

XML Schema Overview

Jeff Offutt

<http://www.cs.gmu.edu/~offutt/>

SWE 432

**Design and Implementation of
Software for the Web**

XML Schemas

- The purpose of an XML Schema is to define the legal building blocks of an XML
- An XML document that conforms to a schema is said to be an instance of the schema
- Schemas have two purposes
 - Describe structure
 - Define data types of elements and attributes
- An XML Schema defines:
 - elements that can appear in a document
 - attributes that can appear in elements
 - child elements

Schema Can Define

- The sequence in which the child elements can appear
- The number of child elements
- Whether an element is empty or can include text
- Data types for elements and attributes
- Default values for elements and attributes

Schemas—XML for Book Example

```
<books>
  <book>
    <ISBN>0471043281</ISBN>
    <title>The Art of Software Testing</title>
    <author>Glen Myers</author>
    <publisher>Wiley</publisher>
    <price>50.00</price>
    <year>1979</year>
  </book>
</books>
```

- XML messages are defined by grammars
 - Schemas and DTDs
- Schemas can define many kinds of types
- Schemas include “facets,” which refine the grammar

XML Schema for Book Example

```
<xs:element name = "books">
  <xs:complexType>
    <xs:sequence>
      <xs:element name = "book" maxOccurs = "unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name = "ISBN" type = "xs:string"/>
            <xs:element name = "title" type = "xs:string"/>
            <xs:element name = "author" type = "xs:string"/>
            <xs:element name = "publisher" type = "xs:string"/>
            <xs:element name = "price" type = "priceType"/>
            <xs:element name = "year" type = "xs:int"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

facets

```
<xs:simpleType name = "priceType">
  <xs:restriction base = "xs:decimal">
    <xs:fractionDigits value = "2" />
    <xs:maxInclusive value = "1000.00" />
  </xs:restriction>
</xs:simpleType>
```

1 December 2011

© Offutt, 2011

5

XML <schema> Element

- <schema> is the root element of every XML Schema
- Attributes to <schema> can define "namespaces"
 - A namespace in XML is a URI that defines elements and attributes that can be used in an XML document
- Typical schema declaration:

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

1 December 2011

© Offutt, 2011

6

Schemas Must Be Well-formed

- A well-formed XML document is a document that conforms to the XML syntax rules:
 - Must begin with the XML declaration
 - Must have one unique root element
 - All start tags must match end-tags
 - XML tags are case sensitive
 - All elements must be closed
 - All elements must be properly nested
 - All attribute values must be quoted
 - XML entities must be used for special characters
- Just like XML, well-formed schemas may not be valid
 - may still contain errors

Schema Motivation: Data Communication

- When data is sent from a sender to a receiver it is essential that both parts have the same “expectations” about the content
- An XML element with a data type like this :
`<date type="date">2017-01-09</date>`
ensures a mutual understanding of the content because the XML data type date requires the format CCYY-MM-DD

Schema Data Types

- Simple and complex types
 - Simple : content is restricted to strings, no attributes or nesting
 - Complex : attributes and nesting allowed
- More than 40 simple types
 - Primitive : string, Boolean, float, time, anyURI
 - Derived builtin : byte, long, decimal, unsignedInt, positiveInteger, NMTOKEN
 - ...

1 December 2011

© Oflutt, 2011

9

Schema Data Types (2)

- User-defined : Specify restrictions on an existing type
 - Called *derived types*
 - Specify restrictions in terms of *facets*
- Facets are ways to *restrict values*
 - Integer facets : totalDigits, maxInclusive, maxExclusive, minInclusive, minExclusive, pattern, enumeration, whitespace
- Facets are based on regular expressions, and the possibilities are endless
- List of predefined data types :
<http://www.w3.org/TR/xmlschema-2/#built-in-datatypes>

1 December 2011

© Oflutt, 2011

10

Common Built-in Data Types

- xs:string – strings surrounded by quotes
- xs:decimal – numbers with decimal points
- xs:integer – integer values
- xs:boolean – true and false
- xs:date – YYYY/MM/DD
- xs:time – HH:MM:SS

XML Restrictions (Facets)

- Restrictions on XML elements are called facets
- Restrictions can be on :
 - Values : minInclusive, maxInclusive, minExclusive, maxExclusive
 - A set of values : enumeration
 - A series of values : pattern
 - Whitespace characters : whitespace
 - Length : length , maxLength, minLength, totalDigits, fractionDigits

Example XML Integer Facets

```
<xs:element name="age">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="16"/>
      <xs:maxInclusive value="34"/>
      <xs:maxLength value="2"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

1 December 2011

© Oflutt, 2011

13

XML Facets—Restricting Strings

```
<xs:simpleType name="isbnType">
  <xs:union>
    <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]{10}"/>
    </xs:restriction>
    <xs:simpleType>
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="TBD"/>
        <xs:enumeration value="NA"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:union>
</xs:simpleType>
```

Composed of numeric digits between 0 and 9

Exactly 10 digits

Name characters (letters, ., -, _, :)

Union of 2 separately defined rules

Additional values for isbnType

1 December 2011

© Oflutt, 2011

14

XML Facets—Enumerations

```
<xs:element name="car">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Hyundai"/>
      <xs:enumeration value="Toyota"/>
      <xs:enumeration value="Volvo"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

These and only
these 3 values

Schema Data Types Examples

```
<xs:simpleType name="pin">
  <xs:restriction base="xs:string">
    <xs:length value="5">
  </xs:restriction>
</xs:simpleType>
```

Exactly 5 digits
long

Complex Elements

- Complex elements can contain other elements and attributes
- The sequence is a “compositor” that defines an ordered sequence of sub-elements

```
<xs:element name="book">
  <xs:complexType>
    <xs:sequence>
      .../...
    </xs:sequence>
    .../...
  </xs:complexType>
</xs:element>
```

1 December 2011

© Offutt, 2011

17

Complex Elements

- Element-only : Other nested elements, but no text
- Text-only : Text, no nested elements
- Mixed content : Nested elements and text
- Empty : No content

1 December 2011

© Offutt, 2011

18

Schema Complex Element

```
<xs:complexType name="Address">  
  <xs:sequence>  
    <xs:element name="street1" type="xs:string" />  
    <xs:element name="street2" type="xs:string"  
      minOccurs="0" maxOccurs="unbounded"/>  
    <xs:element name="city" type="xs:string" />  
    <xs:element name="state" type="xs:string" />  
    <xs:element name="zip" type="xs:decimal" />  
  </xs:sequence>  
</xs:complexType>  
  
<xs:element name = "mailingAddress" type="Address">  
<xs:element name = "billingAddress" type="Address">
```

Elements must appear in this order

Could be constrained as a simple type

1 December 2011

© Offutt, 2011

19

XML Example

This XML is valid by the previous schema definition

```
<? xml version = "1.0"?>  
<Address>  
  <street1>4400 University Blvd</street1>  
  <street2>MS 4A5</street2>  
  <city>Fairfax</city>  
  <state>Virginia</state>  
  <zip>22030</zip>  
</Address>
```

1 December 2011

© Offutt, 2011

20

Validation

- Schemas can be used to validate XML documents
 - By validating XML parsers
 - By standalone tools such as xsv
<http://www.ltg.ed.ac.uk/~ht/xsv-status.html>
- An XML document must be correct according to the XML syntax rules
- An XML document may be valid according to a particular schema or DTD

XML Example : EMail

```
<?xml version="1.0"?>
<note>
  <to>George Burdell</to>
  <from>Buzz</from>
  <heading>Reminder</heading>
  <body>How was your weekend?</body>
</note>
```

XML Schema for EMail

```
<?xml version="1.0"?>
<note>
  <to>George Burdell</to>
  <from>Buzz</from>

  <heading>Reminder</heading>
  >
  <body>How are you?</body>
</note>
```

```
<?xml version="1.0"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema>
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <element name="to" type="xs:string"/>
        <element name="from" type="xs:string"/>
        <element name="heading" type="xs:string"/>
        <element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

1 December 2011

© Offutt, 2011

23

Referencing Schema for EMail

```
<?xml version="1.0"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema>
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <element name="to" type="xs:string"/>
        <element name="from" type="xs:string"/>
        <element name="heading" type="xs:string"/>
        <element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```
<?xml version="1.0"?>
<note xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="http://www.w3schools.com/schema/note.xsd">
  <to>George Burdell</to>
  <from>Buzz</from>
  <heading>Reminder</heading>
  <body>How are you?</body>
</note>
```

1 December 2011

© Offutt, 2011

24

XML Schema Summary

- XML parsers can be downloaded for free
 - They read an XML file and put the contents in a tree whose elements can be accessed through APIs
- Use XML for :
 - Traditional data processing to encode the data
 - Document-driven programs
- Archiving of data for later use