

Computer Science E-259

XML with J2EE

Lecture 10: XML Schema, Continued

13 April 2006

David J. Malan

`malan@post.harvard.edu`

1

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Last Time

XML Schema (Second Edition)

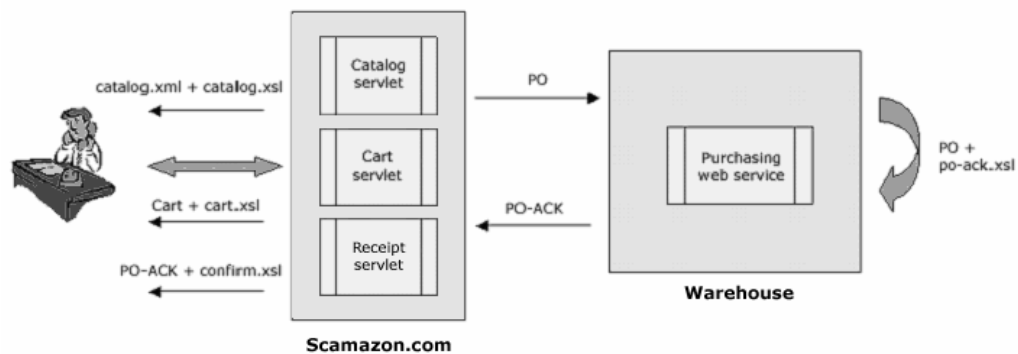
- XML Schema (Second Edition)
- Project 4

2

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Last Time

scamazon.com



3

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Last Time

XML Schema (Second Edition)

- Declarations v. Definitions
- Global v. Local Components
- Element and Attribute Declarations
- Simple v. Complex Types
- Named v. Anonymous Types
- Type Definition Hierarchy
- Simple Types
- Complex Types
- Namespaces
 - Multiple
 - Default
 - Target
- Relating Instances to Schemas

4

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Computer Science E-259

This Time

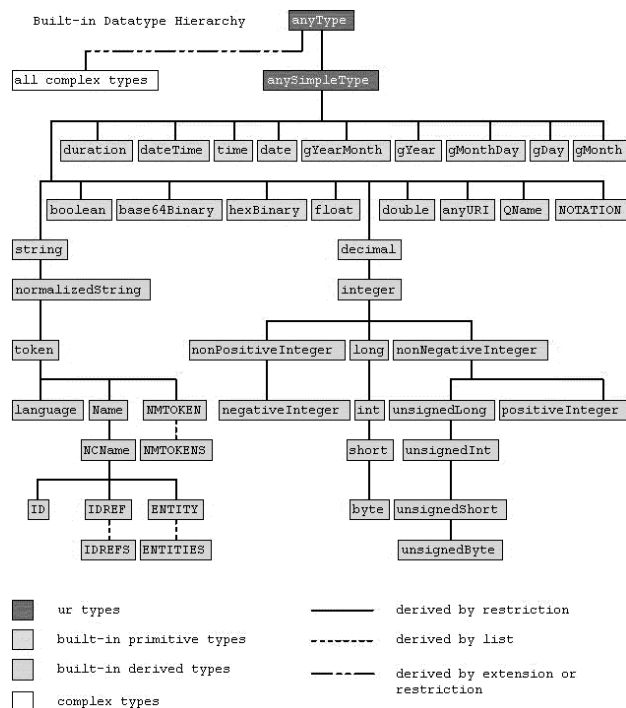
- XML Schema 1.0, Continued

5

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Datatypes

Built-In



6

Image from <http://www.w3.org/TR/xmlschema-2/>.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Structures

Elements

```
<element
  abstract = boolean : false
  block = (#all | List of (extension | restriction | substitution))
  default = string
  final = (#all | List of (extension | restriction))
  fixed = string
  form = (qualified | unqualified)
  id = ID
  maxOccurs = (nonNegativeInteger | unbounded) : 1
  minOccurs = nonNegativeInteger : 1
  name = NCName
  nillable = boolean : false
  ref = QName
  substitutionGroup = QName
  type = QName
  {any attributes with non-schema namespace . . .}>
  Content: (annotation?, ((simpleType | complexType)?, (unique | key |
    keyref)*))
</element>
```

7

Excerpted from <http://www.w3.org/TR/xmlschema-1/>.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Structures

Attributes

```
<attribute
  default = string
  fixed = string
  form = (qualified | unqualified)
  id = ID
  name = NCName
  ref = QName
  type = QName
  use = (optional | prohibited | required) : optional
  {any attributes with non-schema namespace . . .}>
  Content: (annotation?, (simpleType?))
</attribute>
```

8

Excerpted from <http://www.w3.org/TR/xmlschema-1/>.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Elements of Unrestricted, Simple Types

- Canonical Declaration
`<xsd:element name="xxx" type="yyy"/>`
- Examples of Instances
`<lastname>Refsnes</lastname>`
`<age>34</age>`
`<dateborn>1968-03-27</dateborn>`
- Examples of Declarations
`<xsd:element name="lastname" type="xsd:string"/>`
`<xsd:element name="age" type="xsd:integer"/>`
`<xsd:element name="dateborn" type="xsd:date"/>`

Simple Types

Attributes of Unrestricted, Simple Types

- Canonical Declaration
`<xsd:attribute name="xxx" type="yyy"/>`
- Example of an Instance
`<lastname lang="EN">Smith</lastname>`
- Example of a Declaration
`<xsd:attribute name="lang" type="xsd:string"/>`

Simple Types

Attributes with Default or Fixed Values

- In the absence of a value, a default can be assigned
`<xsd:attribute name="lang" type="xsd:string" default="EN"/>`
- A value, if present and declared fixed, must appear as declared;
in the absence of a value, the declared will be assigned
`<xsd:attribute name="lang" type="xsd:string" fixed="EN"/>`

Simple Types

Optional and Required Attributes

- By default, attributes are optional
`<xsd:attribute name="lang" type="xsd:string" use="optional"/>`
- If declared required, attribute must be present
`<xsd:attribute name="lang" type="xsd:string" use="required"/>`

Simple Types

Restrictions through Facets

Facet	Description
enumeration	Defines a list of acceptable values
fractionDigits	Specifies the maximum number of decimal places allowed. Must be equal to or greater than zero
length	Specifies the exact number of characters or list items allowed. Must be equal to or greater than zero
maxExclusive	Specifies the upper bounds for numeric values (the value must be less than this value)
maxInclusive	Specifies the upper bounds for numeric values (the value must be less than or equal to this value)
maxLength	Specifies the maximum number of characters or list items allowed. Must be equal to or greater than zero
minExclusive	Specifies the lower bounds for numeric values (the value must be greater than this value)
minInclusive	Specifies the lower bounds for numeric values (the value must be greater than or equal to this value)
minLength	Specifies the minimum number of characters or list items allowed. Must be equal to or greater than zero
pattern	Defines the exact sequence of characters that are acceptable
totalDigits	Specifies the exact number of digits allowed. Must be greater than zero
whiteSpace	Specifies how white space (line feeds, tabs, spaces, and carriage returns) are handled

13

Adapted from http://www.w3schools.com/schema/schema_facets.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Restricting by Value

```
<xsd:element name="age">
  <xsd:simpleType>
    <xsd:restriction base="xsd:integer">
      <xsd:minInclusive value="1"/>
      <xsd:maxInclusive value="100"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element name="age">
  <xsd:simpleType>
    <xsd:restriction base="xsd:integer">
      <xsd:minExclusive value="0"/>
      <xsd:maxExclusive value="101"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

14

Adapted from http://www.w3schools.com/schema/schema_facets.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Restricting by Value

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

15

Adapted from http://www.w3schools.com/schema/schema_facets.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Restricting by Value

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

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

16

Adapted from http://www.w3schools.com/schema/schema_facets.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Restricting by Pattern

```
<xsd:element name="choice">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[abcd]" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element name="initials">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[A-Z] [A-Z] [A-Z] ?" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Simple Types

Restricting by Pattern

```
<xsd:element name="gender">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="male|female" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element name="password">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[a-zA-Z0-9]{8}" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Simple Types

Restricting by Pattern

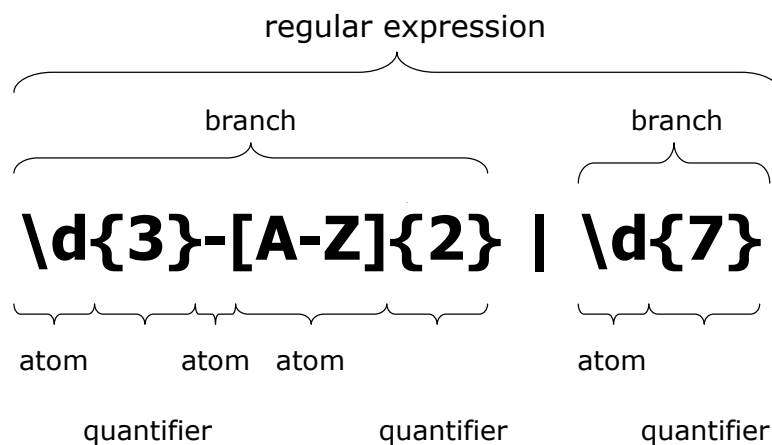
```
<xsd:simpleType name="ProdNumType">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="\d{3}-[A-Z]{2} | \d{7}"/>
  </xsd:restriction>
</xsd:simpleType>
```

19

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Regular Expressions



20

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Regular Expressions

- An atom describes one or more character through
 - a single normal character (e.g., `a` or `c`)
 - a parenthesized regular expression (e.g., `(a|c)`)
 - an escape, such as
 - `\n` for newline, `\?` for `?`, `.` for any character but `\n` and `\r`
 - `\d` for any digit, `\D` for any character but a digit
 - `\s` for any whitespace character
 - a character class expression
 - `[abc]` matches any of a list of characters
 - `[0-9]` or `[a-z]` matched any character from a range
- A quantifier indicates how many times an atom should repeat (e.g., `?`, `*`, `+`, `{n}`, `{n,}`, `{n,m}`)

Simple Types

Restricting Whitespace

```
<xsd:element name="address">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:whiteSpace value="preserve"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element name="address">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:whiteSpace value="replace"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element name="address">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:whiteSpace value="collapse"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Simple Types

Restricting by Length

```
<xsd:element name="password">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:length value="8"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element name="password">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="5"/>
      <xsd:maxLength value="8"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

23

Adapted from http://www.w3schools.com/schema/schema_facets.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Lists

```
<intvalues>100 34 56 -23 1567</intvalues>

<xsd:element name="intvalues" type="valuelist"/>
<xsd:simpleType name="valuelist">
  <xsd:list itemType="xsd:integer"/>
</xsd:simpleType>
```

24

Adapted from http://www.w3schools.com/schema/el_list.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Simple Types

Unions

```
<xsd:element name="jeans_size">
  <xsd:simpleType>
    <xsd:union memberTypes="sizebyno sizebystring"/>
  </xsd:simpleType>
</xsd:element>

<xsd:simpleType name="sizebyno">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:maxInclusive="42"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="sizebystring">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="small"/>
    <xsd:enumeration value="medium"/>
    <xsd:enumeration value="large"/>
  </xsd:restriction>
</xsd:simpleType>
```

25

Adapted from http://www.w3schools.com/schema/el_union.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

Content Types for Elements

- Simple (*i.e.*, children include text only)
`<name>Jerry Seinfeld</name>`
- Element-Only (*i.e.*, children include elements only)
`<name><first>Jerry</first><last>Seinfeld</last></name>`
- Mixed (*i.e.*, children contain text and/or elements)
`<p><name>Jerry Seinfeld</name> is a <title>comedian</title>.</p>`
- Empty (*i.e.*, no children)
`<comedian name="Jerry Seinfeld"/>`

26

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

Simple Content

```
<xsd:element name="somename">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:extension base="basetype">
        ....
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>

<xsd:element name="somename">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:restriction base="basetype">
        ....
      </xsd:restriction>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>
```

27

Adapted from http://www.w3schools.com/schema/schema_complex_text.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

Element-Only Content

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

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

28

Adapted from http://www.w3schools.com/schema/schema_complex_elements.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

Element-Only Content

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

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

Complex Types

Mixed Content

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

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

Complex Types

Empty Content

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

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

Complex Types

Model Groups

- A **sequence** group of element declarations is used to indicate the order in which the elements should appear
- A **choice** group of element declarations is used to indicate that only one of the elements should appear
- An **all** group is used to indicate that all elements should appear, in any order, but no more than once each

Complex Types

The sequence Model

```
<xsd:element name="person">
  <xsd:complexType>
    <xsd:sequence minOccurs="0">
      <xsd:element name="firstname" type="xsd:string"/>
      <xsd:element name="lastname" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

Complex Types

The sequence Model

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

Complex Types

The choice Model

```
<xsd:complexType name="ProductType">
  <xsd:choice minOccurs="0" maxOccurs="3">
    <xsd:element name="size" type="SizeType"/>
    <xsd:element name="color" type="ColorType"/>
  </xsd:choice>
  <xsd:attribute name="effDate" type="xsd:date"/>
</xsd:complexType>
```

35

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

Nesting Models!

```
<xsd:complexType name="ProductType">
  <xsd:sequence>
    <xsd:element name="number" type="xsd:integer"/>
    <xsd:choice minOccurs="0" maxOccurs="3">
      <xsd:element name="size" type="SizeType"/>
      <xsd:element name="color" type="ColorType"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attribute name="effDate" type="xsd:date"/>
</xsd:complexType>
```

36

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

The all Model

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

Complex Types

Defining Model Groups

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

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

Complex Types

Defining Attribute Groups

```
<xsd:attributeGroup name="personattrgroup">
  <xsd:attribute name="firstname" type="xsd:string"/>
  <xsd:attribute name="lastname" type="xsd:string"/>
  <xsd:attribute name="birthday" type="xsd:date"/>
</xsd:attributeGroup>

<xsd:element name="person">
  <xsd:complexType>
    <xsd:attributeGroup ref="personattrgroup"/>
  </xsd:complexType>
</xsd:element>
```

Complex Types

Extending Simple Types

```
<xsd:simpleType name="size">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="small" />
    <xsd:enumeration value="medium" />
    <xsd:enumeration value="large" />
  </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="jeans">
  <xsd:simpleContent>
    <xsd:extension base="size">
      <xsd:attribute name="sex">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="male" />
            <xsd:enumeration value="female" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
```

Complex Types

Extending Complex Types

```
<xsd:complexType name="ProductType">
  <xsd:sequence>
    <xsd:element name="number" type="ProdNumType"/>
    <xsd:element name="name" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ShirtType">
  <xsd:complexContent>
    <xsd:extension base="ProductType">
      <xsd:choice maxOccurs="unbounded">
        <xsd:element name="size" type="SizeType"/>
        <xsd:element name="color" type="ColorType"/>
      </xsd:choice>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

41

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

Allowing for Any Elements

```
<xsd:element name="person">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="firstname" type="xsd:string"/>
      <xsd:element name="lastname" type="xsd:string"/>
      <xsd:any minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

42

Adapted from http://www.w3schools.com/schema/schema_complex_any.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

Allowing for Any Attributes

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

43

Adapted from http://www.w3schools.com/schema/schema_complex_anyattribute.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Complex Types

Allowing for Substitutes

```
<customer>
  <name>John Smith</name>
</customer>

<cliente>
  <nome>Giovanni Smith</nome>
</cliente>

<xsd:element name="name" type="xsd:string"/>
<xsd:element name="nome" substitutionGroup="name"/>
<xsd:complexType name="custinfo">
  <xsd:sequence>
    <xsd:element ref="name"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="customer" type="custinfo"/>
<xsd:element name="cliente" substitutionGroup="customer"/>
```

44

Adapted from http://www.w3schools.com/schema/schema_complex_subst.asp.

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

XML Schema (Second Edition)

Summary

Element	Explanation
<u>all</u>	Specifies that the child elements can appear in any order. Each child element can occur 0 or 1 time
<u>annotation</u>	Specifies the top-level element for schema comments
<u>any</u>	Enables the author to extend the XML document with elements not specified by the schema
<u>anyAttribute</u>	Enables the author to extend the XML document with attributes not specified by the schema
<u>appInfo</u>	Specifies information to be used by the application (must go inside annotation)
<u>attribute</u>	Defines an attribute
<u>attributeGroup</u>	Defines an attribute group to be used in complex type definitions
<u>choice</u>	Allows only one of the elements contained in the <choice> declaration to be present within the containing element
<u>complexContent</u>	Defines extensions or restrictions on a complex type that contains mixed content or elements only
<u>complexType</u>	Defines a complex type element
<u>documentation</u>	Defines text comments in a schema (must go inside annotation)
<u>element</u>	Defines an element
<u>extension</u>	Extends an existing simpleType or complexType element
<u>field</u>	Specifies an XPath expression that specifies the value used to define an identity constraint

XML Schema

Summary

<u>group</u>	Defines a group of elements to be used in complex type definitions
<u>import</u>	Adds multiple schemas with different target namespace to a document
<u>include</u>	Adds multiple schemas with the same target namespace to a document
<u>key</u>	Specifies an attribute or element value as a key (unique, non-nullable, and always present) within the containing element in an instance document
<u>keyref</u>	Specifies that an attribute or element value correspond to those of the specified key or unique element
<u>list</u>	Defines a simple type element as a list of values
<u>notation</u>	Describes the format of non-XML data within an XML document
<u>redefine</u>	Redefines simple and complex types, groups, and attribute groups from an external schema
<u>restriction</u>	Defines restrictions on a simpleType, simpleContent, or a complexContent
<u>schema</u>	Defines the root element of a schema
<u>selector</u>	Specifies an XPath expression that selects a set of elements for an identity constraint
<u>sequence</u>	Specifies that the child elements must appear in a sequence. Each child element can occur from 0 to any number of times
<u>simpleContent</u>	Contains extensions or restrictions on a text-only complex type or on a simple type as content and contains no elements
<u>simpleType</u>	Defines a simple type and specifies the constraints and information about the values of attributes or text-only elements
<u>union</u>	Defines a simple type as a collection (union) of values from specified simple data types
<u>unique</u>	Defines that an element or an attribute value must be unique within the scope

Next Time

Web Services, SOAP 1.2, and WSDL 1.1



47

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

Computer Science E-259

XML with J2EE

Lecture 10: XML Schema, Continued

13 April 2006

David J. Malan

malan@post.harvard.edu

48

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.