Validating XML Documents with W3C XML Schemas

W3C XML Schemas For Data Interchange

5.1.xml: A Very Simple XML Document

```
<name xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="ex_5.1.xsd"
  >Xavier Ampul III</name>
```

5.1.xsd: A Very Simple XML Document

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="name" type="xs:string"/>

</xs:schema>
```

About Namespaces

- **XML namespaces** can prevent two sets of tags used in the same document from having name collisions.
- Elements and attributes defined in a particular namespace can "wake up" a processing program that is looking for elements in that namespace.
- Namespaces are assigned using the form:
  `xmlns:pre="uri-character-string"`
- The letters used for the prefix are arbitrary -- although there are traditional prefix values.
- The URI character string is what identifies the namespace.

About Namespaces (last)

- The URI uniquely identifies the tagset that is being associated with a prefix.
- The creator of the tagset gets to choose the URI character string.
- Their is a tradition of using URLs for this purpose.
- There need not be a Web page available at the URL address that is used.
- Nevertheless, it is polite to place a Web page that describes the tag set and its usage at the address pointed to by the URL.

About Example 5.1

- Instance documents must declare the `XMLSchema-Instance` namespace.
- `xsi:noNamespaceSchemaLocation` is the attribute that connects the schema to the root element.
- This approach of connecting documents to schemas is appropriate for circumstances where the schema does not declare a target namespace.
- Connecting documents to schemas that declare target namespaces will be addressed later in the course.
- Since the elements in our document are not in a namespace, our schema has no namespace associated with it.

About Example 5.1 (cont.)

- Schema documents must declare the `XMLSchema` namespace.
- Elements are declared using the `xs:element` element.
- All elements have a type.
- The `name` element in this example uses the **builtin** type `xs:string`.
- Other builtin types available include `xs:decimal`, `xs:integer`, `xs:double`, `xs:date`, `xs:ID`, and many others.

About Example 5.1 (last)

- The W3C XML Schema Language also supports the creation of user-defined custom types.

5.2.xml: Adding Child Elements

```
<name xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="ex_5.2.xsd">
  <firstname>Xavier</firstname>
  <lastname>Ampul</lastname>
  <suffix>III</suffix>
</name>
```

5.2.xsd: Adding Child Elements

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="name">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="firstname" type="xs:string"/>
        <xs:element name="middleinit" type="xs:string"
          minOccurs="0"/>
        <xs:element name="lastname" type="xs:string"/>
        <xs:element name="suffix" type="xs:string"
          minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

About Example 5.2

- Schema components can be either **global** or **local**.
- The element `name` is **declared** globally because its declaration is at the top level of the document.
- The elements `firstname`, `middleinit`, `lastname` and `suffix` are local because they are declared below the top level of the document.
- We have not defined any types that occur at the top level of the document.
- So, we do not have any global types.
- The type of the `name` element is **defined** inline (not at top level).

About Example 5.2 (last)

- So, `name` has a locally defined type.
- **Complex** types must be used for any type of content model other than just plain text (has child elements and/or attributes).
- `xs:sequence` defines a sequence of child elements.
- Elements can be declared as optional by coding `minOccurs="0"`.

5.3.xml: Defining Global Types

```
<employee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ex_5.3.xsd">
  <name>
    <firstname>Xavier</firstname>
    <lastname>Ampul</lastname>
    <suffix>III</suffix>
  </name>
  <address>
    <street>1234 Main Street</street>
    <street>Apartment 401</street>
    <city>Chicago</city>
    <state>IL</state>
    <zip>60610</zip>
  </address>
</employee>
```

5.3.xsd: Defining Global Types

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="employee">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="NameType"/>
      <xs:element name="address" type="AddressType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>


  <xs:complexType name="NameType">
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="middleinit" type="xs:string"
        minOccurs="0"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:element name="suffix" type="xs:string"
```

5.3.xsd: Defining Global Types (cont.)

```
      minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="AddressType">
  <xs:sequence>
    <xs:element name="street" type="xs:string"
      minOccurs="1" maxOccurs="3"/>
  <xs:element name="city" type="xs:string"/>
  <xs:element name="state" type="StateType"/>
  <xs:element name="zip" type="ZipType"/>
  </xs:sequence>
</xs:complexType>

<xs:simpleType name="StateType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="IL"/>
    <xs:enumeration value="IN"/>
  </xs:restriction>
</xs:simpleType>
```

5.3.xsd: Defining Global Types (last)

```
  <xs:simpleType name="ZipType">
    <xs:restriction base="xs:string">
      <xs:pattern value="\d{5}"/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

About Example 5.3

- `NameType` and `AddressType` are globally defined types because they occur at the top level of the document.
- Both `NameType` and `AddressType` are complex types made up of sequences.
- `StateType` and `ZipType` are defined globally because the occur at the top level of the document.
- Both `StateType` and `ZipType` are **simple** types that are based upon the builtin type `xs:string`.
- `StateType` restricts `xs:string` with an **enumeration**.

About Example 5.3 (last)

- `ZipType` restricts `xs:string` with a **pattern**.
- Patterns are based upon **regular expressions**.

5.4.xml: More Types

```
<?xml version="1.0" encoding="utf-8"?>
<employee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="ex_5.4.xsd">
  <name>
    <firstname>Xavier</firstname>
    <lastname>Ampul</lastname>
    <suffix>III</suffix>
  </name>
  <address>
    <street>1234 Main Street</street>
    <street>Apartment 401</street>
    <city>Chicago</city>
    <state>IL</state>
    <zip>60610</zip>
  </address>
  <phone>555-555-5555</phone>
</employee>
```

5.4.xsd: More Types

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:element name="employee">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="name" type="NameType"/>
            <xs:element name="address" type="AddressType"/>
            <xs:element name="phone" type="PhoneNumberStringType"
               minOccurs="0"
               maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:complexType name="NameType">
      <xs:sequence>
         <xs:element name="firstname" type="xs:string"/>
         <xs:element name="middleinit" type="xs:string"
            minOccurs="0"/>
```

5.4.xsd: More Types (cont.)

```
      <xs:element name="lastname" type="xs:string"/>
      <xs:element name="suffix" type="xs:string"
         minOccurs="0"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="AddressType">
   <xs:sequence>
      <xs:element name="street" type="xs:string"
         minOccurs="1" maxOccurs="3"/>
      <xs:element name="city" type="xs:string"/>
      <xs:element name="state" type="StateType"/>
      <xs:element name="zip" type="ZipType"/>
   </xs:sequence>
</xs:complexType>

<xs:simpleType name="StateType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="IL"/>
      <xs:enumeration value="IN"/>
```

5.4.xsd: More Types (last)

```
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="ZipType">
      <xs:restriction base="xs:string">
         <xs:pattern value="\d{5}"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="PhoneNumberStringType">
      <xs:restriction base="xs:string">
         <xs:pattern value="\d{3}-\d{3}-\d{4}"/>
      </xs:restriction>
   </xs:simpleType>

</xs:schema>
```

About Example 5.4

- The XML declaration has no counterpart in W3C XML Schema.
- In this example, `PhoneNumberStringType` is defined as a **simple** type because it does not have an attribute associated with it.
- `PhoneNumberStringType` is a restriction of `xs:string` using a pattern.

5.5.xml: Adding <employeelist>

```
<?xml version="1.0" encoding="utf-8"?>
<employeelist xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="ex_5.5.xsd">

  <employee>
    <name>
      <firstname>Xavier</firstname>
      <lastname>Ampul</lastname>
      <suffix>III</suffix>
    </name>
    <address>
      <street>1234 Main Street</street>
      <street>Apartment 401</street>
      <city>Chicago</city>
      <state>IL</state>
      <zip>60610</zip>
    </address>
    <phone>555-555-5555</phone>
  </employee>
```

5.5.xml: Adding <employeelist> (cont.)

```
<employee>
  <name>
    <firstname>Frances</firstname>
    <middleinit>R</middleinit>
    <lastname>Smith</lastname>
  </name>
  <address>
    <street>559 Primary Avenue</street>
    <city>Evanston</city>
    <state>IL</state>
    <zip>60201</zip>
  </address>
  <phone>555-555-5590</phone>
  <phone>555-555-5591</phone>
</employee>

<employee>
  <name>
    <firstname>Raymond</firstname>
    <lastname>Jones</lastname>
```

5.5.xml: Adding <employeelist> (last)

```
    </name>
    <address>
      <street>987 Center Street</street>
      <city>Chicago</city>
      <state>IL</state>
      <zip>60610</zip>
    </address>
    <phone>555-555-0987</phone>
  </employee>

</employeelist>
```

5.5.xsd: Adding <employeelist>

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:element name="employeelist">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="employee" minOccurs="1"
               maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="employee">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="name" type="NameType"/>
            <xs:element name="address" type="AddressType"/>
            <xs:element name="phone" type="PhoneNumberStringType"
               minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
```

5.5.xsd: Adding <employeelist> (cont.)

```
   </xs:complexType>
</xs:element>

<xs:complexType name="NameType">
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="middleinit" type="xs:string"
         minOccurs="0"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:element name="suffix" type="xs:string"
         minOccurs="0"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="AddressType">
   <xs:sequence>
      <xs:element name="street" type="xs:string"
         minOccurs="1" maxOccurs="3"/>
      <xs:element name="city" type="xs:string"/>
      <xs:element name="state" type="StateType"/>
```

5.5.xsd: Adding <employeelist> (cont.)

```
      <xs:element name="zip" type="ZipType"/>
   </xs:sequence>
</xs:complexType>

<xs:simpleType name="StateType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="IL"/>
      <xs:enumeration value="IN"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ZipType">
   <xs:restriction base="xs:string">
      <xs:pattern value="\d{5}"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="PhoneNumberStringType">
   <xs:restriction base="xs:string">
      <xs:pattern value="\d{3}-\d{3}-\d{4}"/>
```

5.5.xsd: Adding <employeelist> (last)

```
      </xs:restriction>
   </xs:simpleType>

</xs:schema>
```

About Example 5.5

- Because the `employee` element is declared at the top level of the document (globally), its declaration can be used in the local definition of the `employeelist` element simply by coding the `ref` attribute instead of the `name` attribute.

5.6.xml: Adding the type Attribute

```
<?xml version="1.0" encoding="utf-8"?>
<employeelist xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="ex_5.6.xsd">

  <employee>
    <name>
      <firstname>Xavier</firstname>
      <lastname>Ampul</lastname>
      <suffix>III</suffix>
    </name>
    <address>
      <street>1234 Main Street</street>
      <street>Apartment 401</street>
      <city>Chicago</city>
      <state>IL</state>
      <zip>60610</zip>
    </address>
    <phone type="home">555-555-5555</phone>
  </employee>
```

5.6.xml: Adding the `type` Attribute (cont.)

```
<employee>
  <name>
    <firstname>Frances</firstname>
    <middleinit>R</middleinit>
    <lastname>Smith</lastname>
  </name>
  <address>
    <street>559 Primary Avenue</street>
    <city>Evanston</city>
    <state>IL</state>
    <zip>60201</zip>
  </address>
  <phone type="home">555-555-5590</phone>
  <phone type="mobile">555-555-5591</phone>
</employee>

<employee>
  <name>
    <firstname>Raymond</firstname>
    <lastname>Jones</lastname>
```

5.6.xml: Adding the `type` Attribute (last)

```
    </name>
    <address>
      <street>987 Center Street</street>
      <city>Chicago</city>
      <state>IL</state>
      <zip>60610</zip>
    </address>
    <phone type="mobile">555-555-0987</phone>
  </employee>

</employeelist>
```

5.6.xsd: Adding the type Attribute

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:element name="employeelist">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="employee" minOccurs="1"
               maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="employee">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="name" type="NameType"/>
            <xs:element name="address" type="AddressType"/>
            <xs:element name="phone" type="PhoneElementType"
               minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
```

5.6.xsd: Adding the type Attribute (cont.)

```
   </xs:complexType>
</xs:element>


<xs:complexType name="NameType">
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="middleinit" type="xs:string" minOccurs="0"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:element name="suffix" type="xs:string" minOccurs="0"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="AddressType">
   <xs:sequence>
      <xs:element name="street" type="xs:string" minOccurs="1"
         maxOccurs="3"/>
      <xs:element name="city" type="xs:string"/>
      <xs:element name="state" type="StateType"/>
      <xs:element name="zip" type="ZipType"/>
```

5.6.xsd: Adding the type Attribute (cont.)

```
   </xs:sequence>
</xs:complexType>

<xs:simpleType name="StateType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="IL"/>
      <xs:enumeration value="IN"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ZipType">
   <xs:restriction base="xs:string">
      <xs:pattern value="\d{5}"/>
   </xs:restriction>
</xs:simpleType>

<xs:complexType name="PhoneElementType">
   <xs:simpleContent>
      <xs:extension base="PhoneNumberStringType">
         <xs:attribute name="type" type="TypeAttributeType"/>
```

5.6.xsd: Adding the type Attribute (cont.)

```
      </xs:extension>
   </xs:simpleContent>
</xs:complexType>

<xs:simpleType name="PhoneNumberStringType">
   <xs:restriction base="xs:string">
      <xs:pattern value="\d{3}-\d{3}-\d{4}"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="TypeAttributeType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="office"/>
      <xs:enumeration value="home"/>
      <xs:enumeration value="mobile"/>
      <xs:enumeration value="fax"/>
   </xs:restriction>
</xs:simpleType>
```

5.6.xsd: Adding the `type` Attribute (last)

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

About Example 5.6

- In this example, `PhoneElementType` is a **complex** type because the element now has an attribute.
- This type creates a local definition by using the `xs:simpleContent` element.
- This type is an extension of the simple type `PhoneNumberStringType`.
- The extension in this case is the addition of the `type` attribute.
- The type attribute is declared using the type `TypeAttributeType`.

About Example 5.6 (last)

- `TypeAttributeType` is a restriction of `xs:string` using an enumeration.

5.7.xml: Finishing Touches

```
<?xml version="1.0" encoding="utf-8"?>
<employeelist xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="ex_5.7.xsd">

  <!-- The contents of this document are
       confidential.  Distribution is restricted
       to Xampul Employees and other
       previously authorized persons.   -->

  <listdate>April 4, 2006</listdate>
  <listregion>Central</listregion>

  <employee>
    <name>
      <firstname>Xavier</firstname>
      <lastname>Ampul</lastname>
      <suffix>III</suffix>
    </name>
    <address>
      <street>1234 Main Street</street>
```

5.7.xml: Finishing Touches (cont.)

```
    <street>Apartment 401</street>
    <city>Chicago</city>
    <state>IL</state>
    <zip>60610</zip>
  </address>
  <phone type="home">555-555-5555</phone>
</employee>

<employee>
  <name>
    <firstname>Frances</firstname>
    <middleinit>R</middleinit>
    <lastname>Smith</lastname>
  </name>
  <address>
    <street>559 Primary Avenue</street>
    <city>Evanston</city>
    <state>IL</state>
    <zip>60201</zip>
  </address>
```

5.7.xml: Finishing Touches (last)

```
    <phone type="home">555-555-5590</phone>
    <phone type="mobile">555-555-5591</phone>
  </employee>

  <employee>
    <name>
      <firstname>Raymond</firstname>
      <lastname>Jones</lastname>
    </name>
    <address>
      <street>987 Center Street</street>
      <city>Chicago</city>
      <state>IL</state>
      <zip>60610</zip>
    </address>
    <phone type="mobile">555-555-0987</phone>
  </employee>
</employeelist>
```

5.7.xsd: Finishing Touches

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:element name="employeelist">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="listdate" type="xs:string"/>
            <xs:element name="listregion" type="xs:string"/>
            <xs:element ref="employee" minOccurs="1"
               maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="employee">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="name" type="NameType"/>
            <xs:element name="address" type="AddressType"/>
            <xs:element name="phone" type="PhoneElementType"
```

5.7.xsd: Finishing Touches (cont.)

```
            minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
   </xs:complexType>
</xs:element>

<xs:complexType name="NameType">
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="middleinit" type="xs:string"
         minOccurs="0"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:element name="suffix" type="xs:string"
         minOccurs="0"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="AddressType">
   <xs:sequence>
      <xs:element name="street" type="xs:string"
         minOccurs="1" maxOccurs="3"/>
```

5.7.xsd: Finishing Touches (cont.)

```
      <xs:element name="city" type="xs:string"/>
      <xs:element name="state" type="StateType"/>
      <xs:element name="zip" type="ZipType"/>
   </xs:sequence>
</xs:complexType>

<xs:simpleType name="StateType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="IL"/>
      <xs:enumeration value="IN"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ZipType">
   <xs:restriction base="xs:string">
      <xs:pattern value="\d{5}"/>
   </xs:restriction>
</xs:simpleType>

<xs:complexType name="PhoneElementType">
```

5.7.xsd: Finishing Touches (cont.)

```
   <xs:simpleContent>
      <xs:extension base="PhoneNumberStringType">
         <xs:attribute name="type" type="TypeAttributeType"/>
      </xs:extension>
   </xs:simpleContent>
</xs:complexType>

<xs:simpleType name="PhoneNumberStringType">
   <xs:restriction base="xs:string">
      <xs:pattern value="\d{3}-\d{3}-\d{4}"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="TypeAttributeType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="office"/>
      <xs:enumeration value="home"/>
      <xs:enumeration value="mobile"/>
      <xs:enumeration value="fax"/>
   </xs:restriction>
```

5.7.xsd: Finishing Touches (last)

```
    </xs:simpleType>

</xs:schema>
```

About Example 5.7

- XML comments have no counterpart in the W3C XML Schema.
- Adding element declarations to an existing sequence is easy.