XML Schema Structure

An XML Schema Definition has the following components:

❑Schema Element	<schema></schema>
❑Element Declaration	
❑Attribute Declaration	

Schema Element

- An XML Schema is composed of root element <schema> tag. The <schema> element contain the following namespace.
  http://www.w3.org/2001/XMLSchema
- All the elements and data types used in schema come from the http://www.w3.org/2001/XMLSchema namespace.
- Example:

```
<schema xmlns=”http://www.w3.org/2001/XMLSchema” >

//XMLSchema Element/Attribute declaration

</schema>
```

Element Declaration – Simple Element

How do you define the of an XML document in an XML Schema?

- The general form of Element Declaration is :
  <element name=”element_name” type=”data_type” >
- In XML Schema we can create 2 type of element:
  a) **Simple Element**- Contain only text data.
  b) **Complex Element –** Contain child element or Attribute or Both
- Example: **<element name=”name” type=”string” />**
  <element name=”age” type=”int” />

The corresponding XML Elements are:

<name>Neha</name>
<age>22</age>

Element Declaration - Complex Element

□ Complex element contain child element or an Attribute or Both.

□ EMPTY element are considered to be complex type element.

□ A complex type element is defined by using the <complexType> schema element.

□ To create a complex type element we use complex data type:
So whenever we want to create a complex element with
Child element we use any one complex content.

XML Schema to create Complex type element with some child element

```
<student>
<srollno>545</srollno>
<sname>Soumya</sname>
<student>
```

a.xml

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
<element name="student">
<complexType>
<sequence>
<element name="srollno" type="int" />
<element name="sname" type="string"/>
</sequence>
</complexType>
</element>
</schema>
```

a.xsd

Occurrence indicator

□ In XML document an element can occur zero time, one time, 2 times, 3 times…. N number of times.

□ We can specify the number of times an element occurs in a document using minOccurs and maxOccurs.

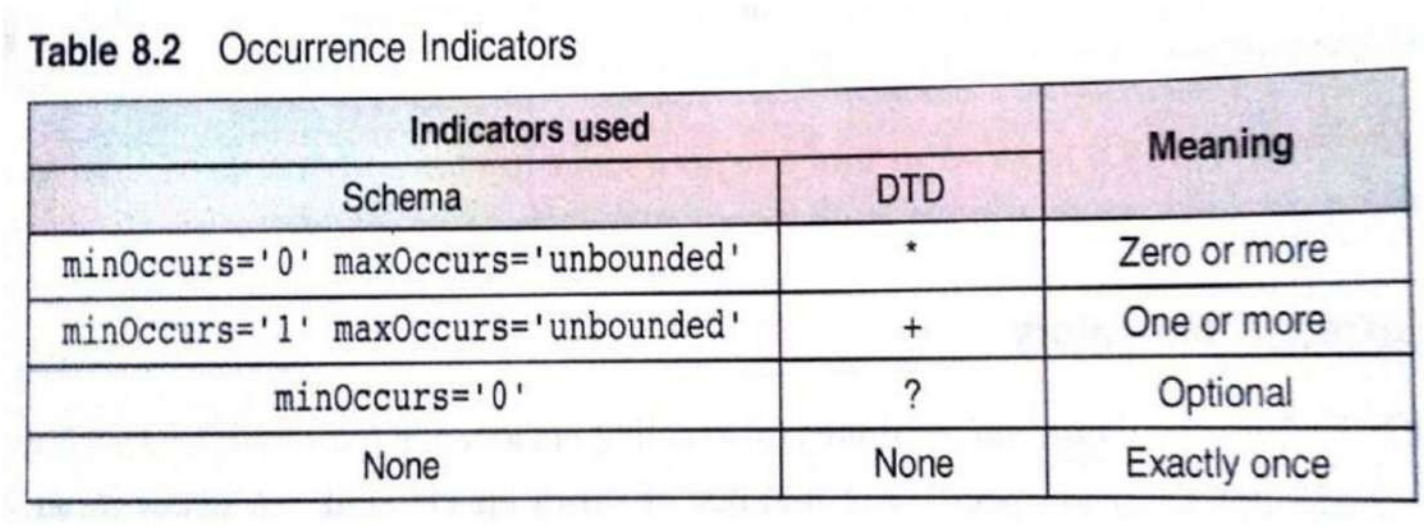


Table 8.2 Occurrence Indicators

Indicators used		**Meaning**
Schema	DTD	
`minOccurs='0' maxOccurs='unbounded'`	*	Zero or more
`minOccurs='1' maxOccurs='unbounded'`	+	One or more
`minOccurs='0'`	?	Optional
None	None	Exactly once

a.xml

```
<student>
<sname>Akansha</sname>
<subject>WT</subject>
<subject>ST</subject>
<subject>AA</subject>
<phone>9542893448</phone>
<phone>8736372632</phone>
</student>
```

a.xsd

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
<element name="student">
<complexType>
<sequence>
<element name="sname" type="string"/>
<element name="subject" type="string" minOccurs="2" maxOccurs="unbounded"/>
<element name="phone" type="string" maxOccurs="2"/>
</sequence>
</complexType>
</element>
</schema>
```

<name> elements occurs from **2 to n** number of times

Order Indicators

❑Order indicator is used to specify which order elements should occur.

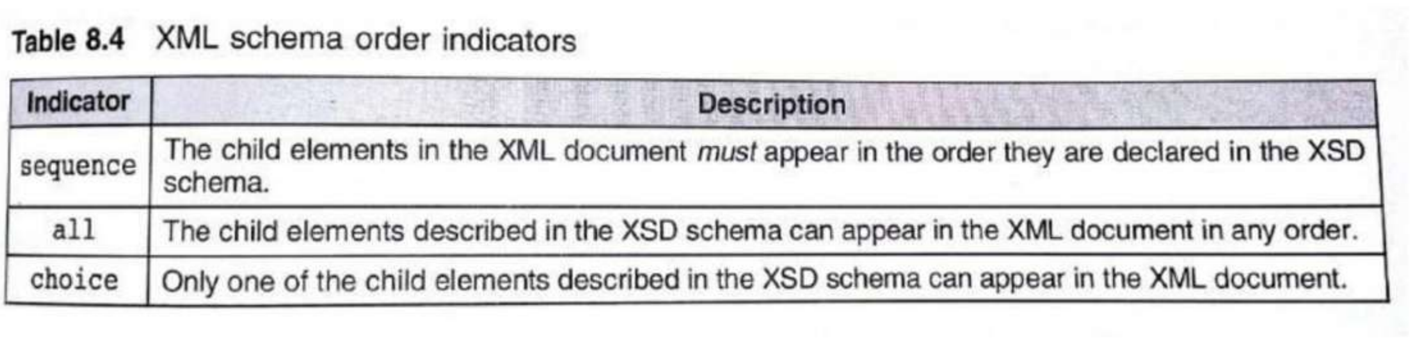


Table 8.4 XML schema order indicators

Indicator	Description
`sequence`	The child elements in the XML document *must* appear in the order they are declared in the XSD schema.
`all`	The child elements described in the XSD schema can appear in the XML document in any order.
`choice`	Only one of the child elements described in the XSD schema can appear in the XML document.

Sequence indicator

❑This indicator specifies that the element within an enclosing element must occur in the order specified in XML Schema.

a.xml

```
<student>
<fname>Ganesh</fname>
<mname>kumar</mname>
<lname>Reddy</lname>
</student>
```

a.xsd

```
<schema
xmlns="http://www.w3.org/2001/XMLSchema">
<element name="student">
<complexType>
<sequence>
<element name="fname" type="string"/>
<element name="mname" type="string"/>
<element name="lname" type="string"/>
</sequence>
</complexType>
</element>
</schema>
```

all indicator

- All indicator indicates element can appear in any order.

Choice indicator

- Choice indicator allows any one element from the set of element.
- We define many number of element but only one element is choosen.

a.xml

```
<student>
<DOB>1994-04-10</DOB>
</student>
```

a.xsd

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
<element name="student">
<complexType>
<choice>
<element name="DOB" type="date"/>
<element name="age" type="integer"/>
</choice>
</complexType>
</element>
</schema>
```

Defining Attributes in XML Schema

To declare an Attribute in XML Schema we use syntax:

```
<attribute name=”attribute_name” type=”attribute_datatype” />
```

- Attribute_datatype can be any simple type i.e built-in data type or any user derived type

Example:Student element have 2 attribute srollno and sname.

a.xml

```
<student srollno="510" sname="Akansha" />
```

a.xsd

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
<element name="student">
<complexType>
<attribute name="srollno" type="decimal" />
<attribute name="sname" type="string" />
</complexType>
</element>
</schema>
```

Attribute element Properties

We can specify whether an attribute is an mandatory attribute or optional attribute.

i) **Mandatory Attribute:**

- we declare an attribute with “use” attribute to specify an attribute as mandatory attribute.
- Attribute will appear one time or not at all, but no other number of times.

ii) **Optional Attribute:**

- If we don’t specify an attribute with “use” attribute, then the attribute is optional attribute.
- By default we have attribute as Optional attribute.

iii) **Default Values**

- Default values of an attribute is declared using the “default” attribute.
- When an attribute is declared with a default value: when we don’t use attribute default value is provided, If we use attribute with value then default value is overrided.

iv) **Fixed Values**

- We can fix a value for an attribute using “fixed” attribute.
- It is used to ensure that the attributes are always set to particular value.

Attribute element Properties

```
<student srollno="510" sname="Akansha" married="true" university="JNTUH" />
```

a.xml

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
<element name="student">
<complexType>
<attribute name="srollno" type="decimal" use="required" />
<attribute name="sname" type="string" />
<attribute name="married" type="boolean" default="false" />
<attribute name="university" type="string" fixed="JNTUH" />
</complexType>
</element>
</schema>
```

Mandatory

default

FIXED

a.xsd

Document Object Model

❑The DOM stands for Document object model.

❑Document object model parser allows us to READ and WRITE XML document by application.

❑In DOM entire XML Document is represented by different kind of nodes.

❑Entire XML document is represented by Document Node. Only one document node exist for each XML Document.

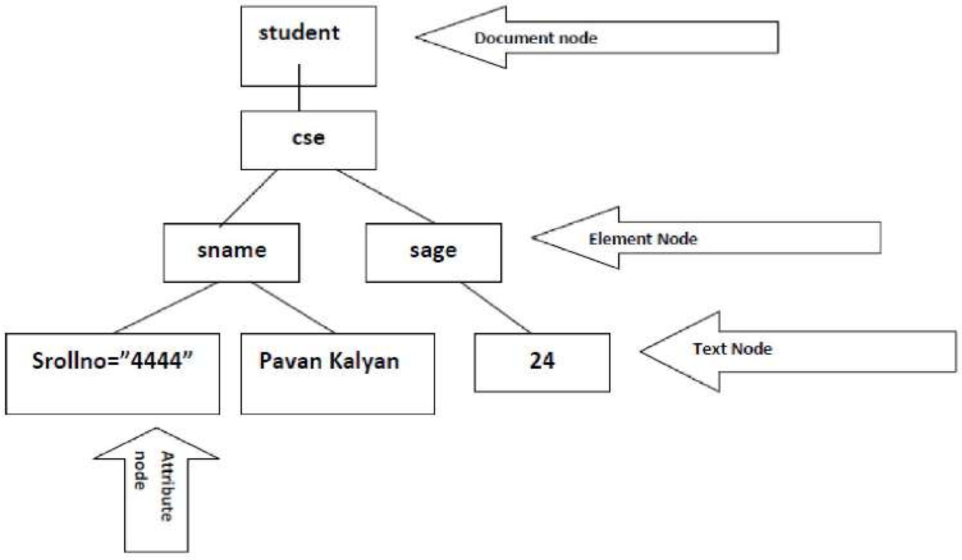
❑In DOM, entire XML document is a node:
o Entire XML document is represented by Document node.
o Elements are represented by element node.
o Attributes are represented by attribute node.
o Content/text between element is Text Node.

XML DOM tree with example

Consider the following XML Dcoumnet:

```
<student>
<cse>
<sname srollno="444">Pavan Kalyan</sname>
<sage>24</sage>
</cse>
</student>
```

The corresponding XML DOM tree is:



Document Object Model

Node Methods and Properties

Method	**Description**
getNodeType() method	JAVA provides the getNodeType() method which returns the node type, as a number. If it returns 1 then is an element node If it returns 2 then the node is an attribute node If it returns 3 then the node is a text node
getNodeName() method	getNodeName() method to read the name of a node.
getNodeValue() method	getNodeValue() method returns the value of the node.

Document Node Properties and Methods

- Document node refers to root node of XML document.
- The XML Document have single root element, which is returned by calling getDocumentElement() like:

```
Element rootElement = doc.getDocumentElement();
```

Difference between DOM and SAX Parser?

DOM	SAX
DOM stands for Document Object Model	SAX stands for Simple API for XML.
DOM parser is a tree-based parser.	SAX parser is a event-based parser
DOM parser load entire XML document in memory and creates a tree structure of XML document	SAX is an event based XML parser and doesn't load entire XML document into memory. It reads node by node.
DOM parser is best suited for small & medium size XML file.	SAX parser is best suited for large XML files
DOM parser can read , insert and delete a node in XML.	SAX parser can't inser or delete node, It can only read an XML.

Servlet

- It is small java program that runs inside JVM on the web server.
- It is used for developing dynamic web applications.

Life Cycle of Servlet

- ❑ init() -

  init()m method is used to create Servlet instance by server.
  init() method is used to initialize the servlet.
  init() is executed only once.

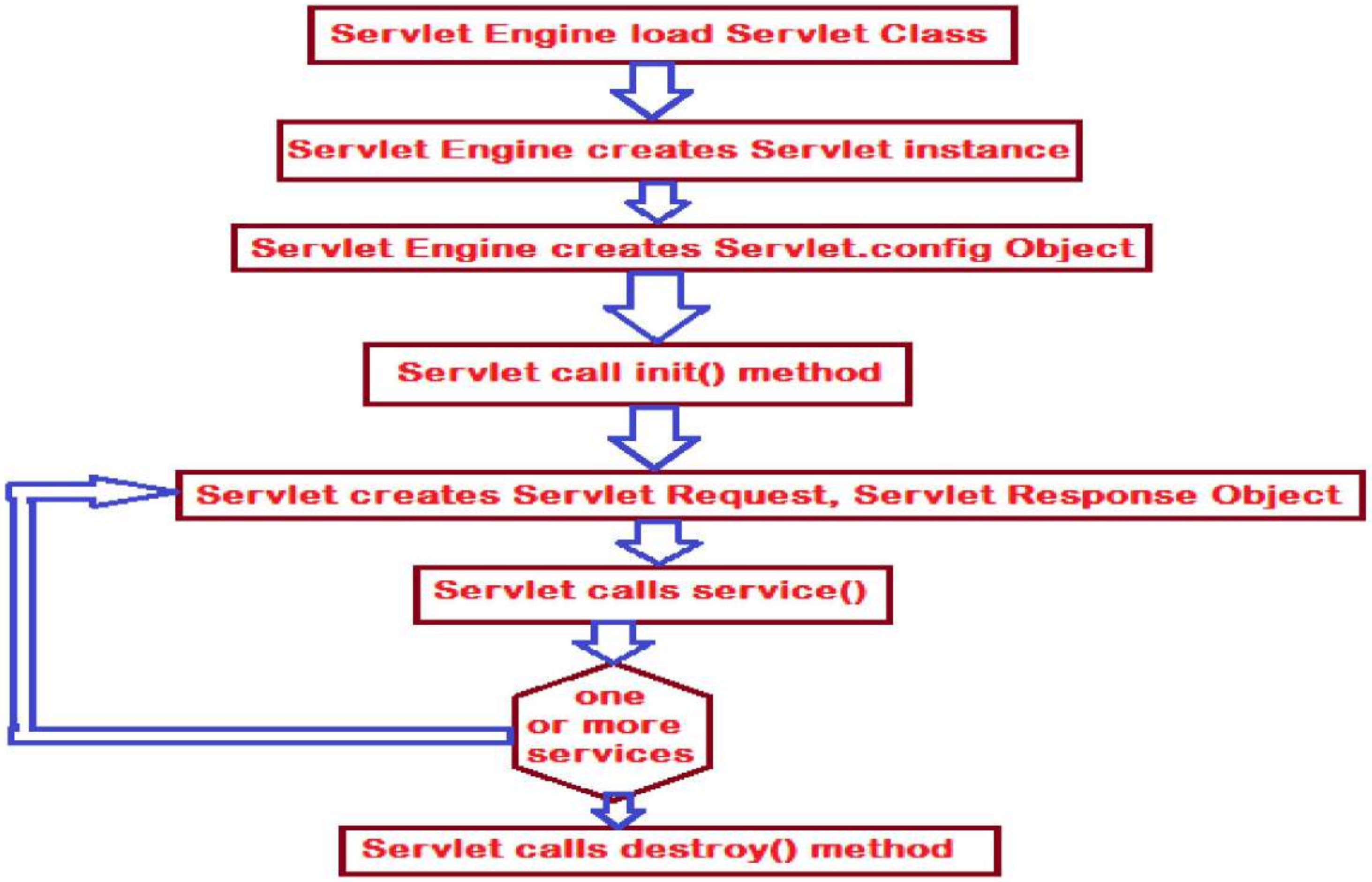
- ❑ service() –

  service() method processes the application request and waits for another request
  service() method is called each time when request for the servlet is received.

- ❑ destroy() -

  destroy() method is called at the end of servlet lifecycle.
  destroy() method cleans up any resources example thread, memory.

Life Cycle of Servlet


Servlet Engine load Servlet Class
Servlet Engine creates Servlet instance
Servlet Engine creates Servlet.config Object
Servlet call init() method
Servlet creates Servlet Request, Servlet Response Object
Servlet calls service()
one or more services
Servlet calls destroy() method