

XML Schema

Andy Clark

What is it?

- A grammar definition language
 - Like DTDs but better
 - Uses XML syntax
 - Defined by W3C
- Primary features
 - Datatypes
 - e.g. integer, float, date, etc...
 - More powerful content models
 - e.g. namespace-aware, type derivation, etc...

XML Schema Types

- Simple types
 - Basic datatypes
 - Can be used for attributes *and* element text
 - Extendable
- Complex types
 - Defines structure of elements
 - Extendable
- Types can be named or “anonymous”

Simple Types

- DTD datatypes
 - Strings, ID/IDREF, NMTOKEN, etc...
- Numbers
 - Integer, long, float, double, etc...
- Other
 - Binary (base64, hex)
 - QName, URI, date/time
 - etc...

Deriving Simple Types

- Apply facets
 - Specify enumerated values
 - Add restrictions to data
 - Restrict lexical space
 - Allowed length, pattern, etc...
 - Restrict value space
 - Minimum/maximum values, etc...
- Extend by list or union

A Simple Type Example (1 of 4)

- Integer with value (1234, 5678]

```
01 <xsd:simpleType name='MyInteger'>
02   <xsd:restriction base='xsd:integer'>
03     <xsd:minExclusive value='1234'/>
04     <xsd:maxInclusive value='5678'/>
05   </xsd:restriction>
06 </xsd:simpleType>
```

A Simple Type Example (2 of 4)

- Integer with value (1234, 5678]

```
01 <xsd:simpleType name='MyInteger'>
02   <xsd:restriction base='xsd:integer'>
03     <xsd:minExclusive value='1234'/>
04     <xsd:maxInclusive value='5678'/>
05   </xsd:restriction>
06 </xsd:simpleType>
```

A Simple Type Example (3 of 4)

- Integer with value (1234, 5678]

```
01 <xsd:simpleType name='MyInteger'>
02   <xsd:restriction base='xsd:integer'>
03     <xsd:minExclusive value='1234'/>
04     <xsd:maxInclusive value='5678'/>
05   </xsd:restriction>
06 </xsd:simpleType>
```


A Simple Type Example (4 of 4)

- Validating integer with value (1234, 5678]

| | | |
|----|---|---------|
| 01 | <code><data xsi:type='MyInteger'></data></code> | INVALID |
| 02 | <code><data xsi:type='MyInteger'>Andy</data></code> | INVALID |
| 03 | <code><data xsi:type='MyInteger'>-32</data></code> | INVALID |
| 04 | <code><data xsi:type='MyInteger'>1233</data></code> | INVALID |
| 05 | <code><data xsi:type='MyInteger'>1234</data></code> | INVALID |
| 06 | <code><data xsi:type='MyInteger'>1235</data></code> | |
| 07 | <code><data xsi:type='MyInteger'>5678</data></code> | |
| 08 | <code><data xsi:type='MyInteger'>5679</data></code> | INVALID |

Complex Types

- Element content models
 - Simple
 - Mixed
 - Unlike DTDs, elements in mixed content can be ordered
 - Sequences and choices
 - Can contain nested sequences and choices
 - All
 - All elements required but order is *not* important

A Complex Type Example (1 of 5)

- Mixed content that allows , <i>, and <u>

```
01 <xsd:complexType name='RichText' mixed='true'>
02   <xsd:choice minOccurs='0' maxOccurs='unbounded'>
03     <xsd:element name='b' type='RichText' />
04     <xsd:element name='i' type='RichText' />
05     <xsd:element name='u' type='RichText' />
06   </xsd:choice>
07 </xsd:complexType>
```

A Complex Type Example (2 of 5)

- Mixed content that allows , <i>, and <u>

```
01 <xsd:complexType name='RichText' mixed='true'>
02   <xsd:choice minOccurs='0' maxOccurs='unbounded'>
03     <xsd:element name='b' type='RichText'/>
04     <xsd:element name='i' type='RichText'/>
05     <xsd:element name='u' type='RichText'/>
06   </xsd:choice>
07 </xsd:complexType>
```

A Complex Type Example (3 of 5)

- Mixed content that allows , <i>, and <u>

```
01 <xsd:complexType name='RichText' mixed='true'>
02   <xsd:choice minOccurs='0' maxOccurs='unbounded'>
03     <xsd:element name='b' type='RichText'/>
04     <xsd:element name='i' type='RichText'/>
05     <xsd:element name='u' type='RichText'/>
06   </xsd:choice>
07 </xsd:complexType>
```

A Complex Type Example (4 of 5)

- Mixed content that allows , <i>, and <u>

```
01 <xsd:complexType name='RichText' mixed='true'>
02   <xsd:choice minOccurs='0' maxOccurs='unbounded'>
03     <xsd:element name='b' type='RichText'/>
04     <xsd:element name='i' type='RichText'/>
05     <xsd:element name='u' type='RichText'/>
06   </xsd:choice>
07 </xsd:complexType>
```

A Complex Type Example (5 of 5)

- Validation of RichText

| | | |
|----|--|---------|
| 01 | <code><content xsi:type='RichText'></content></code> | |
| 02 | <code><content xsi:type='RichText'>Andy</content></code> | |
| 03 | <code><content xsi:type='RichText'>XML is <i>awesome</i>.</content></code> | |
| 04 | <code><content xsi:type='RichText'>bold</content></code> | INVALID |
| 05 | <code><content xsi:type='RichText'><foo/></content></code> | INVALID |

Flexing Our Muscles

- The task:
 - Converting a DTD grammar to XML Schema
- Defining datatypes
 - Beyond what DTDs allow
 - More precise control over “string” values
- Defining content models

Converting DTD (1 of 27)

- Original DTD grammar

01 <!ELEMENT order (item)+ >

02
03 <!ELEMENT item (name,price) >

04 <!ATTLIST item code NMTOKEN #REQUIRED >

05
06 <!ELEMENT name (#PCDATA) >

07
08 <!ELEMENT price (#PCDATA) >

09 <!ATTLIST price currency NMTOKEN 'USD' >

Converting DTD (2 of 27)

- Original DTD grammar

```
01      <!ELEMENT order      (item)+ >
02
03      <!ELEMENT item      (name,price) >
04      <!ATTLIST  item      code      NMTOKEN  #REQUIRED >
05
06      <!ELEMENT name      (#PCDATA) >
07
08      <!ELEMENT price      (#PCDATA) >
09      <!ATTLIST  price      currency  NMTOKEN  'USD' >
```

Converting DTD (3 of 27)

- Original DTD grammar

01 <!ELEMENT order (item)+ >

02
03 <!ELEMENT item (name,price) >

04 <!ATTLIST item code NMTOKEN #REQUIRED >

05
06 <!ELEMENT name (#PCDATA) >

07
08 <!ELEMENT price (#PCDATA) >

09 <!ATTLIST price currency NMTOKEN 'USD' >

Converting DTD (4 of 27)

- Create XML Schema document
 - Grammar with *no* target namespace

01

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

nn

```
</xsd:schema>
```

Converting DTD (5 of 27)

- Create XML Schema document
 - Grammar *with* target namespace

```
01 <xsd:schema xmlns:xsd='http://www.w3.org/2001/XMLSchema'  
02     xmlns:a='NS'  
03     targetNamespace='NS'>
```

```
nn </xsd:schema>
```

- *Note:* It's important to bind the namespace to a prefix because references to names within the XML Schema must be fully qualified.

Converting DTD (6 of 27)

- Declare elements

```
01      <xsd:schema xmlns:xsd='http://www.w3.org/2001/XMLSchema'>
02
03      <xsd:element name='order' type='Order'/>
04      <xsd:element name='item' type='Item'/>
05      <xsd:element name='name' type='Name'/>
06      <xsd:element name='price' type='Price'/>
```

Converting DTD (7 of 27)

- Declare elements

```
01 <xsd:schema xmlns:xsd='http://www.w3.org/2001/XMLSchema'>
02
03 <xsd:element name='order' type='Order'/>
04 <xsd:element name='item' type='Item'/>
05 <xsd:element name='name' type='Name'/>
06 <xsd:element name='price' type='Price'/>
```

- *Note:* In general, it's a good idea to reference named types and avoid anonymous (or “inline”) types. [Beware of qualified form option.]

Converting DTD (8 of 27)

- Define type for <order> element

```
08      <!-- <!ELEMENT item (item)+> -->
09      <xsd:complexType name='Order'>
10        <xsd:sequence>
11          <xsd:element ref='item' minOccurs='1' maxOccurs='unbounded' />
12        </xsd:sequence>
13      </xsd:complexType>
```


Converting DTD (9 of 27)

- Define type for <order> element

```
08 <!-- <!ELEMENT item (item)+> -->
09 <xsd:complexType name='Order'>
10   <xsd:sequence>
11     <xsd:element ref='item' minOccurs='1' maxOccurs='unbounded' />
12   </xsd:sequence>
13 </xsd:complexType>
```

Converting DTD (10 of 27)

- Define type for <order> element

```
08 <!-- <!ELEMENT item (item)+> -->
09 <xsd:complexType name='Order'>
10   <xsd:sequence>
11     <xsd:element ref='item' minOccurs='1' maxOccurs='unbounded' />
12   </xsd:sequence>
13 </xsd:complexType>
```

Converting DTD (11 of 27)

- Define type for <item> element

```
15 <!-- <!ELEMENT item (name,price)> -->
16 <xsd:complexType name='Item'>
17   <xsd:sequence>
18     <xsd:element ref='name'>
19     <xsd:element ref='price'>
20   </xsd:sequence>
21 <!-- <!ATTLIST item code NMTOKEN #REQUIRED> -->
22 <xsd:attribute name='code'>
23   <xsd:simpleType>
24     <xsd:restriction base='xsd:string'>
25       <xsd:pattern value='[A-Z]{2}\d{3}'>
26     </xsd:restriction>
27   </xsd:simpleType>
28 </xsd:attribute>
29 </xsd:complexType>
```

Converting DTD (12 of 27)

- Define type for <item> element

```
15 <!-- <!ELEMENT item (name,price)> -->
16 <xsd:complexType name='Item'>
17   <xsd:sequence>
18     <xsd:element ref='name'>
19     <xsd:element ref='price'>
20   </xsd:sequence>
21 <!-- <!ATTLIST item code NMTOKEN #REQUIRED> -->
22 <xsd:attribute name='code'>
23   <xsd:simpleType>
24     <xsd:restriction base='xsd:string'>
25       <xsd:pattern value='[A-Z]{2}\d{3}'>
26     </xsd:restriction>
27   </xsd:simpleType>
28 </xsd:attribute>
29 </xsd:complexType>
```

Converting DTD (13 of 27)

- Define type for <item> element

```
15 <!-- <!ELEMENT item (name,price)> -->
16 <xsd:complexType name='Item'>
17   <xsd:sequence>
18     <xsd:element ref='name'/'>
19     <xsd:element ref='price'/'>
20   </xsd:sequence>
21 <!-- <!ATTLIST item code NMTOKEN #REQUIRED> -->
22 <xsd:attribute name='code'>
23   <xsd:simpleType>
24     <xsd:restriction base='xsd:string'>
25       <xsd:pattern value='[A-Z]{2}\d{3}'/'>
26     </xsd:restriction>
27   </xsd:simpleType>
28 </xsd:attribute>
29 </xsd:complexType>
```

Converting DTD (14 of 27)

- Define type for <item> element

```
15 <!-- <!ELEMENT item (name,price)> -->
16 <xsd:complexType name='Item'>
17   <xsd:sequence>
18     <xsd:element ref='name'>
19     <xsd:element ref='price'>
20   </xsd:sequence>
21 <!-- <!ATTLIST item code NMTOKEN #REQUIRED> -->
22 <xsd:attribute name='code'>
23   <xsd:simpleType>
24     <xsd:restriction base='xsd:string'>
25       <xsd:pattern value='[A-Z]{2}\d{3}'>
26     </xsd:restriction>
27   </xsd:simpleType>
28 </xsd:attribute>
29 </xsd:complexType>
```

Converting DTD (15 of 27)

- Define type for <item> element

```
15 <!-- <!ELEMENT item (name,price)> -->
16 <xsd:complexType name='Item'>
17   <xsd:sequence>
18     <xsd:element ref='name'/>
19     <xsd:element ref='price'/>
20   </xsd:sequence>
21 <!-- <!ATTLIST item code NMTOKEN #REQUIRED> -->
22 <xsd:attribute name='code'>
23   <xsd:simpleType>
24     <xsd:restriction base='xsd:string'>
25       <xsd:pattern value='[A-Z]{2}\d{3}'/>
26     </xsd:restriction>
27   </xsd:simpleType>
28 </xsd:attribute>
29 </xsd:complexType>
```

Converting DTD (16 of 27)

- Define type for <name> element

```
31 <!-- <!ELEMENT name (#PCDATA)> -->  
32 <xsd:simpleType name='Name'  
33   <xsd:restriction base='xsd:string'/>  
34 </xsd:simpleType>
```


Converting DTD (17 of 27)

- Define type for <name> element

```
31 <!-- <!ELEMENT name (#PCDATA)> -->  
32 <xsd:simpleType name='Name'  
33   <xsd:restriction base='xsd:string'/>  
34 </xsd:simpleType>
```

Converting DTD (18 of 27)

- Define type for <name> element

```
31 <!-- <!ELEMENT name (#PCDATA)> -->  
32 <xsd:simpleType name='Name'  
33 <xsd:restriction base='xsd:string'/>  
34 </xsd:simpleType>
```

Converting DTD (19 of 27)

- Define type for <price> element

```
36 <!-- <!ELEMENT price (#PCDATA)> -->
37 <xsd:complexType name='Price'>
38   <xsd:simpleContent>
39     <xsd:extension base='NonNegativeDouble'>
40       <!-- <!ATTLIST price currency NMTOKEN 'USD'> -->
41       <xsd:attribute name='currency' default='USD'>
42         <xsd:simpleType>
43           <xsd:restriction base='xsd:string'>
44             <xsd:pattern value='[A-Z]{3}'/>
45           </xsd:restriction>
46         </xsd:simpleType>
47       </xsd:attribute>
48     </xsd:extension>
49   </xsd:simpleContent>
50 </xsd:complexType>
```

Converting DTD (20 of 27)

- Define type for <price> element

```
36 <!-- <!ELEMENT price (#PCDATA)> -->
37 <xsd:complexType name='Price'>
38   <xsd:simpleContent>
39     <xsd:extension base='NonNegativeDouble'>
40       <!-- <!ATTLIST price currency NMToken 'USD'> -->
41       <xsd:attribute name='currency' default='USD'>
42         <xsd:simpleType>
43           <xsd:restriction base='xsd:string'>
44             <xsd:pattern value='[A-Z]{3}'/>
45           </xsd:restriction>
46         </xsd:simpleType>
47       </xsd:attribute>
48     </xsd:extension>
49   </xsd:simpleContent>
50 </xsd:complexType>
```

Converting DTD (21 of 27)

- Define type for <price> element

```
36 <!-- <!ELEMENT price (#PCDATA)> -->
37 <xsd:complexType name='Price'>
38   <xsd:simpleContent>
39     <xsd:extension base='NonNegativeDouble'>
40       <!-- <!ATTLIST price currency NMTOKEN 'USD'> -->
41       <xsd:attribute name='currency' default='USD'>
42         <xsd:simpleType>
43           <xsd:restriction base='xsd:string'>
44             <xsd:pattern value='[A-Z]{3}'/>
45           </xsd:restriction>
46         </xsd:simpleType>
47       </xsd:attribute>
48     </xsd:extension>
49   </xsd:simpleContent>
50 </xsd:complexType>
```

Converting DTD (22 of 27)

- Define type for <price> element

```
36 <!-- <!ELEMENT price (#PCDATA)> -->
37 <xsd:complexType name='Price'>
38   <xsd:simpleContent>
39     <xsd:extension base='NonNegativeDouble'>
40       <!-- <!ATTLIST price currency NMTOKEN 'USD'> -->
41       <xsd:attribute name='currency' default='USD'>
42         <xsd:simpleType>
43           <xsd:restriction base='xsd:string'>
44             <xsd:pattern value='[A-Z]{3}'/>
45           </xsd:restriction>
46         </xsd:simpleType>
47       </xsd:attribute>
48     </xsd:extension>
49   </xsd:simpleContent>
50 </xsd:complexType>
```

Converting DTD (23 of 27)

- Define type for <price> element

```
36 <!-- <!ELEMENT price (#PCDATA)> -->
37 <xsd:complexType name='Price'>
38   <xsd:simpleContent>
39     <xsd:extension base='NonNegativeDouble'>
40       <!-- <!ATTLIST price currency NMTOKEN 'USD'> -->
41       <xsd:attribute name='currency' default='USD'>
42         <xsd:simpleType>
43           <xsd:restriction base='xsd:string'>
44             <xsd:pattern value='[A-Z]{3}'/>
45           </xsd:restriction>
46         </xsd:simpleType>
47       </xsd:attribute>
48     </xsd:extension>
49   </xsd:simpleContent>
50 </xsd:complexType>
```

Converting DTD (24 of 27)

- Define simple type for use with Price type

```
52     <xsd:simpleType name='NonNegativeDouble'>
53         <xsd:restriction base='xsd:double'>
54             <xsd:minInclusive value='0.00'/>
55         </xsd:restriction>
56     </xsd:simpleType>
57
58 </xsd:schema>
```


Converting DTD (25 of 27)

- Define simple type for use with Price type

```
52 <xsd:simpleType name='NonNegativeDouble'>
53   <xsd:restriction base='xsd:double'>
54     <xsd:minInclusive value='0.00'/>
55   </xsd:restriction>
56 </xsd:simpleType>
57
58 </xsd:schema>
```

Converting DTD (26 of 27)

- Define simple type for use with Price type

```
52 <xsd:simpleType name='NonNegativeDouble'>
53   <xsd:restriction base='xsd:double'>
54     <xsd:minInclusive value='0.00'/>
55   </xsd:restriction>
56 </xsd:simpleType>
57
58 </xsd:schema>
```

Converting DTD (27 of 27)

- Size comparison

DTD

```
<!ELEMENT order (item)+>
<!ELEMENT item (name,price)
<!ATTLIST item code NMTOKEN #REQUIRED>
<!ELEMENT name (#PCDATA)
<!ELEMENT price (#PCDATA)
<!ATTLIST price currency NMTOKEN 'USD'>
```

XML Schema

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

  <xsd:element name="order" type="Order"/>
  <xsd:element name="item" type="Item"/>
  <xsd:element name="name" type="Name"/>
  <xsd:element name="price" type="Price"/>

  <!-- <ELEMENT order (item)+ -->
  <xsd:complexType name="Order">
    <xsd:sequence>
      <xsd:element ref="item" minOccurs="1" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <!-- <ELEMENT item (name,price) -->
  <xsd:complexType name="Item">
    <xsd:sequence>
      <xsd:element ref="name"/>
      <xsd:element ref="price"/>
    </xsd:sequence>
  </xsd:complexType>

  <!-- <ATTLIST item code NMTOKEN #REQUIRED -->
  <xsd:attribute name="code">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:pattern value="[A-Z]{2}d{3}"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  </xsd:complexType>

  <!-- <ELEMENT name (#PCDATA) -->
  <xsd:simpleType name="Name">
    <xsd:restriction base="xsd:string"/>
  </xsd:simpleType>

  <!-- <ELEMENT price (#PCDATA) -->
  <xsd:complexType name="Price">
    <xsd:simpleContent>
      <xsd:extension base="NonNegativeDouble">
        <!-- <ATTLIST price currency NMTOKEN 'USD' -->
        <xsd:attribute name="currency" default="USD">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:pattern value="[A-Z]{3}"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>

  <xsd:simpleType name="NonNegativeDouble">
    <xsd:restriction base="xsd:double">
      <xsd:minInclusive value="0.00"/>
    </xsd:restriction>
  </xsd:simpleType>

</xsd:schema>
```

**Note the verbosity
of the XML Schema
version of the
grammar.
(I'm using a 4 pt. font!)**

Using XML Schema Grammar to validate Document

- Bind XML Schema “instance” namespace
 - e.g.
xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
- Grammar with *no* target namespace
 - e.g. xsi:noNamespaceSchemaLocation='grammar.xsd'
- Grammar *with* target namespace
 - Namespace URI and systemId pairs
 - e.g. xsi:schemaLocation='NS grammar.xsd'

Example Document (1 of 3)

- Grammar with *no* target namespace

```
01 <?xml version='1.0' encoding='Shift_JIS'?>
02 <order xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
03     xsi:noNamespaceSchemaLocation='grammar.xsd'>
04     <item code='BK123'>
05         <name>ウオムバットを育てる</name>
06         <price currency='JPN'>5460</price>
07     </item>
08 </order>
```

Example Document (2 of 3)

- Grammar *with* target namespace (1 of 2)

```
01 <?xml version='1.0' encoding='Shift_JIS'?>
02 <order xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
03     xsi:schemaLocation='NS grammar.xsd'
04     xmlns='NS'>
05     <item code='BK123'>
06         <name>ウオムバットを育てる</name>
07         <price currency='JPN'>5460</price>
08     </item>
09 </order>
```

Example Document (3 of 3)

- Grammar *with* target namespace (2 of 2)

```
01 <?xml version='1.0' encoding='Shift_JIS'?>
02 <order xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
03     xsi:schemaLocation='NS grammar.xsd'
04     xmlns='NS'>
05   <item code='BK123'>
06     <name>ウオムバットを育てる</name>
07     <price currency='JPN'>5460</price>
08   </item>
09 </order>
```

Useful Links

- XML Schema Specification
 - Part 0: <http://www.w3.org/TR/xmlschema-0/>
 - Part 1: <http://www.w3.org/TR/xmlschema-1/>
 - Part 2: <http://www.w3.org/TR/xmlschema-2/>
- XML.com articles
 - <http://www.xml.com/pub/a/2001/06/06/schemasimple.html>

XML Schema

Andy Clark
