

<simpleType name="carType">
  <restriction base="string">
    <enumeration value="Audi"/>
    <enumeration value="Ferrari"/>
    <enumeration value="BMW"/>
  </restriction>
</simpleType>
```

Restrições

- Sobre uma série de valores - pattern:

```
<element name="letter">

<simpleType>
  <restriction base="string">
    <pattern value="[0-9][0-9][0-9]"/>
  </restriction>
</simpleType>

</element>
```

Restrições

- Outro exemplo com pattern:

```
<simpleType name="SKU">
  <restriction base="string">
    <pattern value="\d{3}-[A-Z]{2}"/>
  </restriction>
</simpleType>
```

Restrições

- Sobre tamanho - lenght:

```
<element name="password">

  <simpleType>
    <restriction base="string">
      <minLength value="5"/>
      <maxLength value="8"/>
    </restriction>
  </simpleType>

</element>
```

Elemento Complexo

- Um elemento complexo é um elemento XML que contém outros elementos e/ou atributos.
- Há quatro tipos:
  - Elementos vazios (empty)
  - Elementos que contêm somente outros elementos
  - Elementos que contêm somente texto
  - Elementos que contêm tanto outros elementos quanto texto

Elemento Complexo - Exemplo

- Elementos complexos XML:

```
<employee>
   <firstname>Smith</firstname>
   <lastname> John </lastname>
</employee>
```

Elemento Complexo - Exemplo

- Definição em XML Schema:

```
<element name="employee">
  <complexType>
   <sequence>
     <element name="firstname" type="string"/>
     <element name="lastname" type="string"/>
   </sequence>
 </complexType>
</element>
```

Elemento Complexo – Outro exemplo

```
<element name="employee" type="teste:personinfo"/>
<element name="student" type="teste:personinfo"/>
<element name="member" type="teste:personinfo"/>

<complexType name="personinfo">
  <sequence>
    <element name="firstname" type="string"/>
    <element name="lastname" type="string"/>
  </sequence>
</complexType>
```

Elemento Complexo Vazio

- Pode conter atributos mas não pode ter qualquer conteúdo entre as tags de abertura e fechamento.
- Elemento XML vazio:

```
<product prodid = "1345" />
```

Elemento Complexo Vazio

- Definição em XML Schema:

```
<element name="product">
  <complexType>
     <attribute name="prodid"
                type="positiveInteger"/>
 </complexType>
</element>
```

Elemento Complexo com apenas elementos

- Definição em XML Schema:

```
<element name="person">
 <complexType>
   <sequence>
     <element name="firstname" type="string"/>
     <element name="lastname"  type="string"/>
   </sequence>
 </complexType>
</element>
```

Elemento Complexo com apenas texto

```
<shoesize country="france">35</shoesize>
```

- Definição em XML Schema:

```
<element name="shoesize">
 <complexType>
  <simpleContent>
   <extension base="integer">
     <attribute name="country" type="string"/>
   </extension>
  </simpleContent>
 </complexType>
</element>
```

Elemento Complexo com conteúdo misto

- Um elemento complexo misto pode conter atributos, elementos e texto.
- Elemento XML:

```
<letter>
     Dear Mr.<name>John Smith</name>.
     Your order <orderid>1032</orderid>
     will be shipped on
    <shipdate>2001-07-13</shipdate>.
</letter>
```

Elemento Complexo com conteúdo misto

- Definição em XML Schema:

```
<element name="letter">
 <complexType mixed="true">
  <sequence>
   <element name="name" type="string"/>
   <element name="orderid"type="positiveInteger"/>
   <element name="shipdate" type="date"/>
  </sequence>
 </complexType>
</element>
```

Indicadores de Tipos Complexos

- Pode-se controlar como os elementos serão usados nos documentos com indicadores.
- Há sete tipos de indicadores:
  - **3 de Ordem**
  - **2 de Ocorrência**
  - 2 de Grupos

Indicadores de Tipos Complexos

- **Ordem:**
  - **All**
  - **Choice**
  - **Sequence**
- **Ocorrência:**
  - **maxOccurs**
  - **minOccurs**
- Grupo:
  - Group name
  - AttributeGroup name

All

- Especifica que os elementos filhos podem aparecer em qualquer ordem e devem ocorrer uma e apenas uma vez.

```
<element name="person">
  <complexType>
    <all>
      <element name="firstname" type="string"/>
      <element name="lastname" type="string"/>
    </all>
  </complexType>
</element>
```

Choice

- Especifica que tanto um elemento filho quanto o outro podem ocorrer.

```
<element name="person">
  <complexType>
    <choice>
      <element name="employee" type="employee"/>
      <element name="member"  type="member"/>
    </choice>
  </complexType>
</element>
```

Sequence

- Especifica que os elementos filhos devem aparecer em um ordem específica.

```
<element name="person">
  <complexType>
    <sequence>
      <element name="firstname" type="string"/>
      <element name="lastname" type="string"/>
    </sequence>
  </complexType>
</element>
```

maxOccurs minOccurs

- Especificam o número máximo e mínimo, respectivamente, que um elemento pode ocorrer.

```
<element name="person">
 <complexType>
   <sequence>
     <element name="full_name" type="string"/>
     <element name="child_name" type="string"
              maxOccurs="10" minOccurs="0"/>
   </sequence>
 </complexType>
</element>
```

Group

- Indicadores de grupo são usados para definir conjuntos de elementos relacionados.
- Elementos:

```
<group name="groupname">
  ...
</group>
```

- Atributos:

```
<attributeGroup name="groupname">
  ...
</attributeGroup>
```

Element Group

```
<group id="persongroup">
        <sequence>
                <element name="firstname" type="string"/>
                <element name="lastname" type="string"/>
                <element name="birthday" type="date"/>
        </sequence>
</group>

<element name="person" type="teste:personinfo/">

<complexType name="personinfo">
        <sequence>
                <group ref="persongroup"/>
                <element name="country" type="string"/>
        </sequence>
</complexType>
```

Attribute Group

```
<attributeGroup id="personattgroup">
        <attribute name="firstname" type="string"/>
        <attribute name="lastname" type="string"/>
        <attribute name="birthday" type="date"/>
</attributeGroup >

<element name="person">
<complexType>
        <attributeGroup ref="personattgroup"/>
</complexType>
</element>
```

Declaração de atributos

- <Person lang="EN">…</Person>
- <attribute name="lang" type="string"/>

- Um atributo pode aparecer uma ou nenhuma vez.
- O atributo use indica se um atributo é `required`, `optional` ou `prohibited`. Exemplo:

```
<attribute name="partNum" type="SKU" use="required"/>
```

- A declaração de um atributo pode contar ainda os atributos default e fixed. Exemplo:

```
<attribute name="country" type="string" fixed="US"/>
```

```
<element name="person">
   <complexType>
     <sequence>
       <element name="firstname" type="string"/>
       <element name="lastname" type="string"/>
     </sequence>
      <attribute name="lang" type="string"/>
   </complexType>
 </element>
```

Vínculo XML -> XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<esquemaTeste xmlns="http://www.example.com/teste"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.example.com/teste teste.xsd">
....

</esquemaTeste>
```

- Elemento Raíz: <esquemaTeste>

Bibliografia

- Site da W3C (www.w3.org)
- XML Schema Part 0: Primer Second Edition.
  W3C Standard 28 October 2004
  http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/