XML Schema

Introduction

What is XML Schema?

- XML Schema is an XML-based alternative to DTD.
- An XML schema describes the structure of an XML document.

What it describes?

- It can define the structure of elements and the attributes of xml document
- It defines the data types of the elements and the attributes

Support Data types

- It is easier to describe allowable document content
- It is easier to validate the correctness of data
- It is easier to work with data from a database
- It is easier to define data facets (restrictions on data)
- It is easier to define data patterns (data formats)
- It is easier to convert data between different data types

XML Schema Element

```
<?xml version="1.0"?>

<xs:schema>
...
...
</xs:schema>
```

XML Schema

-
  ```
  <?xml version="1.0"?>

  <xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.w3schools.com"
  xmlns="http://www.w3schools.com"
  elementFormDefault="qualified">
  ...
  ...
  </xs:schema>
  ```

- xmlns:xs="http://www.w3.org/2001/XMLSchema"
  - indicates that the elements and data types **used in the schema** come from the "http://www.w3.org/2001/XMLSchema" namespace.
  - It also specifies that the elements and data types that come from the "http://www.w3.org/2001/XMLSchema" namespace should be prefixed with **xs:**

- targetNamespace="http://www.w3schools.com"
  - indicates that the elements defined by this schema (note, to, from, heading, body.) come from the "http://www.w3schools.com" namespace.
- xmlns="http://www.w3schools.com"
  - indicates that the default namespace is "http://www.w3schools.com".

- elementFormDefault="qualified“
  - indicates that any elements used by the XML instance document which were declared in this schema must be namespace qualified.

XML Document Instance

```
<?xml version="1.0"?>

<note xmlns="http://www.w3schools.com"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema
   -instance"
   xsi:schemaLocation="http://www.w3schools.com
   “note.xsd">

   <to>Tove</to>
   <from>Jani</from>
   <heading>Reminder</heading>
   <body>Don't forget me this weekend!</body>
   </note>
```

- xmlns="http://www.w3schools.com"
  - Specifies the default namespace
- xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  - Obtain the namespace instance
- xsi:schemaLocation="http://www.w3schools.com “note.xsd"
  - The first value is the namespace to use
  - The second value is the location of the XML schema to use for that namespace

Simple Element

- It contain only texts
- <xs:element name="xxx" type="yyy"/>
- XML Schema has a lot of built-in data types. The most common types are:
  - xs:string
  - xs:decimal
  - xs:integer
  - xs:boolean
  - xs:date
  - xs:time

Example for simple elements

Syntax:

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

Example:

```
<lastname>Ramesh</lastname>
<age>36</age>
<dateborn>1975-03-27</dateborn>
```

Default and fixed values for simple element

- <xs:element name="color" type="xs:string" default="red"/>
- <xs:element name="color" type="xs:string" fixed="red"/>

Defining an Attribute

- <xs:attribute name="xxx" type="yyy"/>
- XML Schema has a lot of built-in data types. The most common types are:
  - xs:string, xs:decimal, xs:integer, xs:boolean
  - xs:date, xs:time

- <xs:attribute name="lang" type="xs:string"/>
  - <lastname lang="EN">Smith</lastname>

…Contd

Syntax

- <xs:attribute name="lang" type="xs:string"/>

Example

- <lastname lang="EN">Smith</lastname>

Attributes

- <xs:attribute name="lang" type="xs:string" default="EN"/>
- <xs:attribute name="lang" type="xs:string" fixed="EN"/>
- <xs:attribute name="lang" type="xs:string" use="required"/>

Restrictions using Data types

-
  ```
  <xs:element name="age">
    <xs:simpleType>
      <xs:restriction base="xs:integer">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="120"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  ```

Restrictions on a Set of Values

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

…Contd

-
  ```
  <xs:element name="car" type="carType"/>
  <xs:simpleType name="carType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Audi"/>
      <xs:enumeration value="Golf"/>
      <xs:enumeration value="BMW"/>
    </xs:restriction>
  </xs:simpleType>
  ```

Restrictions on a Series of Values

-
  ```
  <xs:element name="letter">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="[a-z]"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  ```

…Contd

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

…Contd

-
  ```
  <xs:element name="initials">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  ```

…Contd

-
  ```
  <xs:element name="prodid">
    <xs:simpleType>
      <xs:restriction base="xs:integer">
        <xs:pattern value="[0-9][0-9][0-9][0-
  9][0-9]"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  ```

…Contd

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

…Contd

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

…Contd

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

…Contd

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

What is complex element?

- empty elements
- elements that contain only other elements
- elements that contain only text
- elements that contain both other elements and text

Complex Element - Examples

Example1 – Empty Element

- <product pid="1345"/>

Example2 – Element contains elements

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

...Contd

Example3 – Element contains only text

```
<food type="dessert">Ice cream</food>
```

Example4 – Element contains text & elements

```
<description>
It happened on <date
  lang="norwegian">03.03.99</date> ....
</description>
```

ComplexEmpty Element

XML

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

Schema

```
<xs:element name="product">
  <xs:complexType>
    <xs:complexContent>
      <xs:restriction base="xs:integer">
        <xs:attribute name="prodid"
   type="xs:positiveInteger"/>
      </xs:restriction>
```

Complex elements contain other elements

XML

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

Schema

```
<xs:element name="employee">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname"
```

Complex element contain text only

XML

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

Schema

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

Order Indicator - <all>

They are used to define the order of the elements.

<all> indicator - child elements can appear in any order

Example

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

Order Indicator - <choice>

It specifies that either one child element or another can occur

Example

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

Order Indicator - <sequence>

It specifies that the child elements must appear in a specific order

Example

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

Occurrence Indicator – Min & Max

<maxOccurs> - specifies the maximum number of times an element can occur

<minOccurs> - specifies the maximum number of times an element can occur:

Example

```
<xs:element name="person">
    <xs:complexType>
     <xs:sequence>
         <xs:element name="full_name" type="xs:string"/>
          <xs:element name="child_name" type="xs:string"
    minOccurs=”5”
               maxOccurs="10"/>
```