CS 683 Emerging Technologies
Spring Semester, 2003
Doc 9 XML Namespaces & Schemas

Contents

References

Learning XML, Erik Ray, O'Reilly, 2001

Namespaces in XML, http://www.w3.org/TR/REC-xml-names/

XML Schema Part 0: A Primer, http://www.w3.org/TR/xmlschema-0/

XML Schema Part 1: Structures, http://www.w3.org/TR/xmlschema-1/

XML Schema Part 2: Datatypes, http://www.w3.org/TR/xmlschema-2/

Namespaces

An XML namespace is a group of

- Elements
- Attributes

XML namespaces allows mixing of elements from different DTDs

Namespace Tag name

Format

<namespacePrefix:tagName>

Example

<sdsu:from>Roger</sdsu:from>

Declaring a Namespace

Format

xmlns:namespaceName = ‘url’

The url

- Is a unique identifier for the namespace
- Sometimes points to the defining DTD
- Sometimes just point to definers web site
- Can point to nothing

Example

```
<sdsu:from xmlns:sdsu=”http://www.sdsu.edu”>
  Roger</sdsu:from>
```

Sample Use of Namespace

```
<?xml version="1.0" ?>
<whitney:greetings  xmlns:whitney=”http://www.eli.sdsu”>
  <whitney:from>
       <whitney:firstname>Roger</whitney:firstName>
  </whitney:from>
  <whitney:to>
       <whitney:firstname>John</whitney:firstName>
  </whitney:to>
  <whitney:message>Hi</whitney:message>
</whitney:greetings>
```

Using Multiple Namespaces

```
<?xml version="1.0" ?>
<whitney:greetings  xmlns:whitney=”http://www.eli.sdsu”
        xmlns:godot=”http://www.waiting.com”>
  <whitney:from>
        <godot:firstname>Roger</godot:firstName>
  </whitney:from>
  <whitney:to>
        <godot:firstname>John</godot:firstName>
  </whitney:to>
  <whitney:message>Hi</whitney:message>
</whitney:greetings>
```

Limiting a namespace to part of document

```
<?xml version="1.0" ?>
<whitney:greetings  xmlns:whitney=”http://www.eli.sdsu”
        >
  <godot:from xmlns:godot=”http://www.waiting.com”>
        <godot:firstname>Roger</godot:firstName>
  </godot:from>
  <whitney:to>
        <whitney:firstname>John</whitney:firstName>
  </whitney:to>
  <whitney:message>Hi</whitney:message>
</whitney:greetings>
```

Default Namespace

All tags are in the ”http://www.eli.sdsu” namespace

```
<?xml version="1.0" ?>
<greetings  xmlns=”http://www.eli.sdsu”>
  <from>
      <firstname>Roger</firstName>
  </from>
  <to>
      <firstname>John</firstName>
  </to>
  <message>Hi<message>
<greetings>
```

Multiple and Default Namespace

```
<?xml version="1.0" ?>
<greetings  xmlns=”http://www.eli.sdsu”
        xmlns:godot=”http://www.waiting.com”>
  <from>
        <godot:firstname>Roger</godot:firstName>
  </from>
  <to>
        <firstname>John</firstName>
  </to>
  <message>Hi</message>
</greetings>
```

Schemas

Using a DTD to define an element

- The content is a string
- There is no way restrict the content

<age>21</age>
<age>The cat in the hat is Back</age>

Schemas are a way to define XML documents

- Allow element content to have a type
- Allow element content to be restricted

Defining Schemas

Schemas are defined using XML

- Don’t need DTDs for defining types
- Plain XML parsers do not validate XML documents

W3C Schema

Main W3C site on Schema

http://www.w3.org/XML/Schema

XML Schema Part 0: A Primer

http://www.w3.org/TR/xmlschema-0/

XML Schema Part 1: Structures

http://www.w3.org/TR/xmlschema-1/

XML Schema Part 2: Datatypes

http://www.w3.org/TR/xmlschema-2/

Sample Schema

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

  <xsd:element name="address" type="USAddress"/>

  <xsd:complexType name="USAddress">
    <xsd:sequence>
      <xsd:element name="name"   type="xsd:string"/>
      <xsd:element name="street" type="xsd:string"/>
      <xsd:element name="city"   type="xsd:string"/>
      <xsd:element name="state"  type="StateAbrreviation"/>
      <xsd:element name="zip"    type="xsd:decimal"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:simpleType name="StateAbrreviation">
    <xsd:restriction base="xsd:string">
      <xsd:length value="2" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

XML Using the Schema

```
<?xml version="1.0"?>
  <address>
     <name>Alice Smith</name>
        <street>123 Maple Street</street>
        <city>Mill Valley</city>
        <state>CA</state>
        <zip>90952</zip>
  </address>
```

Note the XML does not explicitly link to the schema defined on the last page. Future examples will show how to link to the schema

Simple Types

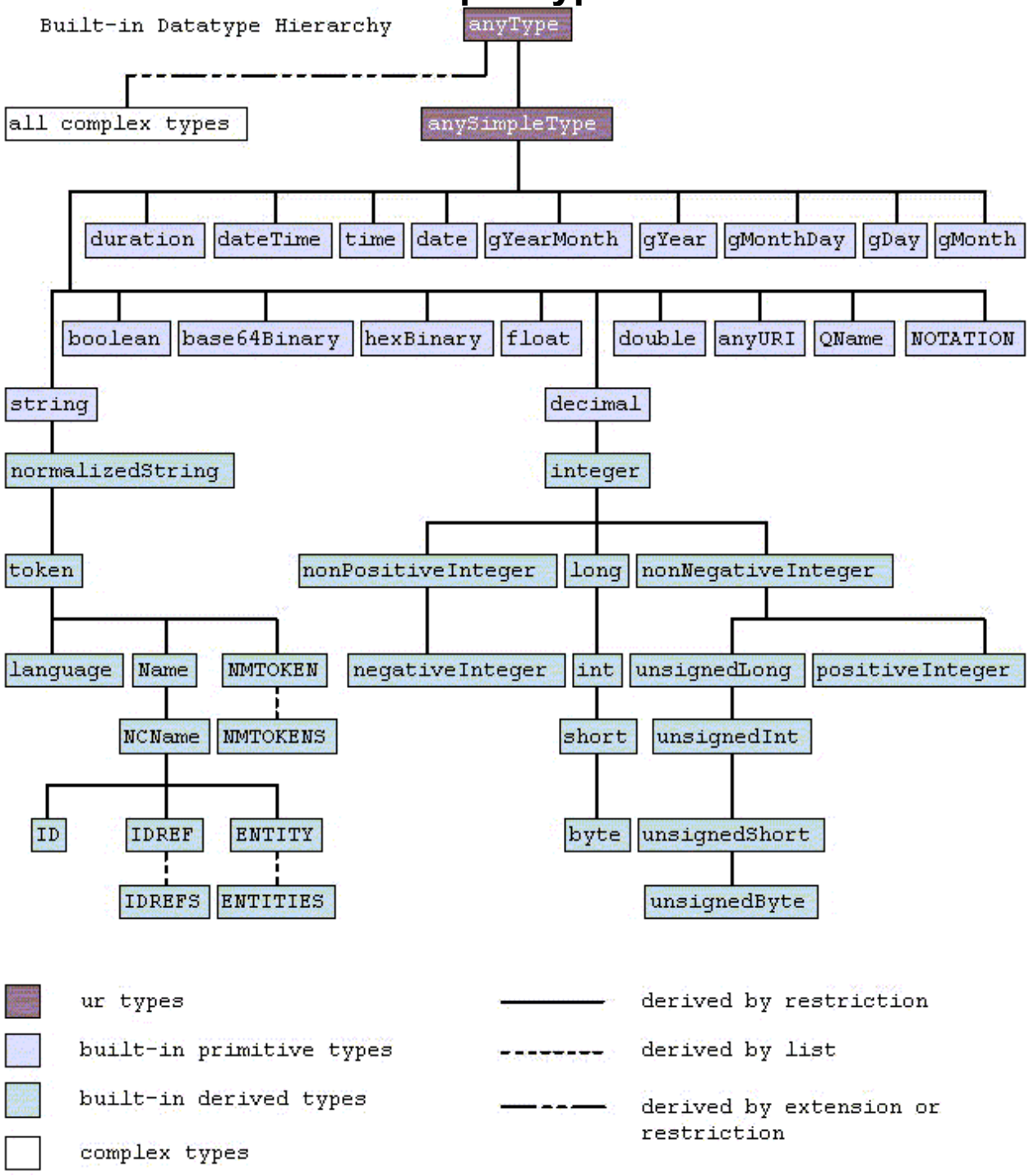

Diagram is from http://www.w3.org/TR/xmlschema-2/

Some Simple Schema Types

Simple Type	Examples (delimited by commas)
string	Confirm this is electric
normalizedString	Confirm this is electric
byte	-1, 126
unsignedByte	0, 126
base64Binary	GpM7
hexBinary	0FB7
integer	-126789, -1, 0, 1, 126789
positiveInteger	1, 126789
negativeInteger	-126789, -1
nonNegativeInteger	0, 1, 126789
nonPositiveInteger	-126789, -1, 0
int	-1, 126789675
unsignedInt	0, 1267896754
long	-1, 12678967543233
unsignedLong	0, 12678967543233
short	-1, 12678
unsignedShort	0, 12678
decimal	-1.23, 0, 123.4, 1000.00
float	-INF, -1E4, -0, 0, 12.78E-2, 12, INF, NaN
double	-INF, -1E4, -0, 0, 12.78E-2, 12, INF, NaN
boolean	true, false
time	13:20:00.000, 13:20:00.000-05:00
dateTime	1999-05-31T13:20:00.000-05:00
duration	P1Y2M3DT10H30M12.3S
date	1999-05-31
gMonth	--05--
gYear	1999
gYearMonth	1999-02
gDay	---31
gMonthDay	--05-31

Table from http://www.w3.org/TR/xmlschema-0/

Some Simple Schema Types Continued

Name	shipTo
QName	po:USAddress
NCName	USAddress
anyURI	http://www.example.com/, http://www.example.com/doc.html#ID5
language	en-GB, en-US, fr

Restricting Types

A base type can be restricted along facets

```
<xsd:simpleType name="StateAbrreviation">
  <xsd:restriction base="xsd:string">
    <xsd:length value="2" />
  </xsd:restriction>
</xsd:simpleType>
```

String Facets

- length

  Exact length of the string

- minLength
- maxLength

  Minimum (maximum) length of the string

- pattern

  Regular expression pattern defining legal values

- enumeration

  List of all possible legal values

- whiteSpace

  Rules for handling whitespace

  preserve
  replace (tab, line feed, carriage return with space)
  collapse