P.PORTO

ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

XML Schema - Namespaces

Bruno Oliveira (bmo@estg.ipp.pt)

PEI Processamento Estruturado de Informação

Agenda

- Referenciar elementos e tipos
- Modularização de um vocabulário XSD em vários documentos XSD;
- XML namespaces;
- Documentação em XSD.

Referenciar elementos e tipos

- Quando os documentos são extensos, torna-se mais difícil a sua legibilidade e manutenção.
- Um XSD mais complexo pode ser “simplificado” através do uso dos atributos “ref” e “type”.

Referenciar elementos e tipos

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="fatura">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="numero_fatura" type="xs:positiveInteger" />
        <xs:element name="produto" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="nome" type="xs:string" />
              <xs:element name="quantidade" type="xs:positiveInteger" />
              <xs:element name="preco" type="xs:decimal" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Referenciar elementos e tipos

- Um dos modos de se aumentar a legibilidade de um XSD consiste em definir-se, inicialmente, os elementos e atributos e, posteriormente, referenciá-los através do atributo “ref”.
- É possível, por exemplo, referenciarem-se elementos complexos que, por sua vez, fazem referência a outros elementos.

Referenciar elementos e tipos

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="produto">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="nome" type="xs:string" />
                <xs:element name="quantidade" type="xs:positiveInteger" />
                <xs:element name="preco" type="xs:decimal" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="fatura">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="numero_fatura" type="xs:positiveInteger" />
                <xs:element ref="produto" maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

Referenciar elementos e tipos

- Outra alternativa (geralmente a aconselhada) passa por definir tipos, que possam ser reutilizados e, de seguida, invocá-los através do atributo “type”.

Referenciar elementos e tipos

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:complexType name="produtoTipo">
        <xs:sequence>
            <xs:element name="nome" type="xs:string" />
            <xs:element name="quantidade" type="xs:positiveInteger" />
            <xs:element name="preco" type="xs:decimal" />
        </xs:sequence>
    </xs:complexType>

    <xs:element name="fatura">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="numero_fatura" type="xs:positiveInteger" />
                <xs:element name="produto" maxOccurs="unbounded" type="produtoTipo" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

Modularização de um XSD em vários ficheiros

- Exemplo:

Fatura.xsd

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:include schemaLocation="produto.xsd"/>

<xs:element name="fatura">
    <xs:complexType>
        <xs:sequence>
          <xs:element name="numero_fatura" type="xs:positiveInteger"/>
          <xs:element name="produto" maxOccurs="unbounded" type="produtoTipo"/>
        </xs:sequence>
      </xs:complexType>
</xs:element>

</xs:schema>
```

Produto.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:complexType name="produtoTipo">
        <xs:sequence>
            <xs:element name="nome" type="xs:string"/>
            <xs:element name="quantidade" type="xs:positiveInteger"/>
            <xs:element name="preco" type="xs:decimal"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

Namespaces

- Até agora aprendemos a criar documentos XML e os respetivos documentos de regras que nos permitem definir o vocabulário admissível para a construção de documentos;
- Considere que ficou encarregue de criar os documentos XSD para definir o conceito de ingrediente, enquanto que outro colega ficou responsável pela definição dos conceitos relacionados com uma Pizza;
- Considere ainda que o nome do ingrediente tem no máximo 50 caracteres e o nome da pizza 30 caracteres.

Namespaces

- Elementos com o mesmo nome!

Ingrediente.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:simpleType name="tipoNome">
        <xs:restriction base="xs:string">
           <xs:maxLength value="50" />
        </xs:restriction>
   </xs:simpleType>
(…)
</xs:schema>
```

Pizza.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:simpleType name=" tipoNome ">
        <xs:restriction base="xs:string">
           <xs:maxLength value="30" />
        </xs:restriction>
   </xs:simpleType>
(…)
</xs:schema>
```

Namespaces

- O que irá originar o seguinte erro:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:include schemaLocation="Ingrediente.xsd"/>
   (…)
   <xs:element name="pizza">
       <xs:complexType>
          <xs:sequence>
             <xs:element name="nome" type="tipoNome" />
          </xs:sequence>
       </xs:complexType>
    </xs:element>
</xs:schema>
```

A schema cannot contain two global components with the same name; this schema contains two occurrences of ',name'.

Utilização de namespaces

- A solução para o problema apresentado nos slides anteriores passa por agrupar um conjunto de elementos de cada documento no seu próprio "espaço";
- Isto significa que a sua referenciação passa a ser feita considerando o espaço ao qual cada elemento reside;

Utilização de namespaces

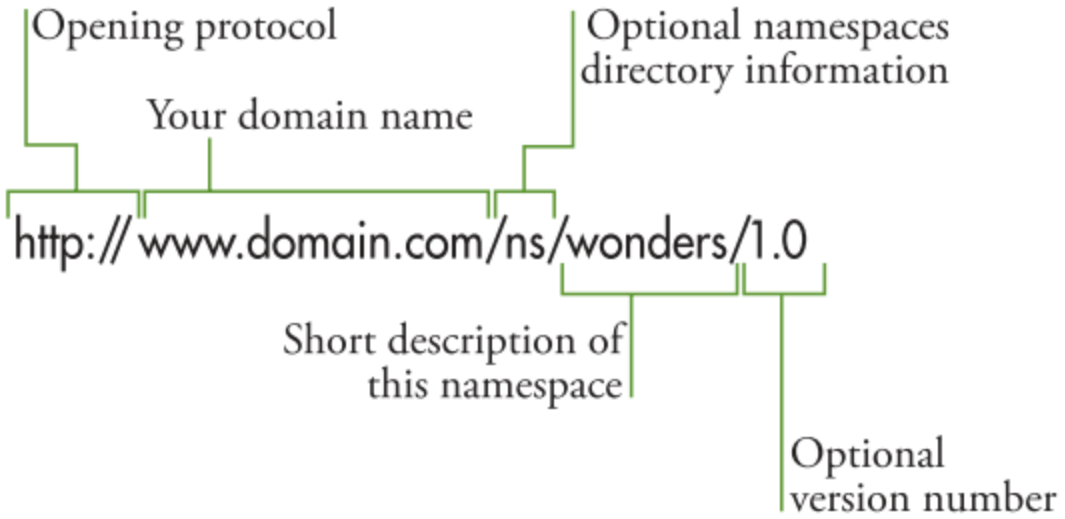
- Este formato permite a distinção dos elementos XML de cada um dos grupos;
- Se por exemplo identificar-mos um determinado “espaço” como ESTG, então (por exemplo) o tipo (ou elemento): ESTG.nome do exemplo anterior, não pode ser confundido com nenhum outro elemento de outro “espaço”.

Utilização de namespaces

- Normalmente não utilizamos nomes como ESTG para nos referirmos a um namespace, mas devemos utilizar nomes mais estruturados para o efeito;
- Em XML, os namespaces são escritos na forma de URI (Uniform Resource Identifier).

Utilização de namespaces

- Estrutura de um URI



Default Namespace

- Após a definição do nome do namespace, podemos declarar o namespace por defeito;
- Exemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<pizza xmlns="http:\\estg.ipp.pt\eoi\2020\Pizza">
  <name>Portuguesa</name>
  <ingredients>
    <ingredient>
      <name>Tomate</name>
      <category>Vegetal</category>
      <unit>grams</unit>
      <calories>200</calories>
    </ingredient>
  </ingredients>
  <price>20</price>
</pizza>
```

Default Namespace

- A declaração de um default namespace para o elemento root significa que todos os elementos (filhos do root) do documento pertencem ao mesmo namespace;
- Um namespace pode ser definido com o atributo xmlns na marcação de início de um elemento;

Default Namespace

- No caso que não ser declarado namespace, todos os elementos são considerados como estando “sem namespace”;
- Até agora quando declaramos num documento XML a inclusão do documento XSD, coloca xsi:noNamespaceSchemaLocation="Exemplo7.xsd"
- Ou seja, sem namespace!

Definição de prefixos

- Na definição do namespace podemos utilizar um identificador para nos referirmos a um conjunto de elementos com um determinado namespace;
- Para isso, podemos definir um prefixo para um namespace e então utilizar o prefixo para referenciar elementos individuais;
- Os prefixos são definidos através de: xmlns:prefix, onde o prefix é o “nickname” para o namespace.

Definição de prefixos

- Após a declaração do prefixo para um namespace, podemos associar elementos para diferentes namespaces no XML;
- Recorremos a prefixos para identificar elementos de diferentes namespaces;
- Neste caso, especificamos o target namespace (namespace de destino) para o qual os elementos e atributos vão pertencer.

Definição de prefixos

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns="http://www.aTascadaESTG.pt/Ingredient"
        targetNamespace="http://www.aTascadaESTG.pt/Ingredient"
        elementFormDefault="qualified">
(…)
```

Definição de prefixos

- Este processo de especificar os elementos e atributos para um namespace é chamado de povoar o namespace;
- Quando povoamos um namespace, apenas os elementos definidos globalmente e atributos ficam associados ao namespace.

Definição de prefixos

- É comum a utilização do atributo: elementFormDefault na definição do XSD, indicando que todos os elementos são qualified (ou seja, estão associadas ao target namespace)
- Desta forma, todos os elementos pertencentes a um namespace, têm de ter especificado o namespace.

Definição de prefixos

- Um elemento qualified é um elemento que está associado a um namespace, utilizando para isso um prefixo;
- Podemos também definir um elemento qualified sem o prefixo (default namespace): xmlns=”<namespace>”;
- Apenas pode existir um namespace associado sem prefixo por documento XML;

Definição de prefixos

Ingrediente.XSD

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns="http://www.aTascadaESTG.pt/Ingrediente"
targetNamespace="http://www.aTascadaESTG.pt/Ingrediente">
(...)
```

Pizza.XSD

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns="http://www.aTascadaESTG.pt/Pizza"
        targetNamespace="http://www.aTascadaESTG.pt/Pizza"
        xmlns:c="http://www.aTascadaESTG.pt/Ingrediente"
        elementFormDefault="qualified">
(...)
<xs:element name="pizza">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="nome" type="name" />
            <xs:element name="ingrediente">
             <xs:complexType>
              <xs:sequence>
                <xs:element name="nome" type="c:tipoNome" />
                (...)
```

Elementos qualificados

Utilização de prefixos

Importar XSD com namespace

- Os namespaces são um mecanismo muito útil para modularizar XML schemas, principalmente quando dividimos a nossa especificação em vários documentos;
- A divisão da especificação XSD em múltiplos ficheiros com namespaces bem definidos, permite-nos reutilizar as nossas definições em diferentes projetos, tornando o vocabulário mais fácil de ler modificar, uma vez que trabalhamos com pequenas partes do vocabulário;

Importar XSD com namespace

Ingrediente.XSD

```
<xs:schema   xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns="http://www.aTascadaESTG.pt/Ingrediente"
 targetNamespace="http://www.aTascadaESTG.pt/Ingrediente">
(...)
```

Com a inclusão de namespaces, o elemento import deverá ser utilizado para importar documentos XSD (ao invés do elemento include)

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns="http://www.aTascadaESTG.pt/Pizza"
        targetNamespace="http://www.aTascadaESTG.pt/Pizza"
        xmlns:c="http://www.aTascadaESTG.pt/Ingrediente"
        elementFormDefault="qualified">
      <xs:import schemaLocation="Ingrediente.xsd"
namespace="http://www.aTascadaESTG.pt/Ingrediente"/>
(...)
<xs:element name="pizza">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="nome" type="name" />
            <xs:element name="ingrediente">
             <xs:complexType>
               <xs:sequence>
                 <xs:element name="nome" type="c:tipoNome" />
               (...)
```

Associação XSD a XML

- xmlns – Descreve o namespace por defeito, indicando que todos os elementos utilizados neste documento estão declarados no namespace;
- xsi:schemaLocation – Este atributo possui dois valores: o nome do namespace a utilizar, (“Espaço”), e o caminho para o XSD utilizado nesse namespace;

Associação XSD a XML

Pizza.xml (excerto)

```
<?xml version="1.0" encoding="UTF-8"?>
<pizza xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://www.aTascadaESTG.pt/Pizza"
    xsi:schemaLocation="http://www.aTascadaESTG.pt/Pizza Pizza.xsd">
        (...)
</pizza>
```

Associação XSD a XML

- O documento XML deverá possuir a identificação dos namespaces envolvidos no XSD utilizado para validação;
- Deve também ser definido um prefixo para cada um desses namespaces (elementFormDefault = “qualified”), com exceção do namespace que seja considerado o default namespace (se for utilizado).

Associação XSD a XML

- Exemplo:

  Pizza.xml (excerto)

```
<?xml version="1.0" encoding="UTF-8"?>
<pizza xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=http://www.estg.ipp.pt/eoi/2020/pizzaria/pizzaTypes PizzaTypes.xsd
xmlns=http://www.estg.ipp.pt/eoi/2020/pizzaria/pizzaTypes  xmlns:i= "http://www.estg.ipp.pt/eoi/2020/pizzaria/ingredientTypes">
   <name>Portuguesa</name>
   <ingredients>
     <i:ingredient>
        <i:name>Tomate</i:name>
        <i:category>Vegetal</i:category>
        <i:unit>grams</i:unit>
        <i:calories>200</i:calories>
     </i:ingredient>
   </ingredients>
   <price>20</price>
</pizza>
```

Definição do prefixo i para o namespace

Utilização do prefixo

Documentação

- O elemento annotation permite documentar os vocabulários produzidos;
- Exemplo:

```
(...)
<xs:simpleType name="priceType">
      <xs:annotation xml:lang="en">
        <xs:documentation>Rules for price definition</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:decimal">
        <xs:totalDigits value="5" />
        <xs:fractionDigits value="2" />
      </xs:restriction>
  </xs:simpleType>
(...)
```

Documentação

- A geração de documentação é fundamental para descrever todas as particularidades do vocabulário:
- Com o Oxygen XML Editor basta aceder a Tools -> Generate Documentation -> XML Schema Documentation;

Bibliografia/referências

- Referências Web:
  - https://www.w3schools.com/;
  - https://www.intertech.com/Blog/xml-schema-elementformdefault-and-attributeformdefault/
  - https://www.liquid-technologies.com/xml-schema-tutorial/xsd-namespaces
- Livro:
  - Anders M. and Michel S., An introduction to XML and Web Technologies, Addison-Wesley, 2006;
  - Eito-Brun, R. XML-based Content Management: Integration, Methodologies, and Tools (1st ed.). Chandos Publishing, 2017;

P.PORTO

ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

XML Schema - Namespaces

Bruno Oliveira (bmo@estg.ipp.pt)

PEI Processamento Estruturado de Informação