

Schemas

Introduction

XSD Elements and Attributes

XSD Data Types

XSD Facets

XSD Indicators

Extending XSD

Objectives

- **Need for Schema**
- **What is an XML Schema?**
- **Exploring XSD**
- **How to write XML Schema**
- **Structuring**
 - XSD files
 - Apply to XML files
 - XSD Elements
 - XSD Simple Types Element
 - XSD Data Types
 - XSD Attributes

Objectives

- **Structuring (cont)**
 - XSD Facets
 - XSD Complex Types Element
 - XSD Indicators
 - Extending Complex Types
 - Abstract Types
 - XSD Any Elements
 - XSD Element Substitution
- **Summary**

Schemas

Need for Schema

- The **limitation** of DTDs
 - DTDs are **written** in a **non-XML syntax**
 - DTDs do not use XML notation (they **use EBNF** to present)
 - The EBNF is **difficult** to **write and use**
 - DTDs **do not support namespace**
 - A namespace cannot be used to refer to an element or an entity declaration
 - If the namespace is used, the DTD has to be modified to include any elements taken from the namespace
 - DTDs **do not provide** a means of specifying element and attribute **data types**
 - DTD can only express the data type of attributes in term of explicit enumeration and a few coarse string formats
 - DTDs do not have a facility to describe numbers, dates, etc. ...
- **Schema** is a **candidate** that specially **aimed** at addressing **DTDs' limitation**

Schemas

What is an XML Schema?

- An XML Schema **specifies** the **structure** of **valid XML documents**
 - **Defining a set of elements**, their **relationships** to each other, and the **attributes** that they can contain
- The **objective** of XML Schema is the **same purpose** with **DTD**
 - **Validating XML data = XML document + XML Schema**
 - A schema defines the **valid building block** of an XML document
 - A schema can be considered as a **common vocabulary** that different organizations **can share** to exchange document
- A schema is **considered** a **valid XML document** because the **schema specification** is **defined** using a **DTD**
- An XML Schema defines
 - **Elements and Attributes**
 - **Child elements** with the **order** and **number** of child elements
 - Whether an element is an **empty** and can **include text**
 - **Data types** for elements and attributes
 - **Default** and **fixed** values for elements and attributes

Schemas

What is an XML Schema?

- XML Schema **features**
 - Schema **supports data types** and **ability** to **create** the required **data type**
 - It is easy to **define**, **validate** valid document **content** and **data formats**, and **implement** the **restrictions** on data
 - Richer data types: Schema **defines many data types**
 - Archetypes: allow to **define own name data type** from **preexisting data types**
 - Attribute grouping: **make relationship** to all attributes
 - Schema is **portable & efficient**: Use **same XML syntax**
 - Schema **secures data communications**
 - The **sender** can **specify** the **data** in a way that the **receiver** will **understand**

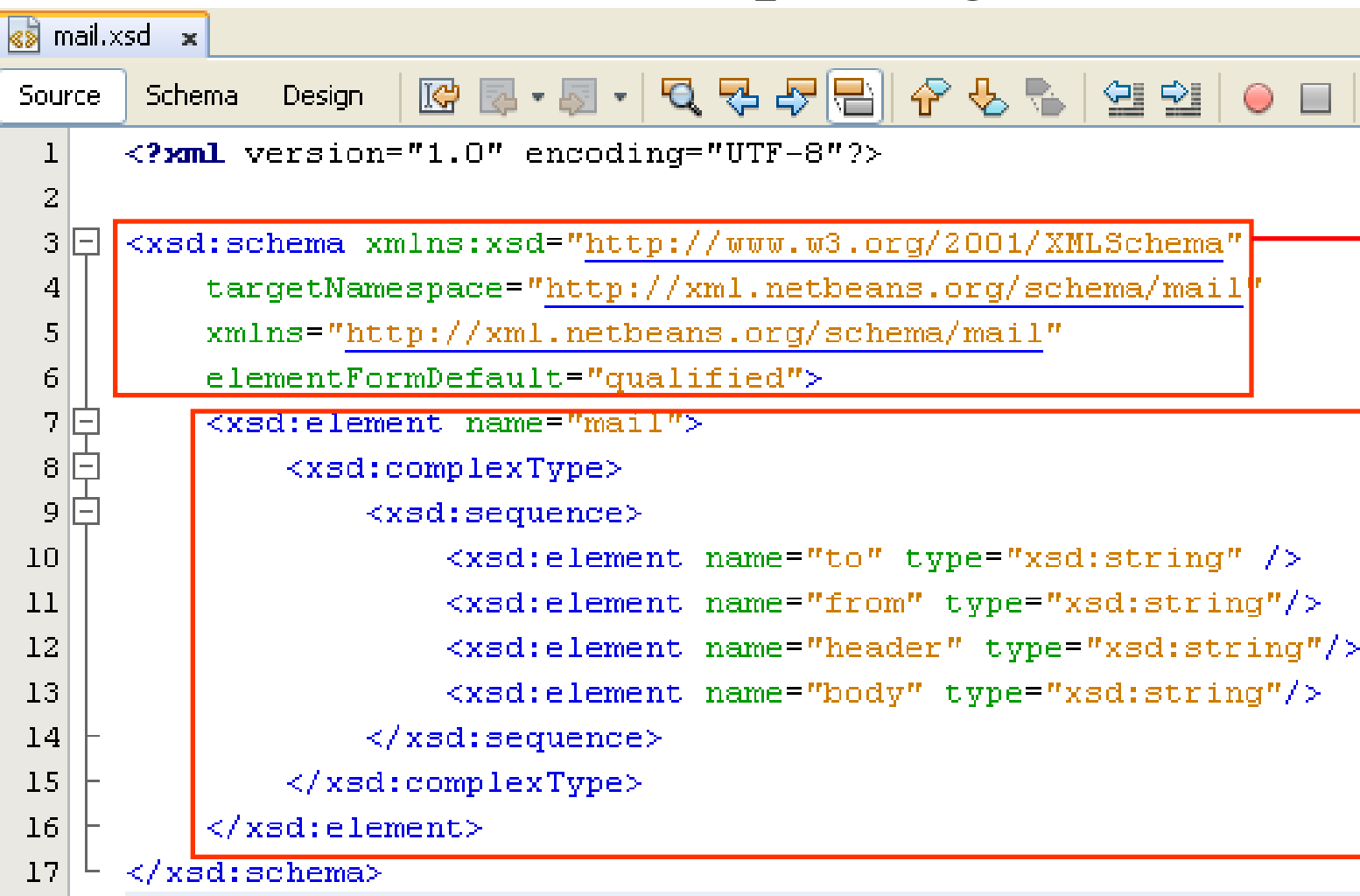
Schemas

What is an XML Schema?

- XML Schema features (cont)
 - Schema is **extendable**
 - **Reusable** and **support reference** of **multiple schemas** in the same document
 - **Refinable archetypes**: allow an “open” content model that elements other than required elements can be presented
 - Schema **catch high-level mistakes**
 - **Checking** a **required** field information is **missing** or in **wrong format**, or an element name is **misspelled** after the XML is **checked** in well-formed
 - Schema **supports namespace: reusable**

Introduction

Exploring XSD



```

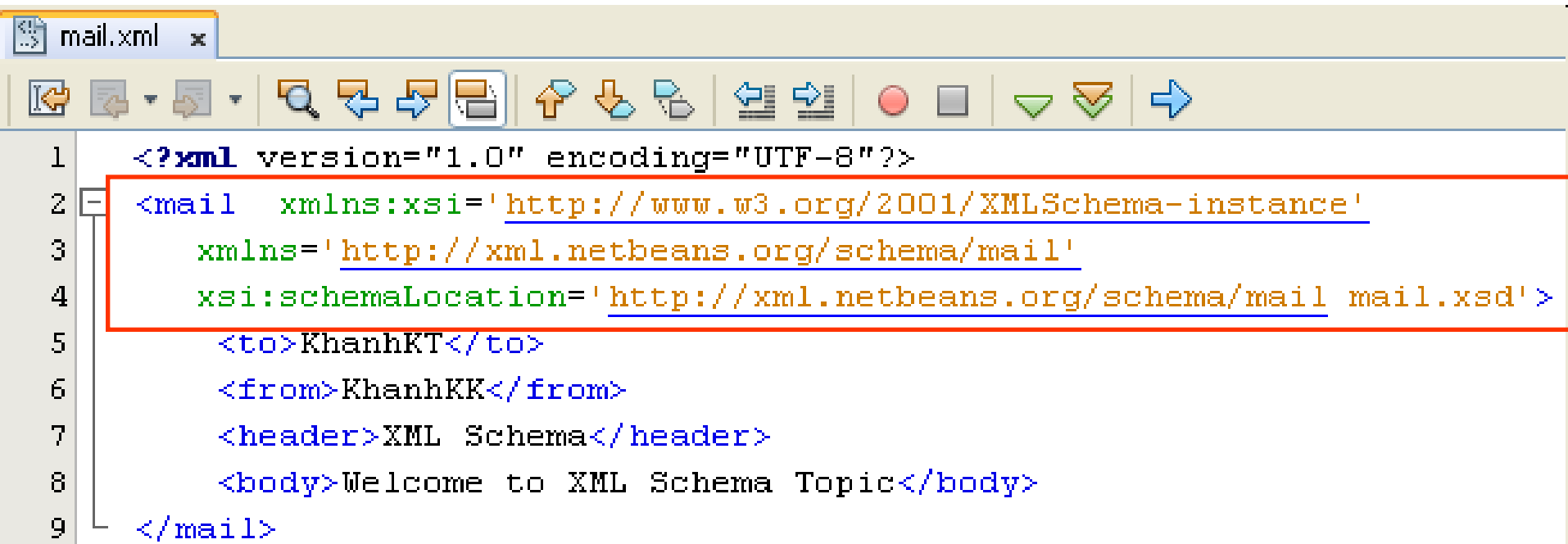
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    targetNamespace="http://xml.netbeans.org/schema/mail"
5    xmlns="http://xml.netbeans.org/schema/mail"
6    elementFormDefault="qualified">
7
8    <xsd:element name="mail">
9      <xsd:complexType>
10        <xsd:sequence>
11          <xsd:element name="to" type="xsd:string" />
12          <xsd:element name="from" type="xsd:string"/>
13          <xsd:element name="header" type="xsd:string"/>
14          <xsd:element name="body" type="xsd:string"/>
15        </xsd:sequence>
16      </xsd:complexType>
17    </xsd:element>
18 </xsd:schema>
  
```

XSD declaration

XSD Contents

Introduction

Exploring XSD



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <mail xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3     xmlns='http://xml.netbeans.org/schema/mail'
4     xsi:schemaLocation='http://xml.netbeans.org/schema/mail mail.xsd'>
5     <to>KhanhKT</to>
6     <from>KhanhKK</from>
7     <header>XML Schema</header>
8     <body>Welcome to XML Schema Topic</body>
9 </mail>
```

Schemas

How to write an XML Schema

- An XML Schema is a **text-only document**, and **begins with a standard XML declaration**.
- Write **file is stored with .xsd extension**
 - A **schema declaration** is **typed** in the **beginning**
 - Define the **elements with their attributes**
 - Define the **restrictions and other constraints** (if any)
- **Reference the schema to xml document**
 - **Using schema declaration in xml document to**
 - **Inform** the schema-validator valid the namespace of particular element
 - **Determine instance namespace** applying to XML document
 - **Locate** the XSD file's location (schemaLocation)
 - **Define XML document's content**

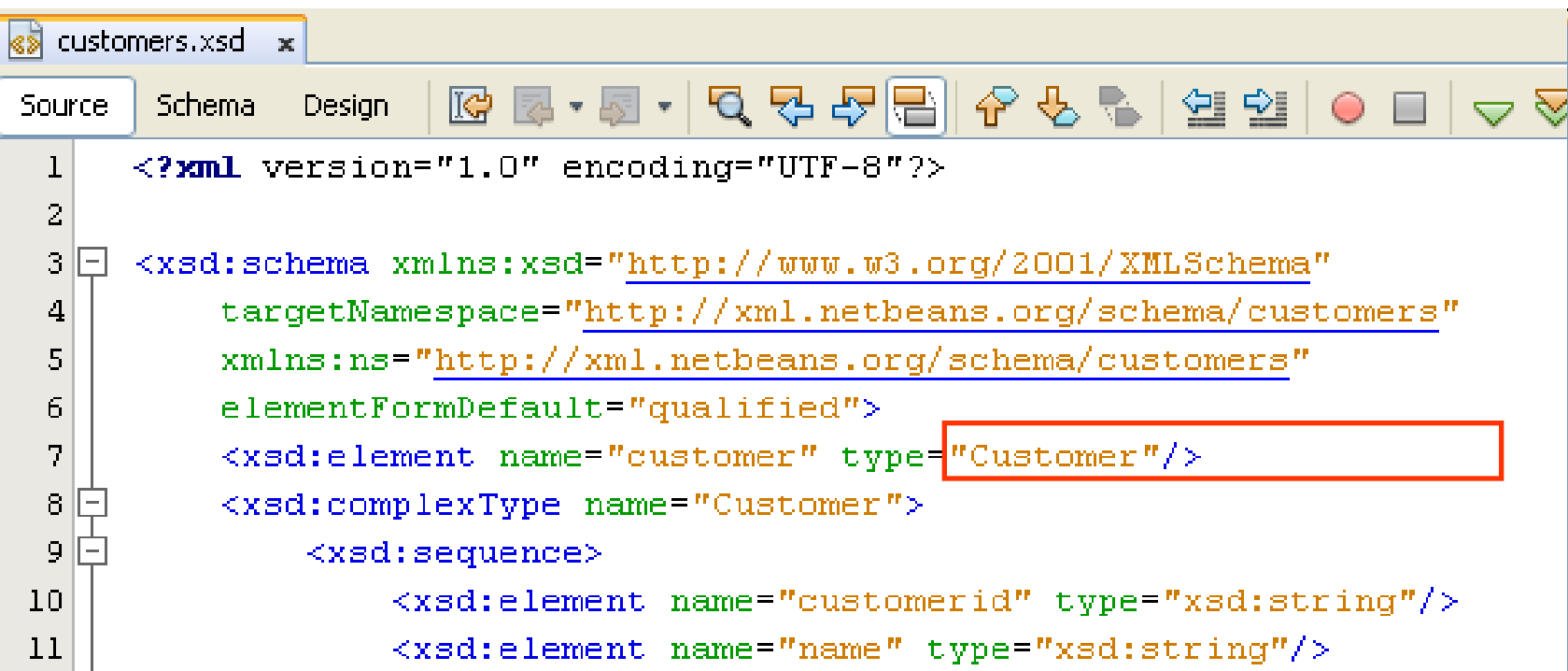
Schemas

Structuring – XSD files

- The schema element is the root element of every XML Schema
 - **xmlns:xsd**="http://www.w3.org/2001/XMLSchema"
 - **Indicates** that the **elements** and **data types** used in the schema **come from** “http://www.w3.org/2001/XMLSchema” namespace
 - All elements and data types should be prefixed with **xsd**
 - **targetNamespace**="namespace_uri"
 - **Indicates** that the **elements defined** by this **schema** (to, from, heading, body) **come from** the “namespace_uri” namespace
 - **xmlns**="namespace_uri"
 - **Indicates** that the **default namespace** is “namespace_uri” that is declared
 - **elementFormDefault**="qualified"
 - **Indicates** that any **elements used** by the XML instance document which were declared in this schema **must be namespace qualified**

Schemas

Structuring – XSD files – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/customers"
5      xmlns:ns="http://xml.netbeans.org/schema/customers"
6      elementFormDefault="qualified">
7      <xsd:element name="customer" type="Customer"/>
8      <xsd:complexType name="Customer">
9          <xsd:sequence>
10             <xsd:element name="customerid" type="xsd:string"/>
11             <xsd:element name="name" type="xsd:string"/>

```

Output - XML check

XML validation started.

[Z:/LapTrinh/Servlet/XML/Day3XML/src/customers.xsd:7,4](#)

ERROR: src-resolve.4.1: Error resolving component 'Customer'. It was detected that 'Customer' has no namespace but components with no target namespace are not referenceable from schema document 'file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/customers.xsd'. If 'Customer' is intended to have a namespace, perhaps a prefix needs to be provided. If it is intended that 'Customer' has no namespace, then an 'import' without a "namespace" attribute should be added to 'file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/customers.xsd'.

1 Error(s), 0 Warning(s).

XML validation finished.

Schemas

Structuring – XSD files – Example

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/customers"
5   xmlns:ns="http://xml.netbeans.org/schema/customers"
6   elementFormDefault="qualified">
7   <xsd:element name="customer" type="ns:Customer"/>
8   <xsd:complexType name="Customer">
9     <xsd:sequence>
10       <xsd:element name="customerid" type="xsd:string"/>
11       <xsd:element name="name" type="xsd:string"/>
12       <xsd:element name="address" type="xsd:string"/>
13       <xsd:element name="city" type="xsd:string"/>
14       <xsd:element name="phone" type="xsd:positiveInteger"/>
15     </xsd:sequence>
16   </xsd:complexType>
17 </xsd:schema>

```

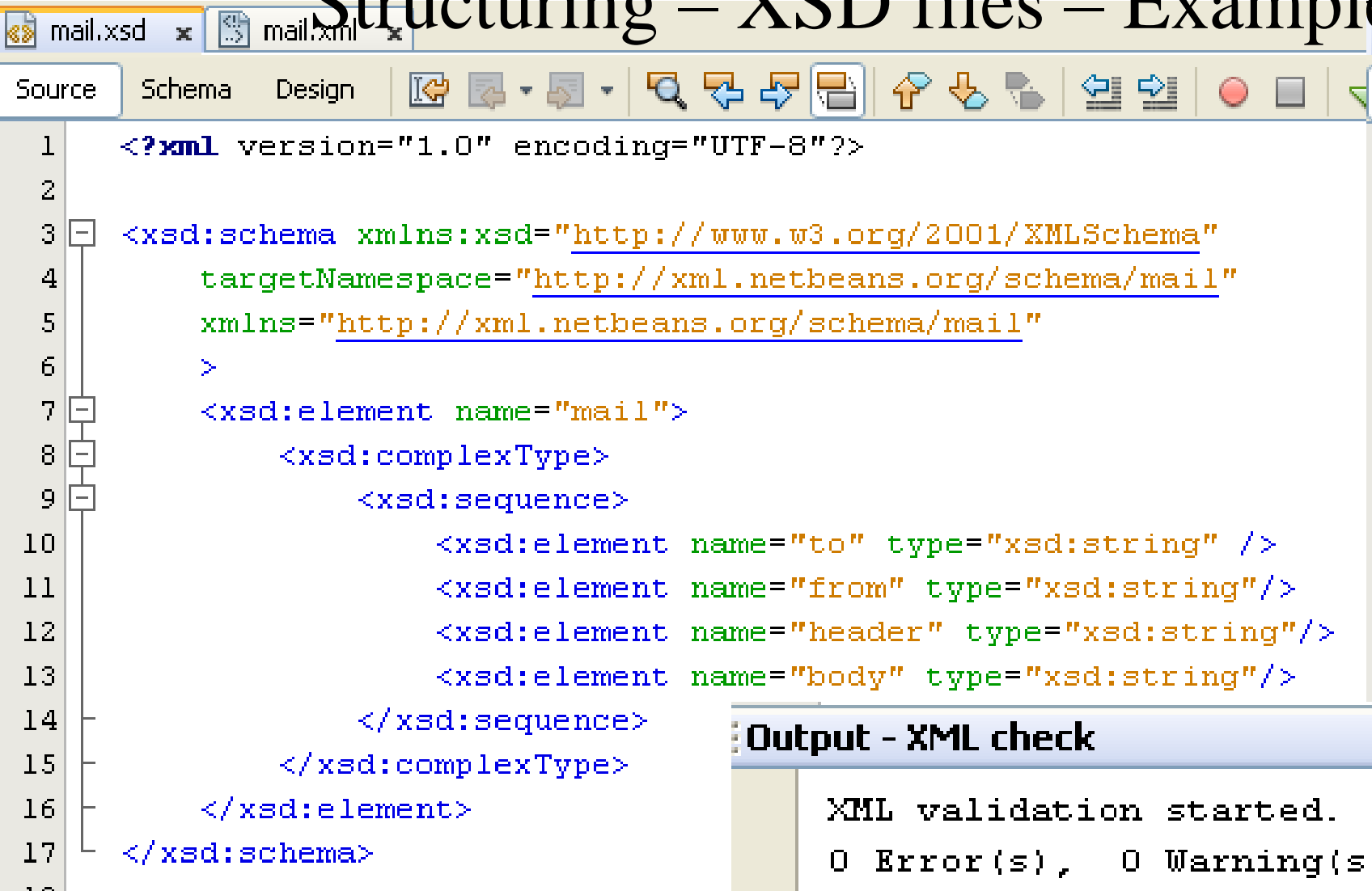
Output - XML check

```

XML validation started.
0 Error(s), 0 Warning(s).
XML validation finished.

```

Structuring – XSD files – Example



The screenshot shows an IDE with two tabs: 'mail.xsd' and 'mail.xml'. The 'mail.xsd' tab is active, displaying the following XML Schema Definition (XSD) code:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/mail"
5      xmlns="http://xml.netbeans.org/schema/mail"
6  >
7      <xsd:element name="mail">
8          <xsd:complexType>
9              <xsd:sequence>
10                 <xsd:element name="to" type="xsd:string" />
11                 <xsd:element name="from" type="xsd:string"/>
12                 <xsd:element name="header" type="xsd:string"/>
13                 <xsd:element name="body" type="xsd:string"/>
14             </xsd:sequence>
15         </xsd:complexType>
16     </xsd:element>
17 </xsd:schema>
  
```

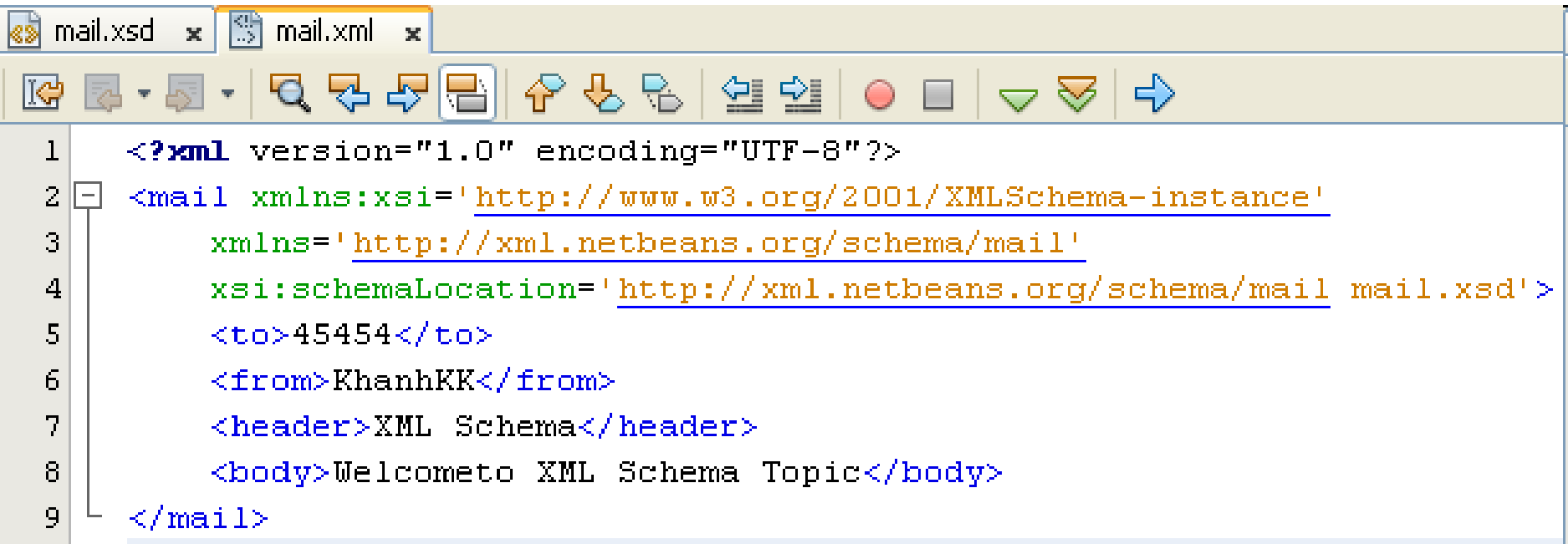
On the right side of the IDE, there is a panel titled 'Output - XML check' which displays the following text:

```

XML validation started.
0 Error(s), 0 Warning(s).
XML validation finished.
  
```

Schemas

Structuring – XSD files – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <mail xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3      xmlns='http://xml.netbeans.org/schema/mail'
4      xsi:schemaLocation='http://xml.netbeans.org/schema/mail mail.xsd'>
5      <to>45454</to>
6      <from>KhanhKK</from>
7      <header>XML Schema</header>
8      <body>Welcometo XML Schema Topic</body>
9  </mail>
  
```

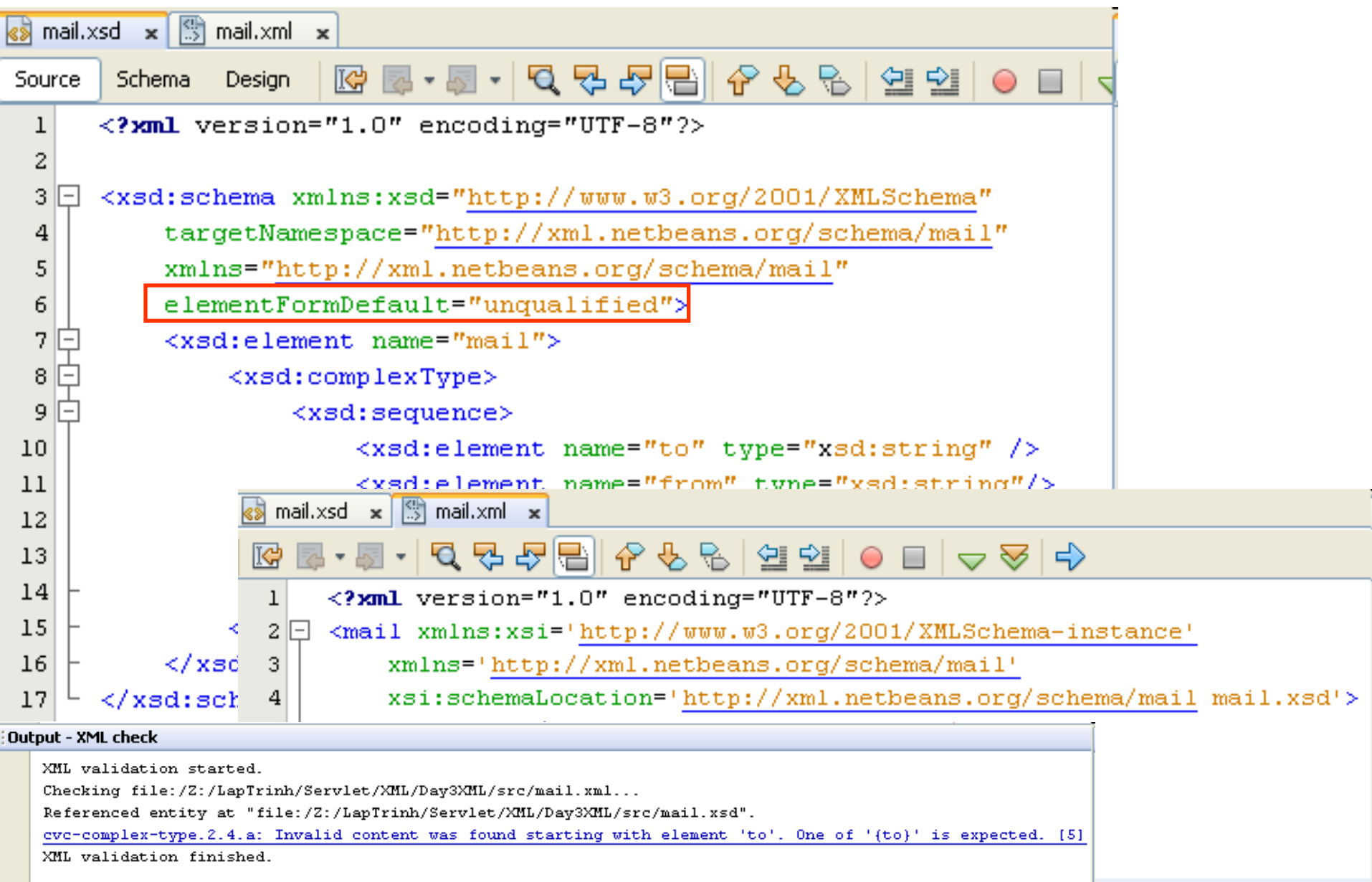
Output - XML check

```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xml...
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xsd".
cvc-complex-type.2.4.a: Invalid content was found starting with element 'to'. One of '{to}' is expected. [5]
XML validation finished.
  
```

Schemas

Structuring – XSD files – Example



The screenshot shows an IDE with two tabs: `mail.xsd` and `mail.xml`. The `mail.xsd` tab is active, displaying the following XML Schema Definition (XSD) code:

```
<?xml version="1.0" encoding="UTF-8"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://xml.netbeans.org/schema/mail"
  xmlns="http://xml.netbeans.org/schema/mail"
  elementFormDefault="unqualified">
  <xsd:element name="mail">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="to" type="xsd:string" />
        <xsd:element name="from" type="xsd:string"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

The `elementFormDefault="unqualified"` attribute is highlighted with a red box. The `mail.xml` tab is also visible, showing the following XML instance code:

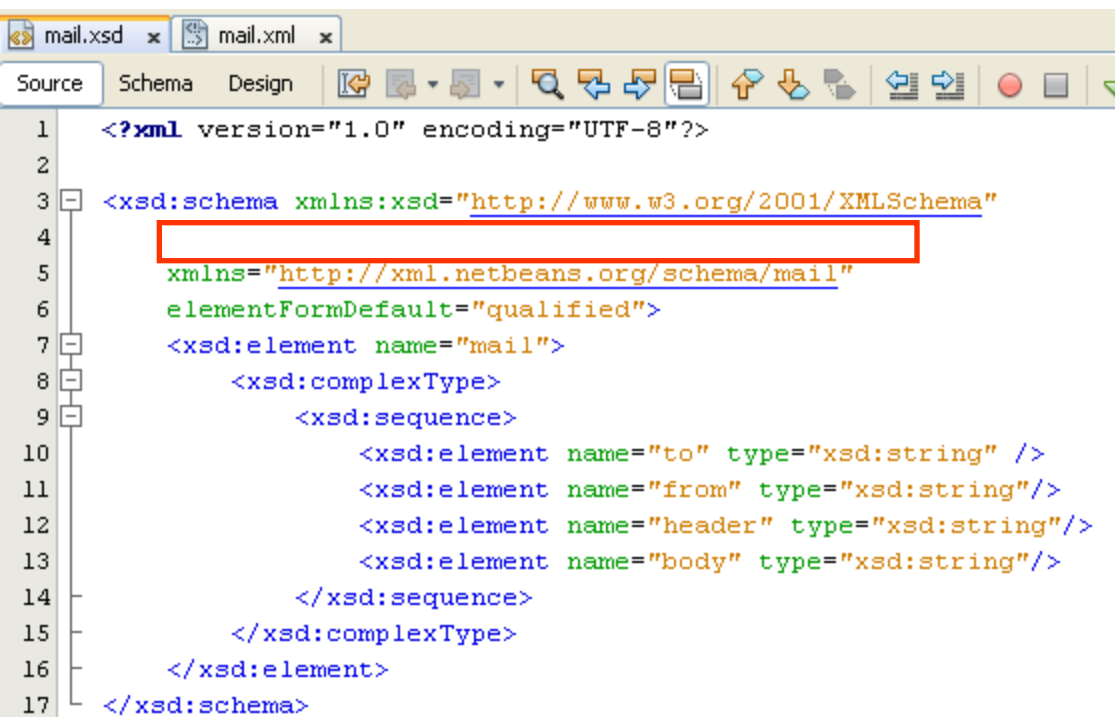
```
<?xml version="1.0" encoding="UTF-8"?>
<mail xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
  xmlns='http://xml.netbeans.org/schema/mail'
  xsi:schemaLocation='http://xml.netbeans.org/schema/mail mail.xsd'>
```

The `mail.xml` tab is partially obscured by a validation error message in the Output window. The error message is:

```
Output - XML check
XML validation started.
Checking file: /Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xml...
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xsd".
cvc-complex-type.2.4.a: Invalid content was found starting with element 'to'. One of '{to}' is expected. [5]
XML validation finished.
```


Schemas

Structuring – XSD files – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    xmlns:mail="http://xml.netbeans.org/schema/mail"
5    elementFormDefault="qualified">
6    <xsd:element name="mail">
7      <xsd:complexType>
8        <xsd:sequence>
9          <xsd:element name="to" type="xsd:string" />
10         <xsd:element name="from" type="xsd:string"/>
11         <xsd:element name="header" type="xsd:string"/>
12         <xsd:element name="body" type="xsd:string"/>
13       </xsd:sequence>
14     </xsd:complexType>
15   </xsd:element>
16 </xsd:schema>

```

Output - XML check

```

XML validation started.
Checking file: /Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xml...
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xsd".
TargetNamespace.1: Expecting namespace 'http://xml.netbeans.org/schema/mail' but the target namespace of the schema document is 'null'. [6]
cvc-elt.1: Cannot find the declaration of element 'mail'. [4]
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xsd".
TargetNamespace.1: Expecting namespace 'http://xml.netbeans.org/schema/mail', but the target namespace of the schema document is 'null'. [6]
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xsd".
TargetNamespace.1: Expecting namespace 'http://xml.netbeans.org/schema/mail', but the target namespace of the schema document is 'null'. [6]
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xsd".
TargetNamespace.1: Expecting namespace 'http://xml.netbeans.org/schema/mail', but the target namespace of the schema document is 'null'. [6]
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xsd".
TargetNamespace.1: Expecting namespace 'http://xml.netbeans.org/schema/mail', but the target namespace of the schema document is 'null'. [6]
XML validation finished.

```

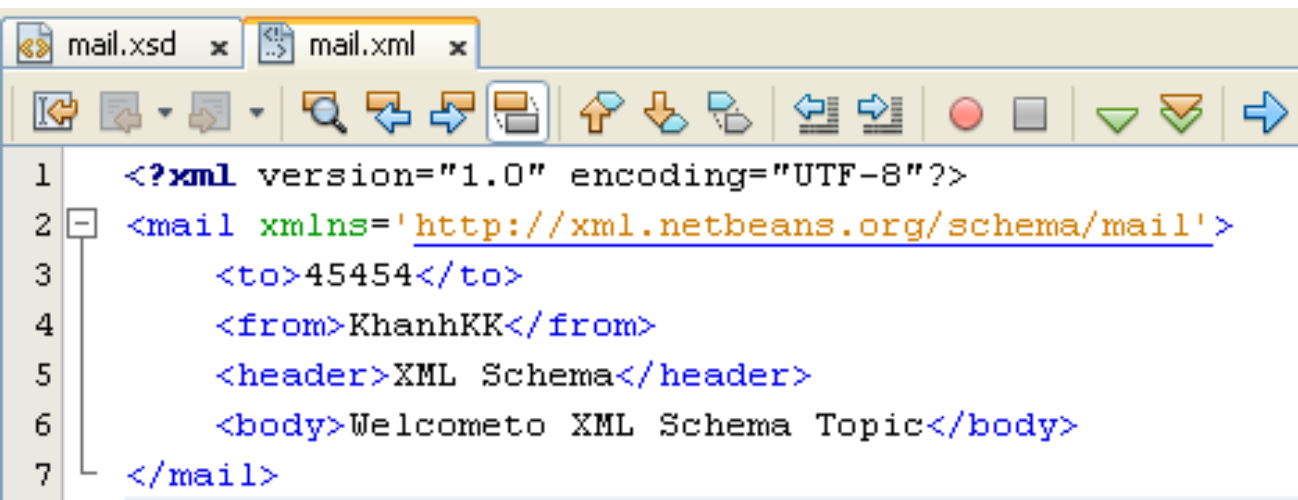
Schemas

Structuring – Apply to XML files

- The root element is defined with some attributes
 - **xmlns:xsi=**
“http://www.w3.org/2001/XMLSchema-instance”
 - XML Schema Definition Language: **Structures defines** several **attributes for direct use in** any **XML** documents. They are defined in Schema Instance Namespace
 - **xmlns=**“http://xml.netbeans.org/schema/mail”
 - **Specifies the default namespace declaration.**
 - This declaration **tells the schema-validator** that all the **elements used** in this XML document **are declared** in the “http://xml.netbeans.org/schema/mail” namespace
 - **xsi:schemaLocation=**“namespace_uri **instance**”>
 - The **first** value is the **namespace** to use.
 - The **second** value is the **location** of the XML **schema** to use for that namespace

Schemas

Structuring – Apply to XML files – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <mail xmlns='http://xml.netbeans.org/schema/mail'>
3      <to>45454</to>
4      <from>KhanhKK</from>
5      <header>XML Schema</header>
6      <body>Welcometo XML Schema Topic</body>
7  </mail>
  
```

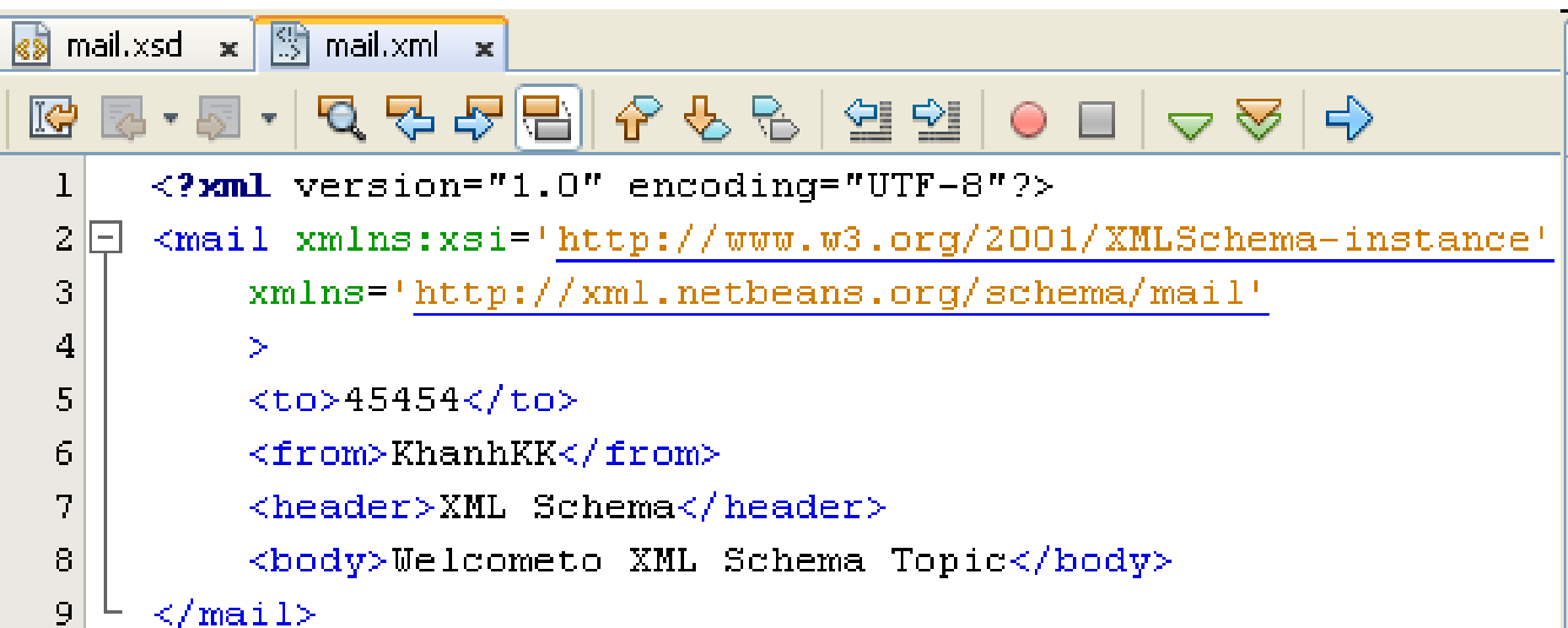
Output - XML check

```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xml...
cvc-elt.1: Cannot find the declaration of element 'mail'. [2]
XML validation finished.
  
```

Schemas

Structuring – Apply to XML files – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <mail xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3      xmlns='http://xml.netbeans.org/schema/mail'
4      >
5      <to>45454</to>
6      <from>KhanhKK</from>
7      <header>XML Schema</header>
8      <body>Welcometo XML Schema Topic</body>
9  </mail>
  
```

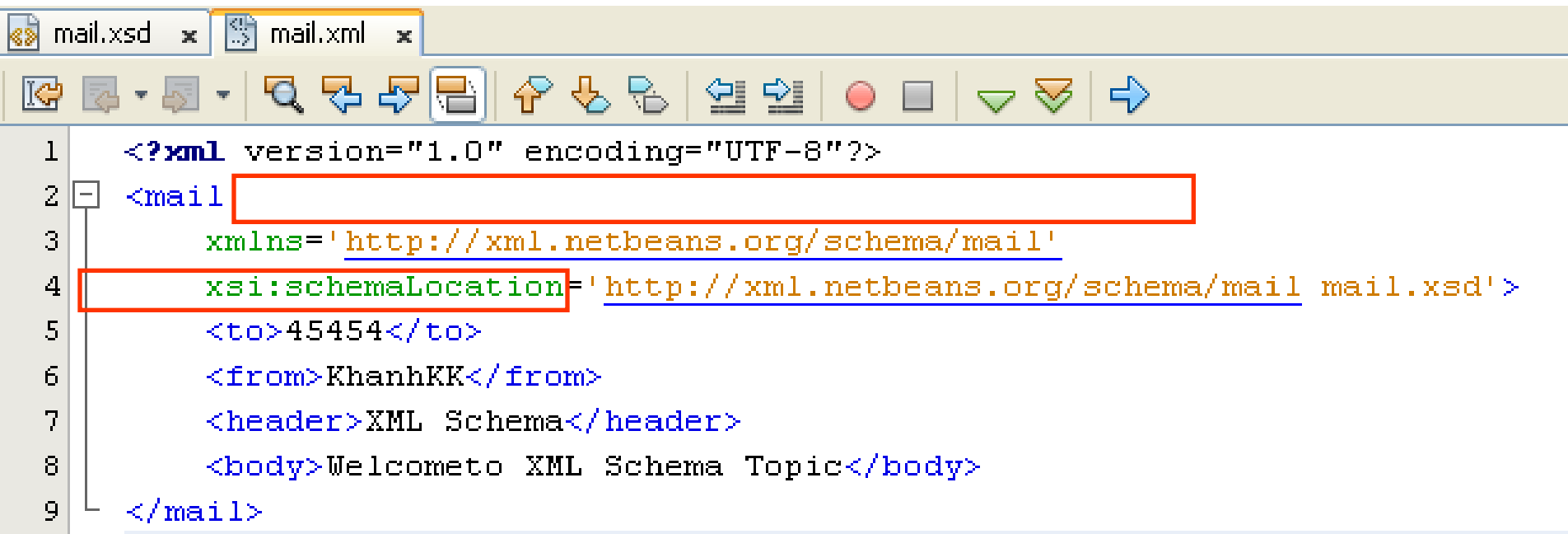
Output - XML check

```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xml...
cvc-elt.1: Cannot find the declaration of element 'mail'. [4]
XML validation finished.
  
```

Schemas

Structuring – Apply to XML files – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <mail
3      xmlns='http://xml.netbeans.org/schema/mail'
4      xsi:schemaLocation='http://xml.netbeans.org/schema/mail mail.xsd'>
5      <to>45454</to>
6      <from>KhanhKK</from>
7      <header>XML Schema</header>
8      <body>Welcometo XML Schema Topic</body>
9  </mail>
  
```

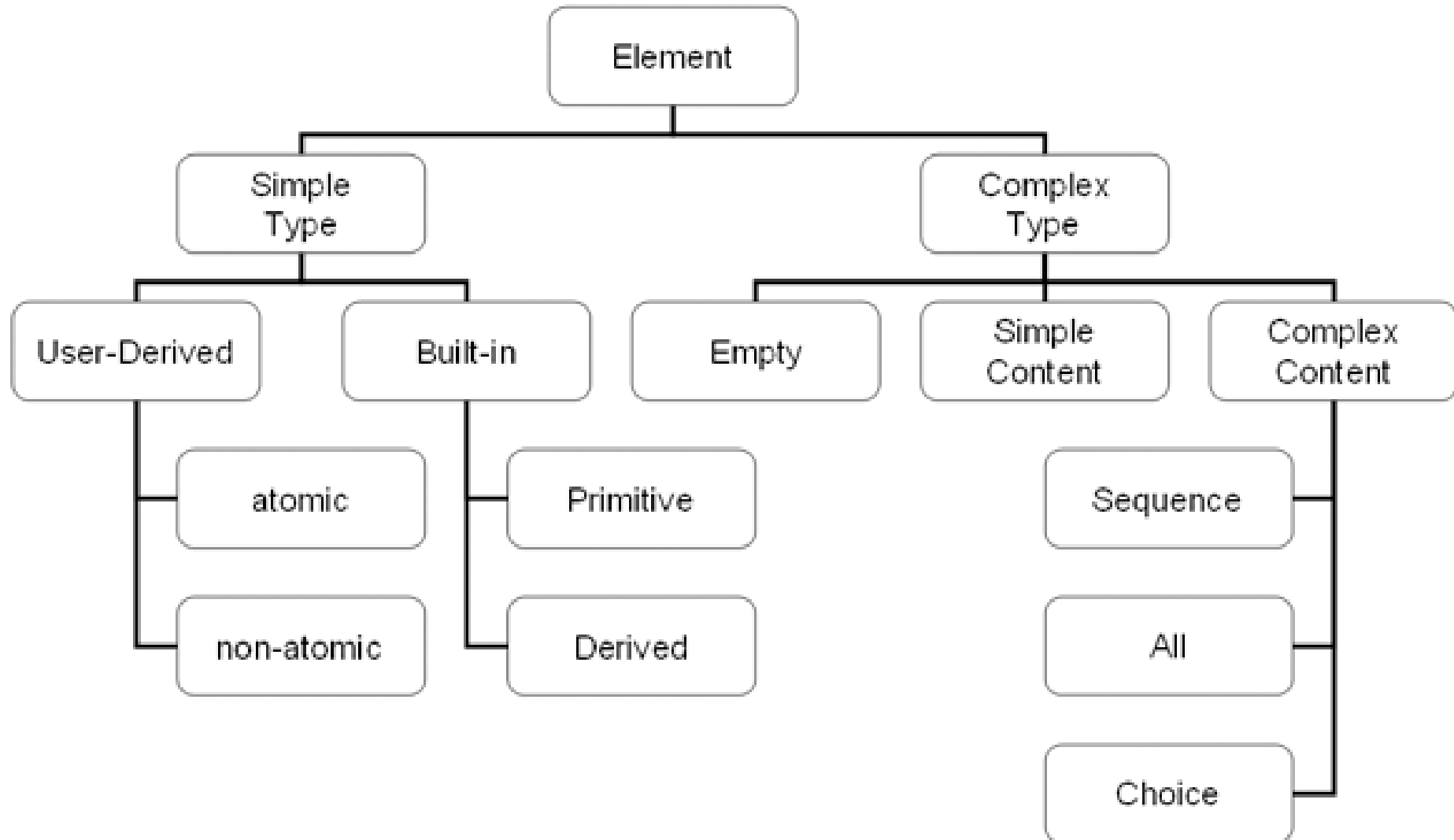
Output - XML check

```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mail.xml...
The prefix "xsi" for attribute "xsi:schemaLocation" associated with an element type "mail" is not bound. [4]
XML validation finished.
  
```

Schemas

Structuring – XSD Elements



Schemas

Structuring – XSD Simple Type Elements

- Is an XML element that **contains only text**.
- **Cannot** contain any other elements or attributes
- **Syntax: declaration** vs. **reference** to global element
 - `<xsd:element name="element_name" type="data_type"/>`
 - `<xsd:element ref="referenced_element_name"/>`
- Default, Fixed, and Nil Values
 - A **default** value is **automatically assigned** to the element **when no other value is specified**

`<xsd:element name="element_name" type="data_type" default="default_value"/>`

- A **fixed** value is also **automatically assigned** to the element, and it **cannot specify another value**

`<xsd:element name="element_name" type="data_type" fixed="fixed_value"/>`

- A **default and fixed value are not existed** in the **same time**
- When an optional element is left out of an XML instance, it has no clear meaning
 - elements in XML can be set to nil by setting the xsi:nil attribute to true

`<xsd:element name="element_name" type="data_type" nillable="true|false"/>`

Schemas

Structuring – XSD Data Types

- XML Schema defines 02 sort of data types
 - **Built-in** data types
 - Are **available** to all XML Schema authors
 - Should be **implemented** by a **conforming** processor
 - Can be used to **specify** and **validate** the **intended data type** of the content
 - **Allow** a user to **create a user-defined data by extending** the built-in data types **using facets** (→ **derived types**)
 - Are specified into groups as
 - string, boolean, numeric, data and time, binary, anyURI
 - **User-derived** data types
 - Are **defined** in individual **schema instances**
 - Are particular to that schema

Schemas

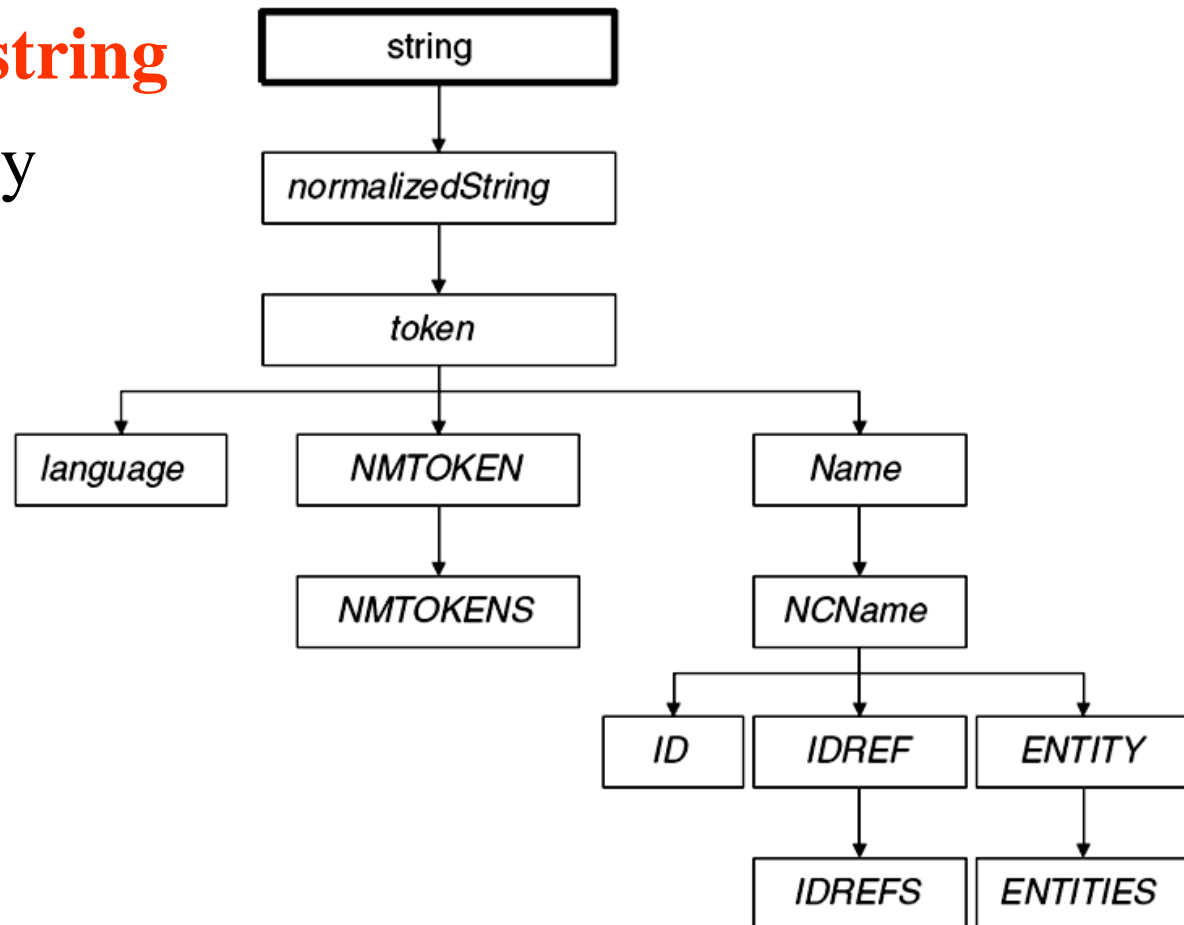
Structuring – XSD Data Types

- Are defined how the values are represented as character in XML following
 - **Basic type**
 - **A value space**
 - The set of values the type can hold
 - **A lexical spaces**
 - The set of lexical is used to declare the represented name
 - **User-derived type**
 - **A basic type**
 - **Canonical representation:** using correctly syntax
 - **List of facets**

Schemas

Structuring – XSD Data Types – string

- A group of characters
 - **Contains** characters, line feeds, carriage returns, and tab characters
 - **Consists** of a **combination** of Unicode characters
- **Syntax: *xsd:string***
- Type hierarchy



Schemas

Structuring – XSD Data Types – string

Type	Description
normalizedString	<ul style="list-style-type: none"> - Contains characters, but the XML processor will remove line feeds, carriage returns, and tab characters. - Syntax: <code>xsd:normalizedString</code>
token	<ul style="list-style-type: none"> - Contains characters, but the XML processor will remove line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces - Syntax: <code>xsd:token</code>
language	- A string that contains a valid language id. Syntax: <code>xsd:language</code>
NMTOKEN	A string that represents the NMTOKEN attribute in XML (only used with schema attributes). Syntax: <code>xsd:NMTOKEN</code>
NMTOKENS	List of NMTOKEN values separated by whitespace (only used with schema attributes). Syntax: <code>xsd:NMTOKENS</code>
Name	A string that contains a valid XML name. Syntax: <code>xsd:Name</code>
QName	<ul style="list-style-type: none"> - Represent qualified names according to Namespace (order:Order) - Syntax: <code>xsd:Qname</code>

Schemas

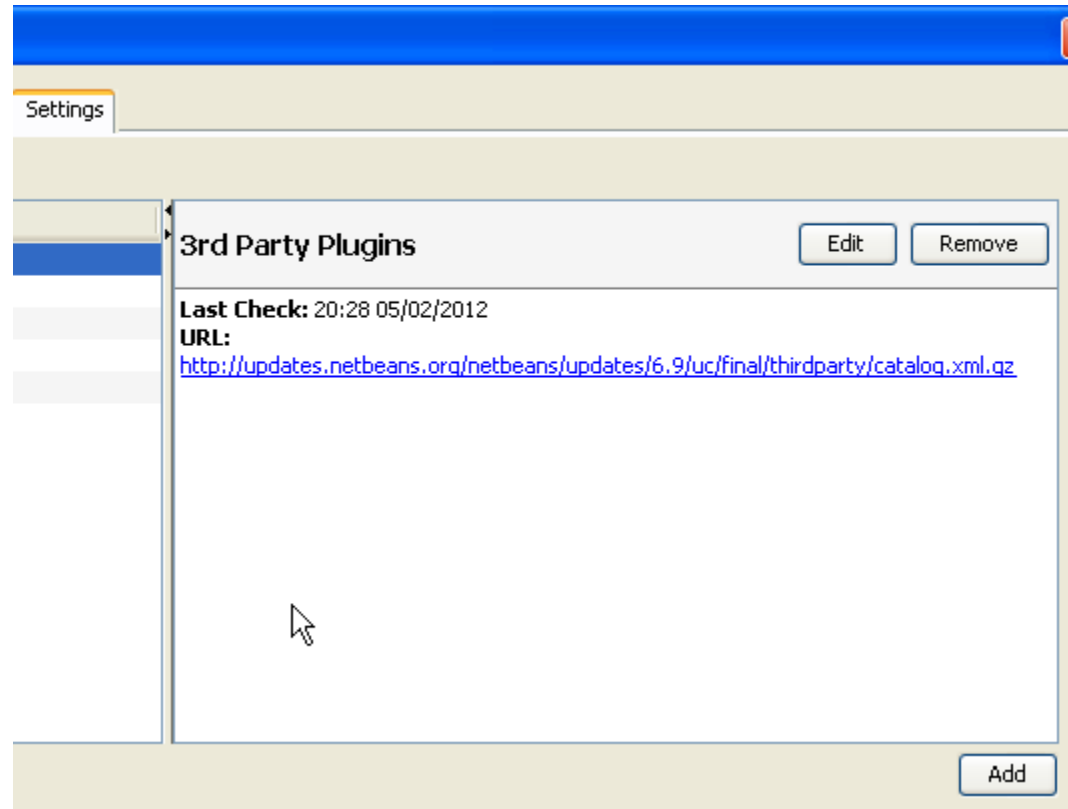
Structuring – XSD Data Types – string

Type	Description
NCNAME	<p>-An XML non-colonized name, typically used for the local names of namespace-qualified elements and attributes; the part after the prefix and the colon (e.g. a name that is not qualified with a namespace-related prefix)</p> <p>-Syntax: xsd:NCName</p>
ID	<p>A string that represents the ID attribute in XML (only used with schema attributes). Syntax: xsd:ID</p>
IDREF	<p>A string must match an ID value elsewhere in XML document (only used with schema attributes). Syntax: xsd:IDREF</p>
IDREFS	<p>List of ID values separated by white space (only used with schema attributes). Syntax: xsd:IDREFS</p>
ENTITY	<p>A String must match the name of an unparsed entity declared (only used with schema attributes). Syntax: xsd:ENTITY</p>
ENTITIES	<p>List of ENTITY names separated by white space (only used with schema attributes). Syntax: xsd:ENTITIES</p>

Schemas

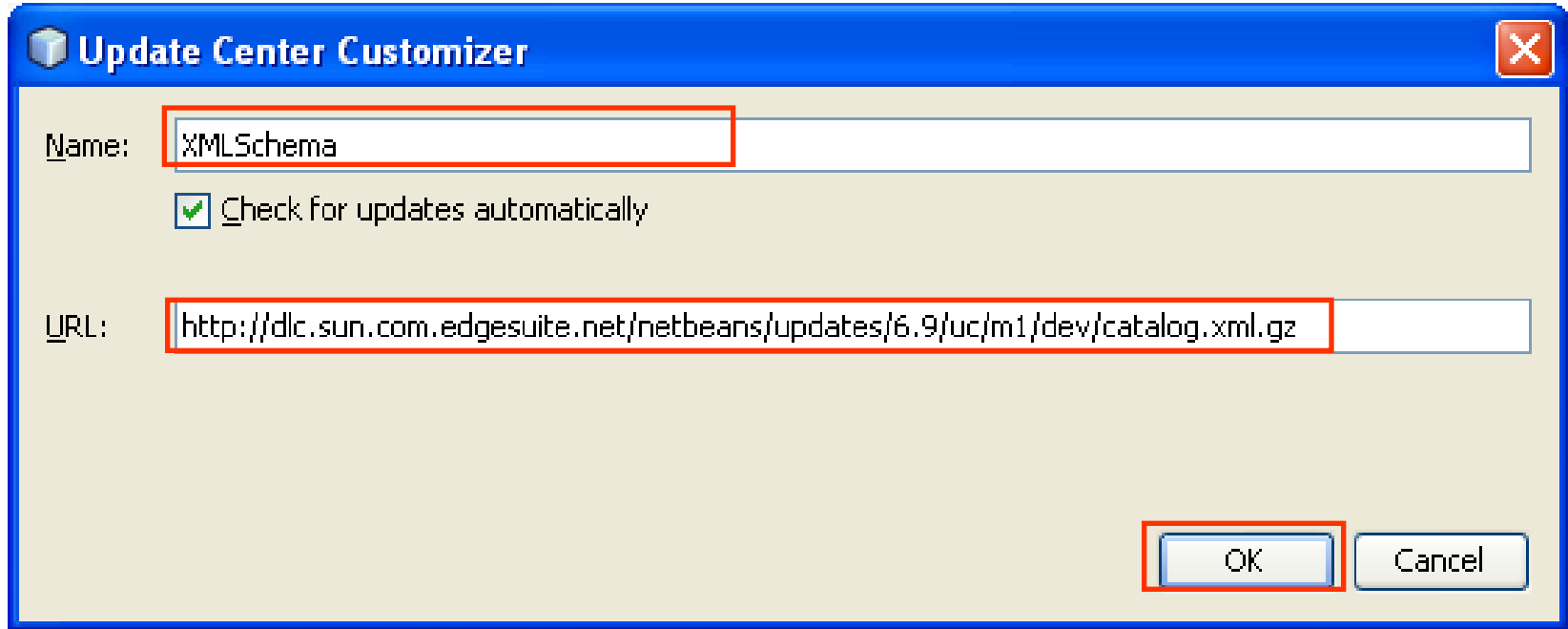
Add Plugin support Schema in Netbeans

- Open menu Tools, choose Plugin
 - Then choose tag Settings, then click Add button



Schemas

Add Plugin support Schema in Netbeans



The image shows a screenshot of the 'Update Center Customizer' dialog box in NetBeans. The dialog has a blue title bar with the text 'Update Center Customizer' and a close button (X) in the top right corner. The main area is light beige. It contains two text input fields, a checkbox, and two buttons. The 'Name' field contains 'XMLSchema'. The 'Check for updates automatically' checkbox is checked. The 'URL' field contains 'http://dlc.sun.com.edgesuite.net/netbeans/updates/6.9/uc/m1/dev/catalog.xml.gz'. The 'OK' and 'Cancel' buttons are at the bottom right. Red rectangular boxes highlight the 'Name' field, the 'URL' field, and the 'OK' button.

Update Center Customizer

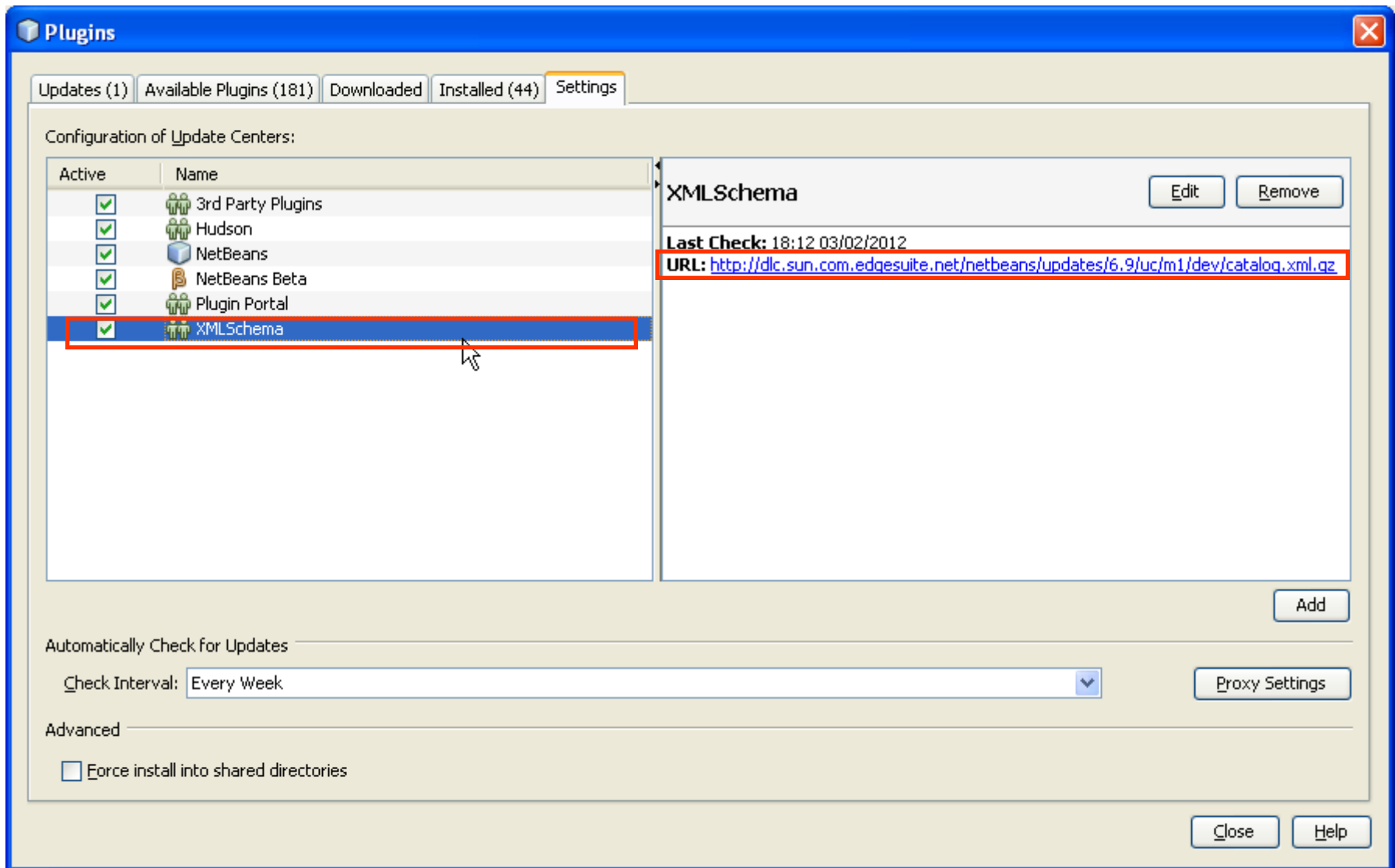
Name:

☒ Check for updates automatically

URL:

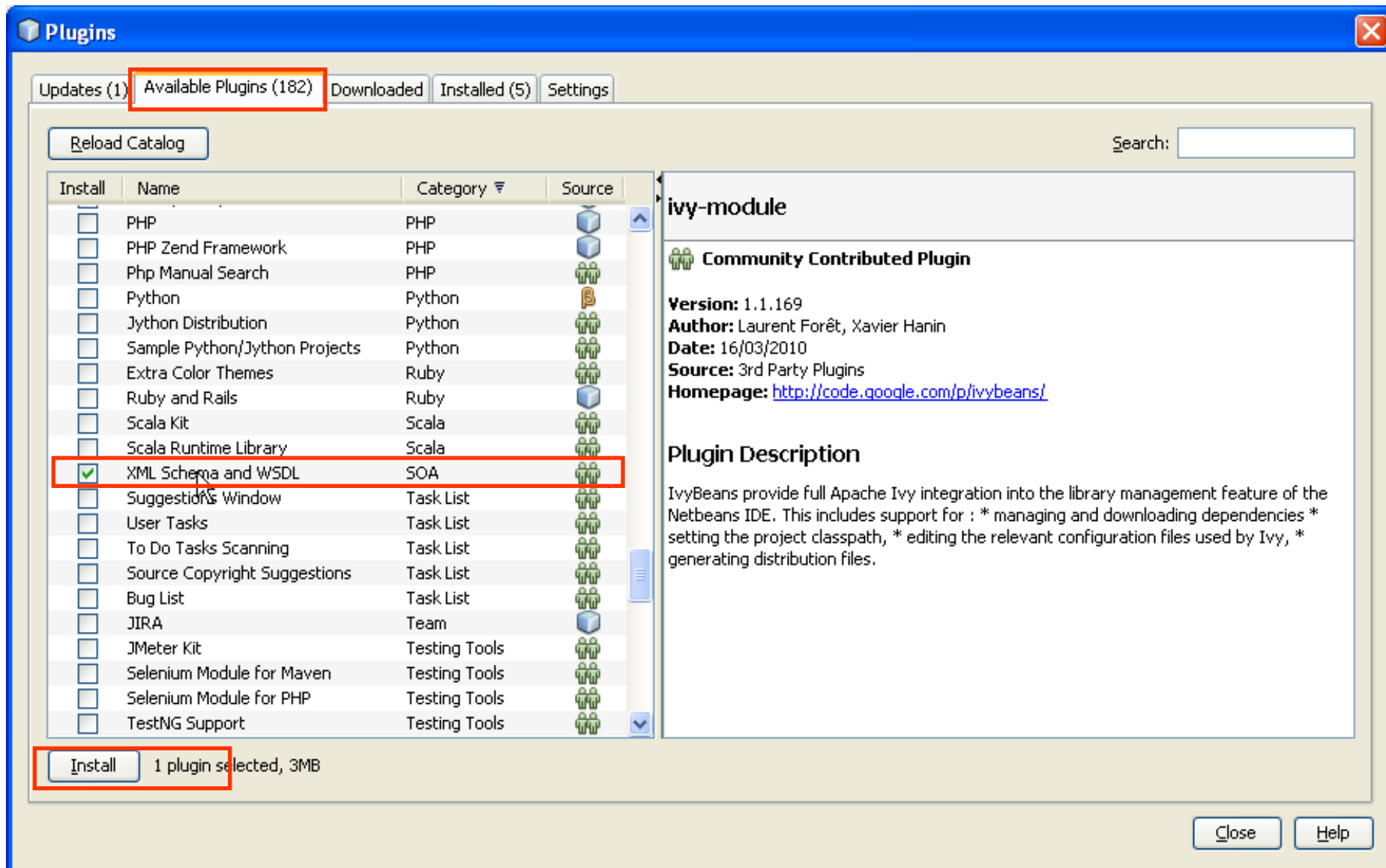
Schemas

Add Plugin support Schema in Netbeans



Schemas

Add Plugin support Schema in Netbeans



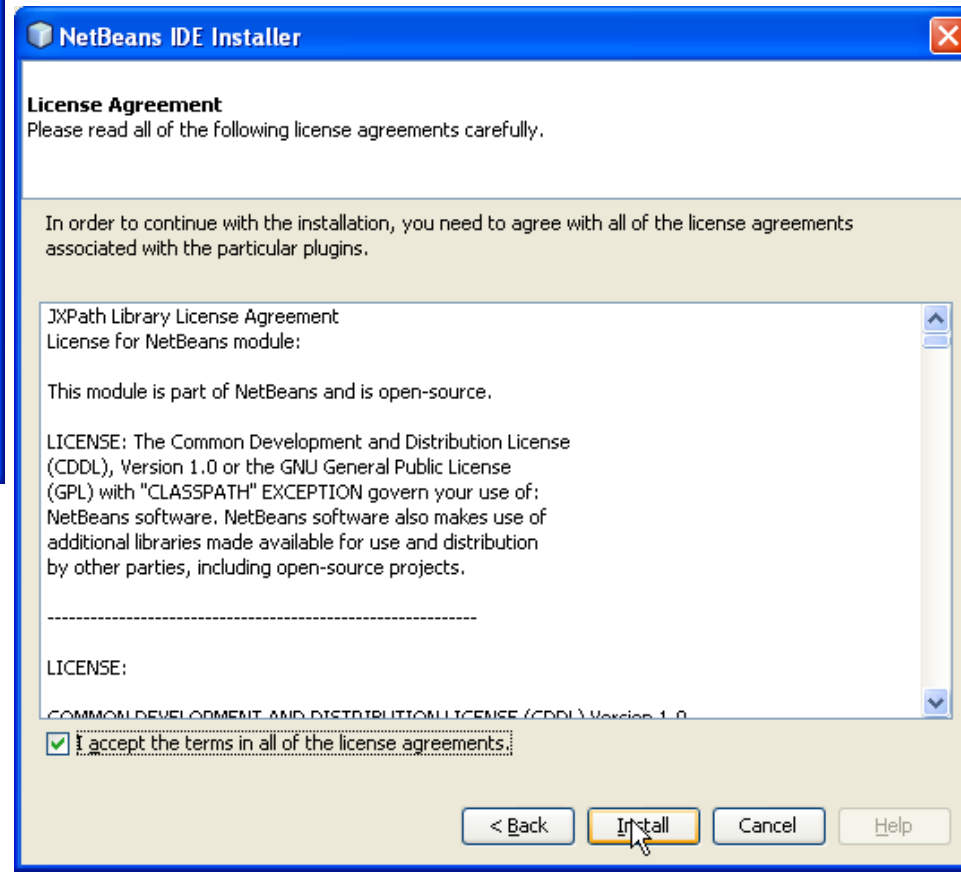
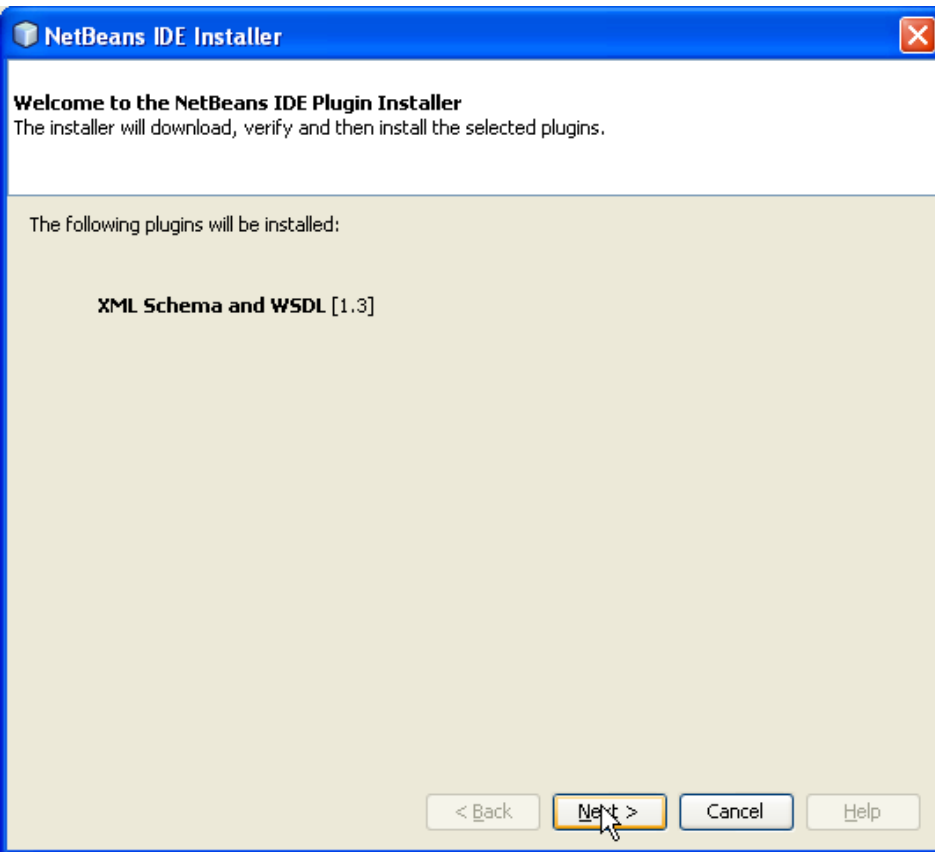
The screenshot shows the NetBeans 'Plugins' window. The 'Available Plugins (182)' tab is selected. The 'XML Schema and WSDL' plugin is highlighted in the list. The right pane shows details for the 'ivy-module' plugin, which is a 'Community Contributed Plugin' with version 1.1.169, authored by Laurent Forêt and Xavier Hanin, dated 16/03/2010. The description states that IvyBeans provide full Apache Ivy integration into the library management feature of the NetBeans IDE.

Install	Name	Category	Source
<input type="checkbox"/>	PHP	PHP	
<input type="checkbox"/>	PHP Zend Framework	PHP	
<input type="checkbox"/>	Php Manual Search	PHP	
<input type="checkbox"/>	Python	Python	
<input type="checkbox"/>	Jython Distribution	Python	
<input type="checkbox"/>	Sample Python/Jython Projects	Python	
<input type="checkbox"/>	Extra Color Themes	Ruby	
<input type="checkbox"/>	Ruby and Rails	Ruby	
<input type="checkbox"/>	Scala Kit	Scala	
<input type="checkbox"/>	Scala Runtime Library	Scala	
<input checked="" type="checkbox"/>	XML Schema and WSDL	SOA	
<input type="checkbox"/>	Suggestions Window	Task List	
<input type="checkbox"/>	User Tasks	Task List	
<input type="checkbox"/>	To Do Tasks Scanning	Task List	
<input type="checkbox"/>	Source Copyright Suggestions	Task List	
<input type="checkbox"/>	Bug List	Task List	
<input type="checkbox"/>	JIRA	Team	
<input type="checkbox"/>	JMeter Kit	Testing Tools	
<input type="checkbox"/>	Selenium Module for Maven	Testing Tools	
<input type="checkbox"/>	Selenium Module for PHP	Testing Tools	
<input type="checkbox"/>	TestNG Support	Testing Tools	

At the bottom of the window, the 'Install' button is highlighted, and the status bar indicates '1 plugin selected, 3MB'.

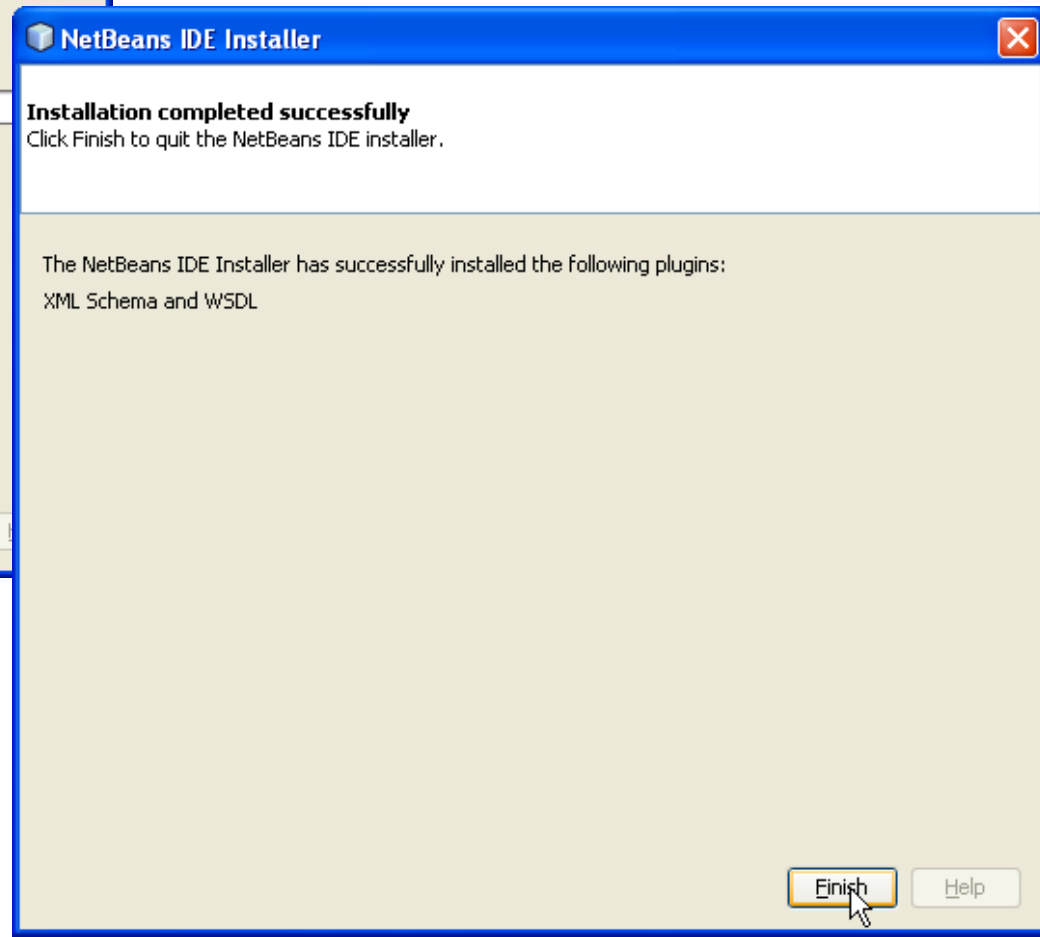
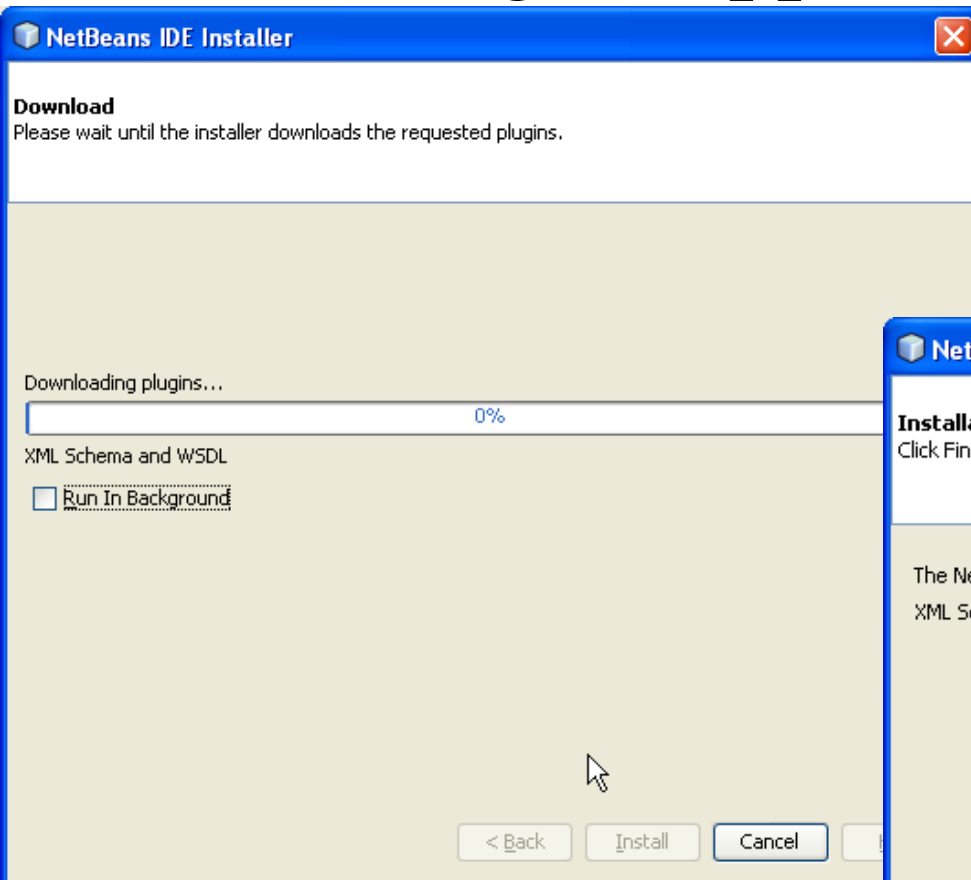
Schemas

Add Plugin support Schema in Netbeans



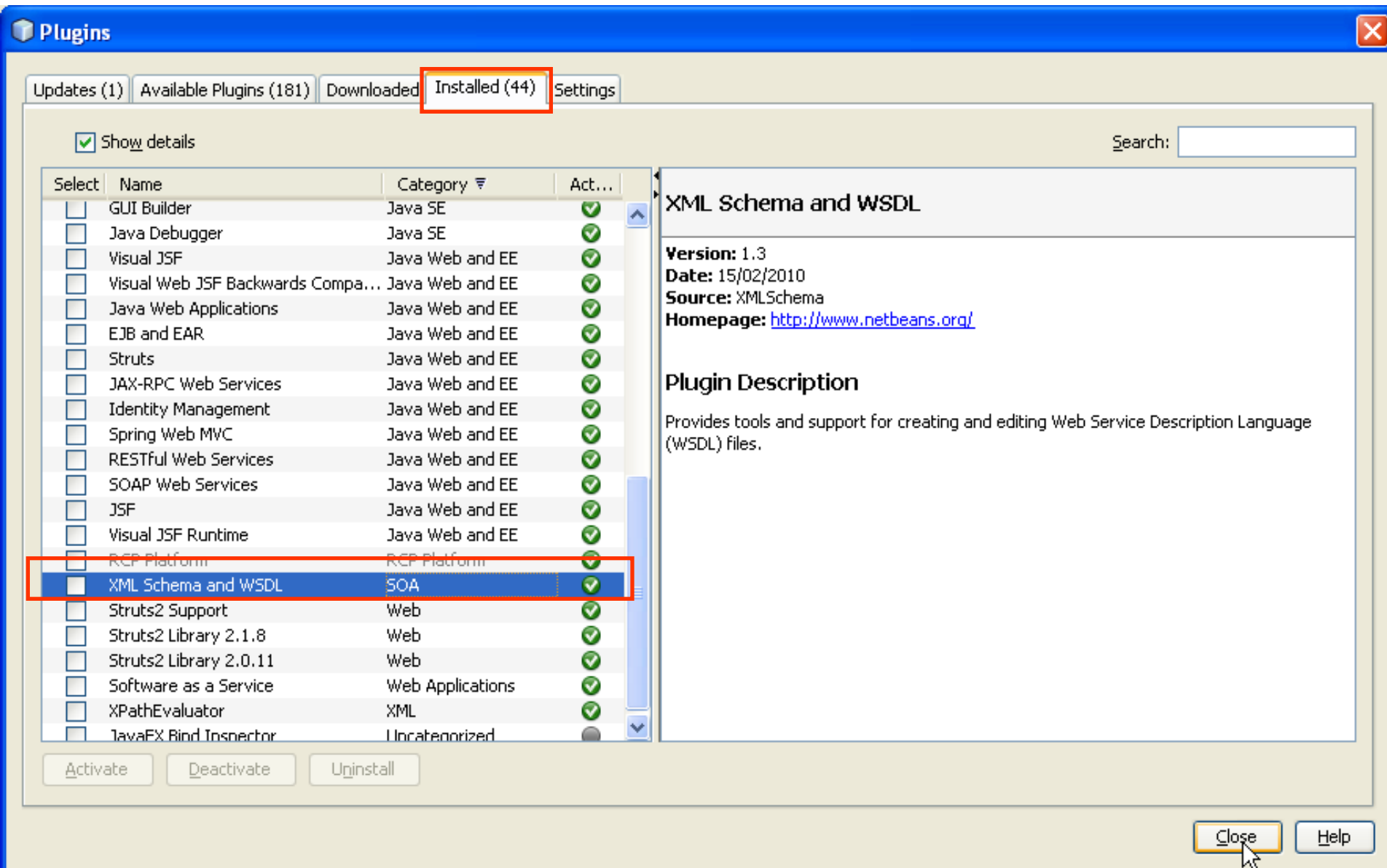
Schemas

Add Plugin support Schema in Netbeans



Schemas

Add Plugin support Schema in Netbeans



The screenshot shows the NetBeans 'Plugins' window. The 'Installed (44)' tab is selected. A list of installed plugins is shown on the left, with 'XML Schema and WSDL' highlighted. The right pane displays the details for this plugin.

Plugins

Updates (1) Available Plugins (181) Downloaded Installed (44) Settings

☒ Show details

Search:

Select	Name	Category	Act...
<input type="checkbox"/>	GUI Builder	Java SE	✓
<input type="checkbox"/>	Java Debugger	Java SE	✓
<input type="checkbox"/>	Visual JSF	Java Web and EE	✓
<input type="checkbox"/>	Visual Web JSF Backwards Compa...	Java Web and EE	✓
<input type="checkbox"/>	Java Web Applications	Java Web and EE	✓
<input type="checkbox"/>	EJB and EAR	Java Web and EE	✓
<input type="checkbox"/>	Struts	Java Web and EE	✓
<input type="checkbox"/>	JAX-RPC Web Services	Java Web and EE	✓
<input type="checkbox"/>	Identity Management	Java Web and EE	✓
<input type="checkbox"/>	Spring Web MVC	Java Web and EE	✓
<input type="checkbox"/>	RESTful Web Services	Java Web and EE	✓
<input type="checkbox"/>	SOAP Web Services	Java Web and EE	✓
<input type="checkbox"/>	JSF	Java Web and EE	✓
<input type="checkbox"/>	Visual JSF Runtime	Java Web and EE	✓
<input type="checkbox"/>	RCP Platform	RCP Platform	✓
<input checked="" type="checkbox"/>	XML Schema and WSDL	SOA	✓
<input type="checkbox"/>	Struts2 Support	Web	✓
<input type="checkbox"/>	Struts2 Library 2.1.8	Web	✓
<input type="checkbox"/>	Struts2 Library 2.0.11	Web	✓
<input type="checkbox"/>	Software as a Service	Web Applications	✓
<input type="checkbox"/>	XPathEvaluator	XML	✓
<input type="checkbox"/>	JavaFX Bind Inspector	Uncategorized	✓

XML Schema and WSDL

Version: 1.3
Date: 15/02/2010
Source: XMLSchema
Homepage: <http://www.netbeans.org/>

Plugin Description

Provides tools and support for creating and editing Web Service Description Language (WSDL) files.

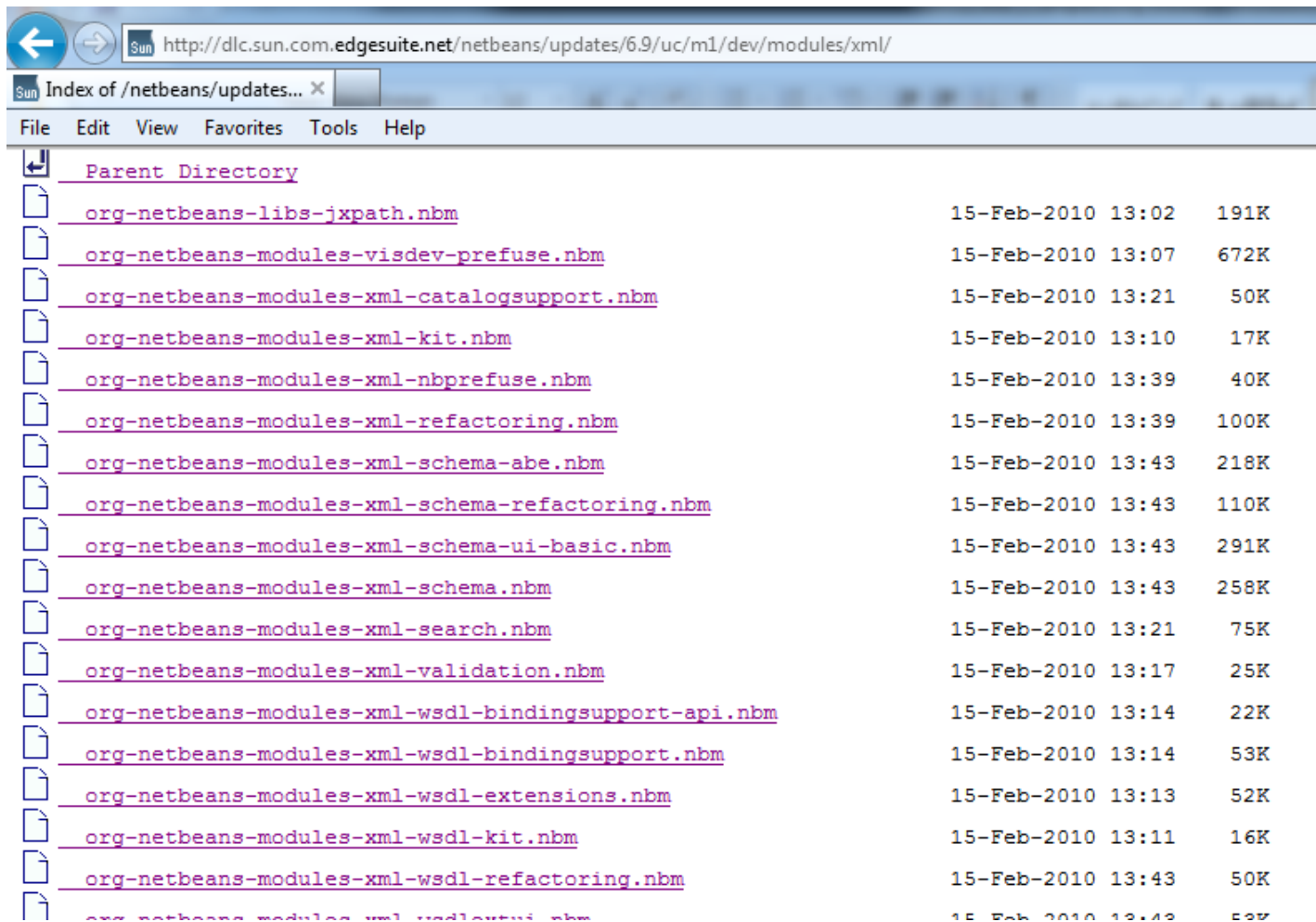
Activate Deactivate Uninstall

Close Help

Schemas

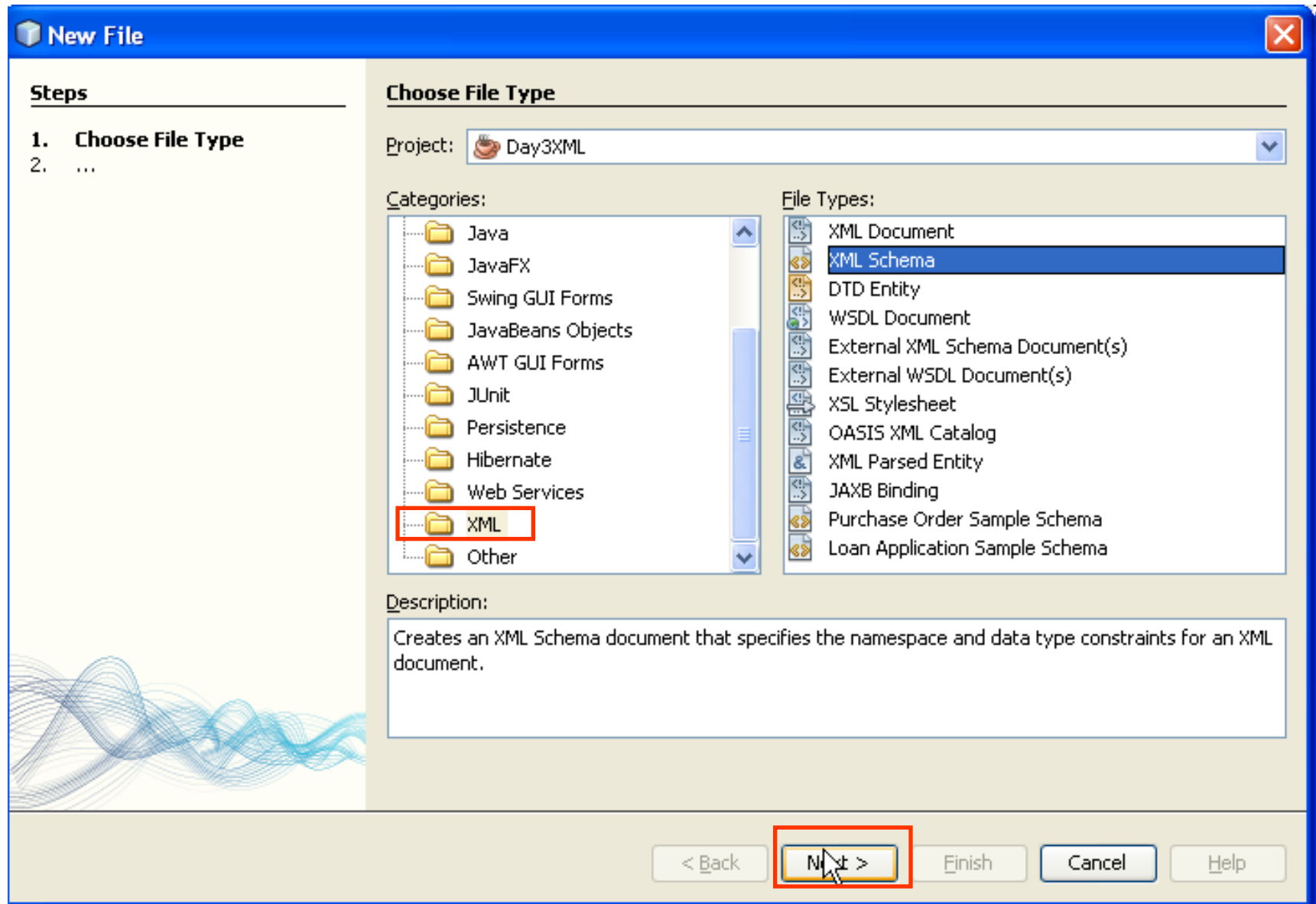
Add Plugin support Schema in Netbeans

- Other ways
 - Download <http://dlc.sun.com.edgesuite.net/netbeans/updates/6.9/uc/m1/dev/modules/xml/> file from
 - Then, copy all to the .netbeans\6.9\update\download, and restart the Netbeans



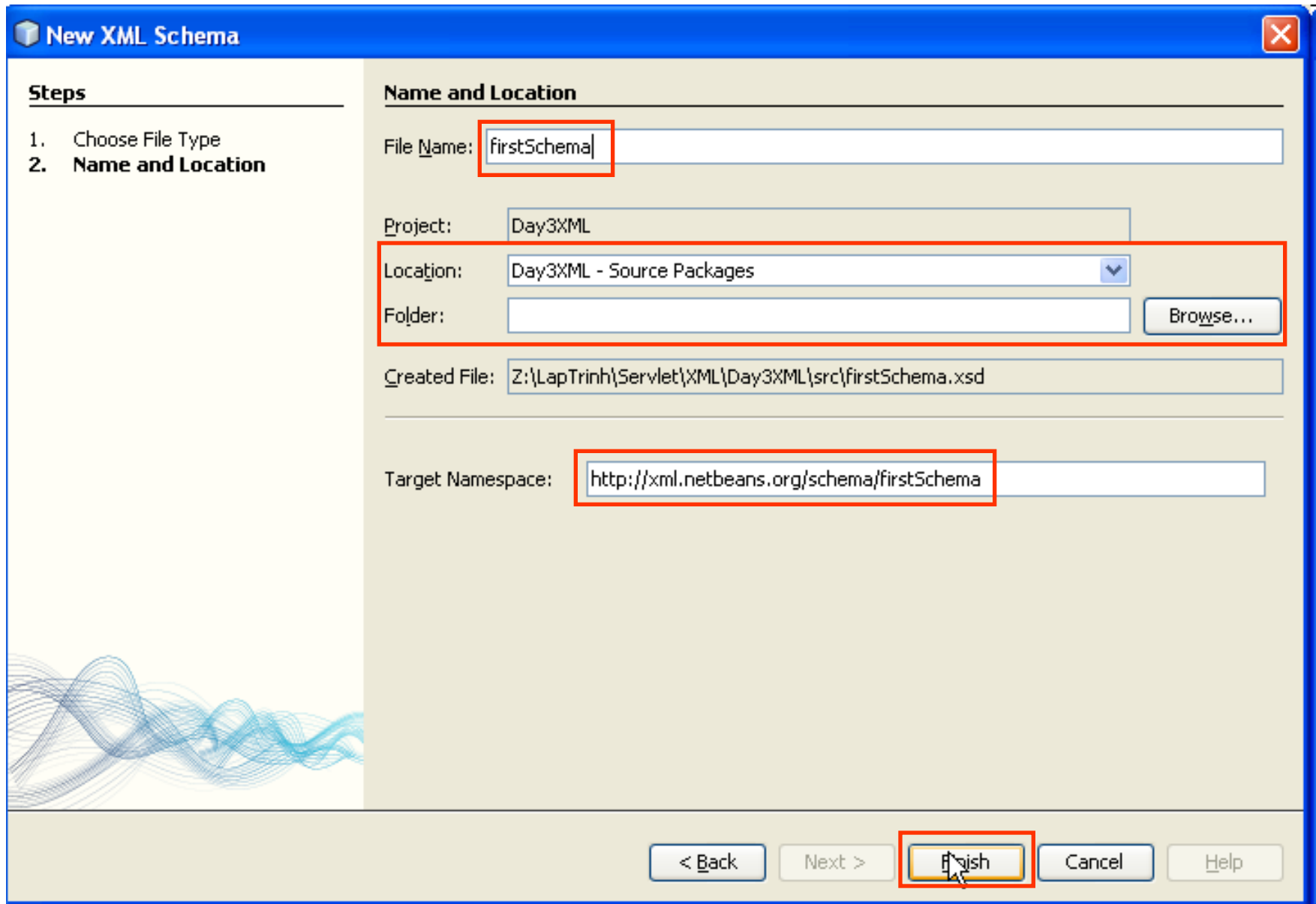
Schemas

How to write Schema in Netbeans



Schemas

How to write Schema in Netbeans



The image shows the 'New XML Schema' dialog box in NetBeans. The dialog has a blue title bar with the text 'New XML Schema' and a close button. On the left, there is a 'Steps' section with two steps: '1. Choose File Type' and '2. Name and Location'. The main area is titled 'Name and Location' and contains several input fields and buttons. The 'File Name' field is set to 'firstSchema'. The 'Project' field is set to 'Day3XML'. The 'Location' field is a dropdown menu set to 'Day3XML - Source Packages'. The 'Folder' field is empty, with a 'Browse...' button next to it. The 'Created File' field shows the path 'Z:\LapTrinh\Servlet\XML\Day3XML\src\firstSchema.xsd'. The 'Target Namespace' field is set to 'http://xml.netbeans.org/schema/firstSchema'. At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Finish' button is highlighted with a red box and a mouse cursor.

New XML Schema

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

File Name: firstSchema

Project: Day3XML

Location: Day3XML - Source Packages

Folder: Browse...

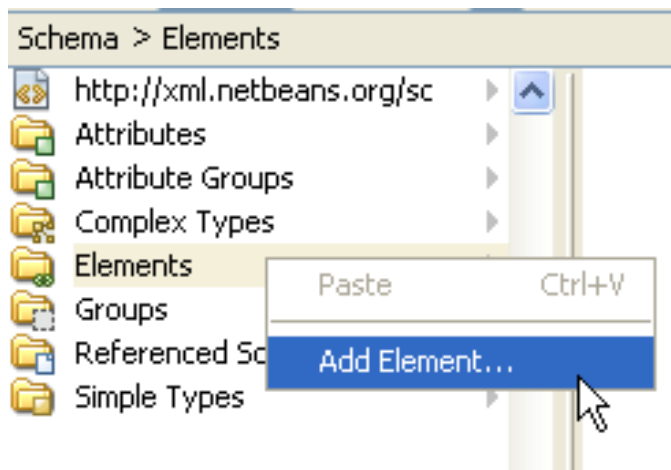
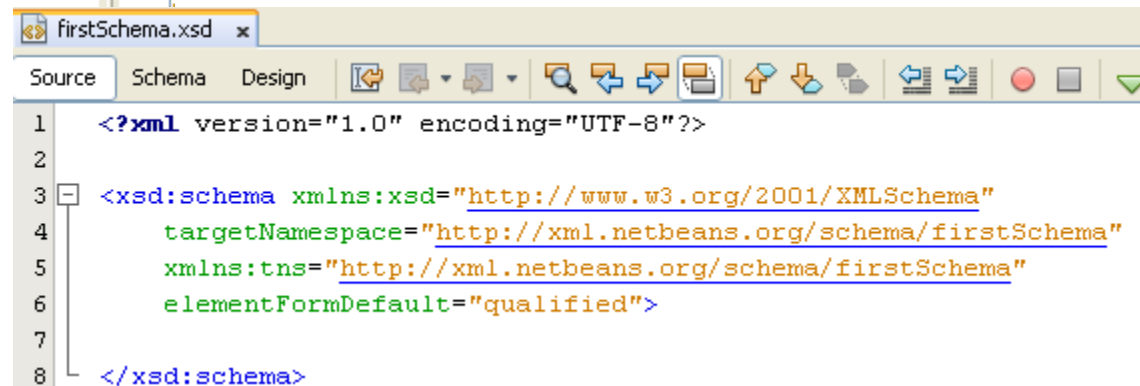
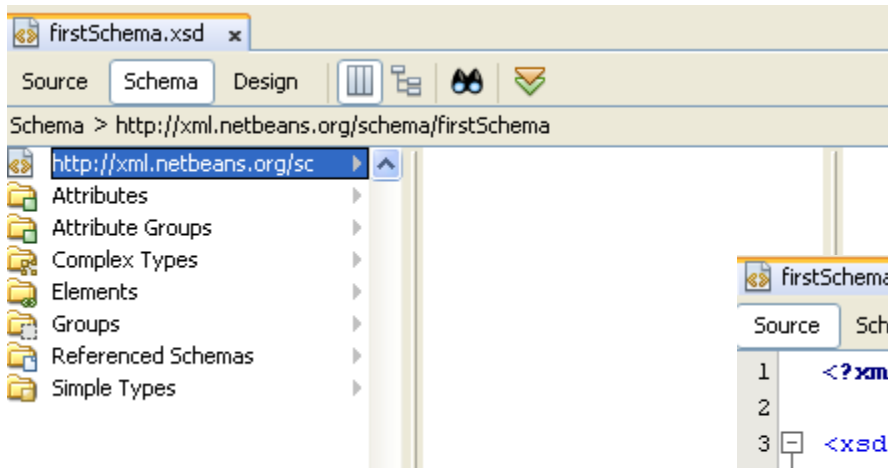
Created File: Z:\LapTrinh\Servlet\XML\Day3XML\src\firstSchema.xsd

Target Namespace: http://xml.netbeans.org/schema/firstSchema

< Back Next > Finish Cancel Help

Schemas

How to write Schema in Netbeans



Click right mouse

Schemas

How to write Schema in Netbeans

The image shows the NetBeans IDE interface for creating an XML schema. The 'Add Element' dialog is open, with 'mail' entered in the 'Name' field. The 'Type' section shows 'Inline Complex Type' selected. The 'Use Existing Type' option is also visible. The 'Schema' tab is active, showing a tree view of the schema structure: Schema > Elements > mail > complexType > sequence. The 'Preview' section shows the generated XML code for the 'mail' element.

Add Element Dialog:

- Name: mail
- Type:
 - ☒ Inline Complex Type
 - ☐ Inline Simple Type
 - ☐ No Type
 - ☐ Use Existing Type
- Schema: firstSchema.xsd
- Schema > Elements > mail > complexType > sequence

Preview:

```
<xsd:element name="mail">
  <xsd:complexType>
    <xsd:sequence/>
  </xsd:complexType>
</xsd:element>
```

firstSchema.xsd File:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/firstSchema"
5   xmlns:tns="http://xml.netbeans.org/schema/firstSchema"
6   elementFormDefault="qualified">
7   <xsd:element name="mail">
8     <xsd:complexType>
9       <xsd:sequence/>
10    </xsd:complexType>
11  </xsd:element>
12 </xsd:schema>
```

The 'OK' button is highlighted in the bottom right corner of the 'Add Element' dialog.

Schemas


Structuring–XSD Data Types–string–Example

```

fixDefault.xsd
Source Schema Design
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/mail"
5   xmlns="http://xml.netbeans.org/schema/mail"
6   elementFormDefault="qualified">
7   <xsd:element name="mail">
8     <xsd:complexType>
9       <xsd:sequence>
10        <xsd:element name="to" type="xsd:string" />
11        <xsd:element name="from" type="xsd:string" fixed="KhanhKT@fpt.edu.vn"/>
12        <xsd:element name="header" type="xsd:string" default="No Subject" />
13        <xsd:element name="body" type="xsd:string" default="Kieu Trong Khanh"/>
14      </xsd:sequence>
15    </xsd:complexType>
16  </xsd:element>
17 </xsd:schema>
  
```

Structuring–XSD Data Types–string–Example

fixDefault.xml



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <mail  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3      xmlns='http://xml.netbeans.org/schema/mail'
4      xsi:schemaLocation='http://xml.netbeans.org/schema/mail fixDefault.xsd'>
5      <to>KhanhKT</to>
6      <from>KhanhKK</from>
7      <header>XML Schema</header>
8      <body>Welcome to XML Schema Topic</body>
9  </mail>

```

Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/fixDefault.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/fixDefault.xsd".
cvc-elt.5.2.2.2.2: The value 'KhanhKK' of element 'from' does not match the {value constraint} value 'KhanhKT@fpt.edu.vn'. [6]
XML validation finished.

```

Structuring–XSD Data Types–string–Example

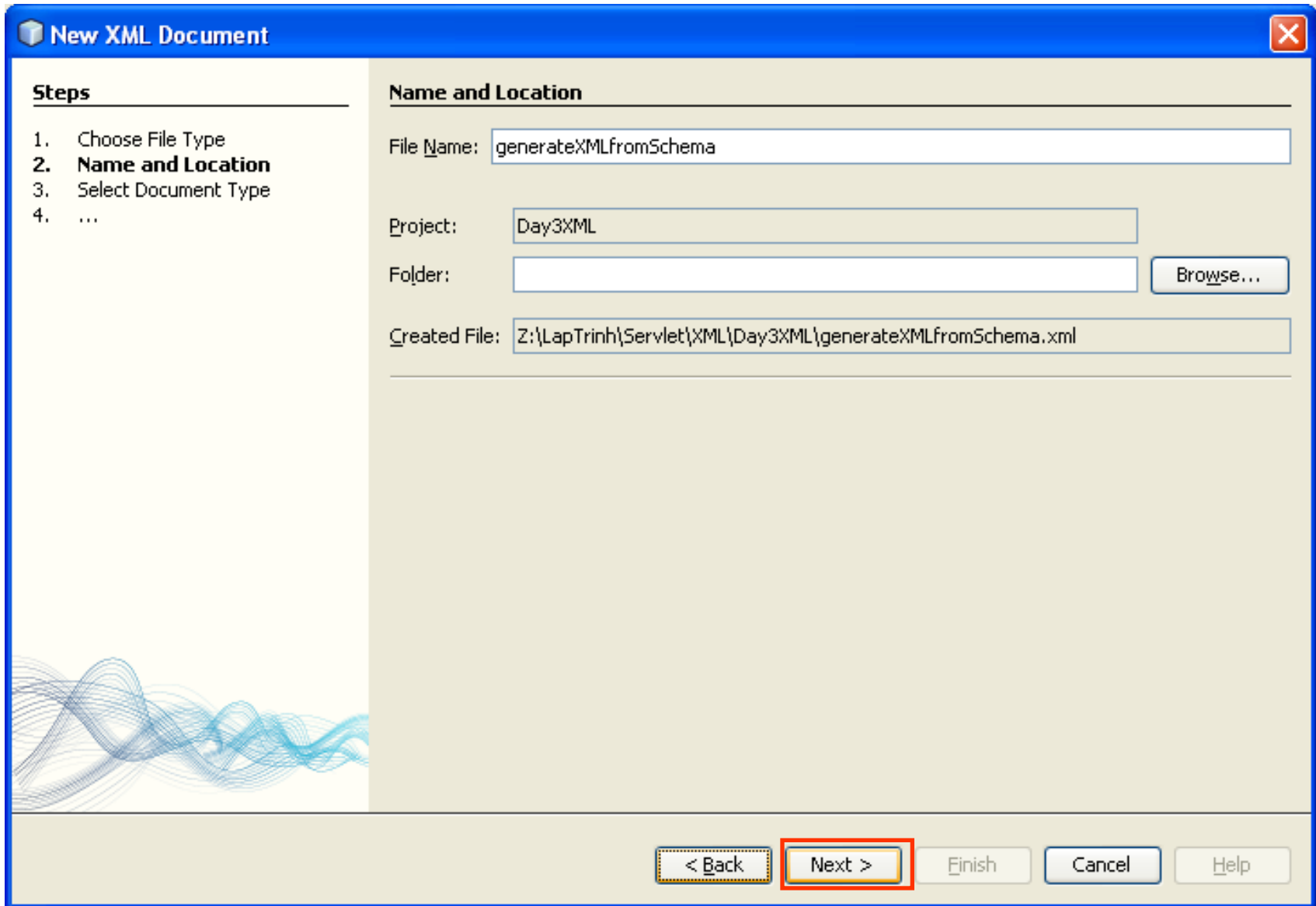
```

fixDefault.xml x
[Icons: Save, Open, Print, Find, Copy, Paste, Undo, Redo, etc.]

1  <?xml version="1.0" encoding="UTF-8"?>
2  <mail xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3    xmlns='http://xml.netbeans.org/schema/mail'
4    xsi:schemaLocation='http://xml.netbeans.org/schema/mail fixDefault.xsd'>
5    <to>KhanhKT</to>
6    <from>KhanhKT@fpt.edu.vn</from>
7    <header>XML Schema</header>
8    <body>Welcome to XML Schema Topic</body>
9  </mail>
  
```

Schemas

How to generate XML from Schema in Netbeans



The image shows the 'New XML Document' dialog box in NetBeans. The dialog has a blue title bar with the text 'New XML Document' and a close button. On the left, there is a 'Steps' section with a list of four steps: 1. Choose File Type, 2. **Name and Location**, 3. Select Document Type, and 4. ... The second step is highlighted. The main area of the dialog is titled 'Name and Location' and contains several input fields and buttons. The 'File Name' field is filled with 'generateXMLfromSchema'. The 'Project' field is filled with 'Day3XML'. The 'Folder' field is empty, and there is a 'Browse...' button next to it. The 'Created File' field is filled with 'Z:\LapTrinh\Servlet\XML\Day3XML\generateXMLfromSchema.xml'. At the bottom of the dialog, there are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'. The 'Next >' button is highlighted with a red rectangle.

New XML Document

Steps

1. Choose File Type
2. **Name and Location**
3. Select Document Type
4. ...

Name and Location

File Name: generateXMLfromSchema

Project: Day3XML

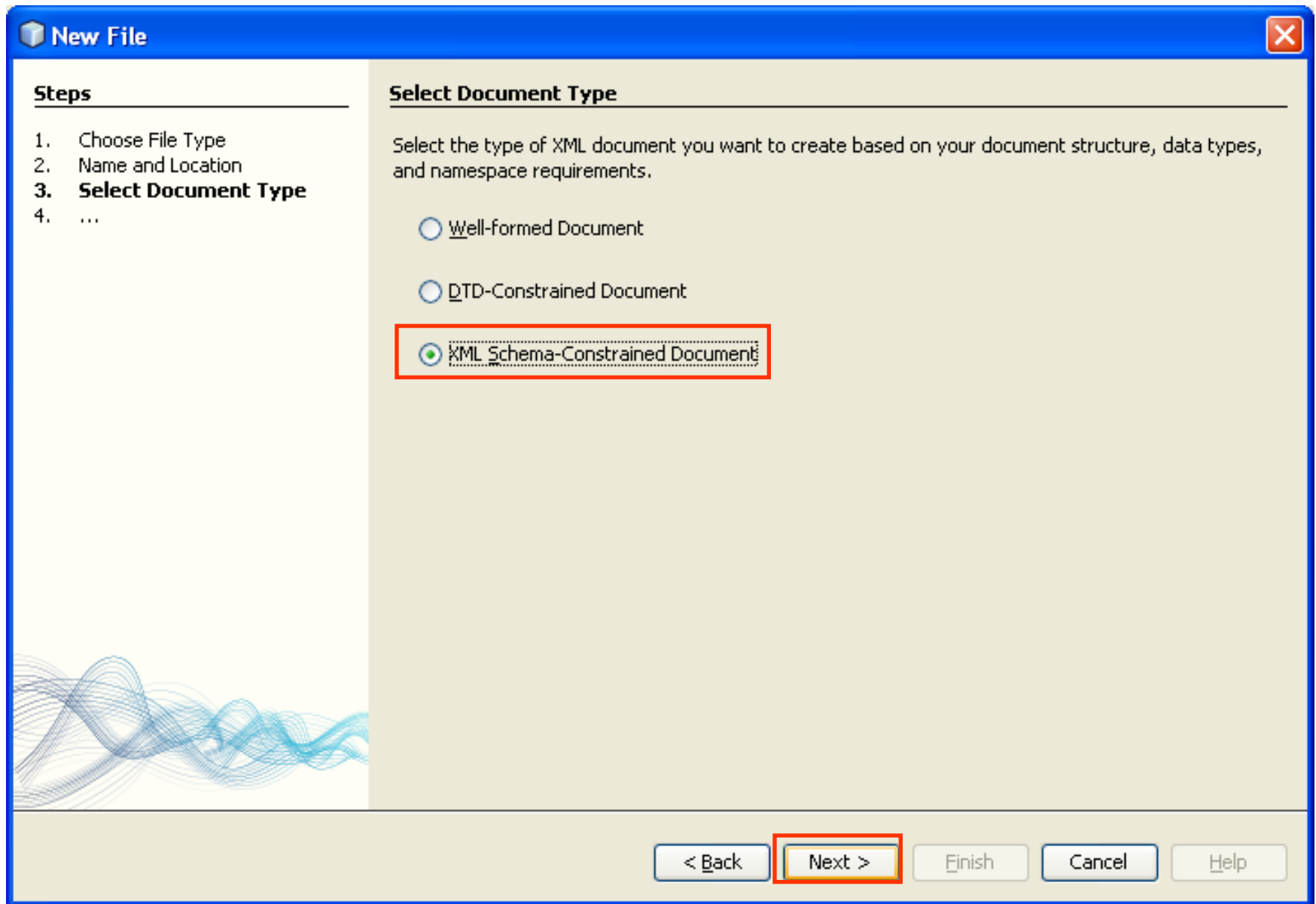
Folder: Browse...

Created File: Z:\LapTrinh\Servlet\XML\Day3XML\generateXMLfromSchema.xml

< Back Next > Finish Cancel Help

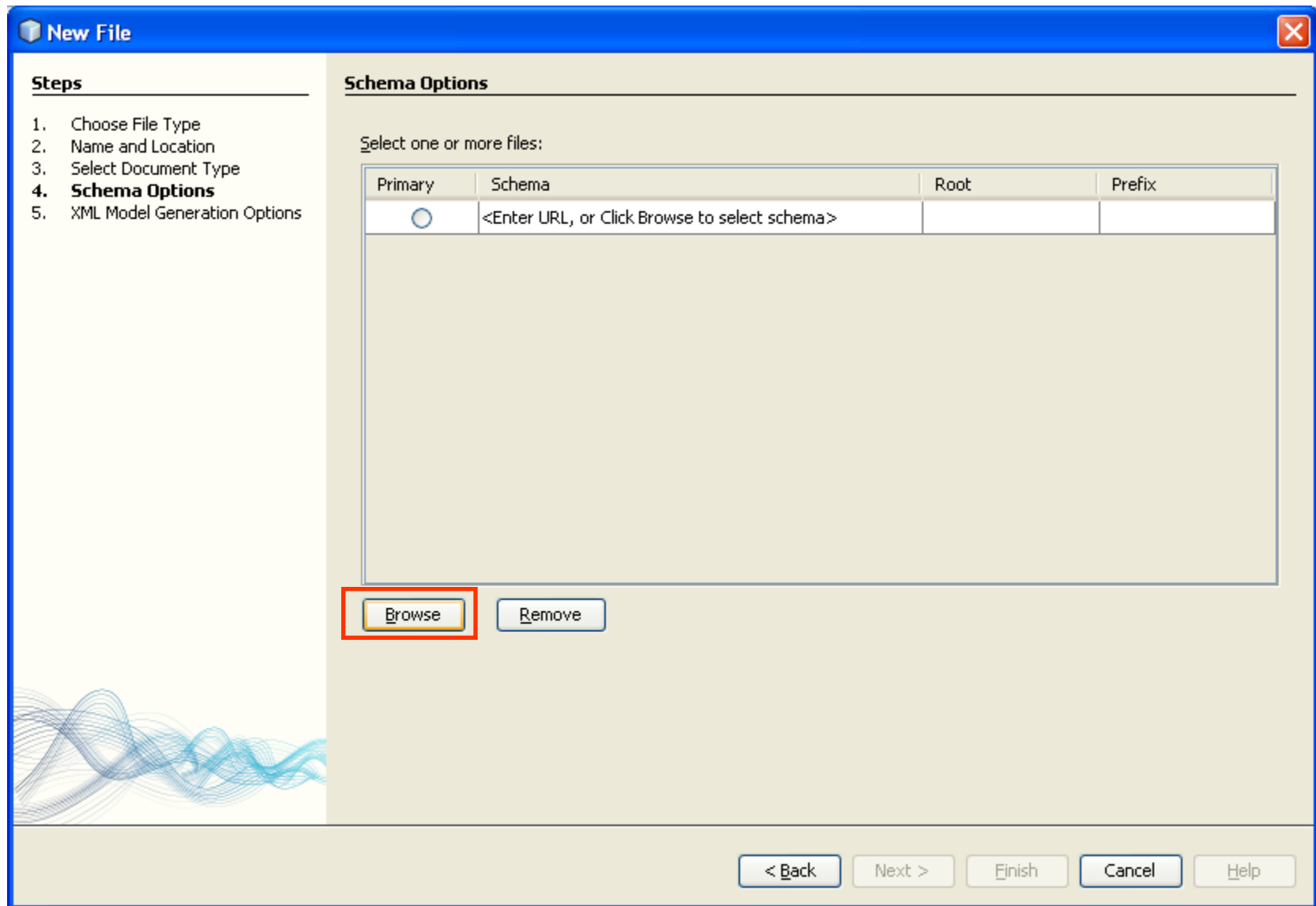
Schemas

How to generate XML from Schema in Netbeans



Schemas

How to generate XML from Schema in Netbeans



New File

Steps

1. Choose File Type
2. Name and Location
3. Select Document Type
- 4. Schema Options**
5. XML Model Generation Options

Schema Options

Select one or more files:

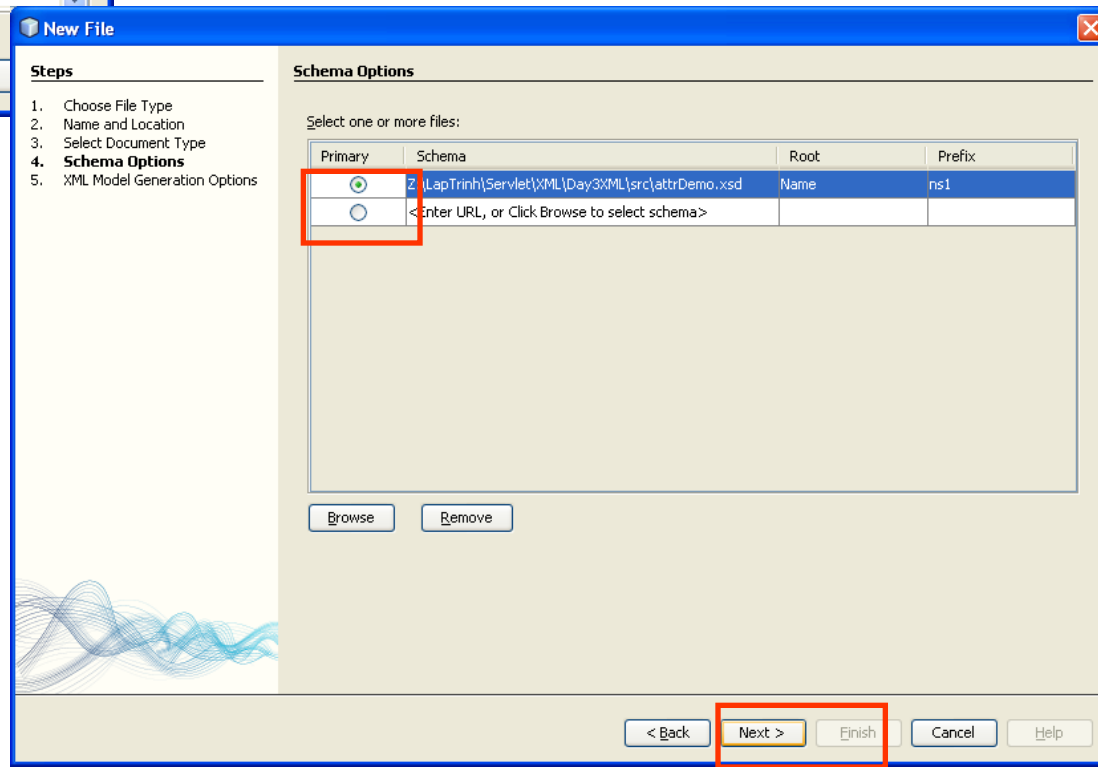
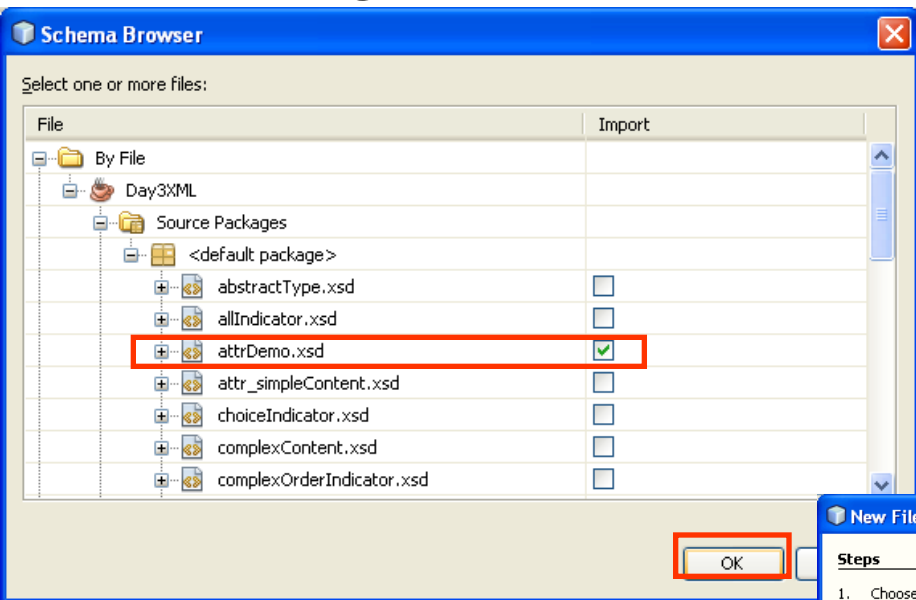
Primary	Schema	Root	Prefix
<input type="radio"/>	<Enter URL, or Click Browse to select schema>		

Browse **Remove**

< Back Next > Finish Cancel Help

Schemas

How to generate XML from Schema in Netbeans



Schemas

How to generate XML from Schema in Netbeans

New File

Steps

1. Choose File Type
2. Name and Location
3. Select Document Type
4. Schema Options
5. **XML Model Generation Options**

XML Model Generation Options

Select options for including optional content in the XML document

Content Options

☒ Generate Optional Elements

☒ Generate Optional Attributes

Repeating Elements

Preferred occurrences: Value to user when "preferred > minOccurs" and "preferred < maxOccurs".

Recursion depth: Depth to which self-referencing elements in the XML document should recurse

< Back Next > **Finish** Cancel Help

Schemas

How to generate XML from Schema in Netbeans

```

generateXMLfromSchema.xml x
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <!--
4     Document    : generateXMLfromSchema.xml
5     Created on  : Ngày 06 tháng 2 năm 2012, 21:12
6     Author     : Trong Khanh
7     Description:
8         Purpose of the document follows.
9 -->
10
11 <ns1:Name xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
12     xmlns:ns1='http://xml.netbeans.org/schema/attrDemo'
13     xsi:schemaLocation='http://xml.netbeans.org/schema/attrDemo src/attrDemo.xsd'
14     Pseudonym="true" HomePage="http://www.w3.org/">
15     <ns1:FirstName></ns1:FirstName>
16     <ns1:LastName></ns1:LastName>
17
18 </ns1:Name>
  
```

```

generateXMLfromSchema.xml x attrDemo.xsd x
Source Schema Design
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4     targetNamespace="http://xml.netbeans.org/schema/attrDemo"
5     xmlns="http://xml.netbeans.org/schema/attrDemo"
6     elementFormDefault="qualified">
7     <xsd:element name="Name">
8         <xsd:complexType>
9             <xsd:sequence>
10                 <xsd:element name="FirstName" type="xsd:string"/>
11                 <xsd:element name="LastName" type="xsd:string"/>
12             </xsd:sequence>
13             <xsd:attribute name="Pseudonym" type="xsd:boolean" fixed="true"
14                 use="required"/>
15             <xsd:attribute name="HomePage" type="xsd:anyURI"
16                 default="http://www.w3.org/" />
17         </xsd:complexType>
18     </xsd:element>
19 </xsd:schema>
  
```

Structuring–XSD Data Types–string–Example

```

fixDefault.xsd
Source Schema Design
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/mail"
5   xmlns="http://xml.netbeans.org/schema/mail"
6   elementFormDefault="qualified">
7   <xsd:element name="mail">
8     <xsd:complexType>
9       <xsd:sequence>
10        <xsd:element name="to" type="xsd:string" />
11        <xsd:element name="from" type="xsd:string" fixed="KhanhKT@fpt.edu.vn"
12          default="KhanhKT@fpt.edu.vn"/>
13        <xsd:element name="header" type="xsd:string" default="No Subject" />
14        <xsd:element name="body" type="xsd:string" default="Kieu Trong Khanh"/>
15      </xsd:sequence>
16    </xsd:complexType>
17  </xsd:element>
18 </xsd:schema>
  
```

Output - XML check

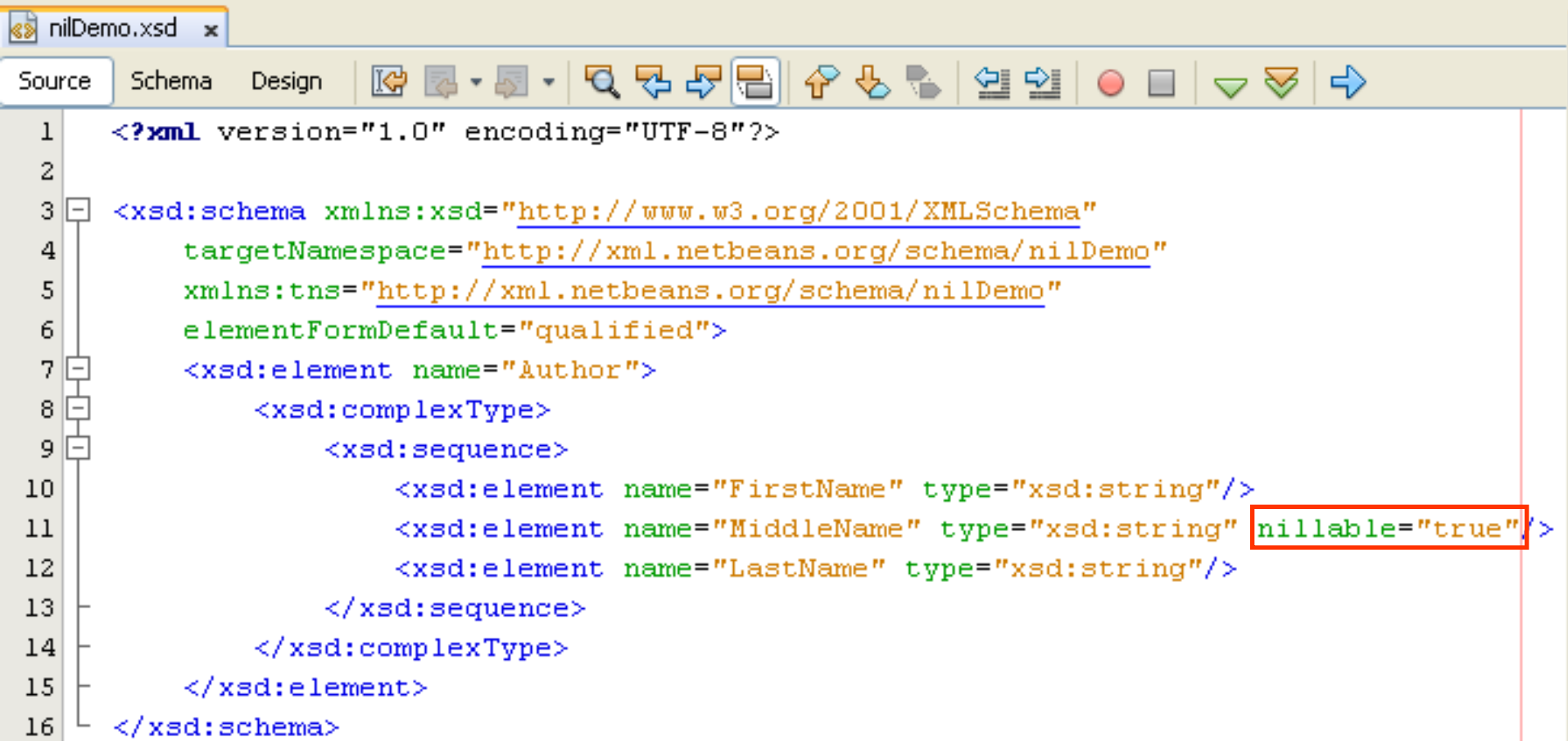
XML validation started.

Z:/Laptrinh/Servlet/Day3/XML/src/fixDefault.xsd:11,16

ERROR: src-element.1: The properties 'default' and 'fixed' cannot both be present in element declaration 'from'. Use only one of them.

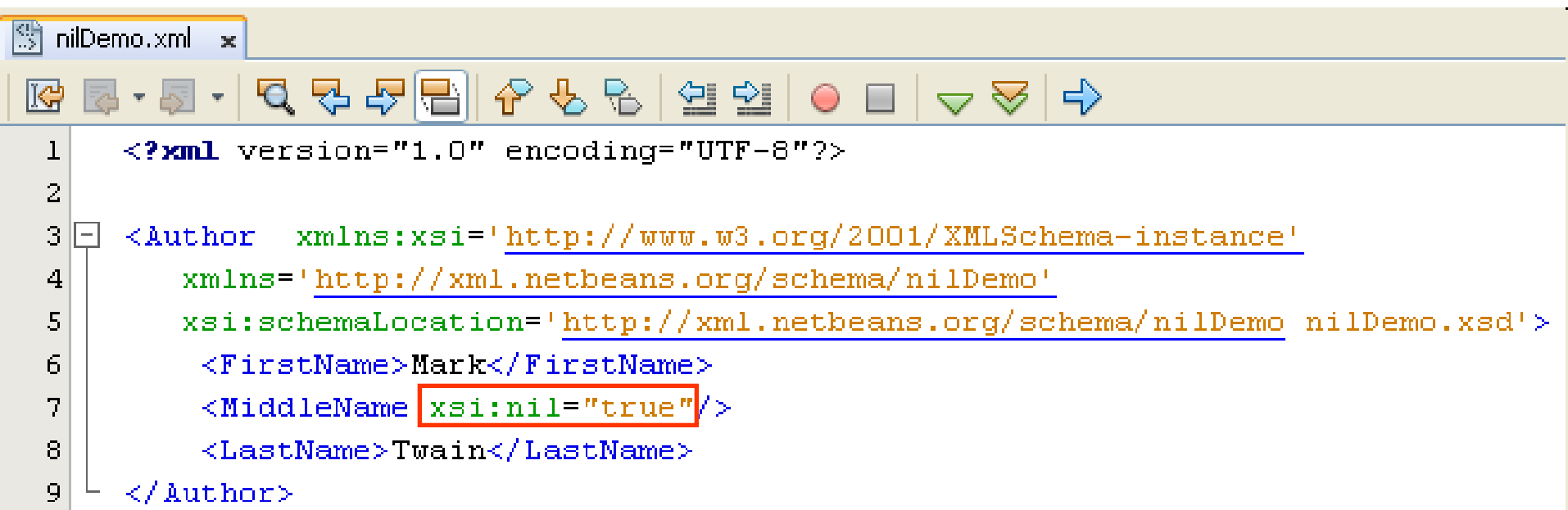
1 Error(s), 0 Warning(s).

XML validation finished.



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/nilDemo"
5      xmlns:tns="http://xml.netbeans.org/schema/nilDemo"
6      elementFormDefault="qualified">
7      <xsd:element name="Author">
8          <xsd:complexType>
9              <xsd:sequence>
10                 <xsd:element name="FirstName" type="xsd:string"/>
11                 <xsd:element name="MiddleName" type="xsd:string" nillable="true"/>
12                 <xsd:element name="LastName" type="xsd:string"/>
13             </xsd:sequence>
14         </xsd:complexType>
15     </xsd:element>
16 </xsd:schema>
  
```


 The image shows a screenshot of an XML editor window titled 'nilDemo.xml'. The editor has a toolbar with various icons for file operations, navigation, and editing. The XML content is displayed in a text area with line numbers on the left. The XML code is as follows:


```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Author xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/nilDemo'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/nilDemo nilDemo.xsd'>
6      <FirstName>Mark</FirstName>
7      <MiddleName xsi:nil="true"/>
8      <LastName>Twain</LastName>
9  </Author>
  
```

 The line number '3' is highlighted in blue. The 'xsi:nil="true"' attribute in the 'MiddleName' element is highlighted with a red rectangle.

Structuring–XSD Data Types–string–Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Author xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns='http://xml.netbeans.org/schema/nilDemo'
5        xsi:schemaLocation='http://xml.netbeans.org/schema/nilDemo nilDemo.xsd'>
6    <FirstName>Mark</FirstName>
7    <MiddleName xsi:nil="true">T.</MiddleName>
8    <LastName>Twain</LastName>
9  </Author>
  
```

Output - XML check

```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/nilDemo.xml...
[ Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/nilD
emo.xsd".
[ cvc-elt.3.2.1: Element 'MiddleName' cannot have character or element
[   information [children], because 'http://www.w3.org/2001/XMLSchema-i
[   nstance,nil' is specified. [7]
XML validation finished.
  
```

Structuring–XSD Data Types–string–Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Author xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns='http://xml.netbeans.org/schema/nilDemo'
5        xsi:schemaLocation='http://xml.netbeans.org/schema/nilDemo nilDemo.xsd'>
6    <FirstName>Mark</FirstName>
7    <MiddleName>T.</MiddleName>
8    <LastName>Twain</LastName>
9  </Author>
  
```

Output - XML check

```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/nilDemo.xml...
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/nilDemo.xsd".
XML validation finished.
  
```

Structuring–XSD Data Types–string–Example

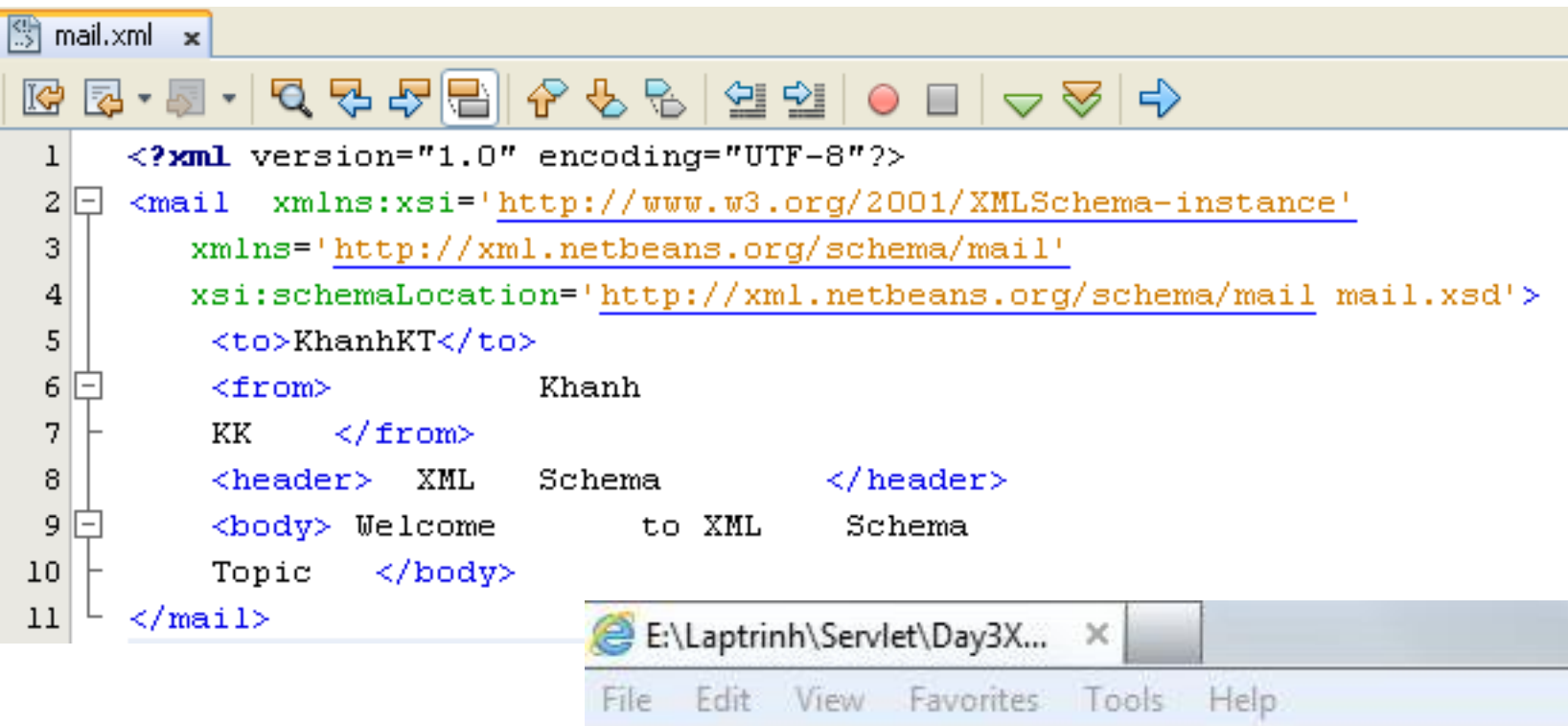
mail.xsd x

Source Schema Design

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/mail"
5      xmlns="http://xml.netbeans.org/schema/mail"
6      elementFormDefault="qualified">
7      <xsd:element name="mail">
8          <xsd:complexType>
9              <xsd:sequence>
10                 <xsd:element name="to" type="xsd:string" />
11                 <xsd:element name="from" type="xsd:token"/>
12                 <xsd:element name="header" type="xsd:string"/>
13                 <xsd:element name="body" type="xsd:normalizedString"/>
14             </xsd:sequence>
15         </xsd:complexType>
16     </xsd:element>
17 </xsd:schema>
  
```

Structuring–XSD Data Types–string–Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <mail xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3      xmlns='http://xml.netbeans.org/schema/mail'
4      xsi:schemaLocation='http://xml.netbeans.org/schema/mail mail.xsd'>
5      <to>KhanhKT</to>
6      <from>Khanh
7      KK</from>
8      <header>XML Schema</header>
9      <body>Welcome to XML Schema
10     Topic</body>
11 </mail>
  
```

```

<?xml version="1.0" encoding="UTF-8"?>
- <mail xsi:schemaLocation="http://xml.netbeans.org/scher
  instance">
    <to>KhanhKT</to>
    <from>KhanhKK</from>
    <header>XML Schema</header>
    <body>Welcome to XML Schema Topic</body>
  </mail>
  
```


Structuring–XSD Data Types–string–Example

mail.xsd


Source Schema Design

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/mail"
5      xmlns="http://xml.netbeans.org/schema/mail"
6      elementFormDefault="qualified">
7      <xsd:element name="mail">
8          <xsd:complexType>
9              <xsd:sequence>
10                 <xsd:element name="to" type="xsd:language" />
11                 <xsd:element name="from" type="xsd:token"/>
12                 <xsd:element name="header" type="xsd:string"/>
13                 <xsd:element name="body" type="xsd:normalizedString"/>
14             </xsd:sequence>
15         </xsd:complexType>
16     </xsd:element>
17 </xsd:schema>
  
```

Structuring–XSD Data Types–string–Example

mail.xml



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <mail  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3      xmlns='http://xml.netbeans.org/schema/mail'
4      xsi:schemaLocation='http://xml.netbeans.org/schema/mail mail.xsd'>
5      <to>%%</to>
6      <from>KhanhKK</from>
7      <header>XML Schema</header>
8      <body>Welcometo XML Schema Topic</body>
9  </mail>

```

Output - XML check

XML validation started.

Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/mail.xml...

Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/mail.xsd".

cvc-pattern-valid: Value '%%' is not facet-valid with respect to pattern '([a-zA-Z]{1,8})(-[a-zA-Z0-9]{1,8})*' for type 'language'. [5]

cvc-type.3.1.3: The value '%%' of element 'to' is not valid. [5]

XML validation finished.

Structuring–XSD Data Types–Miscellaneous

- **Boolean** data types
 - Is used to **specify** a **true** or **false** value
 - **True** can be replaced by the numeric **value of 1** and **false** can be replaced by the **value 0**
 - Is often **used** in **attribute**
 - **Syntax: `xsd:boolean`**
- **Binary** data types
 - Are used to **express binary-formatted data** including **graphic file, executable program** or any other strings
 - There are two binary data types
 - **base64Binary** (Base64-encoded binary data)
 - **hexBinary** (hexadecimal-encoded binary data)
 - **Syntax: `xsd:hexBinary` or `xsd:base64Binary`**

Schemas

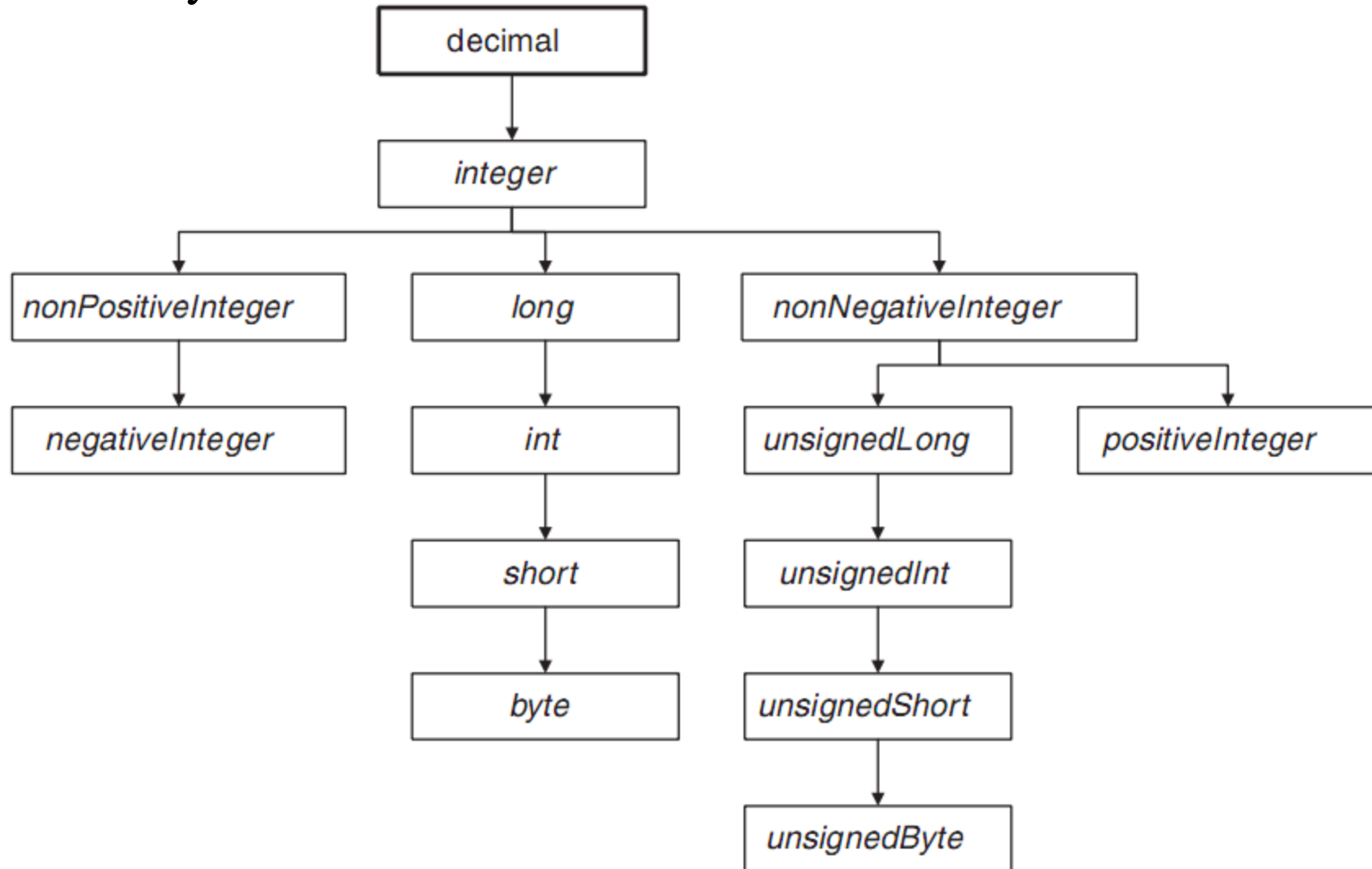
Structuring–XSD Data Types–Miscellaneous

- **anyURI** data types
 - Is used to **specify** a **URI** that represents the file name or location of the file
 - If a URI has **spaces**, **replace** them with **%20**
 - Is often **used in attribute**
 - **Syntax: `xsd:anyURI`**
- **NOTATION** data types
 - Represents a **NOTATION attribute** type. A set of QName
 - **Syntax: `xsd:NOTATION`**

Schemas

Structuring–XSD Data Types–Numeric

- Represents a **numeric** value such as whole numbers & real numbers
- Type hierarchy of numeric as



Schemas

Structuring – XSD Data Types – Numeric

Type	Description
decimal	Is used to specify an arbitrary-precision decimal number that presents exact fraction part. Syntax: <code>xsd:decimal</code>
integer	Is used to specify a numeric value without a fractional component that includes the positive and negative numbers. Syntax: <code>xsd:integer</code>
nonPositiveInteger	An integer containing only non-positive values (...,-2,-1,0). Syntax: <code>xsd:nonPositiveInteger</code>
negativeInteger	An integer containing only negative values (...,-2,-1). Syntax: <code>xsd:negativeInteger</code>
nonNegativeInteger	An integer containing only non-negative values (0,1,2,...). Syntax: <code>xsd:nonNegativeInteger</code>
unsignedLong	An unsigned 64-bit integer. Syntax: <code>xsd:unsignedLong</code>
positiveInteger	An integer containing only positive values (1,2,...). Syntax: <code>xsd:positiveInteger</code>

Schemas

Structuring – XSD Data Types – Numeric

Type	Description
unsignedInt	An unsigned 32-bit integer. Syntax: xsd:unsignedInt
unsignedShort	An unsigned 16-bit integer. Syntax: xsd:unsignedShort
unsignedByte	An unsigned 8-bit integer. Syntax: xsd:unsignedByte
long	A signed 64-bit integer. Syntax: xsd:long
int	A signed 32-bit integer. Syntax: xsd:int
short	A signed 16-bit integer. Syntax: xsd:short
byte	A signed 8-bit integer. Syntax: xsd:byte

Schemas

Structuring – XSD Data Types – Date & Time

Type	Description
dateTime	<p>Represents a specific instance of time.</p> <p>The pattern for dateTime is CCYY-MM-DDThh:mm:ss.sss with T as a separator</p> <p>Syntax: <code>xsd:dateTime</code></p>
date	<p>Represents a calendar date.</p> <p>The pattern for date is CCYY-MM-DD.</p> <p>Syntax: <code>xsd:date</code></p>
time	<p>Represents an instance of time that recurs every day.</p> <p>The pattern for time is hh:mm:ss.sss.</p> <p>Syntax: <code>xsd:time</code></p>
duration	<p>Represents a duration of time. The pattern for duration is PnYnMnDTnHnMnS. Syntax: <code>xsd:duration</code></p>

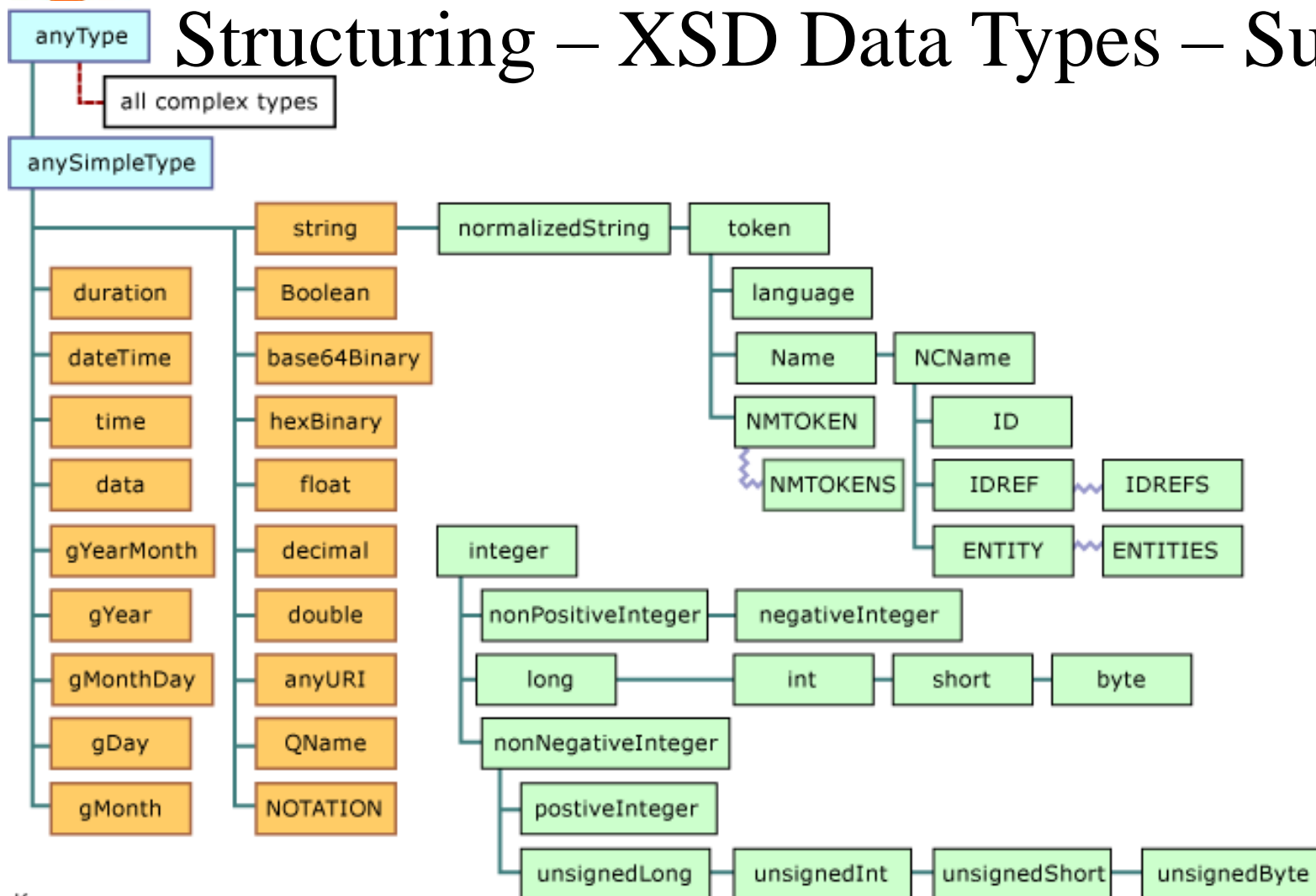
Schemas

Structuring – XSD Data Types – Date & Time

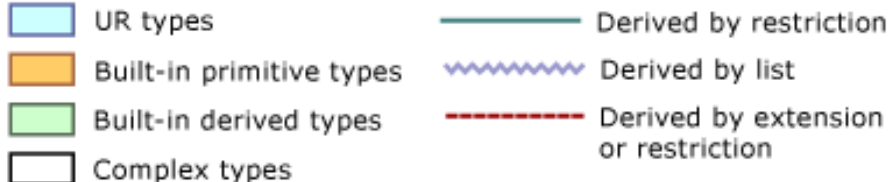
Type	Description
gYearMonth	Represents a specific Gregorian month in a specific Gregorian year. The pattern for gYearMonth is CCYY-MM . Syntax: xsd:gYearMonth
gYear	Represents a specific Gregorian year. The pattern for gYear is CCYY . Syntax: xsd:gYear
gMonthDay	Represents a specific Gregorian date. The pattern for gMonthDay is --MM-DD . Syntax: xsd:gMonthDay
gDay	Represents a Gregorian day. The pattern for gDay is ---DD . Syntax: xsd:gDay
gMonth	Represents a Gregorian month. The pattern for gMonth is --MM-- . Syntax: xsd:gMonth

Schemas

Structuring – XSD Data Types – Summary



Key



Structuring – User Derived Simple Type

- Supports to create custom user-defined data types

- **Syntax:**

<xsd:simpleType name="newName">

restriction type, or list type, or union type

</xsd:simpleType>

- Simple types are **derived** by
 - **Restricting built-in** simple types, or, other **user-derived** simple types.
 - **Ex:**

<xsd:simpleType name="Password">

<xsd:restriction base="xsd:string">

<xsd:length value="8"/>

</xsd:restriction>

</xsd:simpleType>

- Simple types can be derived by applying one or more of the facets

Schemas

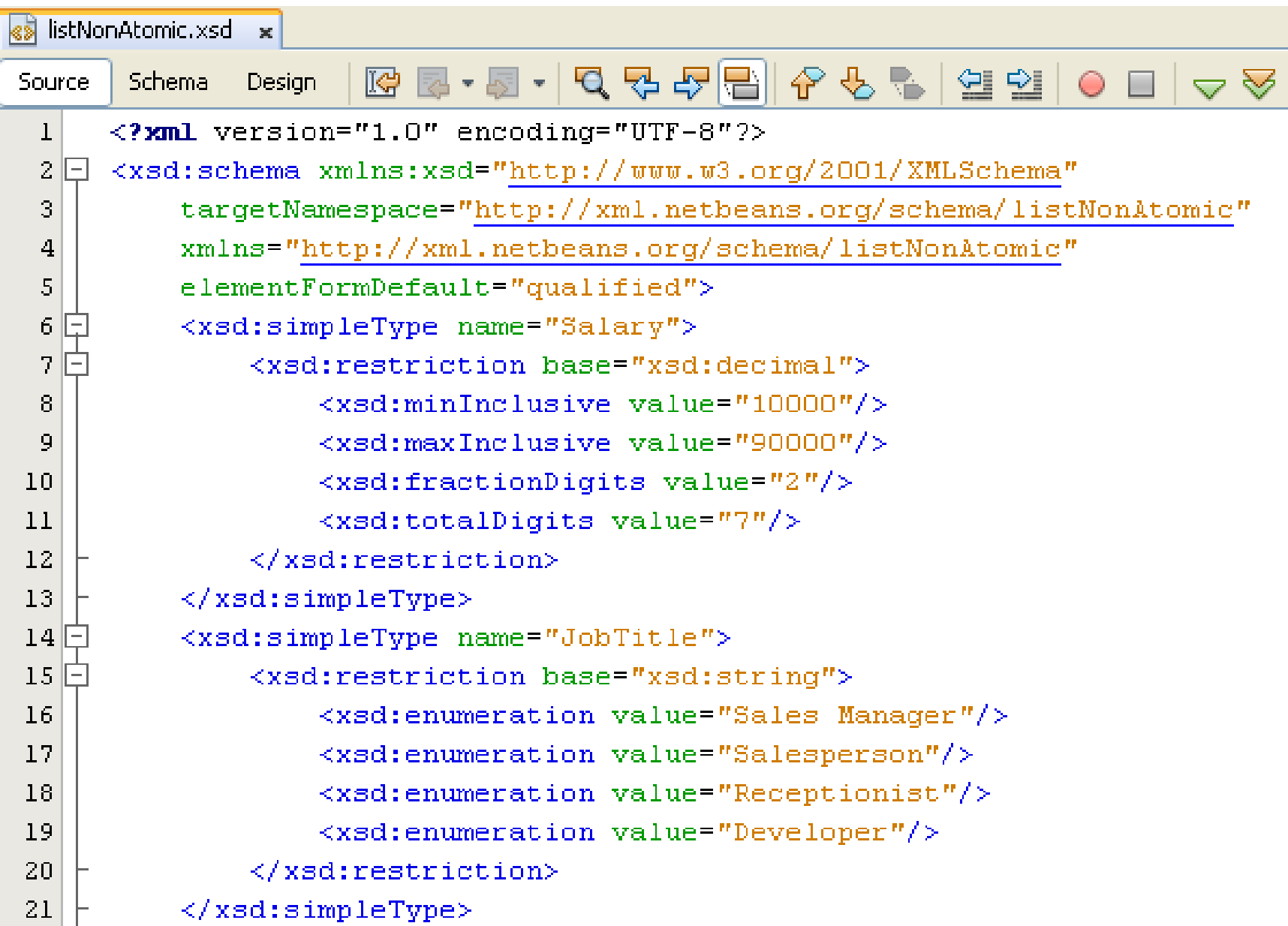
Structuring – XSD Data Types – Non-Atomic

- All of XML Schema's **built-in types are atomic**
 - Meaning that they **cannot** be **broken down** into meaningful bits
- XML Schema provides for **two non-atomic** types
 - **lists**
 - Are **sequences of atomic types separated by whitespace**
 - Represent a **single value** within an element
 - **Syntax**

```
<xsd:simpleType name="type_name">  
  <xsd:list itemType="data_type"/>  
</xsd:simpleType>
```
 - **unions**
 - Are **groupings of types**, essentially allowing for the value of an element to be of **more than one type**
 - **Syntax**

```
<xsd:simpleType name="type_name">  
  <xsd:union memberTypes="list of type separating with spaces"/>  
</xsd:simpleType>
```

Structuring – Non-Atomic – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3      targetNamespace="http://xml.netbeans.org/schema/listNonAtomic"
4      xmlns="http://xml.netbeans.org/schema/listNonAtomic"
5      elementFormDefault="qualified">
6      <xsd:simpleType name="Salary">
7          <xsd:restriction base="xsd:decimal">
8              <xsd:minInclusive value="10000"/>
9              <xsd:maxInclusive value="90000"/>
10             <xsd:fractionDigits value="2"/>
11             <xsd:totalDigits value="7"/>
12         </xsd:restriction>
13     </xsd:simpleType>
14     <xsd:simpleType name="JobTitle">
15         <xsd:restriction base="xsd:string">
16             <xsd:enumeration value="Sales Manager"/>
17             <xsd:enumeration value="Salesperson"/>
18             <xsd:enumeration value="Receptionist"/>
19             <xsd:enumeration value="Developer"/>
20         </xsd:restriction>
21     </xsd:simpleType>
  
```

Schemas

Structuring – Non-Atomic – Example

```

22 <xsd:simpleType name="DateList">
23     <xsd:list itemType="xsd:date"/>
24 </xsd:simpleType>
25 <xsd:element name="Employee">
26     <xsd:complexType>
27         <xsd:sequence>
28             <xsd:element name="Salary" type="Salary"/>
29             <xsd:element name="Title" type="JobTitle"/>
30             <xsd:element name="VacationDays" type="DateList"/>
31         </xsd:sequence>
32     </xsd:complexType>
  
```

listNonAtomic.xml

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <Employee xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4     xmlns='http://xml.netbeans.org/schema/listNonAtomic'
5     xsi:schemaLocation='http://xml.netbeans.org/schema/listNonAtomic listNonAtomic.xsd'>
6     <Salary>44000</Salary>
7     <Title>Salesperson</Title>
8     <VacationDays>2006-08-13 2006-08-14 2006-08-15</VacationDays>
9 </Employee>
  
```

Schemas

Structuring – Non-Atomic – Example

```

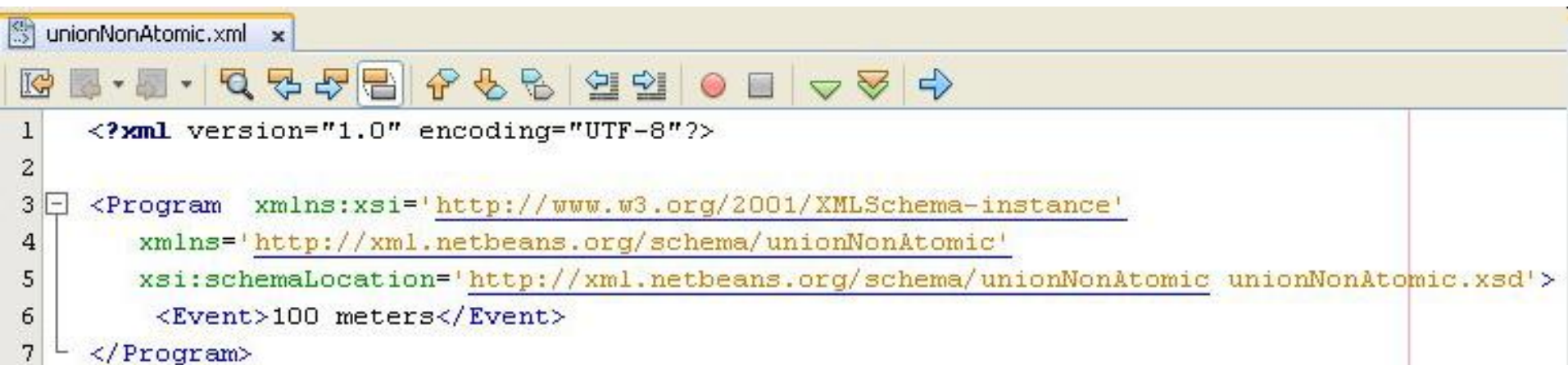
Source Schema Design
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/unionNonAtomic"
5   xmlns="http://xml.netbeans.org/schema/unionNonAtomic"
6   elementFormDefault="qualified">
7   <xsd:simpleType name="RunningRace">
8     <xsd:restriction base="xsd:string">
9       <xsd:enumeration value="100 meters"/>
10      <xsd:enumeration value="10 kilometers"/>
11      <xsd:enumeration value="440 yards"/>
12      <xsd:enumeration value="10 miles"/>
13      <xsd:enumeration value="Marathon"/>
14    </xsd:restriction>
15  </xsd:simpleType>
16  <xsd:simpleType name="Gymnastics">
17    <xsd:restriction base="xsd:string">
18      <xsd:enumeration value="Vault"/>
19      <xsd:enumeration value="Floor"/>
20      <xsd:enumeration value="Rings"/>
21      <xsd:enumeration value="Beam"/>
22      <xsd:enumeration value="Uneven Bars"/>
23    </xsd:restriction>
24  </xsd:simpleType>
  
```

Schemas

Structuring – Non-Atomic – Example

```

25 <xsd:simpleType name="Event">
26     <xsd:union memberTypes="RunningRace Gymnastics"/>
27 </xsd:simpleType>
28 <xsd:element name="Program">
29     <xsd:complexType>
30         <xsd:sequence>
31             <xsd:element name="Event" type="Event"/>
32         </xsd:sequence>
33     </xsd:complexType>
34 </xsd:element>
35 </xsd:schema>
  
```

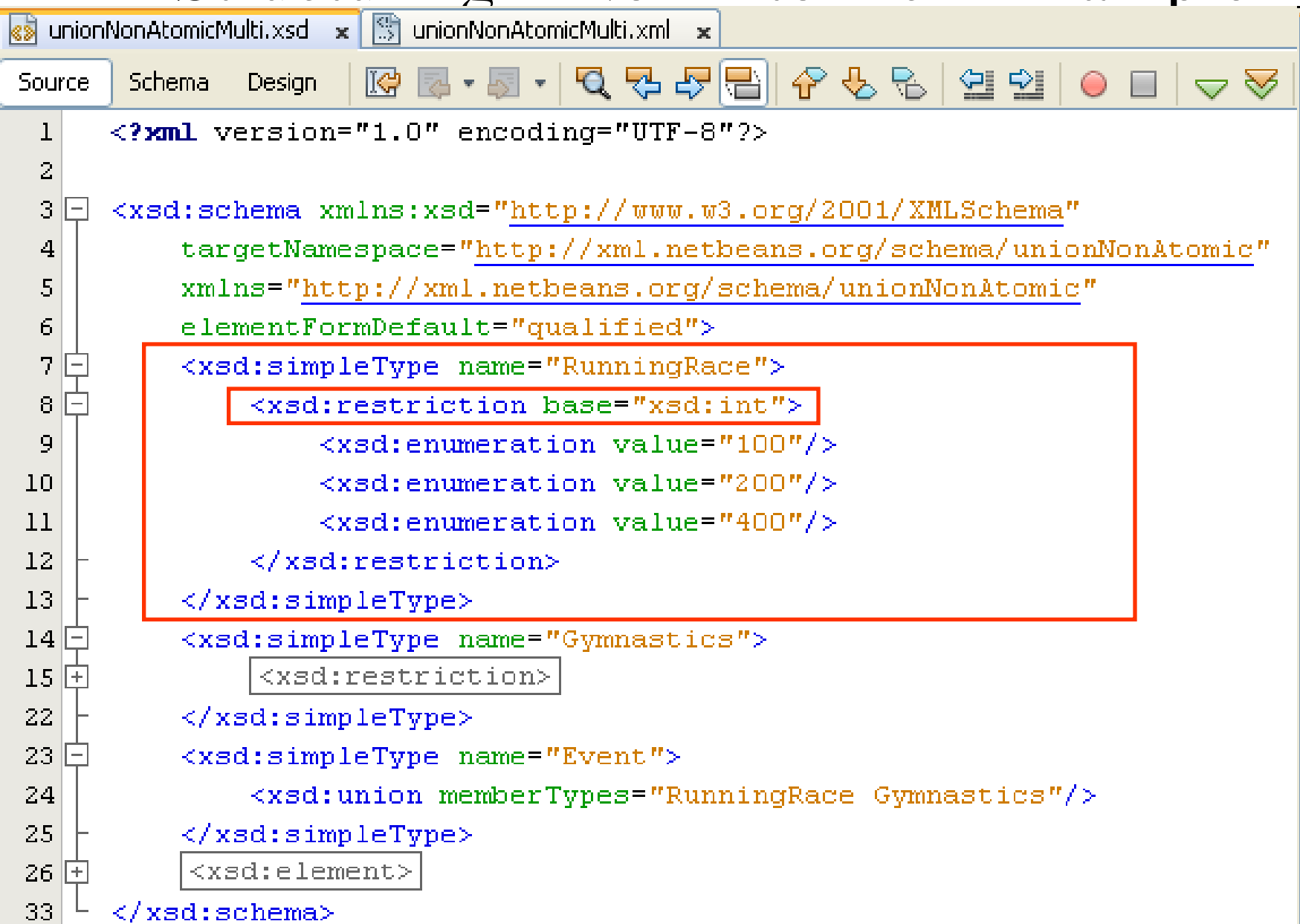

 A screenshot of an XML editor window titled 'unionNonAtomic.xml'. The editor shows an XML document with a root element 'Program' containing an 'Event' element. The 'Event' element has the value '100 meters'. The XML document is well-formed and includes namespace declarations for xsi and the schema location.


```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <Program xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4     xmlns='http://xml.netbeans.org/schema/unionNonAtomic'
5     xsi:schemaLocation='http://xml.netbeans.org/schema/unionNonAtomic unionNonAtomic.xsd'>
6     <Event>100 meters</Event>
7 </Program>
  
```


Schemas

Structuring – Non-Atomic – Example



The screenshot shows an XML Schema editor with two tabs: 'unionNonAtomicMulti.xsd' and 'unionNonAtomicMulti.xml'. The 'Source' tab is active, displaying the following XML Schema code:

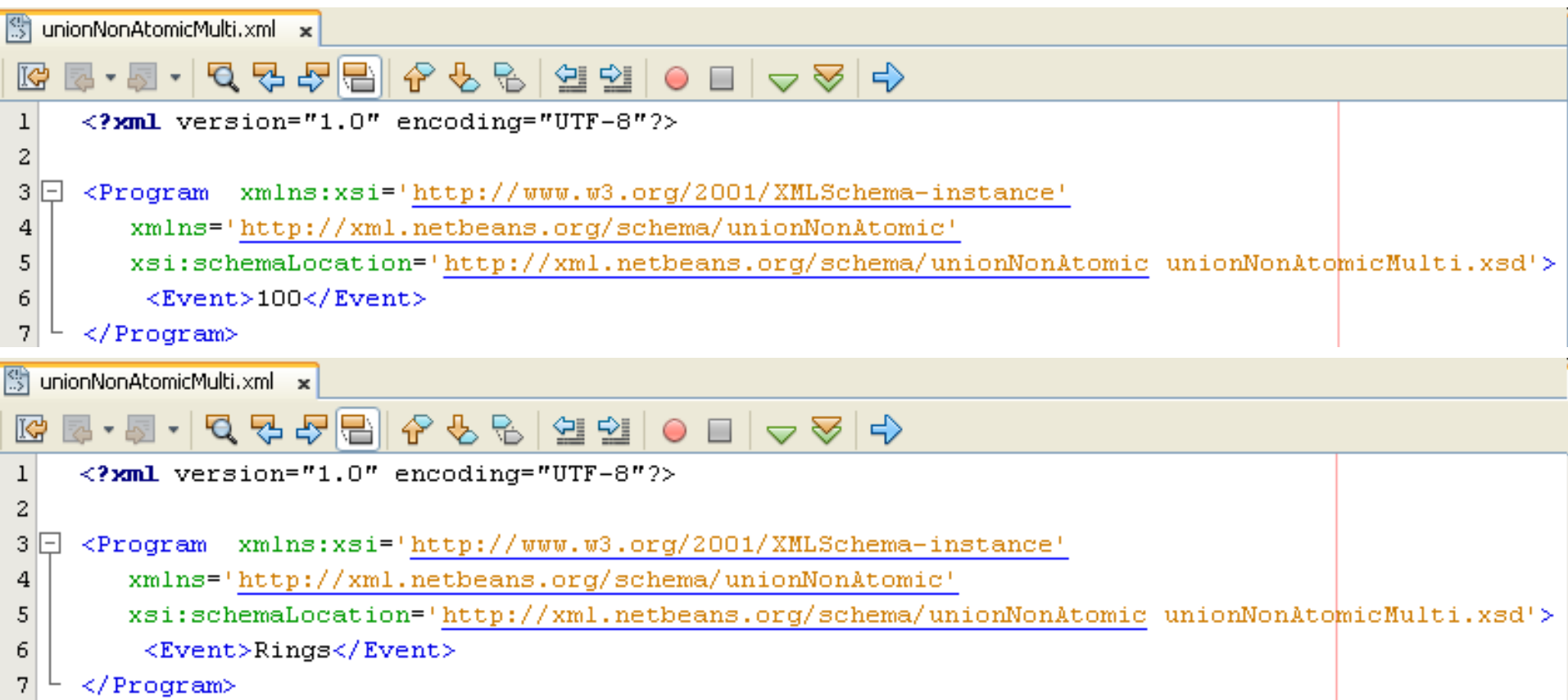
```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/unionNonAtomic"
5      xmlns="http://xml.netbeans.org/schema/unionNonAtomic"
6      elementFormDefault="qualified">
7      <xsd:simpleType name="RunningRace">
8          <xsd:restriction base="xsd:int">
9              <xsd:enumeration value="100"/>
10             <xsd:enumeration value="200"/>
11             <xsd:enumeration value="400"/>
12         </xsd:restriction>
13     </xsd:simpleType>
14     <xsd:simpleType name="Gymnastics">
15         <xsd:restriction>
22     </xsd:simpleType>
23     <xsd:simpleType name="Event">
24         <xsd:union memberTypes="RunningRace Gymnastics"/>
25     </xsd:simpleType>
26     <xsd:element>
33 </xsd:schema>
  
```

A red rectangular box highlights the definition of the 'RunningRace' simple type, specifically the 'xsd:restriction' and its 'xsd:enumeration' values. The 'Gymnastics' simple type and the 'Event' simple type (which uses a union of 'RunningRace' and 'Gymnastics') are also visible in the code.

Schemas

Structuring – Non-Atomic – Example



The image shows two screenshots of an XML editor. The top screenshot shows an XML file named 'unionNonAtomicMulti.xml' with the following content:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Program xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/unionNonAtomic'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/unionNonAtomic unionNonAtomicMulti.xsd'>
6      <Event>100</Event>
7  </Program>

```

The bottom screenshot shows the same XML file with the following content:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Program xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/unionNonAtomic'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/unionNonAtomic unionNonAtomicMulti.xsd'>
6      <Event>Rings</Event>
7  </Program>

```

Output - XML check

```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/unionNonAtomicMulti.xml...
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/unionNonAtomicMulti.xsd".
XML validation finished.

```

Schemas

Structuring – XSD Attributes

- The **attributes** themselves **must** be of **simple type**
 - Simple elements cannot have attributes.
 - If an **element** has **attributes**, it is considered to be of a **complex type**
- **Syntax**
 - `<xsd:attribute name="attr_name" type="data_types"/>`
- **Default** and **Fixed** values are **same as elements**
 - **Default** syntax
 - `<xsd:attribute name="attr_name" type="data_types" default="value"/>`
 - **Fixed** syntax
 - `<xsd:attribute name="attr_name" type="data_types" fixed="value"/>`
- **Optional** and **Required** Attributes
 - Attributes are **optional by default**. To specify that the attribute is required, use the "**use**" attribute to **set required** value
 - **Syntax** (default and use are not used in the same time)
 - `<xsd:attribute name="attr_name" type="data_types" use="required" [fixed="value"]/>`

Structuring – XSD Attributes – Example

attrDemo.xsd

Source Schema Design

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/attrDemo"
5      xmlns="http://xml.netbeans.org/schema/attrDemo"
6      elementFormDefault="qualified">
7      <xsd:element name="Name">
8          <xsd:complexType>
9              <xsd:sequence>
10                 <xsd:element name="FirstName" type="xsd:string"/>
11                 <xsd:element name="LastName" type="xsd:string"/>
12             </xsd:sequence>
13             <xsd:attribute name="Pseudonym" type="xsd:boolean"/>
14             <xsd:attribute name="HomePage" type="xsd:anyURI"/>
15          </xsd:complexType>
16      </xsd:element>
17  </xsd:schema>
  
```

Schemas

Structuring – XSD Attributes – Example

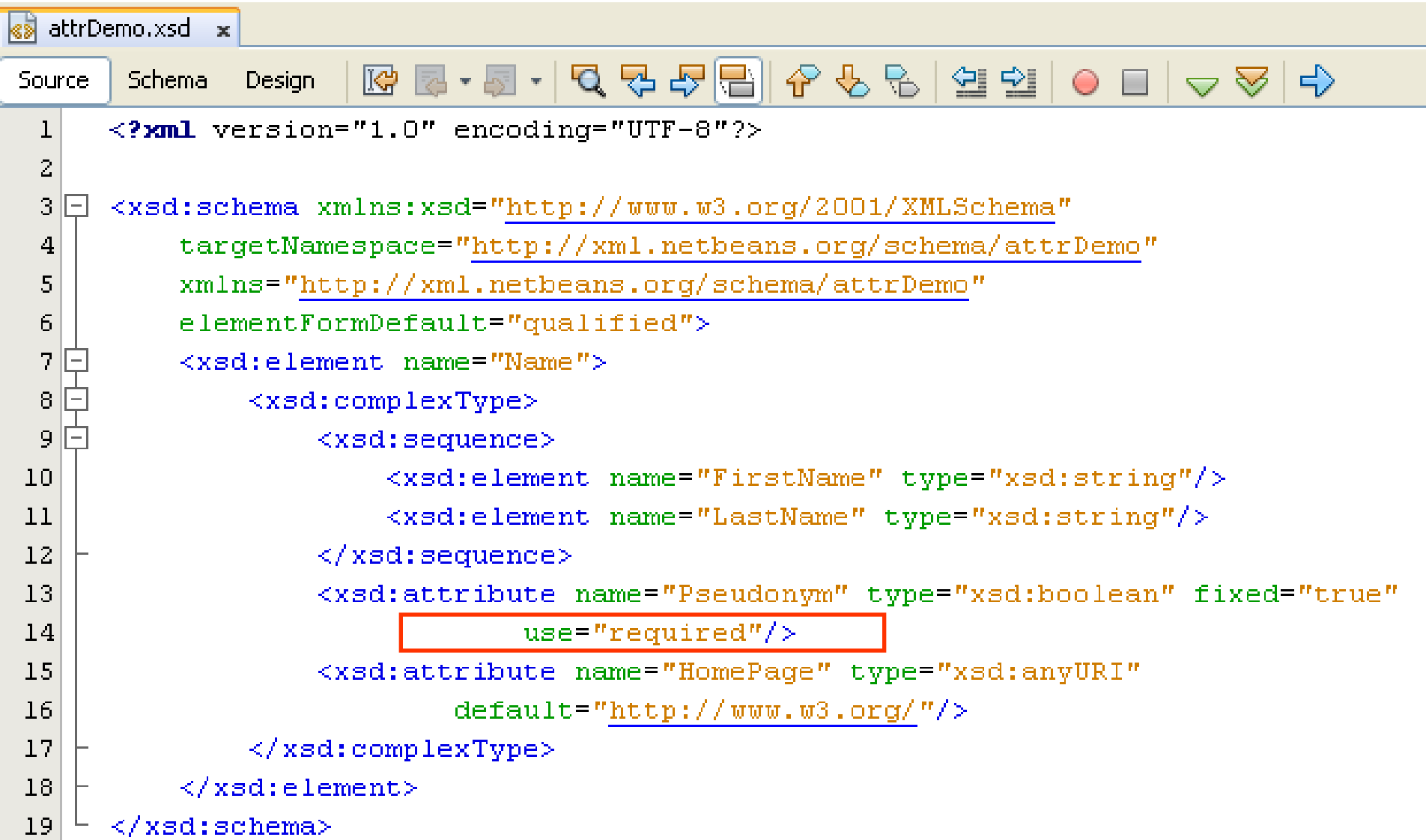
```

attrDemo.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Name  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/attrDemo'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/attrDemo attrDemo.xsd'
6      Pseudonym="true"
7      HomePage="http://www.marktwain.com">
8      <FirstName>Mark</FirstName>
9      <LastName>Twain</LastName>
10 </Name>
  
```

```

attrDemo.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Name  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/attrDemo'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/attrDemo attrDemo.xsd'
6
7      HomePage="http://www.marktwain.com">
8      <FirstName>Mark</FirstName>
9      <LastName>Twain</LastName>
10 </Name>
  
```

Structuring – XSD Attributes – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    targetNamespace="http://xml.netbeans.org/schema/attrDemo"
5    xmlns="http://xml.netbeans.org/schema/attrDemo"
6    elementFormDefault="qualified">
7    <xsd:element name="Name">
8      <xsd:complexType>
9        <xsd:sequence>
10          <xsd:element name="FirstName" type="xsd:string"/>
11          <xsd:element name="LastName" type="xsd:string"/>
12        </xsd:sequence>
13        <xsd:attribute name="Pseudonym" type="xsd:boolean" fixed="true"
14          use="required"/>
15        <xsd:attribute name="HomePage" type="xsd:anyURI"
16          default="http://www.w3.org/" />
17      </xsd:complexType>
18    </xsd:element>
19  </xsd:schema>
  
```

Schemas

Structuring – XSD Attributes – Example

attrDemo.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Name xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/attrDemo'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/attrDemo attrDemo.xsd'
6      Pseudonym="true">
7      <FirstName>Mark</FirstName>
8      <LastName>Twain</LastName>
9  </Name>

```

attrDemo.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Name xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/attrDemo'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/attrDemo attrDemo.xsd'
6
7      HomePage="http://xml.netbeans.org/schema/attrDemo"
8      <FirstName>Mark</FirstName>
9      <LastName>Twain</LastName>
10 </Name>

```

Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/attrDemo.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/attrDemo.xsd".
cvc-complex-type.4: Attribute 'Pseudonym' must appear on element 'Name'. [7]
XML validation finished.

```

Schemas

Structuring – XSD Facets

- Facets are called **restrictions**
- Are used to **define acceptable values** for XML elements or attributes
 - Are used to **restrict** the **set of values** a data type contain
 - Are **ensured** that the **new value range** must be **equal** to or **narrower than range** of the **base type**
- Syntax

```
<xsd:restriction base="base_type">  
  <xsd:facet_element value="value"/>  
  ...  
</xsd:restriction>
```
- There are **12 facet** elements

Schemas

Structuring – XSD Facets

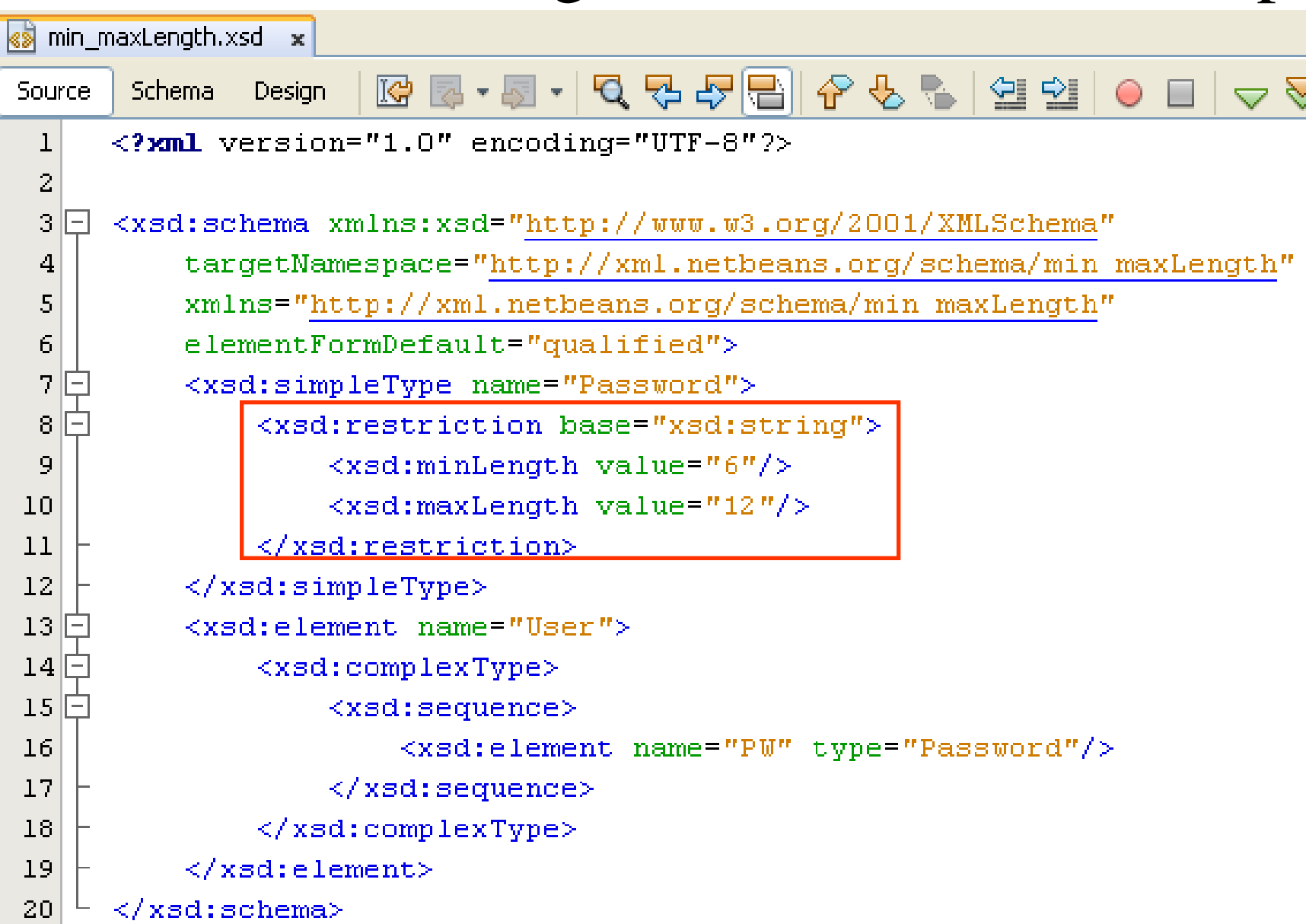
Constraint	Description
enumeration	Defines a list of acceptable values. Syntax: <code>xsd:enumeration</code>
fractionDigits	Specifies the maximum number of decimal places allowed. Must be equal to or greater than zero. Syntax: <code>xsd:fractionDigits</code>
totalDigits	Specifies the exact number of digits allowed. Must be greater than zero. Syntax: <code>xsd:totalDigits</code>
length	Specifies the exact number of characters or list items allowed. Must be equal to or greater than zero. Syntax: <code>xsd:length</code>
maxExclusive	Specifies the upper bounds for numeric values (the value must be less than this value). Syntax: <code>xsd:maxExclusive</code>
minExclusive	Specifies the lower bounds for numeric values (the value must be greater than this value). Syntax: <code>xsd:minExclusive</code>
maxInclusive	Specifies the upper bounds for numeric values (the value must be less than or equal to this value). Syntax: <code>xsd:maxInclusive</code>
minInclusive	Specifies the lower bounds for numeric values (the value must be greater than or equal to this value). Syntax: <code>xsd:minInclusive</code>

Schemas

Structuring – XSD Facets

Constraint	Description
minLength	<p>Specifies the minimum number of characters or list items allowed. Must be equal to or greater than zero.</p> <p>Syntax: <code>xsd:minLength</code></p>
maxLength	<p>Specifies the maximum number of characters or list items allowed. Must be equal to or greater than zero.</p> <p>Syntax: <code>xsd:maxLength</code></p>
pattern	<p>Specific pattern that the data type's values must match. This constrains the data type to literals that match the specified pattern. The pattern value must be a regular expression.</p> <p>Syntax: <code>xsd:pattern</code></p>
whitespace	<p>Value must be one of preserve (No normalization is performed), replace (replace tab, line feed, carriage return with space), or collapse (After the processing implied by replace, multiple space is replace with space and the leading or trailing space is removed). The whiteSpace facet cannot be changed for most numeric data types. Syntax: <code>xsd:whitespace</code></p>

Structuring – XSD Facets – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/min_maxLength"
5      xmlns="http://xml.netbeans.org/schema/min_maxLength"
6      elementFormDefault="qualified">
7      <xsd:simpleType name="Password">
8          <xsd:restriction base="xsd:string">
9              <xsd:minLength value="6"/>
10             <xsd:maxLength value="12"/>
11         </xsd:restriction>
12     </xsd:simpleType>
13     <xsd:element name="User">
14         <xsd:complexType>
15             <xsd:sequence>
16                 <xsd:element name="PW" type="Password"/>
17             </xsd:sequence>
18         </xsd:complexType>
19     </xsd:element>
20 </xsd:schema>
  
```

Schemas

Structuring – XSD Facets – Example

```

min_maxLength.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <User  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/min_maxLength'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/min_maxLength min_maxLength.xsd'>
6      <PW>aaa</PW>
7  </User>
  
```

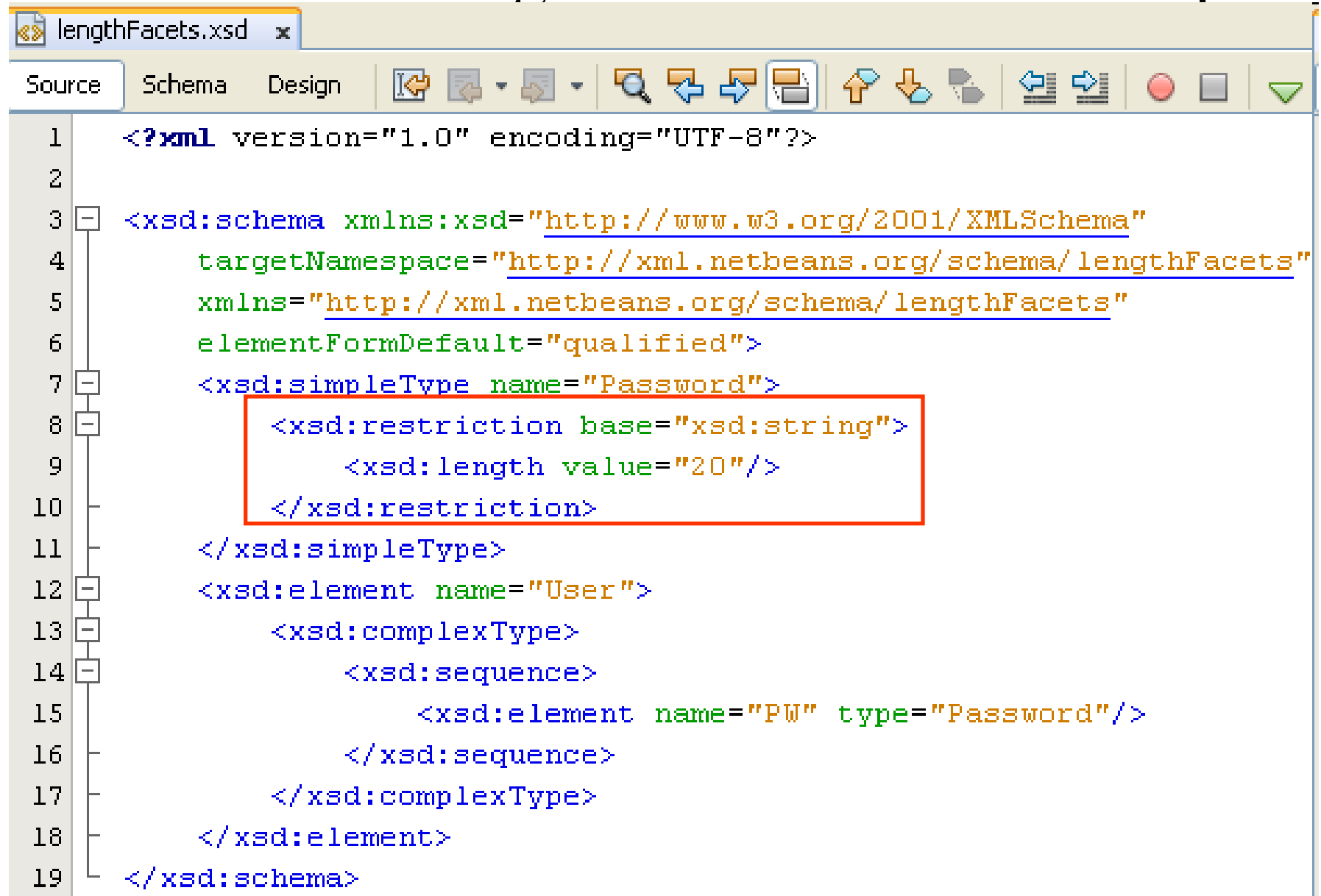
Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/min_maxLength.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/min_maxLength.xsd".
cvc-minLength-valid: Value 'aaa' with length = '3' is not facet-valid with respect to minLength '6' for type 'Password'. [6]
cvc-type.3.1.3: The value 'aaa' of element 'PW' is not valid. [6]
XML validation finished.
  
```

Schemas

Structuring – XSD Facets – Example



The screenshot shows an XML editor window titled 'lengthFacets.xsd'. The editor has tabs for 'Source', 'Schema', and 'Design'. The 'Source' tab is active, displaying the following XML code:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/lengthFacets"
5      xmlns="http://xml.netbeans.org/schema/lengthFacets"
6      elementFormDefault="qualified">
7      <xsd:simpleType name="Password">
8          <xsd:restriction base="xsd:string">
9              <xsd:length value="20"/>
10         </xsd:restriction>
11     </xsd:simpleType>
12     <xsd:element name="User">
13         <xsd:complexType>
14             <xsd:sequence>
15                 <xsd:element name="PW" type="Password"/>
16             </xsd:sequence>
17         </xsd:complexType>
18     </xsd:element>
19 </xsd:schema>
  
```

A red rectangular box highlights the facet definition on lines 8 through 10:

```

<xsd:restriction base="xsd:string">
    <xsd:length value="20"/>
</xsd:restriction>
  
```

Schemas

Structuring – XSD Facets – Example

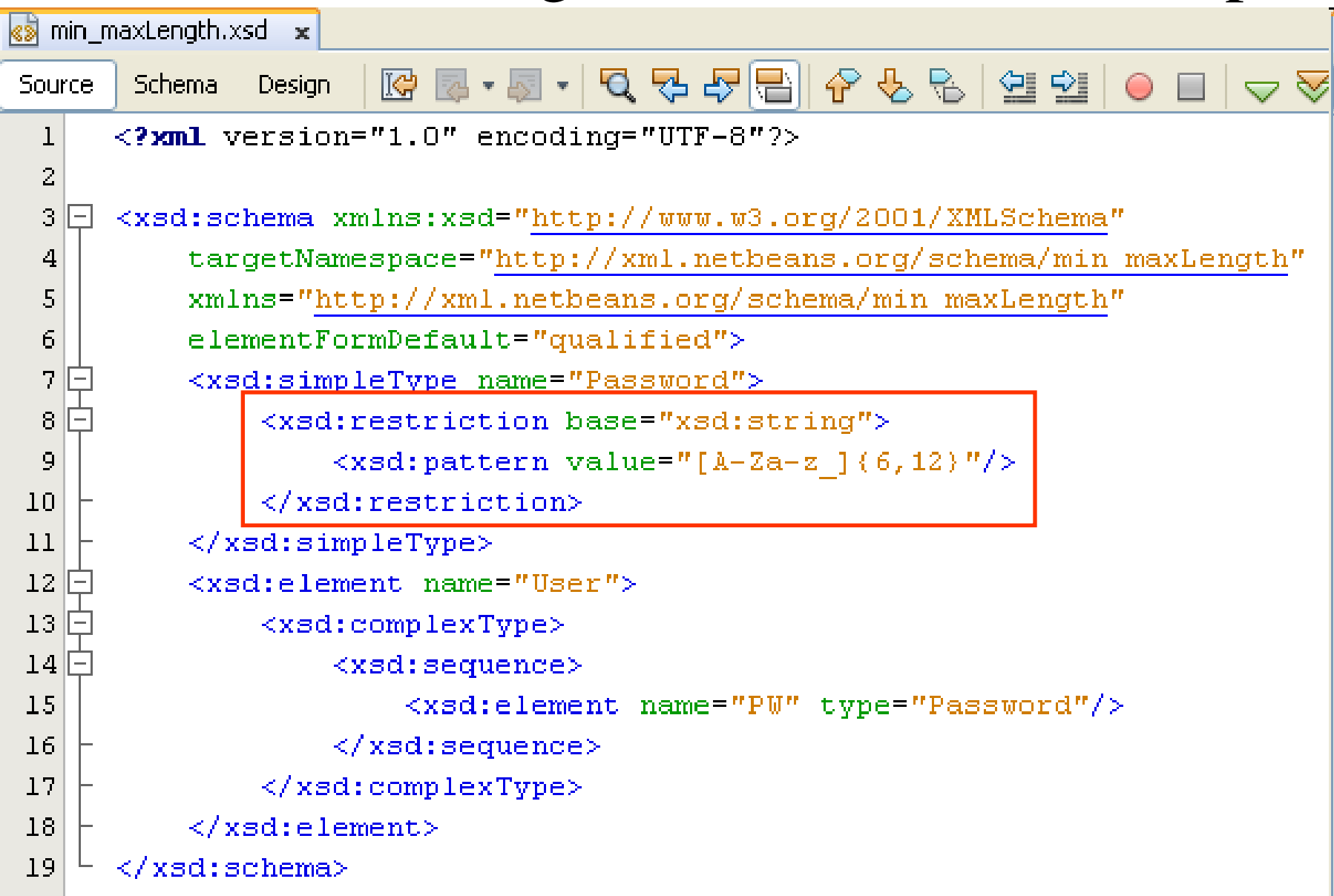
```
lengthFacets.xml x
<?xml version="1.0" encoding="UTF-8"?>
<User xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
      xmlns='http://xml.netbeans.org/schema/lengthFacets'
      xsi:schemaLocation='http://xml.netbeans.org/schema/lengthFacets lengthFacets.xsd'>
  <PW>1222222222222</PW>
</User>
```

Output - XML check

```
XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/lengthFacets.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/lengthFacets.xsd".
cvc-length-valid: Value '1222222222222' with length = '13' is not facet-valid with respect to length '20' for type 'Password'. [6]
cvc-type.3.1.3: The value '1222222222222' of element 'PW' is not valid. [6]
XML validation finished.
```

Schemas

Structuring – XSD Facets – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/min_maxLength"
5      xmlns="http://xml.netbeans.org/schema/min_maxLength"
6      elementFormDefault="qualified">
7      <xsd:simpleType name="Password">
8          <xsd:restriction base="xsd:string">
9              <xsd:pattern value="[A-Za-z_]{6,12}" />
10         </xsd:restriction>
11     </xsd:simpleType>
12     <xsd:element name="User">
13         <xsd:complexType>
14             <xsd:sequence>
15                 <xsd:element name="PW" type="Password" />
16             </xsd:sequence>
17         </xsd:complexType>
18     </xsd:element>
19 </xsd:schema>
  
```

Schemas

Structuring – XSD Facets – Example

```

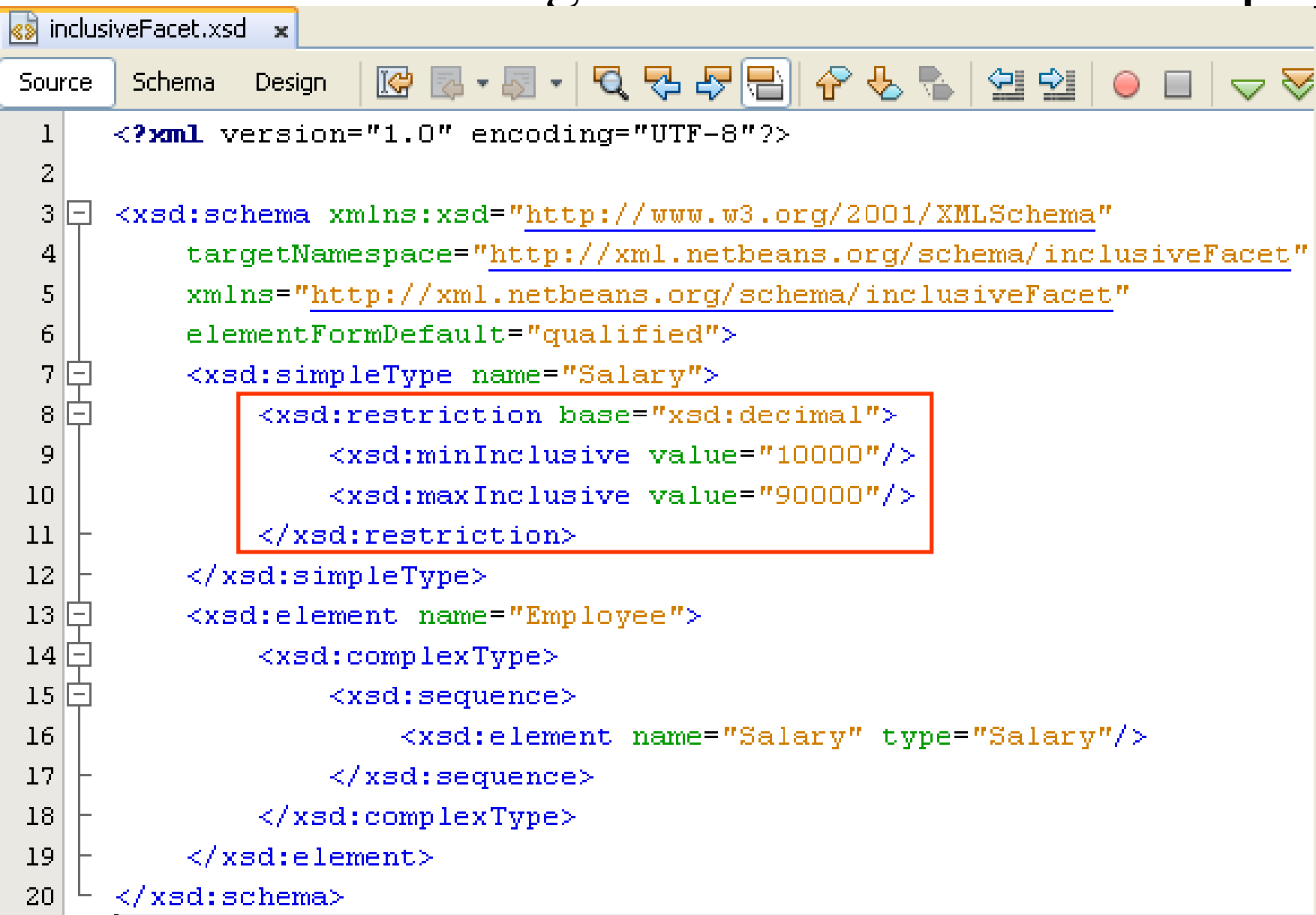
min_maxLength.xml x
[Icons]
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <User  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/min_maxLength'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/min_maxLength min_maxLength.xsd'>
6      <PW>aa@ad</PW>
7  </User>
  
```

Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/min_maxLength.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/min_maxLength.xsd".
cvc-pattern-valid: Value 'aa@ad' is not facet-valid with respect to pattern '[A-Za-z_]{6,12}' for type 'Password'. [6]
cvc-type.3.1.3: The value 'aa@ad' of element 'PW' is not valid. [6]
XML validation finished.
  
```


Structuring – XSD Facets – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/inclusiveFacet"
5      xmlns="http://xml.netbeans.org/schema/inclusiveFacet"
6      elementFormDefault="qualified">
7      <xsd:simpleType name="Salary">
8          <xsd:restriction base="xsd:decimal">
9              <xsd:minInclusive value="10000"/>
10             <xsd:maxInclusive value="90000"/>
11         </xsd:restriction>
12     </xsd:simpleType>
13     <xsd:element name="Employee">
14         <xsd:complexType>
15             <xsd:sequence>
16                 <xsd:element name="Salary" type="Salary"/>
17             </xsd:sequence>
18         </xsd:complexType>
19     </xsd:element>
20 </xsd:schema>
  
```

Schemas

Structuring – XSD Facets – Example

```

inclusiveFacet.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Employee  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/inclusiveFacet'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/inclusiveFacet inclusiveFacet.xsd'>
6      <Salary>9000</Salary>
7  </Employee>
  
```

Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/inclusiveFacet.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/inclusiveFacet.xsd".
cvc-minInclusive-valid: Value '9000' is not facet-valid with respect to minInclusive '10000.0' for type 'Salary'. [6]
cvc-type.3.1.3: The value '9000' of element 'Salary' is not valid. [6]
XML validation finished.
  
```

Schemas

Structuring – XSD Facets – Example

inclusiveFacet.xsd

Source Schema Design


```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    targetNamespace="http://xml.netbeans.org/schema/inclusiveFacet"
5    xmlns="http://xml.netbeans.org/schema/inclusiveFacet"
6    elementFormDefault="qualified">
7    <xsd:simpleType name="Salary">
8      <xsd:restriction base="xsd:decimal">
9        <xsd:minInclusive value="10000"/>
10       <xsd:maxInclusive value="90000"/>
11       <xsd:fractionDigits value="2"/>
12       <xsd:totalDigits value="7"/>
13     </xsd:restriction>
14   </xsd:simpleType>
15   <xsd:element name="Employee">
16     <xsd:complexType>
17       <xsd:sequence>
18         <xsd:element name="Salary" type="Salary"/>
19       </xsd:sequence>
20     </xsd:complexType>
21   </xsd:element>
22 </xsd:schema>
  
```

Schemas

Structuring – XSD Facets – Example

inclusiveFacet.xml



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Employee  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4          xmlns='http://xml.netbeans.org/schema/inclusiveFacet'
5          xsi:schemaLocation='http://xml.netbeans.org/schema/inclusiveFacet inclusiveFacet.xsd'>
6      <Salary>13000.234</Salary>
7  </Employee>

```

Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/inclusiveFacet.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/inclusiveFacet.xsd".
cvc-fractionDigits-valid: Value '13000.234' has 3 fraction digits, but the number of fraction digits has been limited to 2. [6]
cvc-type.3.1.3: The value '13000.234' of element 'Salary' is not valid. [6]
XML validation finished.

```

Schemas

Structuring – XSD Facets – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/enumerationFacet"
5      xmlns="http://xml.netbeans.org/schema/enumerationFacet"
6      elementFormDefault="qualified">
7      <xsd:simpleType name="JobTitle">
8          <xsd:restriction base="xsd:string">
9              <xsd:enumeration value="Sales Manager"/>
10             <xsd:enumeration value="Salesperson"/>
11             <xsd:enumeration value="Receptionist"/>
12             <xsd:enumeration value="Developer"/>
13         </xsd:restriction>
14     </xsd:simpleType>
15     <xsd:element name="Employee">
16         <xsd:complexType>
17             <xsd:sequence>
18                 <xsd:element name="Title" type="JobTitle"/>
19             </xsd:sequence>
20         </xsd:complexType>
21     </xsd:element>
22 </xsd:schema>
  
```

Schemas

Structuring – XSD Facets – Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Employee xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/enumerationFacet'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/enumerationFacet enumerationFacet.xsd'>
6      <Title>Tester</Title>
7  </Employee>
  
```

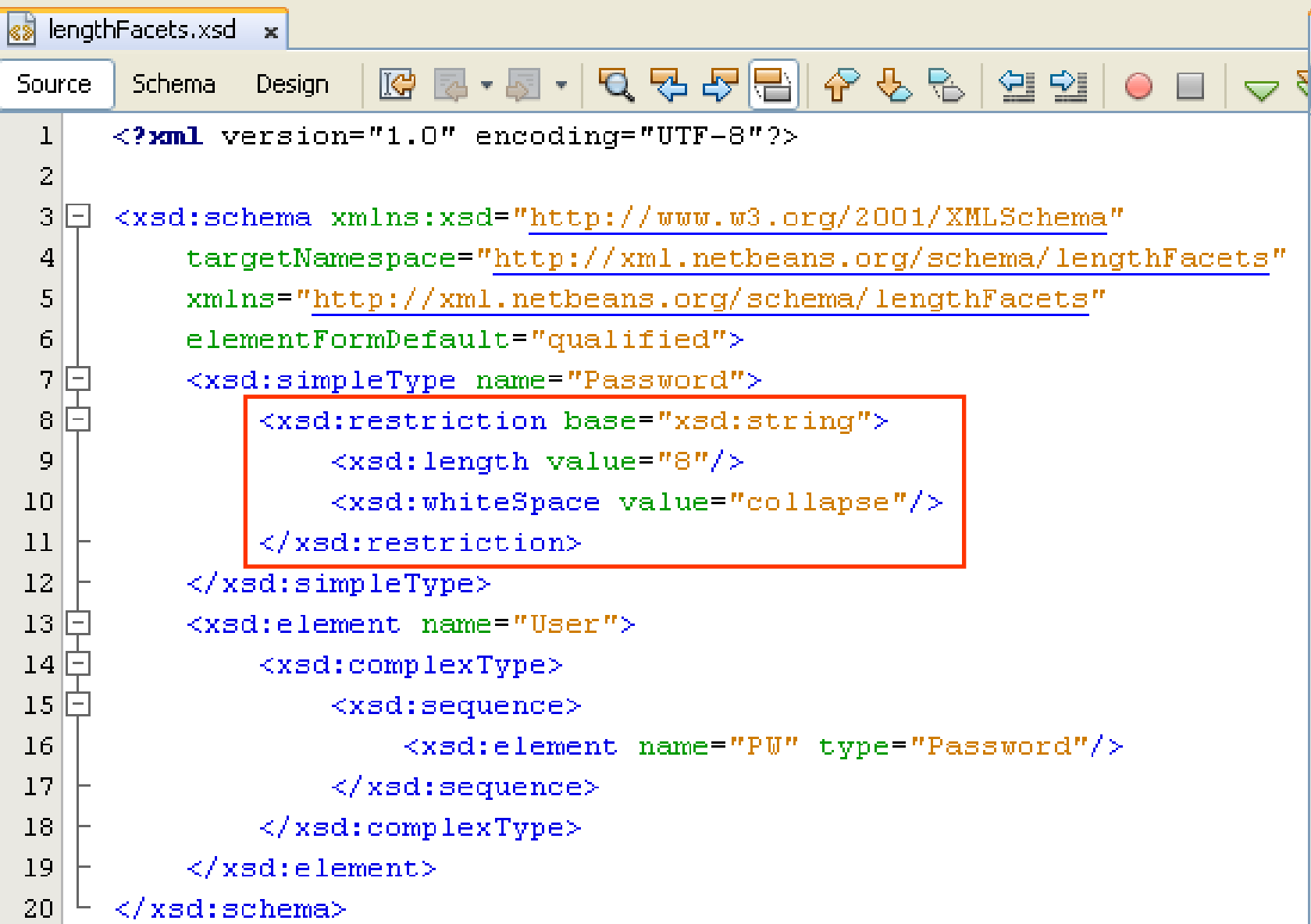
Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/enumerationFacet.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/enumerationFacet.xsd".
cvc-enumeration-valid: Value 'Tester' is not facet-valid with respect to enumeration '[Sales Manager, Salesperson, Receptionist, Developer]'. It must be a value from the enumeration. [6]
cvc-type.3.1.3: The value 'Tester' of element 'Title' is not valid. [6]
XML validation finished.
  
```

Schemas

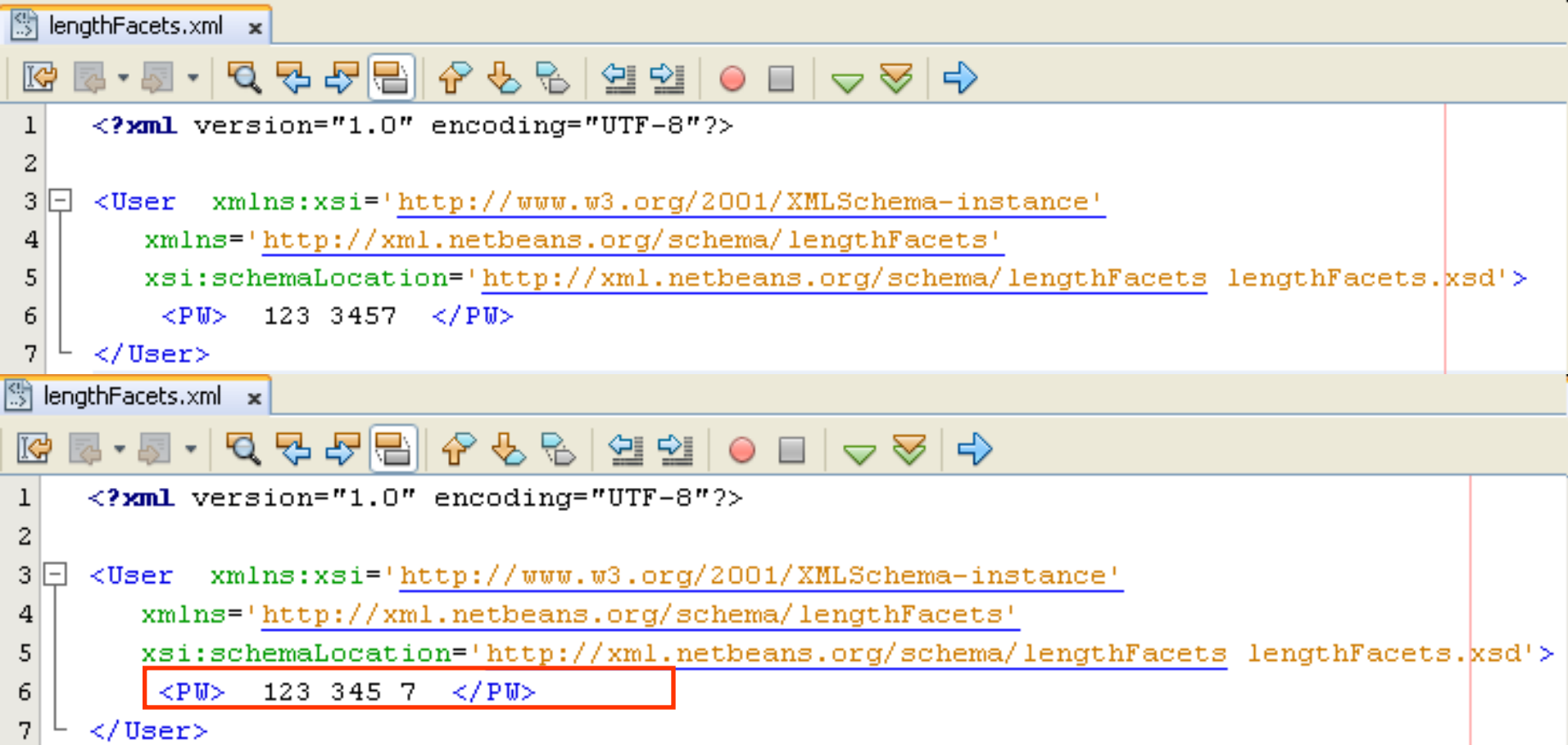
Structuring – XSD Facets – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    targetNamespace="http://xml.netbeans.org/schema/lengthFacets"
5    xmlns="http://xml.netbeans.org/schema/lengthFacets"
6    elementFormDefault="qualified">
7    <xsd:simpleType name="Password">
8      <xsd:restriction base="xsd:string">
9        <xsd:length value="8"/>
10       <xsd:whiteSpace value="collapse"/>
11     </xsd:restriction>
12   </xsd:simpleType>
13   <xsd:element name="User">
14     <xsd:complexType>
15       <xsd:sequence>
16         <xsd:element name="PW" type="Password"/>
17       </xsd:sequence>
18     </xsd:complexType>
19   </xsd:element>
20 </xsd:schema>
  
```

Structuring – XSD Facets – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <User  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns='http://xml.netbeans.org/schema/lengthFacets'
5        xsi:schemaLocation='http://xml.netbeans.org/schema/lengthFacets lengthFacets.xsd'>
6    <PW> 123 3457 </PW>
7  </User>
  
```

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <User  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns='http://xml.netbeans.org/schema/lengthFacets'
5        xsi:schemaLocation='http://xml.netbeans.org/schema/lengthFacets lengthFacets.xsd'>
6    <PW> 123 345 7 </PW>
7  </User>
  
```

Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/lengthFacets.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/lengthFacets.xsd".
cvc-length-valid: Value '123 345 7' with length = '9' is not facet-valid with respect to length '8' for type 'Password'. [6]
cvc-type.3.1.3: The value '123 345 7' of element 'PW' is not valid. [6]
XML validation finished.
  
```


Schemas

Structuring – XSD Complex Type Elements

- **Have** attributes, child elements, or some **combination** of the two
 - May contain **nested elements and have attributes**
- **Have 04 variations**
 - **Empty** elements
 - Optionally specify attribute types, but **do not permit content**
 - **Only** elements (**simple** element)
 - Can only contain elements and **do not contain** any attributes
 - **However**, either they can be contained attribute, or **their sub elements** are
 - **Text-Only** (simple content)
 - Contains only **simple content** (text and attributes)
 - A **simpleContent** element must be around the content
 - A **extension** or **restriction** tag must be defined within a simpleContent
 - **Mixed**
 - Can contain **text** content as well as **sub-elements within element**
 - They may or may not attributes

Schemas

Structuring – XSD Content Elements

- Is used to **define** an **extension** or a **restriction** of simple or complex type
- **There are 2 types**
 - **Simple Content**
 - Are created by **adding** a **list of attributes** to a **simple type**
 - Appear right **after** we **declare complexType** element
 - Only contains character data (**text**) and/or **attributes (if any)**.
 - Can be derived by,
 - extensions (**extending simple type by adding attribute**)
 - Or, restrictions (**allows not only restriction of the scope of the text node, but also the restriction of the scope of the attribute**)

Structuring – XSD Content Elements

- **There are 2 types (cont)**
 - **Complex Content**
 - Elements that **have child elements**.
 - Appear right **after** we **declare complexType element**
 - **Attributes** for such elements are declared after the element's model group.
 - Can be **derived, by extension or by restriction**, from complex types
 - All the instance structures that match the restricted complex type must also match the base complex type
 - Can be contained simple content in complex content

Schemas

Structuring – XSD Complex Type Elements

- There are **02 ways** to define a complex type
 - The element can be **declared directly** by naming the element

<xsd:element name="element_name">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="element_name" type="data types"/>

...

</xsd:sequence>

</xsd:complexType>

< /xsd:element>

- The **disadvantage** of this is **only the declared element** can use the specified complex type
 - The element can have a type attribute that **refers** to the name of the complex type to use

<xsd:element name="element_name" type="new_data_type"/>

<xsd:complexType name="new_data_type">

<xsd:sequence>

<xsd:element name="element_name" type="data types"/>

...

</xsd:sequence>

< /xsd:complexType>

- The **advantage** is **reusable** and **extensible**

Schemas

Structuring – Empty Complex Type – Example

simpleComplexType.xsd

Source Schema Design

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/simpleComplexType"
5   xmlns="http://xml.netbeans.org/schema/simpleComplexType"
6   elementFormDefault="qualified">
7   <xsd:element name="product">
8     <xsd:complexType>
9       <xsd:attribute name="prodid" type="xsd:positiveInteger"/>
10    </xsd:complexType>
11  </xsd:element>
12 </xsd:schema>
  
```

simpleComplexType.xml

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <product xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4   xmlns='http://xml.netbeans.org/schema/simpleComplexType'
5   xsi:schemaLocation='http://xml.netbeans.org/schema/simpleComplexType simpleComplexType.xsd'
6   prodid="12345"/>
  
```

Schemas

Structuring – Empty Complex Type – Example

simpleComplexType.xsd

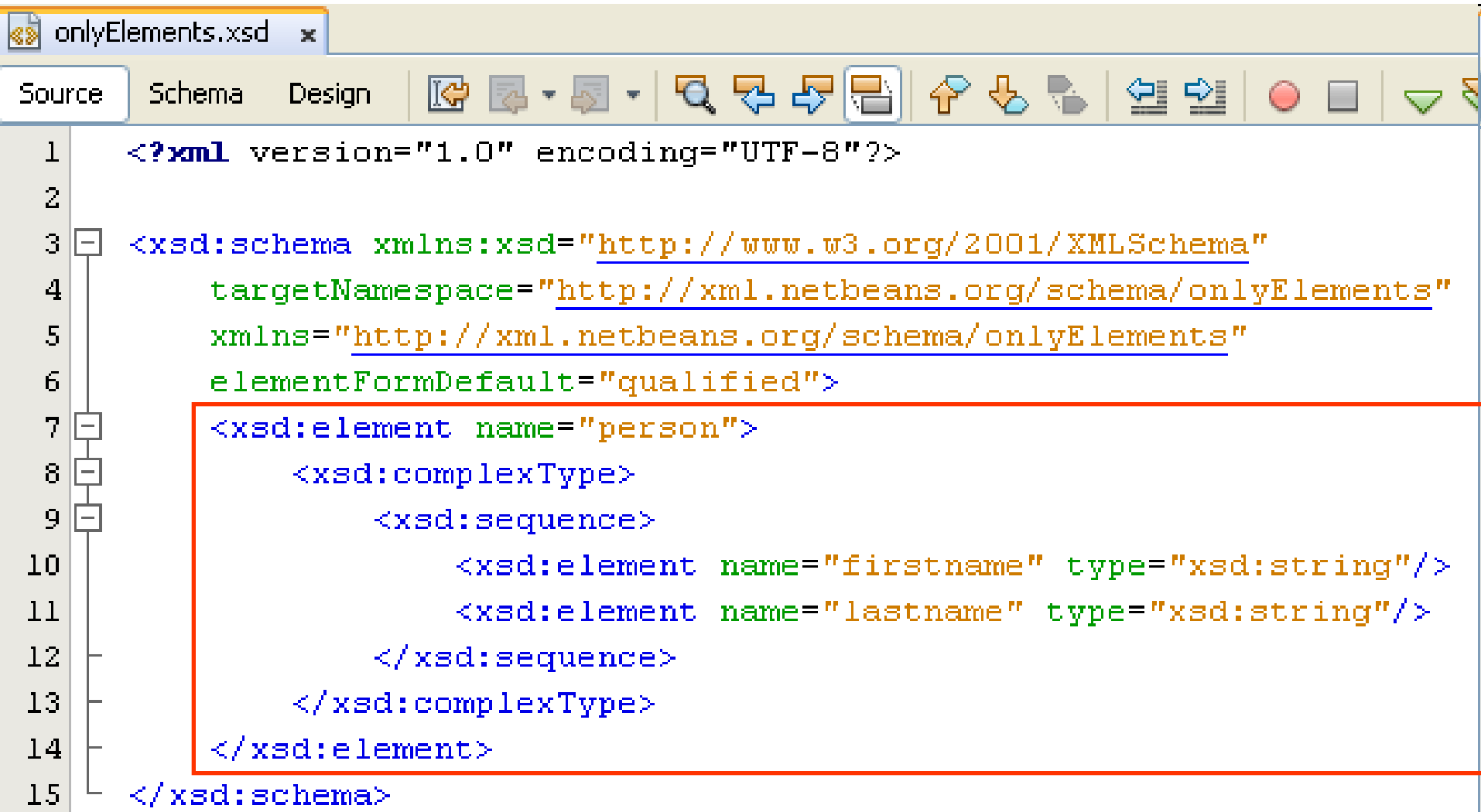
Source Schema Design

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/simpleComplexType"
5      xmlns="http://xml.netbeans.org/schema/simpleComplexType"
6      elementFormDefault="qualified">
7      <xsd:element name="product" type="prodtype"/>
8
9      <xsd:complexType name="prodtype">
10         <xsd:attribute name="prodid" type="xsd:positiveInteger"/>
11     </xsd:complexType>
12 </xsd:schema>
  
```

Schemas

Structuring – Only Elements – Example

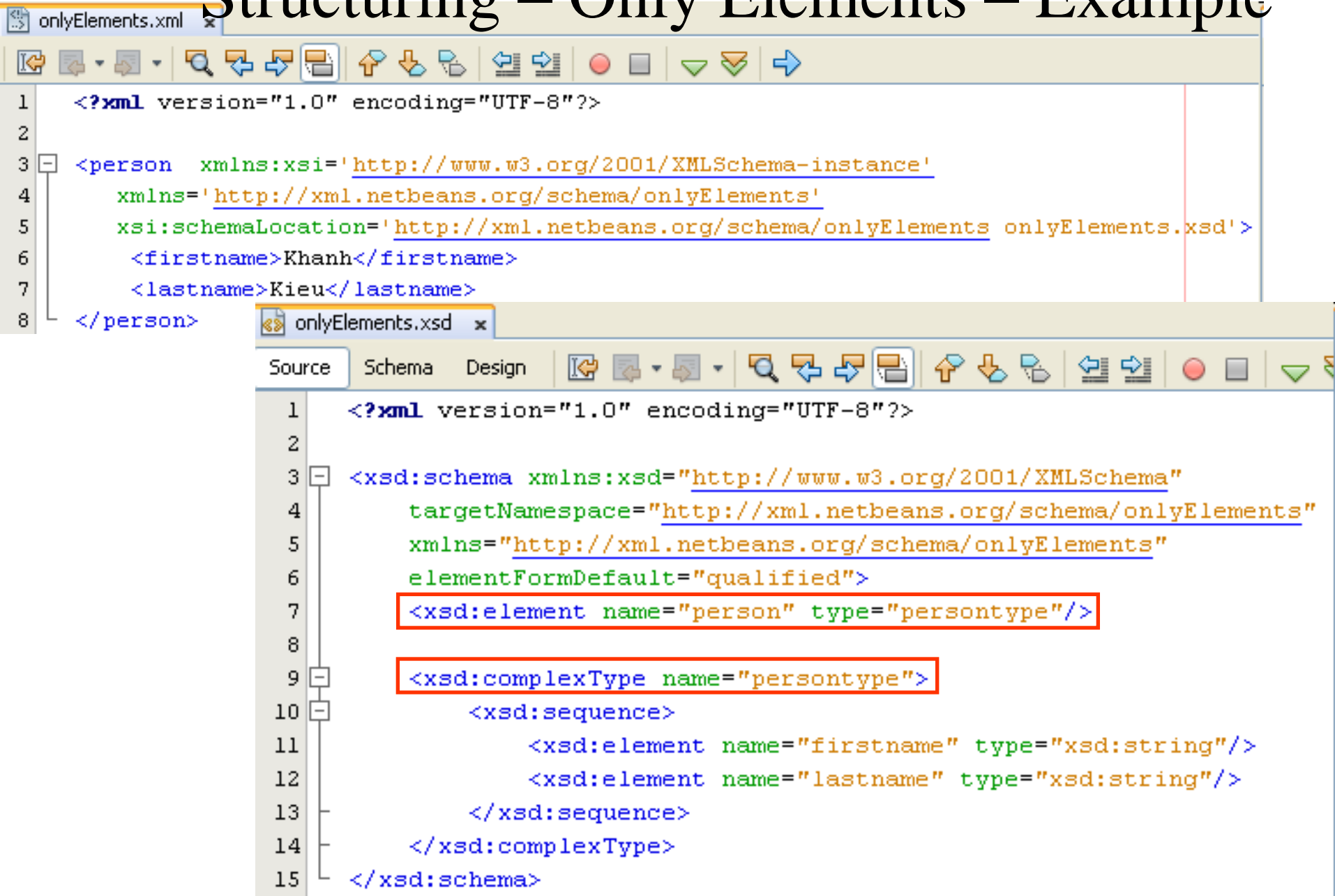


```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/onlyElements"
5      xmlns="http://xml.netbeans.org/schema/onlyElements"
6      elementFormDefault="qualified">
7      <xsd:element name="person">
8          <xsd:complexType>
9              <xsd:sequence>
10                 <xsd:element name="firstname" type="xsd:string"/>
11                 <xsd:element name="lastname" type="xsd:string"/>
12             </xsd:sequence>
13         </xsd:complexType>
14     </xsd:element>
15 </xsd:schema>
  
```

Schemas

Structuring – Only Elements – Example



The image shows two XML files in an editor. The top file, `onlyElements.xml`, contains an XML document with a root element `<person>` and two child elements, `<firstname>` and `<lastname>`. The bottom file, `onlyElements.xsd`, is an XSD schema that defines the structure of the XML document. It includes a `<xsd:schema>` element with various attributes and two nested elements: `<xsd:element>` and `<xsd:complexType>`. The `<xsd:element>` element is highlighted with a red box, and the `<xsd:complexType>` element is also highlighted with a red box. The `<xsd:complexType>` element contains a `<xsd:sequence>` element, which in turn contains two `<xsd:element>` elements, `<xsd:element name="firstname" type="xsd:string"/>` and `<xsd:element name="lastname" type="xsd:string"/>`.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <person xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/onlyElements'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/onlyElements onlyElements.xsd'>
6      <firstname>Khanh</firstname>
7      <lastname>Kieu</lastname>
8  </person>

```

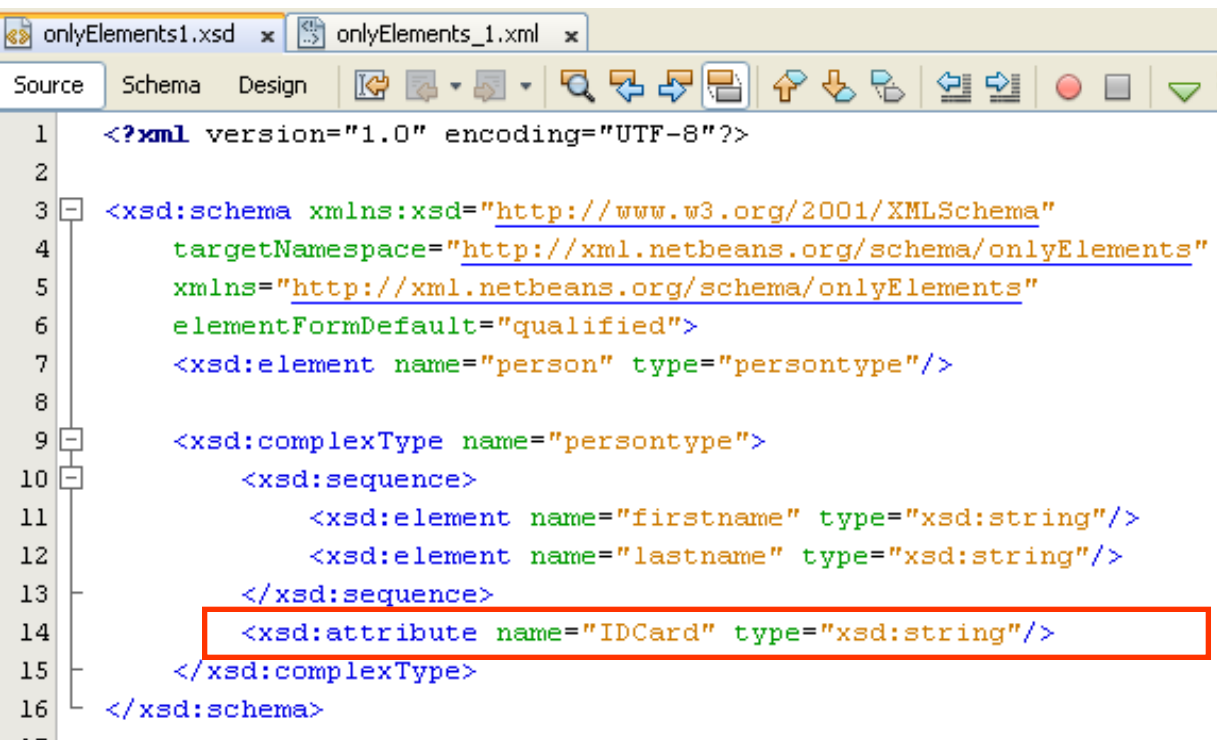
```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/onlyElements"
5      xmlns="http://xml.netbeans.org/schema/onlyElements"
6      elementFormDefault="qualified">
7      <xsd:element name="person" type="persontype"/>
8
9      <xsd:complexType name="persontype">
10         <xsd:sequence>
11             <xsd:element name="firstname" type="xsd:string"/>
12             <xsd:element name="lastname" type="xsd:string"/>
13         </xsd:sequence>
14     </xsd:complexType>
15 </xsd:schema>

```

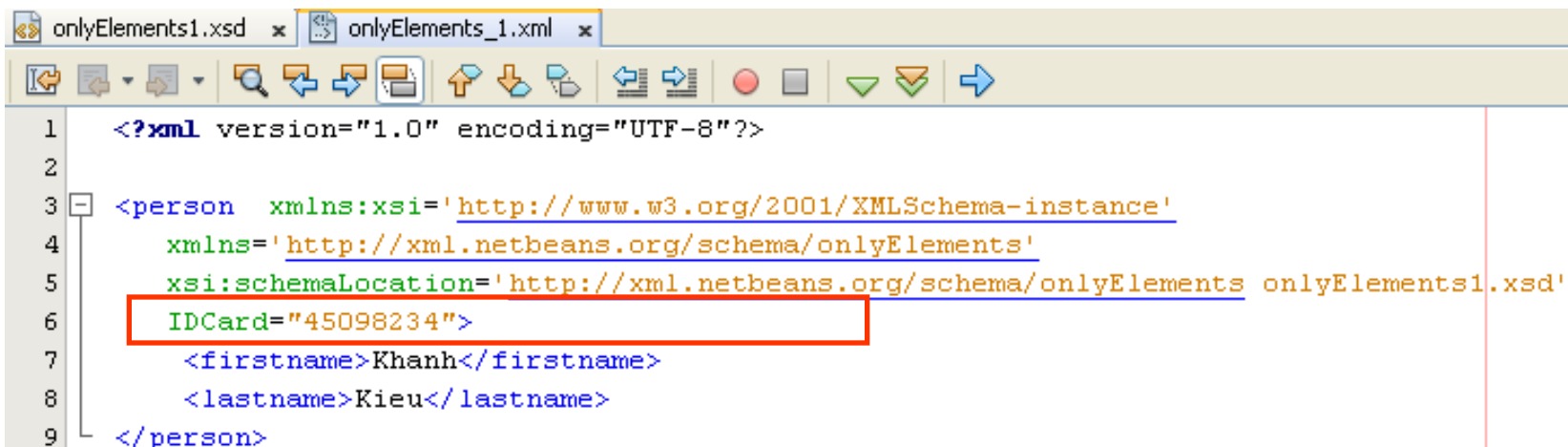

Schemas

Structuring – Only Elements – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/onlyElements"
5      xmlns="http://xml.netbeans.org/schema/onlyElements"
6      elementFormDefault="qualified">
7      <xsd:element name="person" type="persontype"/>
8
9      <xsd:complexType name="persontype">
10         <xsd:sequence>
11             <xsd:element name="firstname" type="xsd:string"/>
12             <xsd:element name="lastname" type="xsd:string"/>
13         </xsd:sequence>
14         <xsd:attribute name="IDCard" type="xsd:string"/>
15     </xsd:complexType>
16 </xsd:schema>
  
```

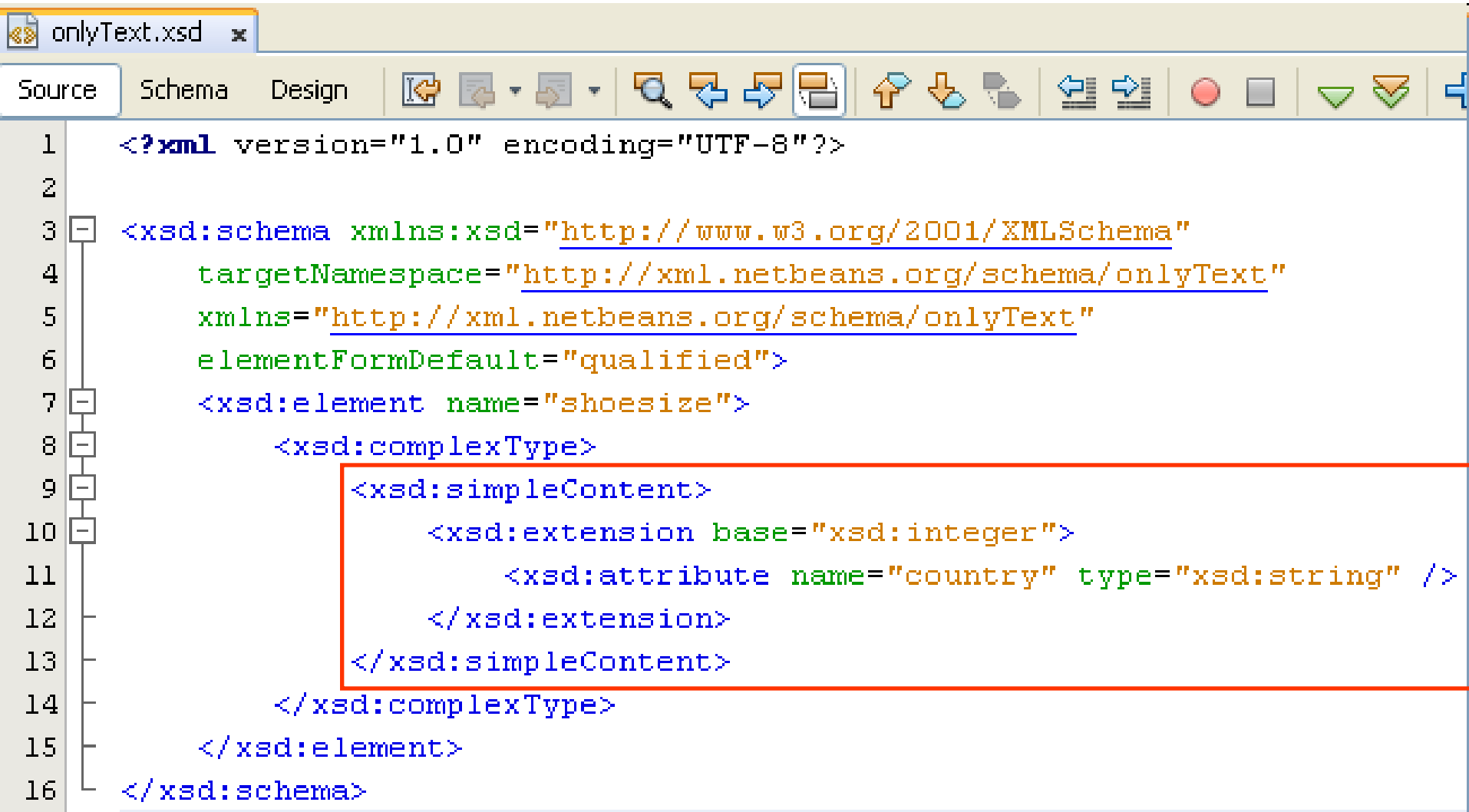


```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <person xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/onlyElements'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/onlyElements onlyElements1.xsd'
6      IDCard="45098234">
7      <firstname>Khanh</firstname>
8      <lastname>Kieu</lastname>
9  </person>
  
```

Schemas

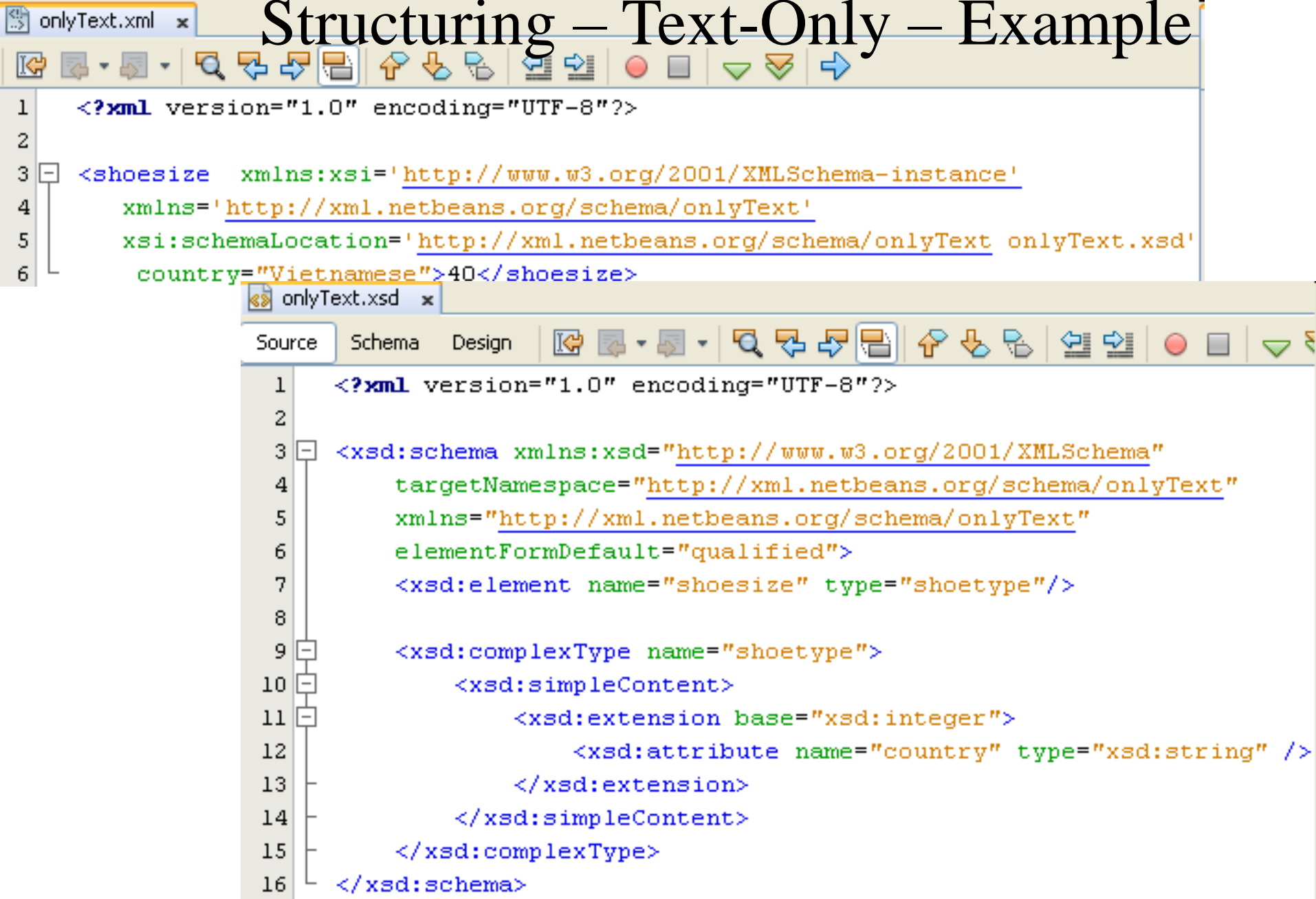
Structuring – Text-Only – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/onlyText"
5      xmlns="http://xml.netbeans.org/schema/onlyText"
6      elementFormDefault="qualified">
7      <xsd:element name="shoesize">
8          <xsd:complexType>
9              <xsd:simpleContent>
10                 <xsd:extension base="xsd:integer">
11                     <xsd:attribute name="country" type="xsd:string" />
12                 </xsd:extension>
13             </xsd:simpleContent>
14         </xsd:complexType>
15     </xsd:element>
16 </xsd:schema>
  
```

Structuring – Text-Only – Example



The screenshot displays the NetBeans IDE with two XML files open: `onlyText.xml` and `onlyText.xsd`.

onlyText.xml (lines 1-6):

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <shoesize xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/onlyText'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/onlyText onlyText.xsd'
6      country="Vietnamese">40</shoesize>

```

onlyText.xsd (lines 1-16):

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/onlyText"
5      xmlns="http://xml.netbeans.org/schema/onlyText"
6      elementFormDefault="qualified">
7      <xsd:element name="shoesize" type="shoetype"/>
8
9      <xsd:complexType name="shoetype">
10         <xsd:simpleContent>
11             <xsd:extension base="xsd:integer">
12                 <xsd:attribute name="country" type="xsd:string" />
13             </xsd:extension>
14         </xsd:simpleContent>
15     </xsd:complexType>
16 </xsd:schema>

```

Structuring – Text-Only – Example



```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/attr_simpleContent"
5   xmlns="http://xml.netbeans.org/schema/attr_simpleContent"
6   elementFormDefault="qualified">
7   <xsd:element name="Author">
8     <xsd:complexType>
9       <xsd:sequence>
10        <xsd:element name="Name">
11          <xsd:complexType>
12            <xsd:sequence>
13              <xsd:element name="FirstName">
14                <xsd:complexType>
15                  <xsd:simpleContent>
16                    <xsd:extension base="xsd:string">
17                      <xsd:attribute name="Full" type="xsd:boolean"/>
18                    </xsd:extension>
19                  </xsd:simpleContent>
20                </xsd:complexType>
21              </xsd:element>
22              <xsd:element name="LastName" type="xsd:string"/>
23            </xsd:sequence>
24            <xsd:attribute name="Pseudonym" type="xsd:boolean"/>
25            <xsd:attribute name="HomePage" type="xsd:anyURI"/>
26          </xsd:complexType>
27        </xsd:element>
28      </xsd:sequence>
29    </xsd:complexType>
30  </xsd:element>
31 </xsd:schema>
  
```

Schemas

Structuring – Text-Only – Example

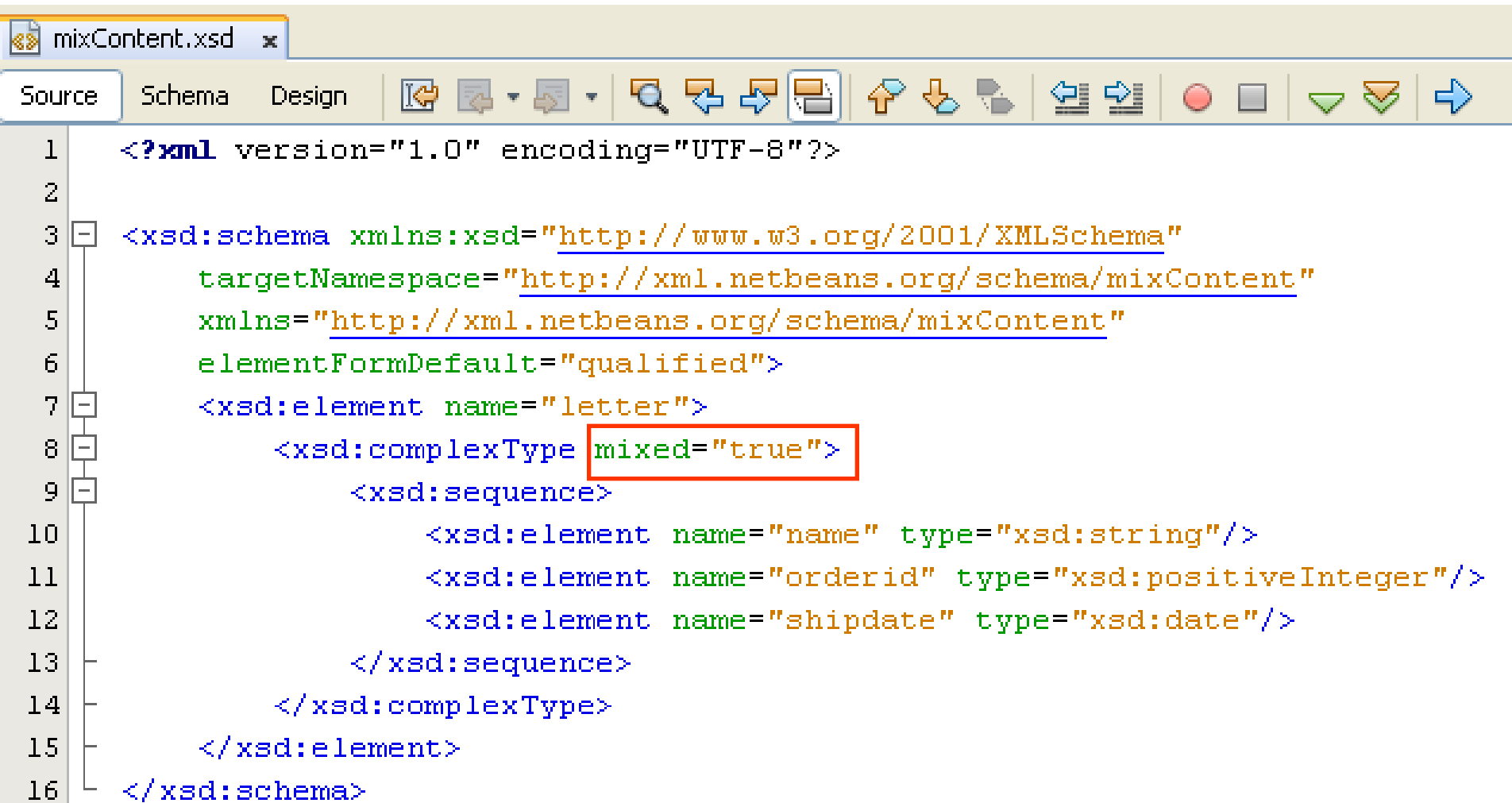


```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Author xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns='http://xml.netbeans.org/schema/attr_simpleContent'
5        xsi:schemaLocation='http://xml.netbeans.org/schema/attr_simpleContent attr_simpleContent.xsd'>
6    <Name Pseudonym="true" HomePage="http://www.marktwain.com">
7        <FirstName Full="true">Mark Twain</FirstName>
8        <LastName>Twain</LastName>
9    </Name>
10 </Author>
  
```

Schemas

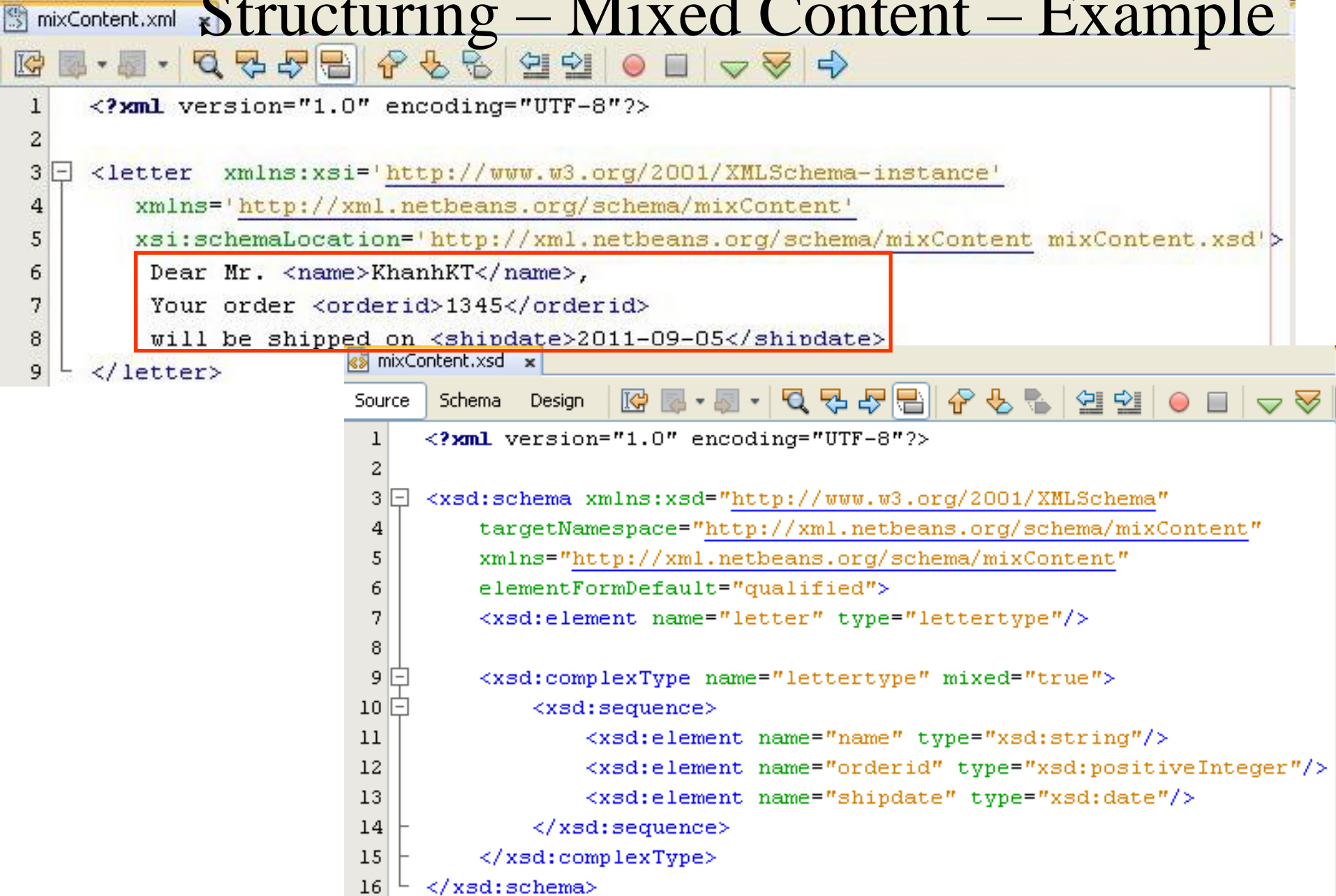
Structuring – Mixed Content – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/mixContent"
5      xmlns="http://xml.netbeans.org/schema/mixContent"
6      elementFormDefault="qualified">
7      <xsd:element name="letter">
8          <xsd:complexType mixed="true">
9              <xsd:sequence>
10                 <xsd:element name="name" type="xsd:string"/>
11                 <xsd:element name="orderid" type="xsd:positiveInteger"/>
12                 <xsd:element name="shipdate" type="xsd:date"/>
13             </xsd:sequence>
14         </xsd:complexType>
15     </xsd:element>
16 </xsd:schema>
  
```

Structuring – Mixed Content – Example



The image shows two XML files in an IDE. The top file, `mixContent.xml`, is an XML document with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<letter xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
  xmlns='http://xml.netbeans.org/schema/mixContent'
  xsi:schemaLocation='http://xml.netbeans.org/schema/mixContent mixContent.xsd'>
  Dear Mr. <name>KhanhKT</name>,
  Your order <orderid>1345</orderid>
  will be shipped on <shipdate>2011-09-05</shipdate>
</letter>
```

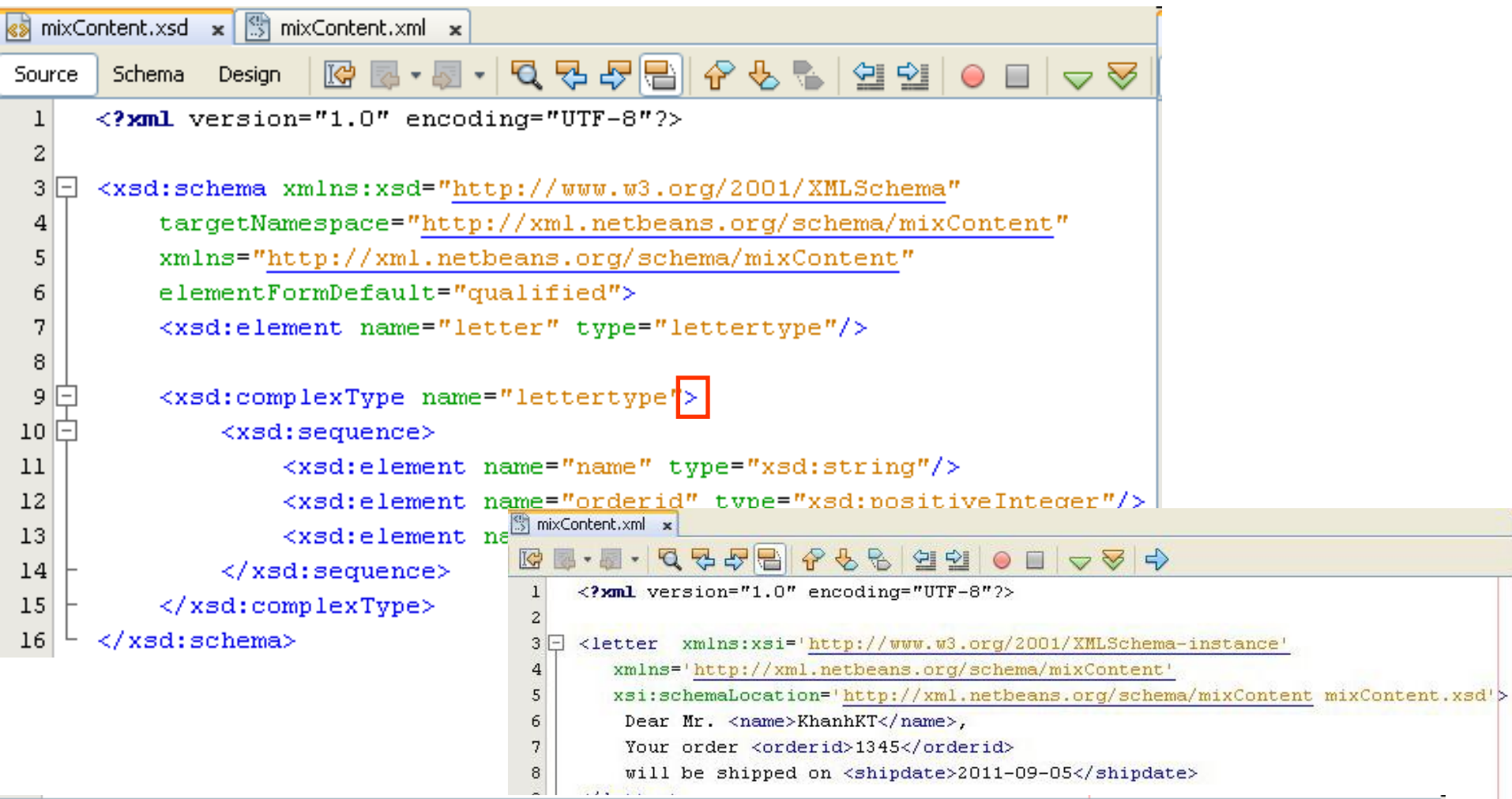
The bottom file, `mixContent.xsd`, is an XSD schema defining the structure of the XML document:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://xml.netbeans.org/schema/mixContent"
  xmlns="http://xml.netbeans.org/schema/mixContent"
  elementFormDefault="qualified">
  <xsd:element name="letter" type="lettertype"/>

  <xsd:complexType name="lettertype" mixed="true">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string"/>
      <xsd:element name="orderid" type="xsd:positiveInteger"/>
      <xsd:element name="shipdate" type="xsd:date"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

Schemas

Structuring – Mixed Content – Example



The screenshot shows the NetBeans IDE with two tabs: `mixContent.xsd` and `mixContent.xml`. The `mixContent.xsd` tab is active, displaying the following XML Schema code:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/mixContent"
5      xmlns="http://xml.netbeans.org/schema/mixContent"
6      elementFormDefault="qualified">
7      <xsd:element name="letter" type="lettertype"/>
8
9      <xsd:complexType name="lettertype">
10         <xsd:sequence>
11             <xsd:element name="name" type="xsd:string"/>
12             <xsd:element name="orderid" type="xsd:positiveInteger"/>
13             <xsd:element name="shipdate" type="xsd:date"/>
14         </xsd:sequence>
15     </xsd:complexType>
16 </xsd:schema>

```

The `mixContent.xml` tab is also visible, showing the following XML instance code:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <letter xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/mixContent'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/mixContent mixContent.xsd'>
6      Dear Mr. <name>KhanhKT</name>,
7      Your order <orderid>1345</orderid>
8      will be shipped on <shipdate>2011-09-05</shipdate>

```

Output - XML check

```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mixContent.xml...
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/mixContent.xsd".
cvc-complex-type.2.3: Element 'letter' cannot have character [children], because the type's content type is element-only. [9]
XML validation finished.

```


Schemas

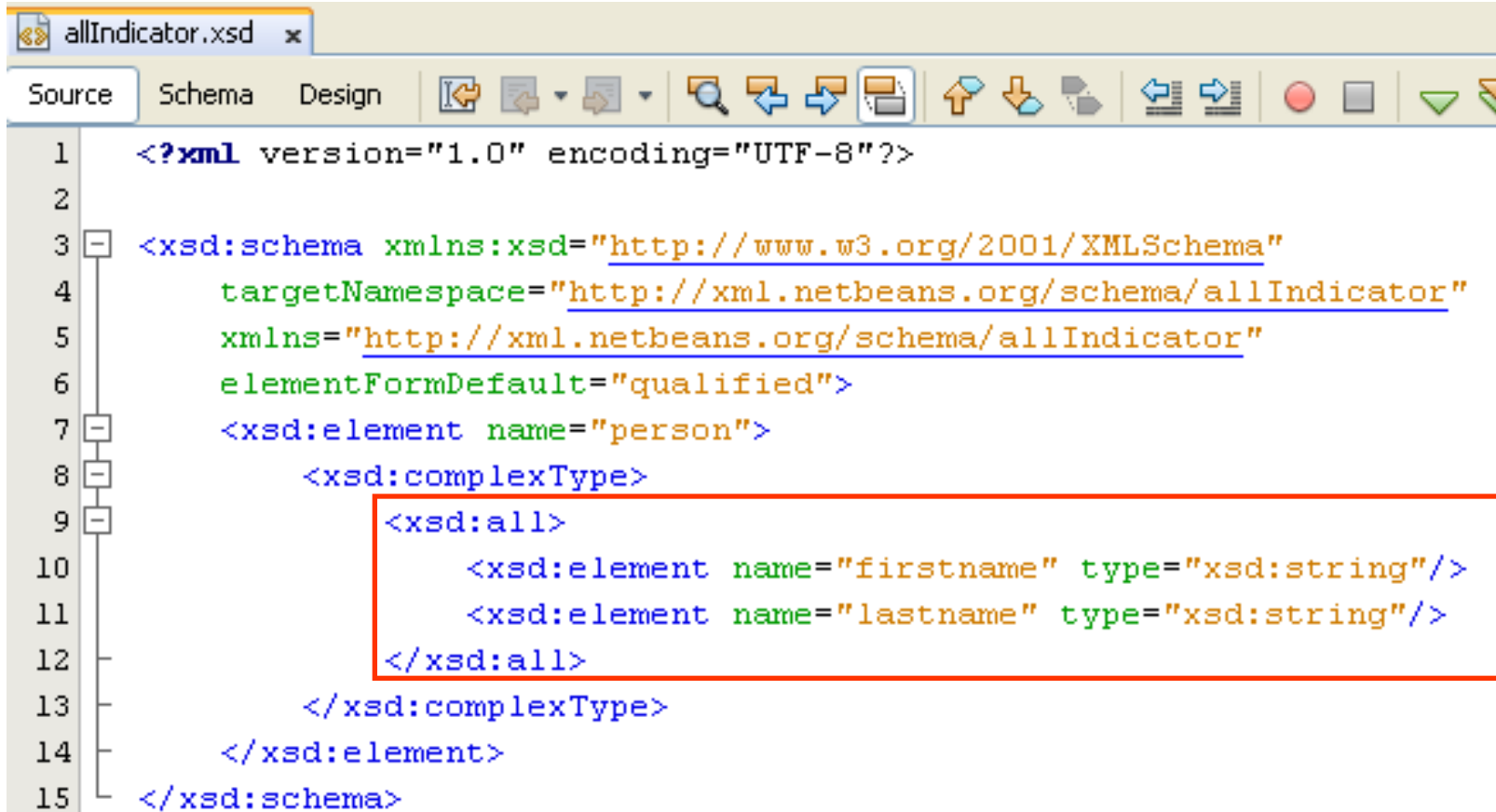
Structuring – XSD Indicators

- **Control how elements** are to be **used** in documents
- **Indicate the structure and order** in which **child elements** can appear within their parent element
- There are seven indicators:
 - **Order** indicators:
 - All
 - Choice
 - Sequence
 - **Occurrence** indicators:
 - maxOccurs
 - minOccurs
 - **Group** indicators:
 - Group name
 - attributeGroup name

Schemas

Structuring – Order Indicators

- Are used to define the **order of the elements**
- **All** indicator
 - **Specifies** that the child elements can appear in **any order**, and that **each** child element **must occur only once**
 - **Ex:**



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4             targetNamespace="http://xml.netbeans.org/schema/allIndicator"
5             xmlns="http://xml.netbeans.org/schema/allIndicator"
6             elementFormDefault="qualified">
7    <xsd:element name="person">
8      <xsd:complexType>
9        <xsd:all>
10         <xsd:element name="firstname" type="xsd:string"/>
11         <xsd:element name="lastname" type="xsd:string"/>
12        </xsd:all>
13      </xsd:complexType>
14    </xsd:element>
15  </xsd:schema>
  
```

Schemas

Structuring – Order Indicators – Example

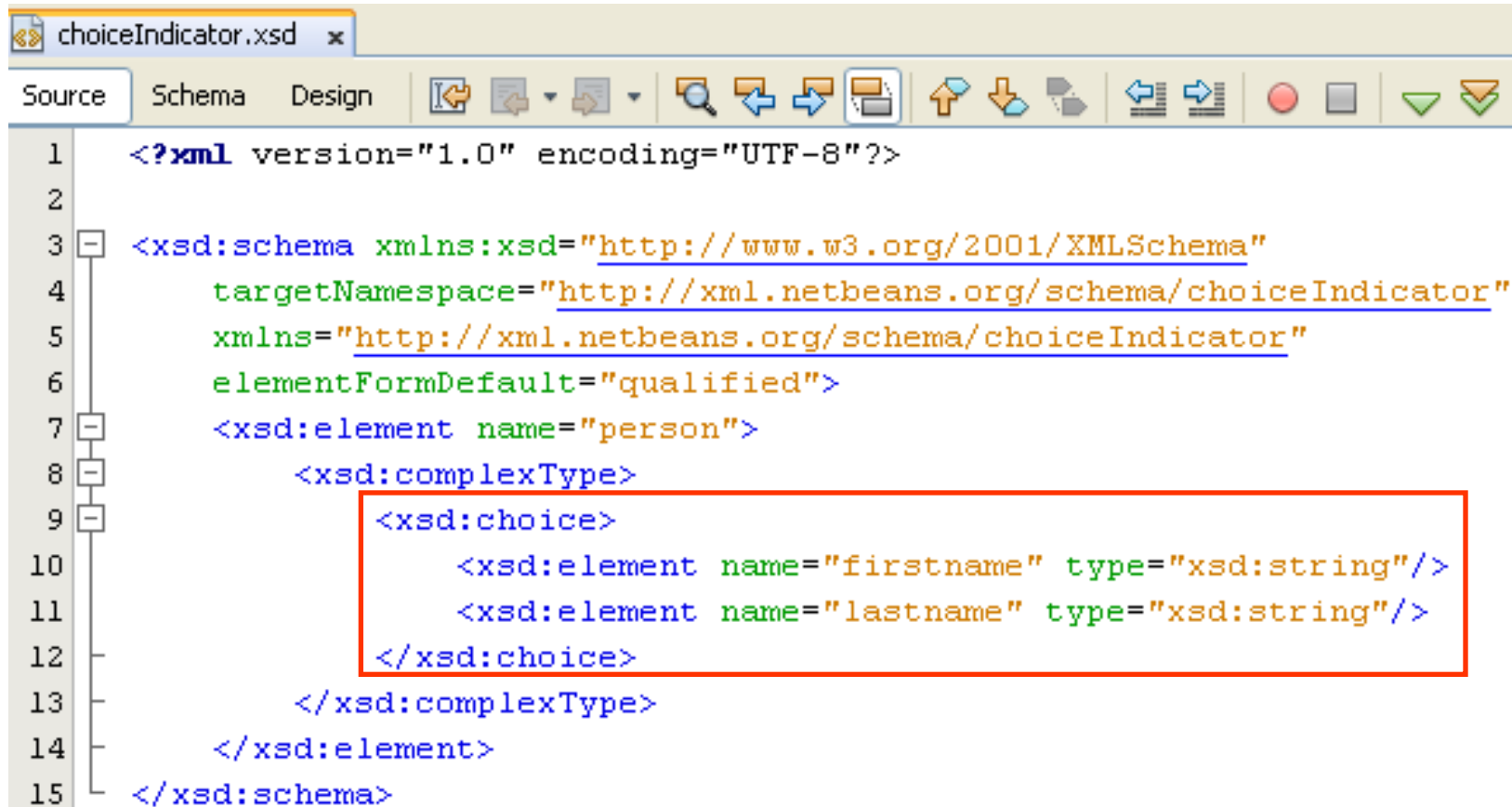
```
allIndicator.xml x
[Icons] [Tools] [Navigation] [Run] [Stop] [Debug] [Test] [Run] [Run] [Run]
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <person xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/allIndicator'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/allIndicator allIndicator.xsd'>
6      <firstname>Khanh</firstname>
7      <lastname>Kieu</lastname>
8  </person>
```

```
allIndicator.xml x
[Icons] [Tools] [Navigation] [Run] [Stop] [Debug] [Test] [Run] [Run] [Run]
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <person xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/allIndicator'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/allIndicator allIndicator.xsd'>
6      <lastname>Kieu</lastname>
7      <firstname>Khanh</firstname>
8  </person>
```

Schemas

Structuring – Order Indicators

- **Choice indicator**
 - Specifies that **either one** child element **or another** can **occur** (meaning that **only one** of the child elements **may show up**)
 - Ex:



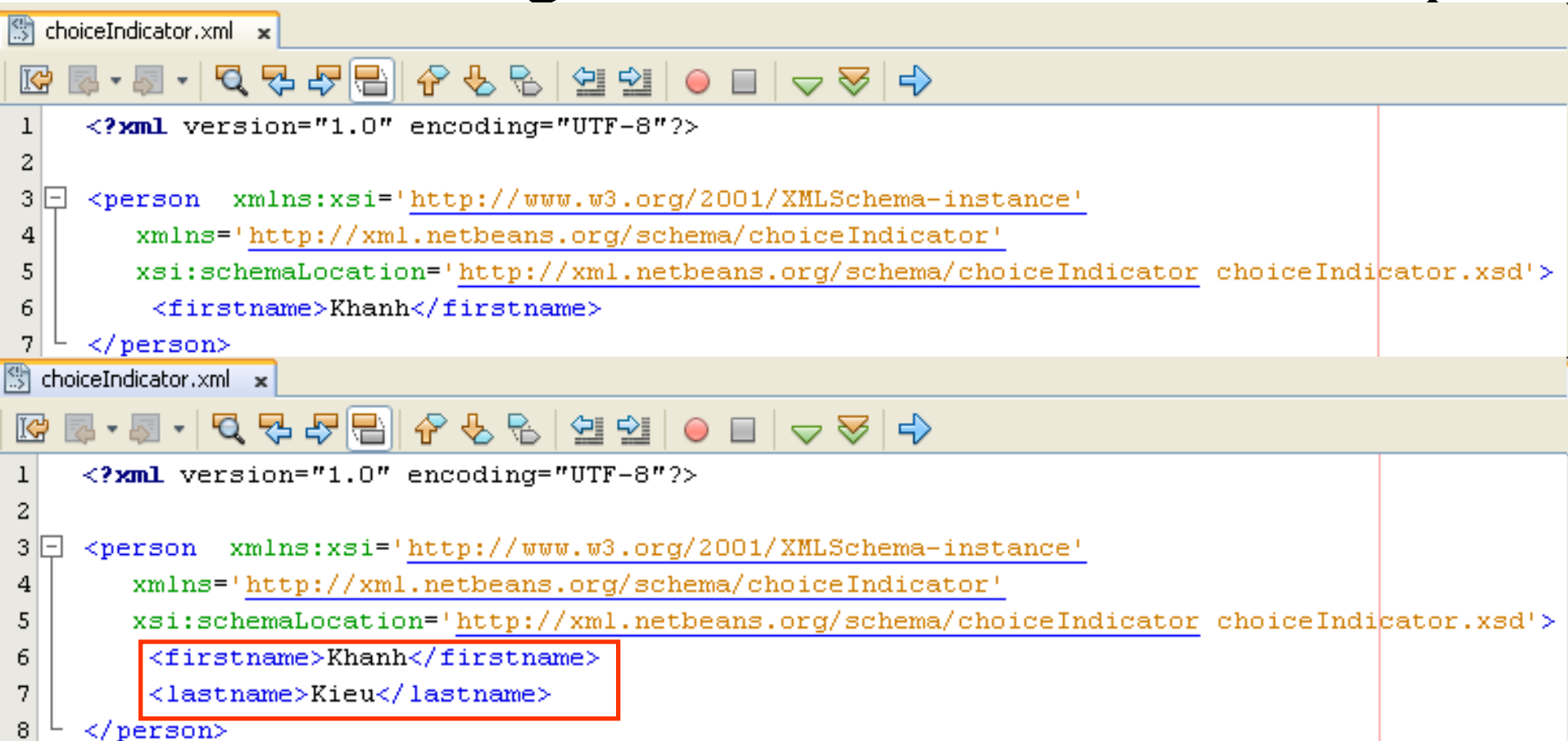
The screenshot shows an XML Schema editor window titled "choiceIndicator.xsd". The editor has tabs for "Source", "Schema", and "Design". The "Source" tab is active, displaying the following XML Schema code:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/choiceIndicator"
5      xmlns="http://xml.netbeans.org/schema/choiceIndicator"
6      elementFormDefault="qualified">
7      <xsd:element name="person">
8          <xsd:complexType>
9              <xsd:choice>
10                 <xsd:element name="firstname" type="xsd:string"/>
11                 <xsd:element name="lastname" type="xsd:string"/>
12             </xsd:choice>
13          </xsd:complexType>
14      </xsd:element>
15  </xsd:schema>
  
```

The code defines an XML Schema for a "person" element. The "person" element is of type "xsd:complexType" and contains a "choice" indicator. The "choice" indicator specifies that either the "firstname" element or the "lastname" element can occur, but only one. The "firstname" and "lastname" elements are of type "xsd:string".

Structuring – Order Indicators – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <person xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/choiceIndicator'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/choiceIndicator choiceIndicator.xsd'>
6      <firstname>Khanh</firstname>
7  </person>

```

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <person xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/choiceIndicator'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/choiceIndicator choiceIndicator.xsd'>
6      <firstname>Khanh</firstname>
7      <lastname>Kieu</lastname>
8  </person>

```

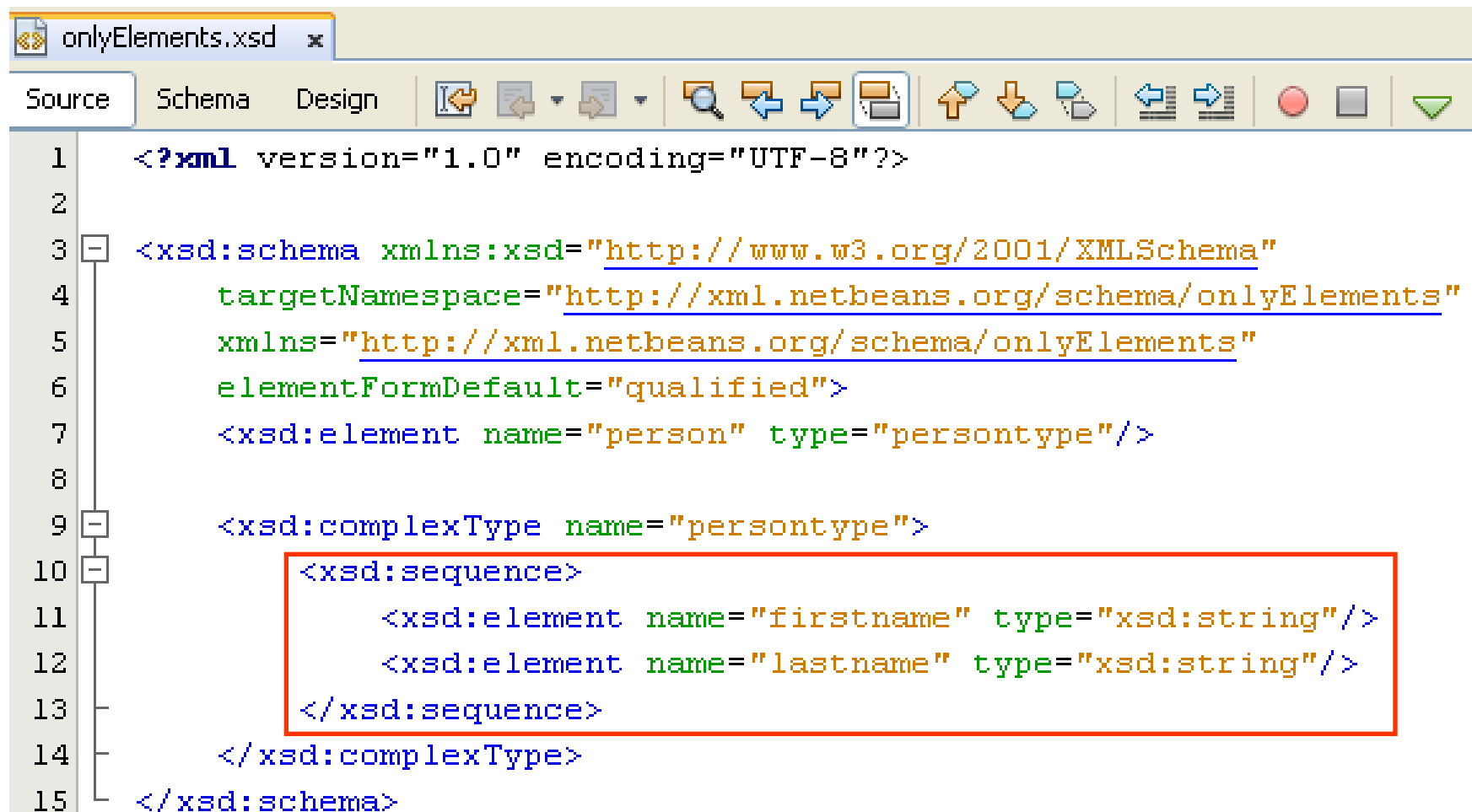
Output - XML check

XML validation started.
 Checking file: /Z:/Laptrinh/Servlet/Day3XML/src/choiceIndicator.xml...
 Referenced entity at "file: /Z:/Laptrinh/Servlet/Day3XML/src/choiceIndicator.xsd".
[cvc-complex-type.2.4.d: Invalid content was found starting with element 'lastname'. No child element is expected at this point. \[7\]](#)
 XML validation finished.

Schemas

Structuring – Order Indicators

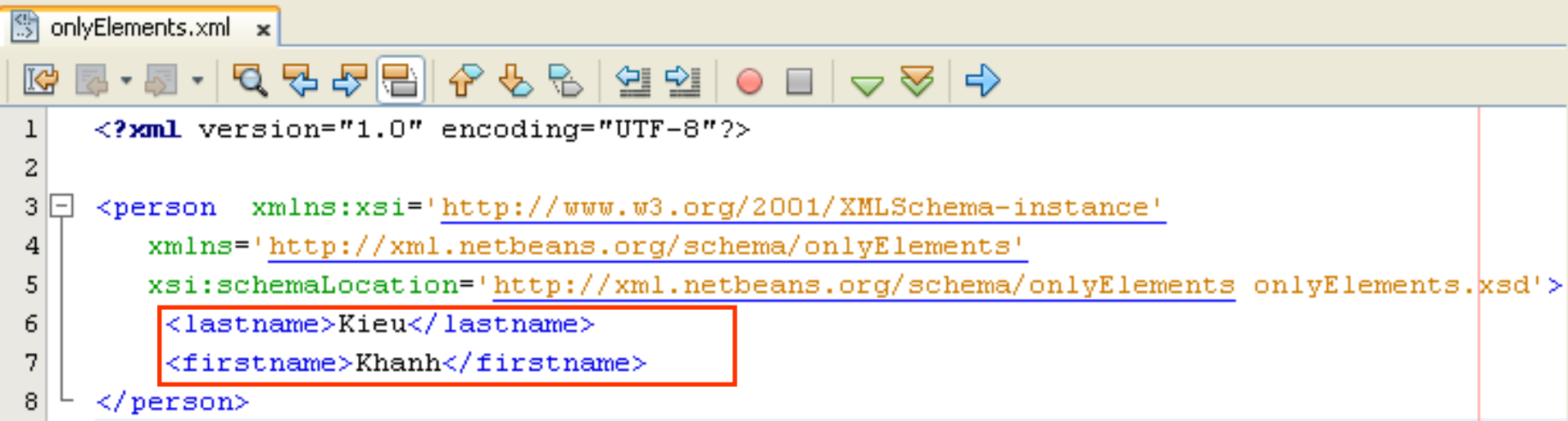
- Sequence indicator
 - Specifies** that the **child** elements must **appear in** a specific **order**
 - Ex:**



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/onlyElements"
5      xmlns="http://xml.netbeans.org/schema/onlyElements"
6      elementFormDefault="qualified">
7      <xsd:element name="person" type="persontype"/>
8
9      <xsd:complexType name="persontype">
10         <xsd:sequence>
11             <xsd:element name="firstname" type="xsd:string"/>
12             <xsd:element name="lastname" type="xsd:string"/>
13         </xsd:sequence>
14     </xsd:complexType>
15 </xsd:schema>
    
```

Structuring – Order Indicators – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <person xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/onlyElements'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/onlyElements onlyElements.xsd'>
6      <lastname>Kieu</lastname>
7      <firstname>Khanh</firstname>
8  </person>

```

Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/onlyElements.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/onlyElements.xsd".
cvc-complex-type.2.4.a: Invalid content was found starting with element 'lastname'. One of '{"http://xml.netbeans.org/schema/onlyElements":firstname}' is expected. [6]
XML validation finished.

```

Structuring – Order Indicators – Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3      targetNamespace="http://xml.netbeans.org/schema/complexOrderIndicator"
4      xmlns="http://xml.netbeans.org/schema/complexOrderIndicator"
5      elementFormDefault="qualified">
6      <xsd:simpleType name="Salary">
7          <xsd:restriction base="xsd:decimal">
8              <xsd:minInclusive value="10000"/>
9              <xsd:maxInclusive value="90000"/>
10         </xsd:restriction>
11     </xsd:simpleType>
12     <xsd:element name="Employee">
13         <xsd:complexType>
14             <xsd:sequence>
15                 <xsd:element name="Name">
16                     <xsd:complexType>
17                         <xsd:sequence>
18                             <xsd:element name="FirstName"/>
19                             <xsd:element name="LastName"/>
20                         </xsd:sequence>
21                     </xsd:complexType>
22                 </xsd:element>
23                 <xsd:choice>
24                     <xsd:element name="Salary" type="Salary"/>
25                     <xsd:element name="Wage" type="xsd:decimal"/>
26                 </xsd:choice>
27             </xsd:sequence>
28         </xsd:complexType>
29     </xsd:element>
30 </xsd:schema>

```


Schemas

Structuring – Order Indicators – Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Employee xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4          xmlns='http://xml.netbeans.org/schema/complexOrderIndicator'
5          xsi:schemaLocation='http://xml.netbeans.org/schema/complexOrderIndicator complexOrderIndicator.xsd'>
6      <Name>
7          <FirstName>Khanh</FirstName>
8          <LastName>Kieu</LastName>
9      </Name>
10     <Salary>45000</Salary>
11 </Employee>
  
```

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Employee xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4          xmlns='http://xml.netbeans.org/schema/complexOrderIndicator'
5          xsi:schemaLocation='http://xml.netbeans.org/schema/complexOrderIndicator complexOrderIndicator.xsd'>
6      <Name>
7          <FirstName>Khanh</FirstName>
8          <LastName>Kieu</LastName>
9      </Name>
10     <Wage>4333</Wage>
11 </Employee>
  
```

Schemas

Structuring – Occurrence Indicators

- Are used to **define how often** an element can **occur**
- Are same ideas as the DTD that uses the markers *, ?, and + to indicate the number of time a particular child element
- The **default value** of indicators for **maxOccurs** and **minOccurs** is **1**
- **maxOccurs** indicator
 - Specifies the **maximum number of times** an element can occur
- **minOccurs** indicator
 - Specifies the **minimum number of times** an element can occur

Schemas

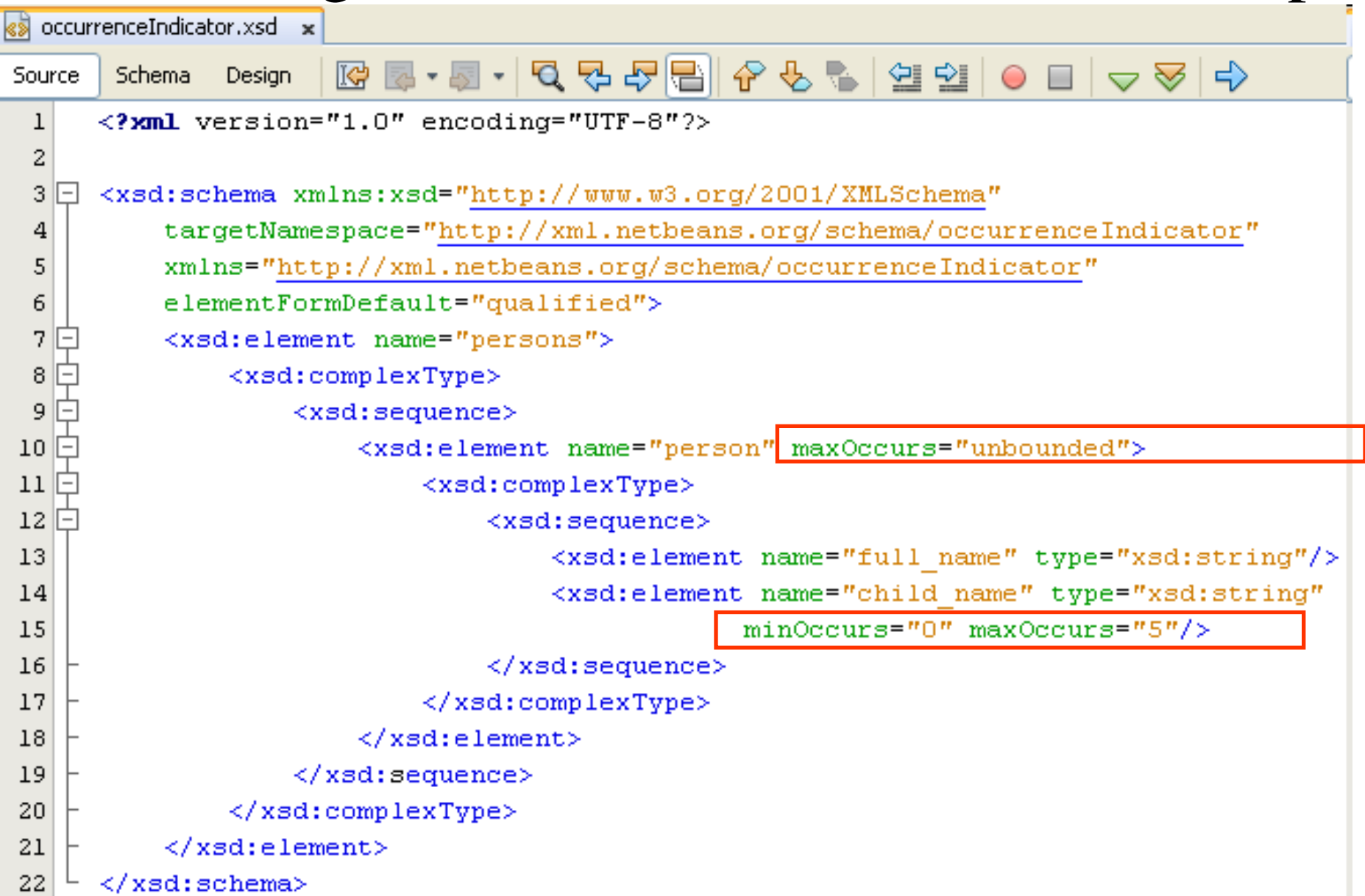
Structuring – Occurrence Indicators

The relationship between minOccurs and maxOccurs attributes

minOccurs	maxOccurs	Number of times an element can occur
0	1	0 or 1
1	1	1
0	unbounded	Infinite – at least zero
1	unbounded	At least one
>0	unbounded	At least minOccurs times
>maxOccurs	Any value	0 – error
Any value	<minOccurs	0 – error

Schemas

Structuring – Occurrence Indicators – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    targetNamespace="http://xml.netbeans.org/schema/occurrenceIndicator"
5    xmlns="http://xml.netbeans.org/schema/occurrenceIndicator"
6    elementFormDefault="qualified">
7    <xsd:element name="persons">
8      <xsd:complexType>
9        <xsd:sequence>
10         <xsd:element name="person" maxOccurs="unbounded">
11           <xsd:complexType>
12             <xsd:sequence>
13               <xsd:element name="full_name" type="xsd:string"/>
14               <xsd:element name="child_name" type="xsd:string"
15                 minOccurs="0" maxOccurs="5"/>
16             </xsd:sequence>
17           </xsd:complexType>
18         </xsd:element>
19       </xsd:sequence>
20     </xsd:complexType>
21   </xsd:element>
22 </xsd:schema>
  
```

Schemas

Structuring – Occurrence Indicators – Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <persons xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3      xmlns='http://xml.netbeans.org/schema/occurrenceIndicator'
4      xsi:schemaLocation='http://xml.netbeans.org/schema/occurrenceIndicator occurrenceIndicator.xsd'>
5      <person>
6          <full_name>Kim Dung</full_name>
7          <child_name>Hu Truc</child_name>
8          <child_name>Doan Du</child_name>
9          <child_name>Kieu Phong</child_name>
10     </person>
11     <person>
12         <full_name>Doan Thuan</full_name>
13         <child_name>Doan Quan</child_name>
14         <child_name>Doan Duan</child_name>
15     </person>
16     <person>
17         <full_name>Kieu Phong</full_name>
18     </person>
19 </persons>
  
```

Structuring – Occurrence Indicators – Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <persons xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3      xmlns='http://xml.netbeans.org/schema/occurrenceIndicator'
4      xsi:schemaLocation='http://xml.netbeans.org/schema/occurrenceIndicator occurrenceIndicator.xsd'>
5      <person>
6          <full_name>Kim Dung</full_name>
7          <child_name>Hu Truc</child_name>
8          <child_name>Doan Du</child_name>
9          <child_name>Kieu Phong</child_name>
10         <child_name>Ho Phi</child_name>
11         <child_name>Vi Tieu Bao</child_name>
12         <child_name>Hiep Khach</child_name>
13     </person>
14     <person>
15         <full_name>Doan Thuan</full_name>
16         <child_name>Doan Quan</child_name>
17         <child_name>Doan Duan</child_name>
18     </person>
19     <person>
20
21 </person>
  
```

Output - XML check

XML validation started.

Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/occurrenceIndicator.xml...

Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/occurrenceIndicator.xsd".

cvc-complex-type.2.4.d: Invalid content was found starting with element 'person'. No child element '{"http://xml.netbeans.org/schema/occurrenceIndicator":child_name}' is expected at this point. [13]

cvc-complex-type.2.4.b: The content of element 'person' is not complete. One of '{"http://xml.netbeans.org/schema/occurrenceIndicator":full_name}' is expected. [21]

XML validation finished.

Structuring – Occurrence Indicators – Example

The screenshot shows an XML editor window titled 'occurrenceIndicator.xsd'. The editor is in 'Source' mode and displays the following XML schema code:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/occurrenceIndicator"
5   xmlns="http://xml.netbeans.org/schema/occurrenceIndicator"
6   elementFormDefault="qualified">
7   <xsd:element name="persons">
8     <xsd:complexType>
9       <xsd:sequence>
10        <xsd:element name="person" maxOccurs="unbounded">
11          <xsd:complexType>
12            <xsd:sequence>
13              <xsd:element name="full_name" type="xsd:string"/>
14              <xsd:element name="child_name" type="xsd:string"
15                maxOccurs="0"/>
16            </xsd:sequence>
17          </xsd:complexType>
18        </xsd:element>
19      </xsd:sequence>
20    </xsd:complexType>
21  </xsd:element>
22 </xsd:schema>
  
```

The line containing the `maxOccurs="0"/>` attribute is highlighted with a red box.

Below the editor, an 'Output - XML check' window displays the following validation message:

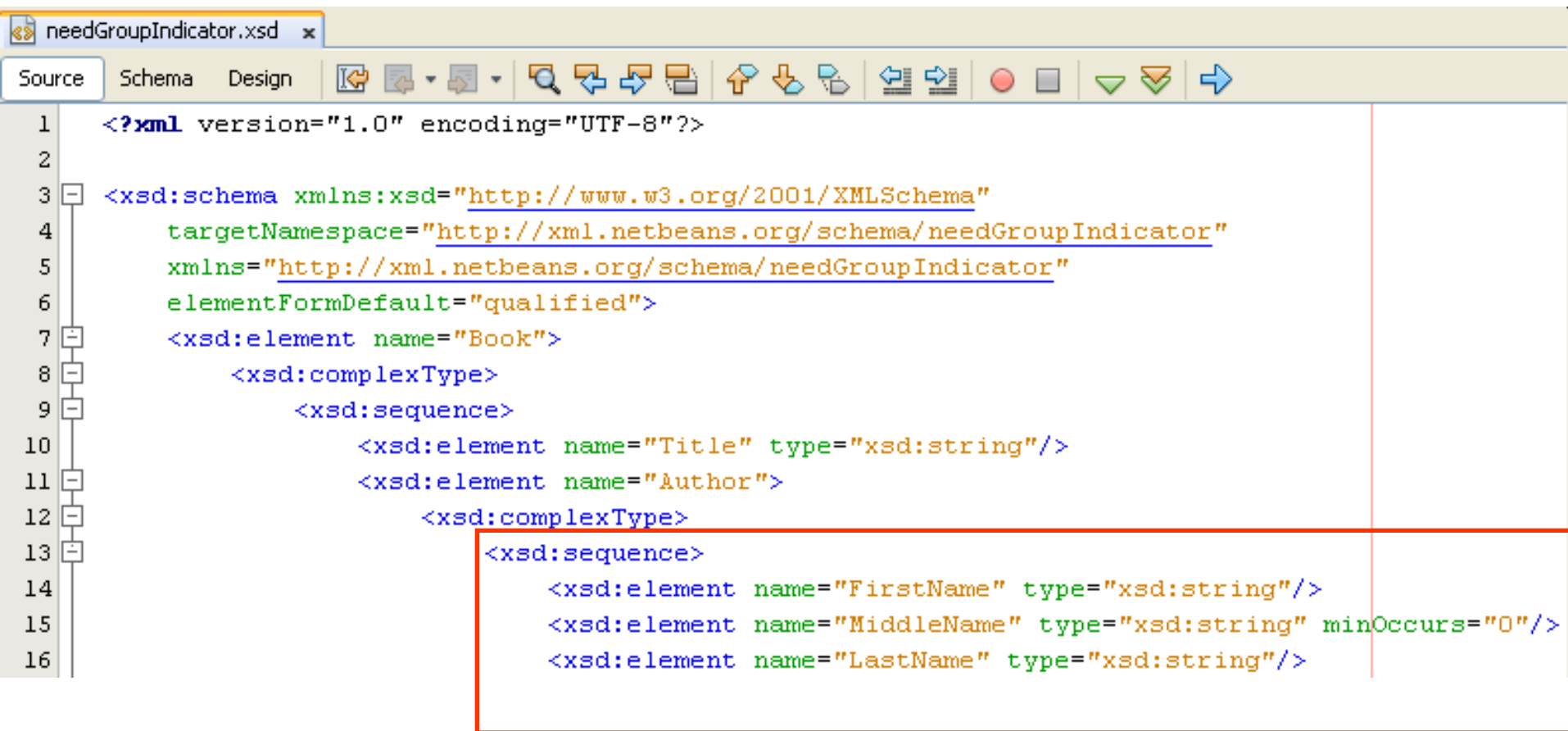
```

XML validation started.
E:/LapTrinh/Servlet/XML/Day3XML/src/occurrenceIndicator.xsd:14,28
ERROR: p-props-correct.2.1: In the declaration of 'element', the value of 'minO
ccurs' is '1', but it must not be greater than the value of 'maxOccurs', which
is '0'.

1 Error(s), 0 Warning(s).
XML validation finished.
  
```

Schemas

Structuring – Need for Group Indicators



The screenshot shows an XML Schema editor window titled 'needGroupIndicator.xsd'. The editor has tabs for 'Source', 'Schema', and 'Design'. The 'Source' tab is active, displaying the following XML Schema code:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    targetNamespace="http://xml.netbeans.org/schema/needGroupIndicator"
5    xmlns="http://xml.netbeans.org/schema/needGroupIndicator"
6    elementFormDefault="qualified">
7    <xsd:element name="Book">
8      <xsd:complexType>
9        <xsd:sequence>
10         <xsd:element name="Title" type="xsd:string"/>
11         <xsd:element name="Author">
12           <xsd:complexType>
13             <xsd:sequence>
14               <xsd:element name="FirstName" type="xsd:string"/>
15               <xsd:element name="MiddleName" type="xsd:string" minOccurs="0"/>
16               <xsd:element name="LastName" type="xsd:string"/>

```

The code defines an XML Schema for a 'Book' element. The 'Book' element is of type 'complexType' and contains a 'sequence' of elements: 'Title' (type 'xsd:string'), 'Author' (type 'complexType'), and 'FirstName' (type 'xsd:string'). The 'Author' element is also of type 'complexType' and contains a 'sequence' of elements: 'MiddleName' (type 'xsd:string', minOccurs='0') and 'LastName' (type 'xsd:string').

Schemas

Structuring – Need for Group Indicators

```

17 <xsd:element name="Specialty">
18   <xsd:simpleType>
19     <xsd:restriction base="xsd:string">
20       <xsd:enumeration value="Mystery"/>
21       <xsd:enumeration value="Humor"/>
22       <xsd:enumeration value="Horror"/>
23       <xsd:enumeration value="Childrens"/>
24     </xsd:restriction>
25   </xsd:simpleType>
26 </xsd:element>
27 </xsd:sequence>
28 <xsd:attribute name="Title">
29   <xsd:simpleType>
30     <xsd:restriction base="xsd:string">
31       <xsd:enumeration value="Mr."/>
32       <xsd:enumeration value="Ms."/>
33       <xsd:enumeration value="Dr."/>
34     </xsd:restriction>
35   </xsd:simpleType>
36 </xsd:attribute>
37 <xsd:attribute name="BirthYear" type="xsd:gYear"/>
38 </xsd:complexType>
39 </xsd:element>
  
```

Schemas

Structuring – Need for Group Indicators

```

40 <xsd:element name="Illustrator" minOccurs="0">
41   <xsd:complexType>
42     <xsd:sequence>
43       <xsd:element name="FirstName" type="xsd:string"/>
44       <xsd:element name="MiddleName" type="xsd:string" minOccurs="0"/>
45       <xsd:element name="LastName" type="xsd:string"/>
46     </xsd:sequence>
47     <xsd:attribute name="Title">
48       <xsd:simpleType>
49         <xsd:restriction base="xsd:string">
50           <xsd:enumeration value="Mr."/>
51           <xsd:enumeration value="Ms."/>
52           <xsd:enumeration value="Dr."/>
53         </xsd:restriction>
54       </xsd:simpleType>
55     </xsd:attribute>
56     <xsd:attribute name="BirthYear" type="xsd:gYear"/>
57   </xsd:complexType>
58 </xsd:element>
59 </xsd:sequence>
60 </xsd:complexType>
61 </xsd:element>
62 </xsd:schema>
  
```

Schemas

Structuring – Need for Group Indicators

```

needGroupIndicator.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Book xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/needGroupIndicator'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/needGroupIndicator needGroupIndicator.xsd'>
6      <Title>Winnie the Pooh</Title>
7      <Author Title="Mr." BirthYear="1882">
8          <FirstName>A.</FirstName>
9          <MiddleName>A.</MiddleName>
10         <LastName>Milne</LastName>
11         <Specialty>Childrens</Specialty>
12     </Author>
13     <Illustrator Title="Mr." BirthYear="1879">
14         <FirstName>Ernest</FirstName>
15         <MiddleName>H.</MiddleName>
16         <LastName>Shepard</LastName>
17     </Illustrator>
18 </Book>
  
```

Schemas

Structuring – Group Indicators

- Are used to **define related sets of elements and/or attributes**
- Can be used to create a set structure **for reuse**
- **Syntax**
 - **Element groups**

```
<xsd:group name="group_name">  
...  
</xsd:group>
```

 - An **all, choice, or sequence element** must be **defined inside** the group declaration
 - **Attribute groups**

```
<xsd:attributeGroup name="group_name">  
...  
</xsd:attributeGroup>
```
- **After the group have been defined, it can be referenced in another definition**
 - **Syntax:** `<xsd:group ref="group_name"/>` or `<xsd:attributeGroup ref=""/>`

```

groupIndicator.xsd x
Source Schema Design
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/groupIndicator"
5   xmlns="http://xml.netbeans.org/schema/groupIndicator"
6   elementFormDefault="qualified">
7   <xsd:group name="person">
8     <xsd:sequence>
9       <xsd:element name="FirstName" type="xsd:string"/>
10      <xsd:element name="MiddleName" type="xsd:string" minOccurs="0"/>
11      <xsd:element name="LastName" type="xsd:string"/>
12    </xsd:sequence>
13  </xsd:group>
14  <xsd:attributeGroup name="attGroupPerson">
15    <xsd:attribute name="Title">
16      <xsd:simpleType>
17        <xsd:restriction base="xsd:string">
18          <xsd:enumeration value="Mr."/>
19          <xsd:enumeration value="Ms."/>
20          <xsd:enumeration value="Dr."/>
21        </xsd:restriction>
22      </xsd:simpleType>
23    </xsd:attribute>
24    <xsd:attribute name="BirthYear" type="xsd:gYear"/>
25  </xsd:attributeGroup>
  
```

Structuring – Group Indicators – Example

```

26 <xsd:element name="Book">
27   <xsd:complexType>
28     <xsd:sequence>
29       <xsd:element name="Title" type="xsd:string"/>
30       <xsd:element name="Author">
31         <xsd:complexType>
32           <xsd:sequence>
33             <xsd:group ref="person"/>
34             <xsd:element name="Specialty">
35               <xsd:simpleType>
36                 <xsd:restriction base="xsd:string">
37                   <xsd:enumeration value="Mystery"/>
38                   <xsd:enumeration value="Humor"/>
39                   <xsd:enumeration value="Horror"/>
40                   <xsd:enumeration value="Childrens"/>
41                 </xsd:restriction>
42               </xsd:simpleType>
43             </xsd:element>
44           </xsd:sequence>
45           <xsd:attributeGroup ref="attGroupPerson"/>
46         </xsd:complexType>
47       </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
  
```

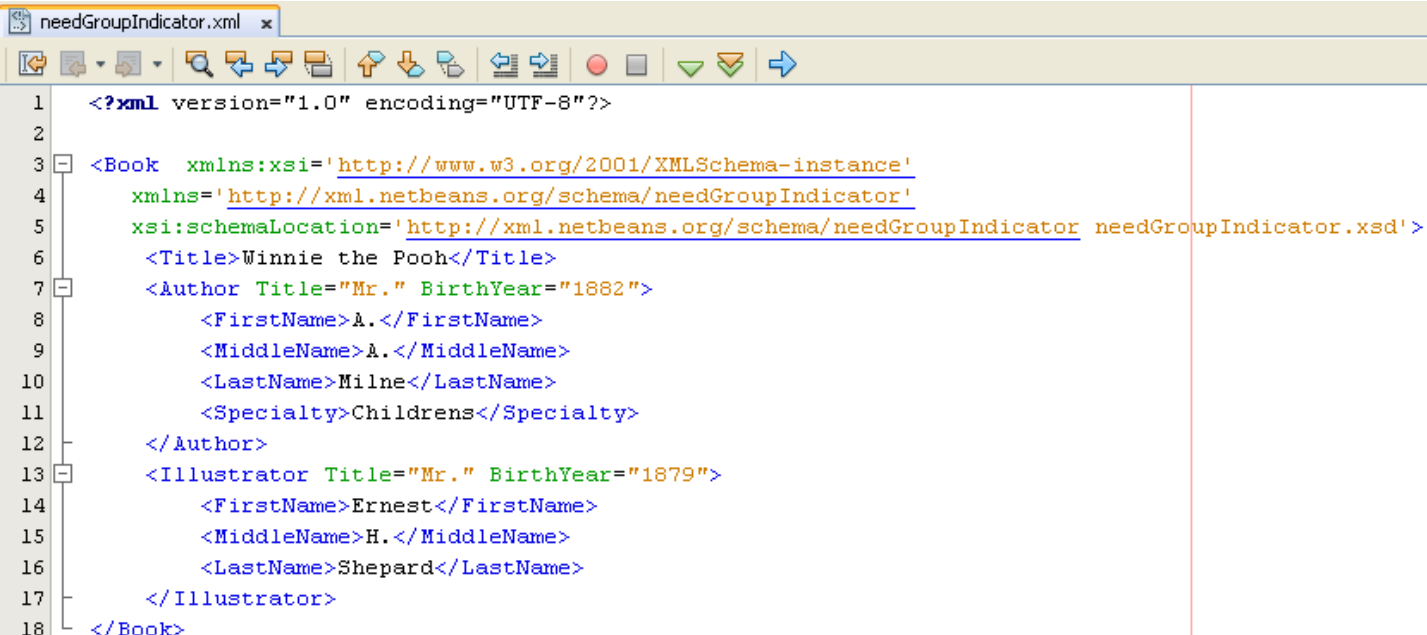
Schemas

Structuring – Group Indicators – Example

```

48 <xsd:element name="Illustrator" minOccurs="0">
49   <xsd:complexType>
50     <xsd:sequence>
51       <xsd:group ref="person"/>
52     </xsd:sequence>
53     <xsd:attributeGroup ref="attGroupPerson"/>
54   </xsd:complexType>
55 </xsd:element>
56 </xsd:sequence>
57 </xsd:schema>
58
59 </xsd:schema>

```



The screenshot shows an XML editor with two files open: 'needGroupIndicator.xml' and 'needGroupIndicator.xml'. The 'needGroupIndicator.xml' file contains the schema definition for the 'Illustrator' element, which is a complex type with a sequence of a group named 'person' and an attribute group named 'attGroupPerson'. The 'needGroupIndicator.xml' file contains an XML instance of a 'Book' element, which has a 'Title' of 'Winnie the Pooh', an 'Author' with 'Title' 'Mr.' and 'BirthYear' '1882', and an 'Illustrator' with 'Title' 'Mr.', 'BirthYear' '1879', 'FirstName' 'Ernest', 'MiddleName' 'H.', and 'LastName' 'Shepard'.

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <Book xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4       xmlns='http://xml.netbeans.org/schema/needGroupIndicator'
5       xsi:schemaLocation='http://xml.netbeans.org/schema/needGroupIndicator needGroupIndicator.xsd'>
6   <Title>Winnie the Pooh</Title>
7   <Author Title="Mr." BirthYear="1882">
8     <FirstName>A.</FirstName>
9     <MiddleName>A.</MiddleName>
10    <LastName>Milne</LastName>
11    <Specialty>Childrens</Specialty>
12  </Author>
13  <Illustrator Title="Mr." BirthYear="1879">
14    <FirstName>Ernest</FirstName>
15    <MiddleName>H.</MiddleName>
16    <LastName>Shepard</LastName>
17  </Illustrator>
18 </Book>

```

Schemas

Structuring – Extending Complex Type

- **New complex types can be derived by extending existing complex types**
 - **Both elements and attributes can be added in the new type, but nothing in the existing type can be overridden**
 - **New elements are appended to the content model**, such that the original elements and new elements act as two groups that must appear in sequence
- **Syntax**

```
<xsd:extension base="base_type">  
...  
</xsd:extension>
```


Schemas

Structuring–Extending Complex Type–Example

extending.xsd

Source Schema Design



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/extending"
5      xmlns="http://xml.netbeans.org/schema/extending"
6      elementFormDefault="qualified">
7      <xsd:group name="person">
8          <xsd:sequence>
13
14      </xsd:group>
15      <xsd:attributeGroup name="attGroupPerson">
16          <xsd:attribute
24              name="BirthYear" type="xsd:gYear"/>
25      </xsd:attributeGroup>
26      <xsd:complexType name="Person">
27          <xsd:sequence>
28              <xsd:group ref="person"/>
29          </xsd:sequence>
30          <xsd:attributeGroup ref="attGroupPerson"/>
31      </xsd:complexType>

```

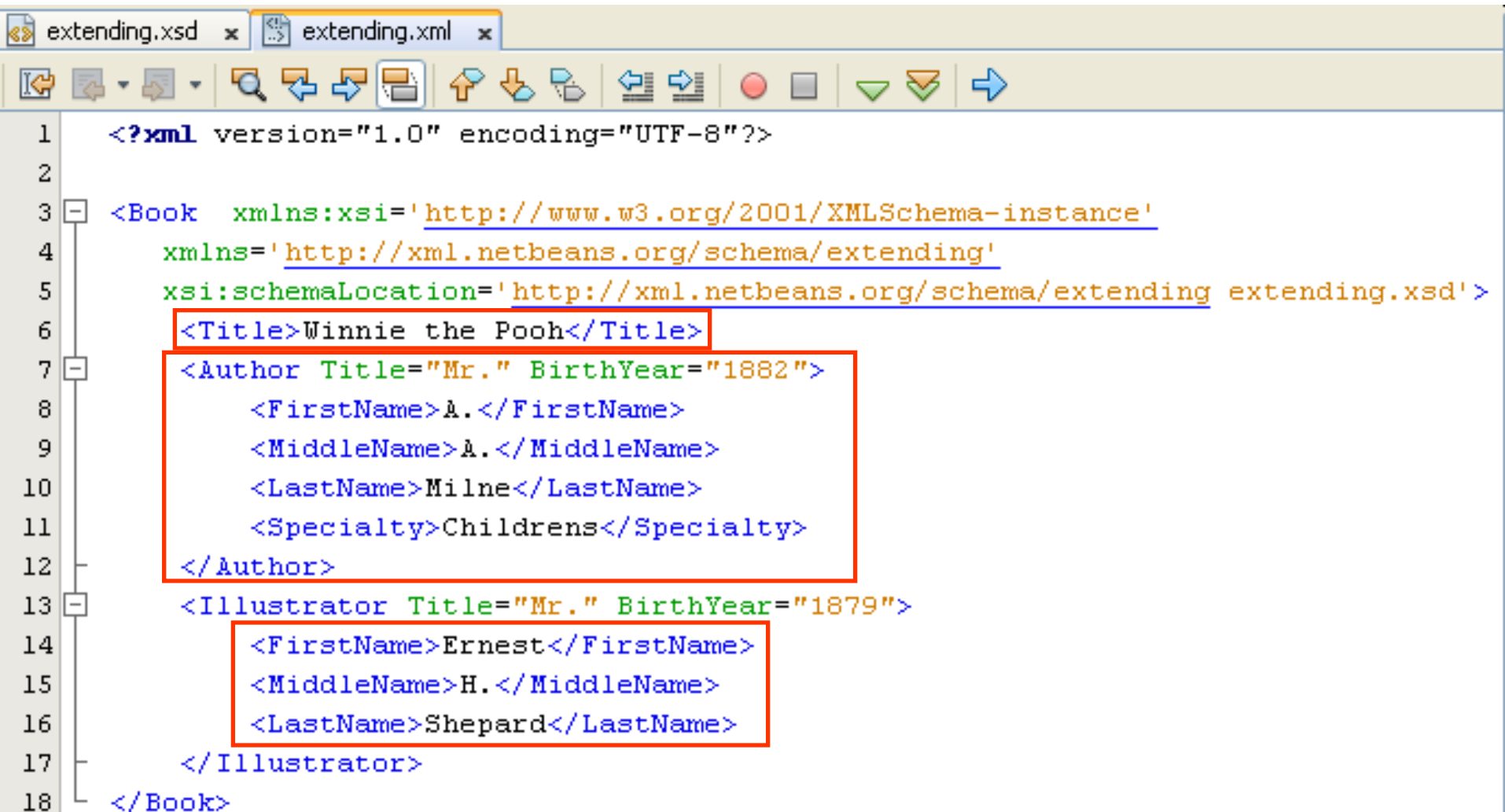
Schemas

Structuring–Extending Complex Type–Example

```

32 <xsd:complexType name="PersonExtended">
33   <xsd:complexContent>
34     <xsd:extension base="Person">
35       <xsd:sequence>
36         <xsd:element name="Specialty">
37           <xsd:simpleType>
38             <xsd:restriction base="xsd:string">
39               <xsd:enumeration value="Mystery"/>
40               <xsd:enumeration value="Humor"/>
41               <xsd:enumeration value="Horror"/>
42               <xsd:enumeration value="Childrens"/>
43             </xsd:restriction>
44           </xsd:simpleType>
45         </xsd:element>
46       </xsd:sequence>
47     </xsd:extension>
48   </xsd:complexContent>
49 </xsd:complexType>
50
51 <xsd:element name="Book">
52   <xsd:complexType>
53     <xsd:sequence>
54       <xsd:element name="Title" type="xsd:string"/>
55       <xsd:element name="Author" type="PersonExtended"/>
56       <xsd:element name="Illustrator" type="Person" minOccurs="0"/>
57     </xsd:sequence>
58   </xsd:complexType>
59 </xsd:element>
60 </xsd:schema>
  
```

Structuring–Extending Complex Type–Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Book  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns='http://xml.netbeans.org/schema/extending'
5        xsi:schemaLocation='http://xml.netbeans.org/schema/extending extending.xsd'>
6    <Title>Winnie the Pooh</Title>
7    <Author Title="Mr." BirthYear="1882">
8        <FirstName>A.</FirstName>
9        <MiddleName>A.</MiddleName>
10       <LastName>Milne</LastName>
11       <Specialty>Childrens</Specialty>
12    </Author>
13    <Illustrator Title="Mr." BirthYear="1879">
14        <FirstName>Ernest</FirstName>
15        <MiddleName>H.</MiddleName>
16        <LastName>Shepard</LastName>
17    </Illustrator>
18 </Book>
  
```

Schemas

Structuring – Abstract Types

- Is used to **define** the **arbitrary complex type** that will be **defined** at **used time**
- When a type is made abstract, it **cannot** be **used directly** in an XML instance
 - One of its **derived types** must be **used** instead
- The **derived** type is **identified** in the **instance** document using the **xsi:type attribute**
- **Syntax**
 - Abstract Type Declaration
<xsd:complexType name="name_type" abstract="true">
...
</xsd:complexType>
 - Derived Type Declaration from Abstract Type: using **extending complex type** mechanism
 - Using in XML document:
<element_Name xsi:type="derivedType">...</element_Name>

Structuring – Abstract Types – Example



The screenshot shows an XML Schema editor with a toolbar and a source code view. The code defines an XML Schema with a target namespace of `http://xml.netbeans.org/schema/abstractType`. It includes a complex type `Measurement` that extends `xsd:integer` and has an attribute `units` of type `xsd:string`. It also defines two elements: `Weight` of type `Measurement` and `Name` of type `xsd:string`. Finally, it defines an abstract complex type `Animal` with the attribute `abstract="true"`. The `Animal` type contains a sequence of two elements: `Name` (referenced by `ref="Name"`) and `Weight` (referenced by `ref="Weight"`).

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    targetNamespace="http://xml.netbeans.org/schema/abstractType"
5    xmlns="http://xml.netbeans.org/schema/abstractType"
6    elementFormDefault="qualified">
7    <xsd:complexType name="Measurement">
8      <xsd:simpleContent>
9        <xsd:extension base="xsd:integer">
10          <xsd:attribute name="units" type="xsd:string"/>
11        </xsd:extension>
12      </xsd:simpleContent>
13    </xsd:complexType>
14    <xsd:element name="Weight" type="Measurement"/>
15    <xsd:element name="Name" type="xsd:string"/>
16    <!-- Abstract Type -->
17    <xsd:complexType name="Animal" abstract="true">
18      <xsd:sequence>
19        <xsd:element ref="Name"/>
20        <xsd:element ref="Weight"/>
21      </xsd:sequence>
22    </xsd:complexType>
  
```

Structuring – Abstract Types – Example

```

23 <xsd:complexType name="Dog">
24   <xsd:complexContent>
25     <xsd:extension base="Animal"/>
26   </xsd:complexContent>
27 </xsd:complexType>
28 <xsd:complexType name="Bird">
29   <xsd:complexContent>
30     <xsd:extension base="Animal">
31       <xsd:sequence>
32         <xsd:element name="WingSpan" type="Measurement"/>
33       </xsd:sequence>
34     </xsd:extension>
35   </xsd:complexContent>
36 </xsd:complexType>
37 <xsd:element name="Animals">
38   <xsd:complexType>
39     <xsd:sequence>
40       <xsd:element name="Animal" type="Animal" maxOccurs="unbounded"/>
41     </xsd:sequence>
42   </xsd:complexType>
43 </xsd:element>
44 </xsd:schema>
  
```



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Animals xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/abstractType'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/abstractType abstractType.xsd'>
6      <Animal xsi:type="Dog">
7          <Name>Rover</Name>
8          <Weight units="pounds">80</Weight>
9      </Animal>
10     <Animal xsi:type="Bird">
11         <Name>Tweetie</Name>
12         <Weight units="grams">15</Weight>
13         <WingSpan units="cm">20</WingSpan>
14     </Animal>
15 </Animals>
  
```

Schemas

Structuring – Abstract Types – Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Animals xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/abstractType'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/abstractType abstractType.xsd'>
6      <Animal>
7          <Name>Rover</Name>
8          <Weight units="pounds">80</Weight>
9      </Animal>
10     <Animal xsi:type="Bird">
11         <Name>Tweety</Name>
12         <Weight units="grams">15</Weight>
13         <WingSpan units="cm">20</WingSpan>
14     </Animal>
15 </Animals>
  
```

Output - XML check

XML validation started.

Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/abstractType.xml...

Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/abstractType.xsd".

cvc-type.2: The type definition cannot be abstract for element Animal. [6]

XML validation finished.

Structuring – Abstract Types – Example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <Animals xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/abstractType'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/abstractType abstractType.xsd'>
6      <Animal xsi:type="Animal">
7          <Name>Rover</Name>
8          <Weight units="pounds">80</Weight>
9      </Animal>
10     <Animal xsi:type="Bird">
11         <Name>Tweetie</Name>
12         <Weight units="grams">15</Weight>
13         <WingSpan units="cm">20</WingSpan>
14     </Animal>
15 </Animals>
  
```

Output - XML check

XML validation started.

Checking file:/Z:/Laptrinh/Servlet/Day3XML/src/abstractType.xml...

Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3XML/src/abstractType.xsd".

cvc-type.2: The type definition cannot be abstract for element Animal. [6]

XML validation finished.

Schemas

Structuring – Any Elements

- **Allows to extend** the XML document with **elements** or **attributes not specified** by the schema
- **Allows documents to contain additional elements or attributes** that are **not declared** in the main XML schema
- **Syntax**
 - **Elements**
<xsd:any/>
 - **Attributes**
<xsd:anyAttribute/>

Schemas

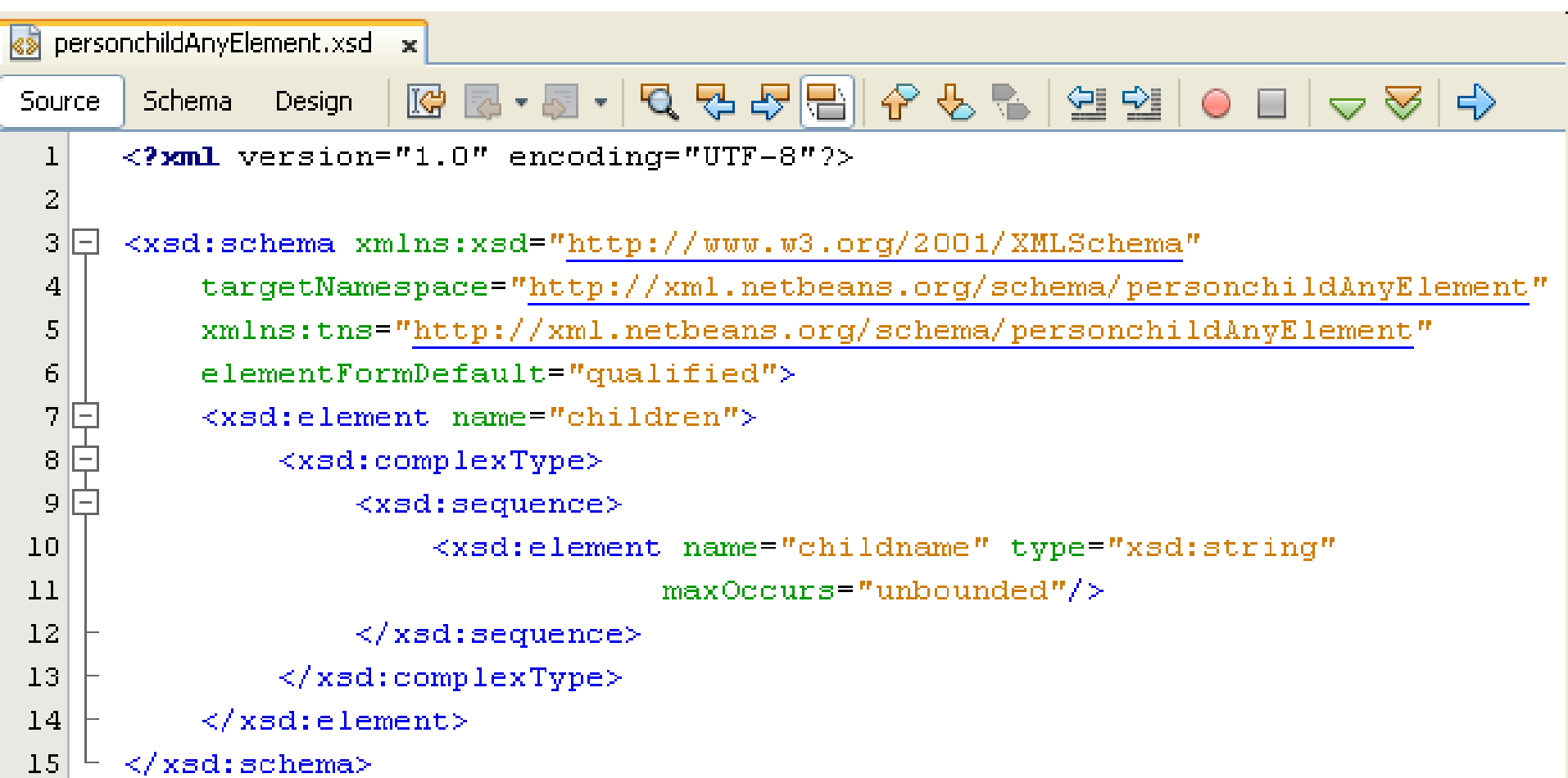
Structuring – Any Elements – Example

```

personAnyElement.xsd
Source Schema Design
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/personAnyElement"
5   xmlns="http://xml.netbeans.org/schema/personAnyElement"
6   elementFormDefault="qualified">
7   <xsd:element name="persons">
8     <xsd:complexType>
9       <xsd:sequence>
10        <xsd:element name="person" maxOccurs="unbounded">
11          <xsd:complexType>
12            <xsd:sequence>
13              <xsd:element name="firstname" type="xsd:string"/>
14              <xsd:element name="lastname" type="xsd:string"/>
15              <xsd:any minOccurs="0"/>
16            </xsd:sequence>
17          </xsd:complexType>
18        </xsd:element>
19      </xsd:sequence>
20    </xsd:complexType>
21  </xsd:element>
22 </xsd:schema>
  
```

Schemas

Structuring – Any Elements – Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/personchildAnyElement"
5      xmlns:tns="http://xml.netbeans.org/schema/personchildAnyElement"
6      elementFormDefault="qualified">
7      <xsd:element name="children">
8          <xsd:complexType>
9              <xsd:sequence>
10                 <xsd:element name="childname" type="xsd:string"
11                     maxOccurs="unbounded"/>
12             </xsd:sequence>
13         </xsd:complexType>
14     </xsd:element>
15 </xsd:schema>
  
```

Schemas

Structuring – Any Elements – Example

```

personAnyElement.xml x
[Icons]
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <persons  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/personAnyElement'
5      xmlns:ns1='http://xml.netbeans.org/schema/personchildAnyElement'
6      xsi:schemaLocation='http://xml.netbeans.org/schema/personAnyElement personAnyElement.xsd
7      http://xml.netbeans.org/schema/personchildAnyElement personchildAnyElement.xsd'>
8      <person>
9          <firstname>Hege</firstname>
10         <lastname>Refsnes</lastname>
11         <ns1:children>
12             <ns1:childname>Cecilie</ns1:childname>
13         </ns1:children>
14     </person>
15
16     <person>
17         <firstname>Stale</firstname>
18         <lastname>Refsnes</lastname>
19     </person>
20 </persons>
  
```

Schemas

Structuring – Any Elements – Example

```

personAnyElement.xsd
Source Schema Design
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4   targetNamespace="http://xml.netbeans.org/schema/personAnyElement"
5   xmlns="http://xml.netbeans.org/schema/personAnyElement"
6   elementFormDefault="qualified">
7   <xsd:element name="persons">
8     <xsd:complexType>
9       <xsd:sequence>
10        <xsd:element name="person" maxOccurs="unbounded">
11          <xsd:complexType>
12            <xsd:sequence>
13              <xsd:element name="firstname" type="xsd:string"/>
14              <xsd:element name="lastname" type="xsd:string"/>
15              <xsd:any minOccurs="0"/>
16            </xsd:sequence>
17            <xsd:anyAttribute/>
18          </xsd:complexType>
19        </xsd:element>
20      </xsd:sequence>
21    </xsd:complexType>
22  </xsd:element>
23 </xsd:schema>
  
```

Schemas

Structuring – Any Elements – Example

personAttrAnyElement.xsd

Source Schema Design

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/personAttrAnyElement"
5      xmlns="http://xml.netbeans.org/schema/personAttrAnyElement"
6      elementFormDefault="qualified">
7      <xsd:attribute name="gender">
8          <xsd:simpleType>
9              <xsd:restriction base="xsd:string">
10                 <xsd:pattern value="male|female"/>
11             </xsd:restriction>
12         </xsd:simpleType>
13     </xsd:attribute>
14 </xsd:schema>
  
```

Schemas

Structuring – Any Elements – Example

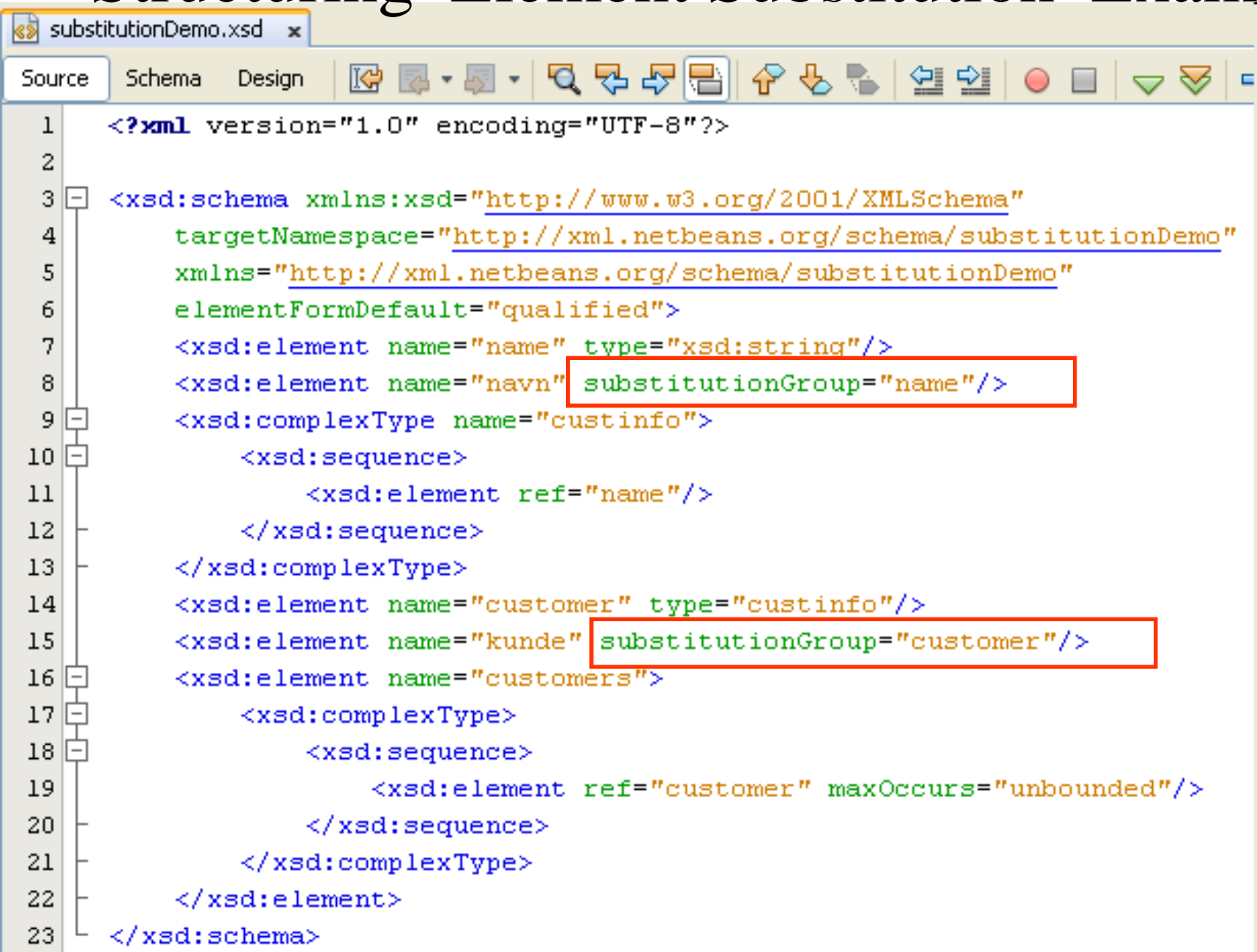
```

personAnyElement.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <persons  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/personAnyElement'
5      xmlns:ns1='http://xml.netbeans.org/schema/personchildAnyElement'
6      xmlns:ns2='http://xml.netbeans.org/schema/personAttrAnyElement'
7      xsi:schemaLocation='http://xml.netbeans.org/schema/personAnyElement personAnyElement.xsd
8      http://xml.netbeans.org/schema/personchildAnyElement personchildAnyElement.xsd
9      http://xml.netbeans.org/schema/personAttrAnyElement personAttrAnyElement.xsd'>
10     <person ns2:gender="male">
11         <firstname>Hege</firstname>
12         <lastname>Refsnes</lastname>
13         <ns1:children>
14             <ns1:childname>Cecilie</ns1:childname>
15         </ns1:children>
16     </person>
17
18     <person ns2:gender="female">
19         <firstname>Stale</firstname>
20         <lastname>Refsnes</lastname>
21     </person>
22 </persons>
  
```


Structuring – XSD Element Substitution

- One element can **substitute** another element
- A **substitutionGroup** is a solution
 - First, a **head element** is declared
 - Then, the **other elements** which **state** that they are **substitutable** for the **head element** are declared
- **Syntax**
 - **First:** declare the **key element**
`<xsd:element name="element_name" type="data_type" [block="substitution"]/>`
 - **block** keyword is used to block element substitution
 - **Second:** declare the **substitution element** with **substitutionGroup** attribute
`<xsd:element name="alternative_name" substitutionGroup="element_name" />`
- **Notes**
 - The **type** of the **substitutable elements** must be the **same as**, or **derived from**, the type of the head element.
 - All elements in the **substitutionGroup** (the head element and the substitutable elements) must be **declared** as **global elements**

Structuring–Element Substitution–Example



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/substitutionDemo"
5      xmlns="http://xml.netbeans.org/schema/substitutionDemo"
6      elementFormDefault="qualified">
7      <xsd:element name="name" type="xsd:string"/>
8      <xsd:element name="navn" substitutionGroup="name"/>
9      <xsd:complexType name="custinfo">
10         <xsd:sequence>
11             <xsd:element ref="name"/>
12         </xsd:sequence>
13     </xsd:complexType>
14     <xsd:element name="customer" type="custinfo"/>
15     <xsd:element name="kunde" substitutionGroup="customer"/>
16     <xsd:element name="customers">
17         <xsd:complexType>
18             <xsd:sequence>
19                 <xsd:element ref="customer" maxOccurs="unbounded"/>
20             </xsd:sequence>
21         </xsd:complexType>
22     </xsd:element>
23 </xsd:schema>
  
```

Structuring–Element Substitution–Example

```

substitutionDemo.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <customers xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/substitutionDemo'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/substitutionDemo substitutionDemo.xsd'>
6      <customer>
7          <name>KhanhKT</name>
8      </customer>
9      <kunde>
10         <navn>KhanhKK</navn>
11     </kunde>
12 </customers>
  
```

Structuring–Element Substitution–Example

substitutionDemo.xsd

Source Schema Design

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/substitutionDemo"
5      xmlns="http://xml.netbeans.org/schema/substitutionDemo"
6      elementFormDefault="qualified">
7      <xsd:element name="name" type="xsd:string" block="substitution"/>
8      <xsd:element name="navn" substitutionGroup="name"/>
9      <xsd:complexType>
14     <xsd:element name="customer" type="custinfo"/>
15     <xsd:element name="kunde" substitutionGroup="customer"/>
16     <xsd:element name="customers">
17         <xsd:complexType>
22         </xsd:element>
23 </xsd:schema>
  
```

Structuring–Element Substitution–Example

```

substitutionDemo.xml x
[Icons]
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <customers  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/substitutionDemo'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/substitutionDemo substitutionDemo.xsd'>
6      <customer>
7          <name>KhanhKT</name>
8      </customer>
9      <kunde>
10         <navn>KhanhKK</navn>
11     </kunde>
12 </customers>
  
```

Output - XML check

```

XML validation started.
Checking file:/Z:/Laptrinh/Servlet/Day3/XML/src/substitutionDemo.xml...
Referenced entity at "file:/Z:/Laptrinh/Servlet/Day3/XML/src/substitutionDemo.xsd".
cvc-complex-type.2.4.a: Invalid content was found starting with element 'navn'. One of '{"http://xml.netbeans.org/s
chema/substitutionDemo":name}' is expected. [10]
XML validation finished.
  
```

Structuring–Element Substitution–Example

```

substitutionDemo.xml x
[Icons]
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <customers  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4      xmlns='http://xml.netbeans.org/schema/substitutionDemo'
5      xsi:schemaLocation='http://xml.netbeans.org/schema/substitutionDemo substitutionDemo.xsd' >
6      <customer>
7          <name>KhanhKT</name>
8      </customer>
9      <kunde>
10         <name>KhanhKK</name>
11     </kunde>
12 </customers>
  
```

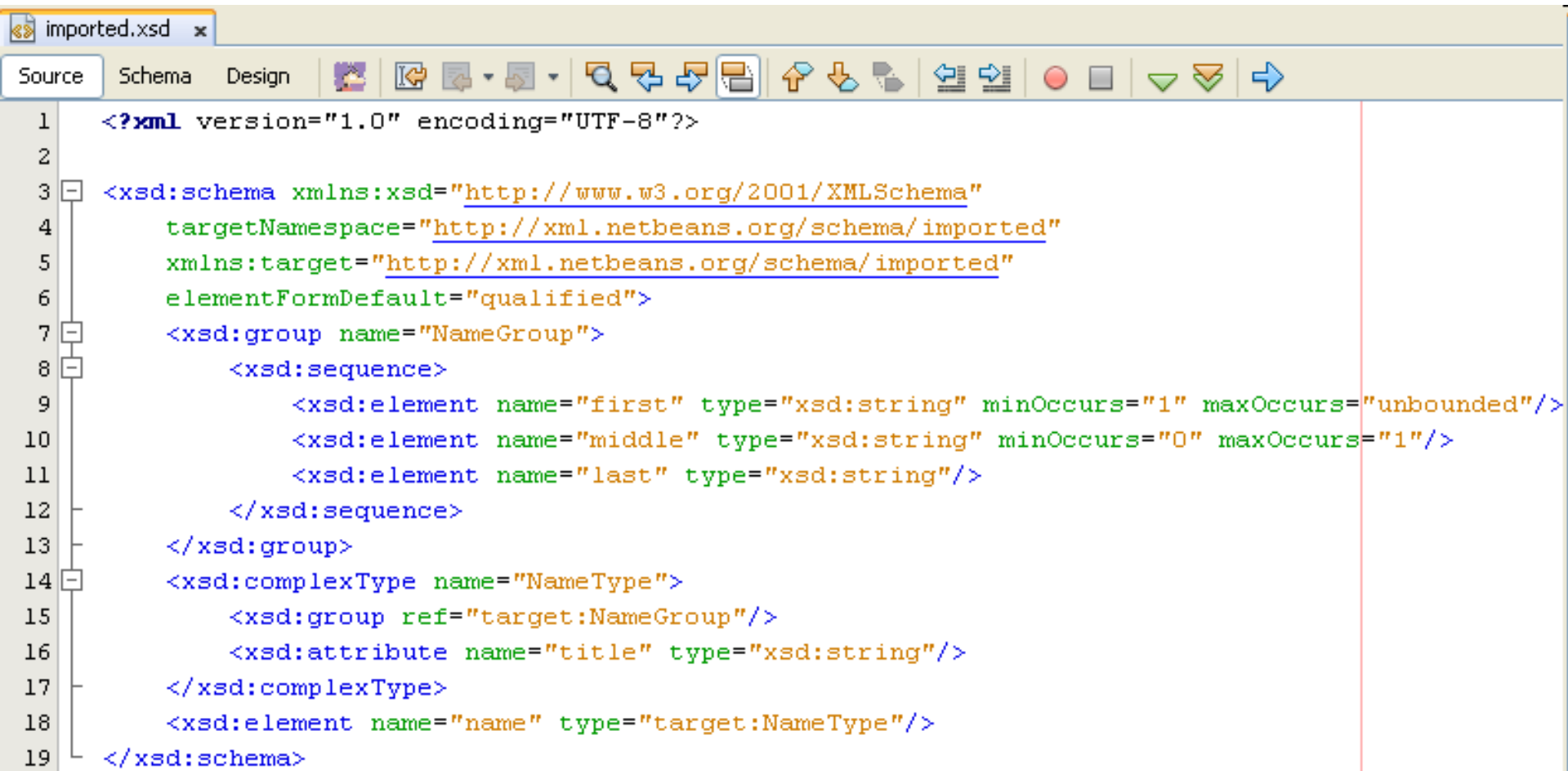
Schemas

Creating schema from multiple documents

- This is **mechanisms** for **combining XML Schemas** and **reusing definitions**
- **Import declaration**
 - Allows to **import global declarations** from other XML Schemas.
 - Is used **primarily** for **combining XML Schemas** that have **different targetNamespaces**
 - Allows to **refer to declarations only within** other XML Schemas
 - **Must be a direct child** of the <schema> element
 - **Syntax:** `<xsd:import namespace="URI" schemaLocation="url"/>`
- **Include declaration**
 - Is very **similar** to the **import** declaration
 - Allows to **combine XML Schemas** that are designed for the **same targetNamespace** (or no targetNamespace) much more effectively
 - **Syntax:** `<xsd:include schemaLocation="url"/>`
 - Can be used only on **documents** with the **same targetNamespace**, or **no targetNamespace**

Schemas

Creating schema from multiple documents



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/imported"
5      xmlns:target="http://xml.netbeans.org/schema/imported"
6      elementFormDefault="qualified">
7      <xsd:group name="NameGroup">
8          <xsd:sequence>
9              <xsd:element name="first" type="xsd:string" minOccurs="1" maxOccurs="unbounded"/>
10             <xsd:element name="middle" type="xsd:string" minOccurs="0" maxOccurs="1"/>
11             <xsd:element name="last" type="xsd:string"/>
12          </xsd:sequence>
13      </xsd:group>
14      <xsd:complexType name="NameType">
15          <xsd:group ref="target:NameGroup"/>
16          <xsd:attribute name="title" type="xsd:string"/>
17      </xsd:complexType>
18      <xsd:element name="name" type="target:NameType"/>
19  </xsd:schema>
  
```


Creating schema from multiple documents

```

importDemo.xsd
Source Schema Design
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4     targetNamespace="http://xml.netbeans.org/schema/importDemo"
5     xmlns="http://xml.netbeans.org/schema/importDemo"
6     xmlns:target="http://xml.netbeans.org/schema/imported"
7     elementFormDefault="qualified">
8     <xsd:import namespace="http://xml.netbeans.org/schema/imported" schemaLocation="imported.xsd"/>
9     <xsd:element name="contacts">
10         <xsd:complexType>
11             <xsd:sequence>
12                 <xsd:element name="contact" minOccurs="0" maxOccurs="unbounded">
13                     <xsd:complexType>
14                         <xsd:sequence>
15                             <xsd:element ref="target:name"/>
16                             <xsd:element name="location" type="xsd:string"/>
17                             <xsd:element name="phone" type="xsd:string"/>
18                             <xsd:element name="knows" type="xsd:string"/>
19                             <xsd:element name="description" type="xsd:string"/>
20                         </xsd:sequence>
21                         <xsd:attribute name="person" type="xsd:string"/>
22                         <xsd:attribute name="tags" type="xsd:token"/>
23                     </xsd:complexType>
24                 </xsd:element>
25             </xsd:sequence>
26         </xsd:complexType>
27     </xsd:element>
28 </xsd:schema>
  
```

Schemas

Creating schema from multiple documents

importDemo.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <contacts xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns='http://xml.netbeans.org/schema/importDemo'
5        xmlns:ns1="http://xml.netbeans.org/schema/imported"
6        xsi:schemaLocation='http://xml.netbeans.org/schema/importDemo importDemo.xsd'>
7    <contact person="John" tags="456">
8        <ns1:name title="Mr.">
9            <ns1:first>David</ns1:first>
10           <ns1:last>John</ns1:last>
11        </ns1:name>
12        <location>234 Phan Van Tri</location>
13        <phone>01276543</phone>
14        <knows>3</knows>
15        <description>Developer</description>
16    </contact>
17 </contacts>

```

Output - XML check

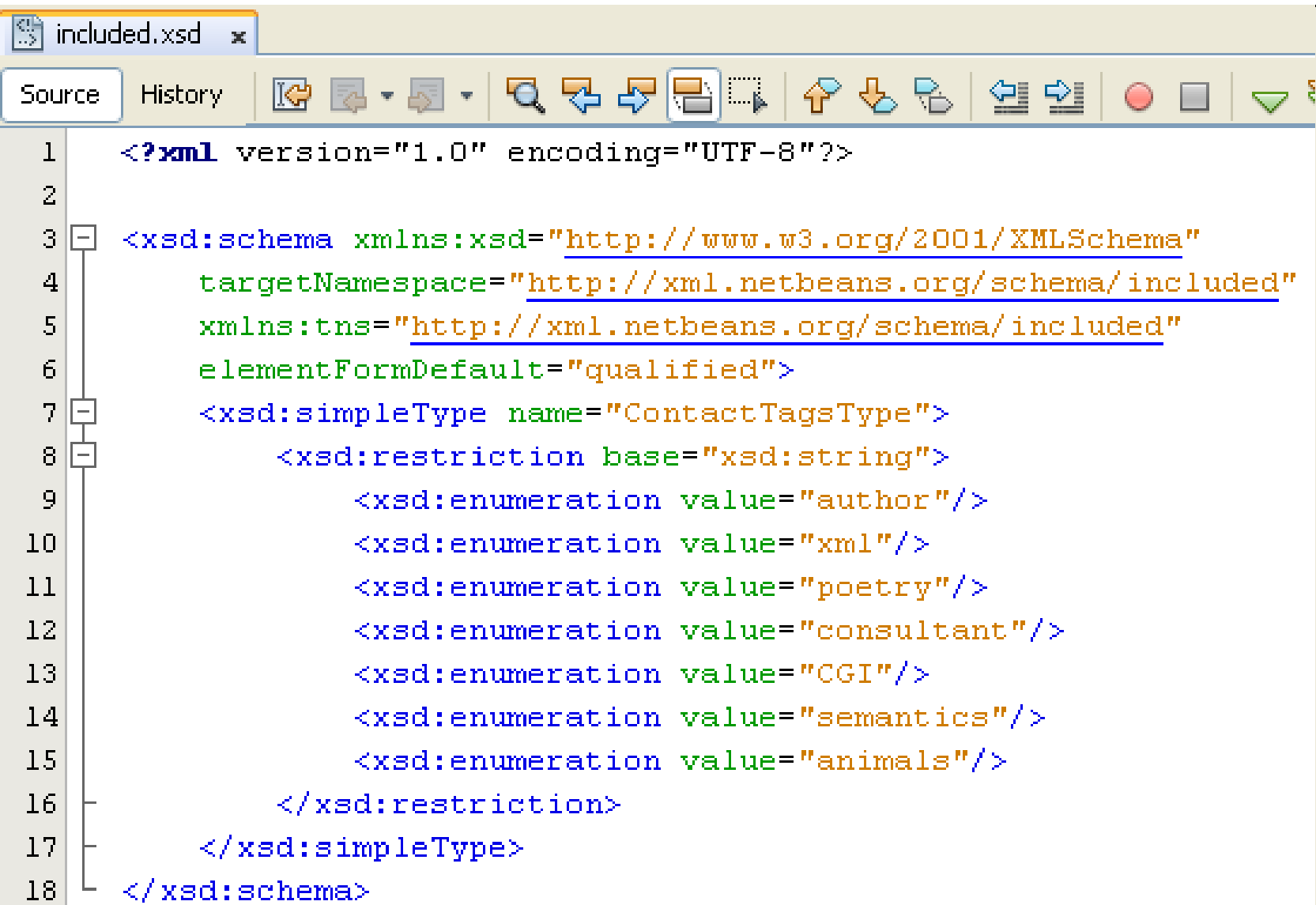
```

XML validation started.
Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/importDemo.xml...
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/importDemo.xsd".
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/imported.xsd".
XML validation finished.

```

Schemas

Creating schema from multiple documents



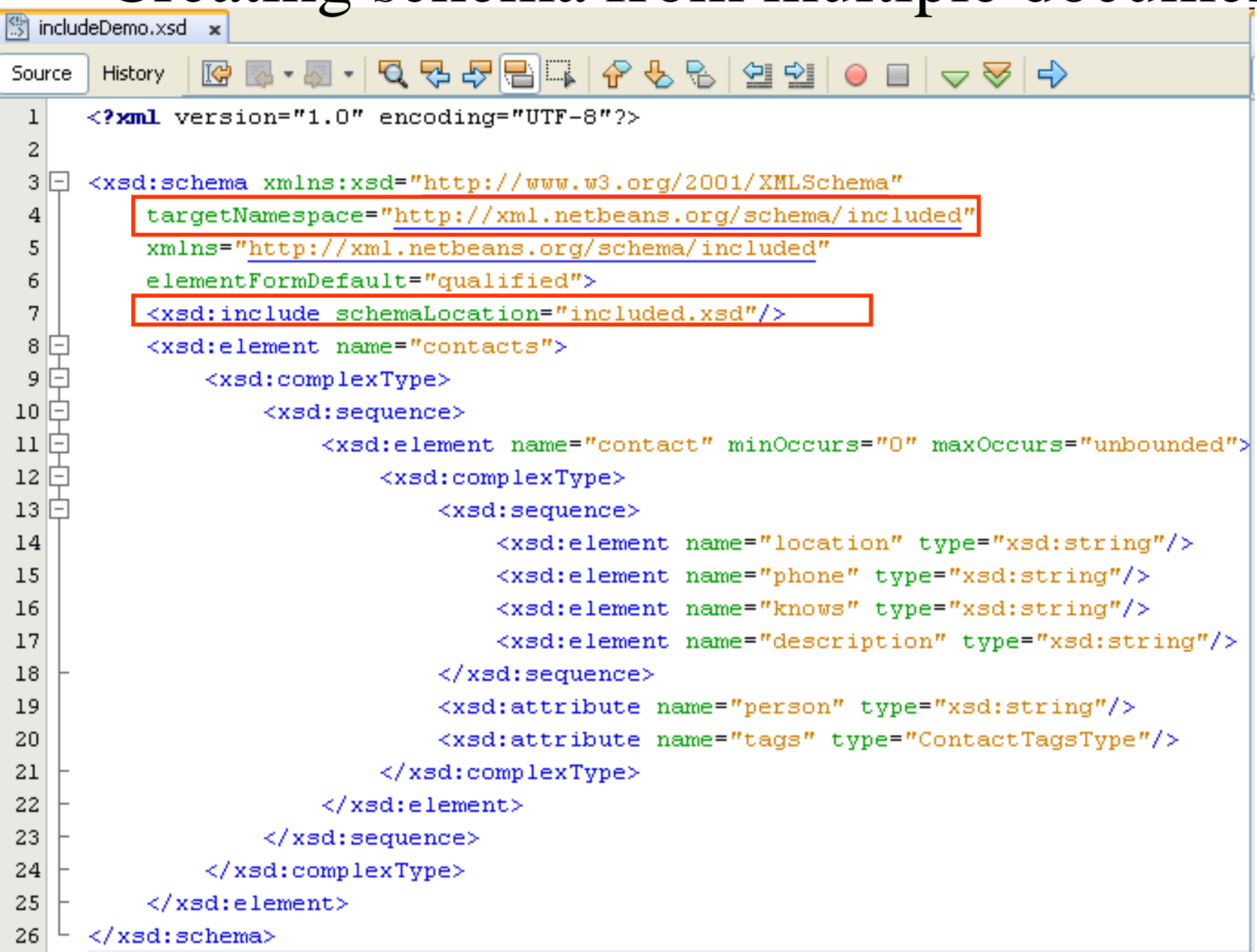
The screenshot shows an XML editor window with a tab titled 'included.xsd'. The editor has a 'Source' tab selected and a toolbar with various icons. The XML code is as follows:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4      targetNamespace="http://xml.netbeans.org/schema/included"
5      xmlns:tns="http://xml.netbeans.org/schema/included"
6      elementFormDefault="qualified">
7      <xsd:simpleType name="ContactTagsType">
8          <xsd:restriction base="xsd:string">
9              <xsd:enumeration value="author"/>
10             <xsd:enumeration value="xml"/>
11             <xsd:enumeration value="poetry"/>
12             <xsd:enumeration value="consultant"/>
13             <xsd:enumeration value="CGI"/>
14             <xsd:enumeration value="semantics"/>
15             <xsd:enumeration value="animals"/>
16         </xsd:restriction>
17     </xsd:simpleType>
18 </xsd:schema>
  
```

Schemas

Creating schema from multiple documents

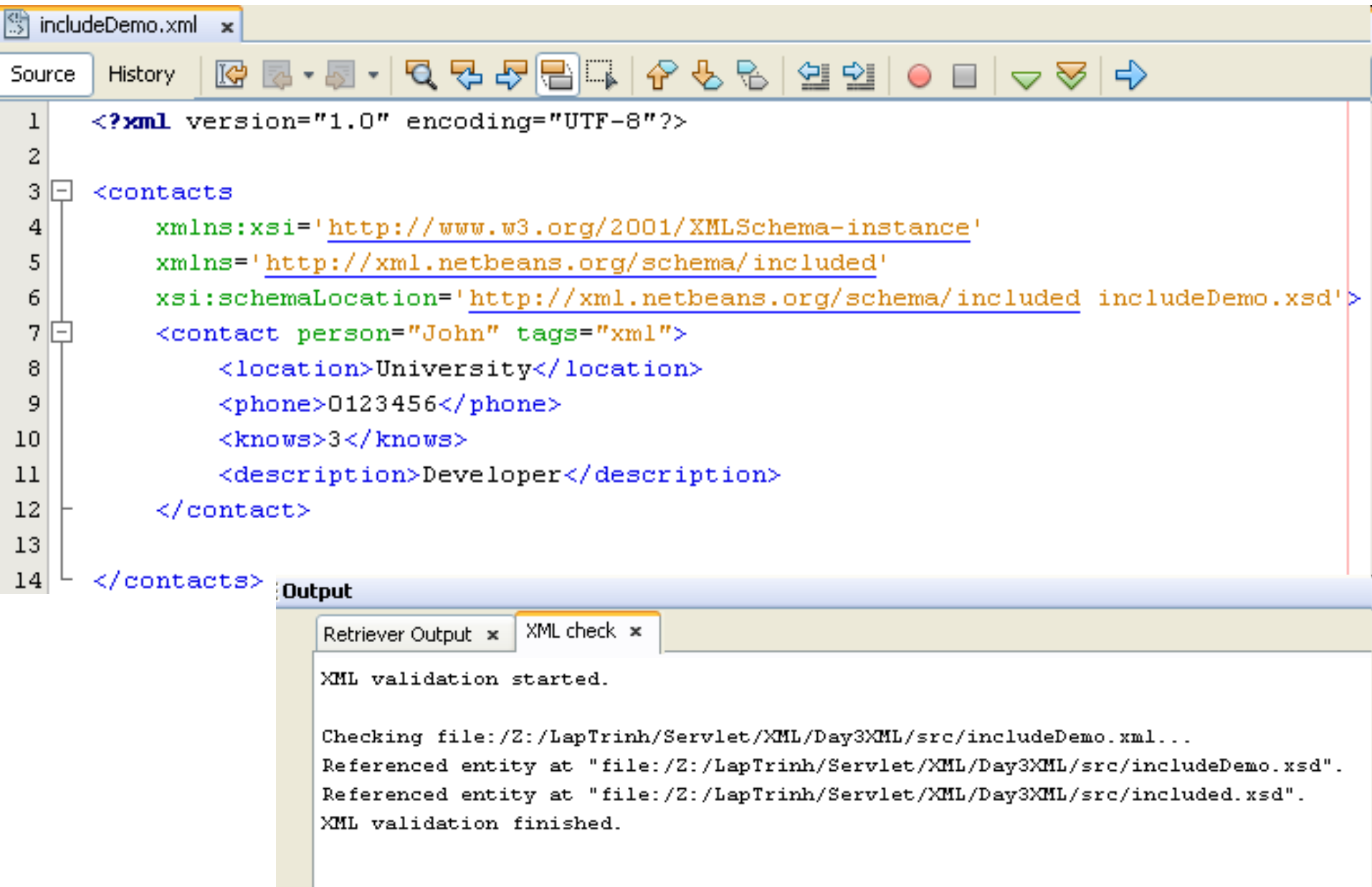


```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4    targetNamespace="http://xml.netbeans.org/schema/included"
5    xmlns="http://xml.netbeans.org/schema/included"
6    elementFormDefault="qualified">
7    <xsd:include schemaLocation="included.xsd"/>
8    <xsd:element name="contacts">
9      <xsd:complexType>
10        <xsd:sequence>
11          <xsd:element name="contact" minOccurs="0" maxOccurs="unbounded">
12            <xsd:complexType>
13              <xsd:sequence>
14                <xsd:element name="location" type="xsd:string"/>
15                <xsd:element name="phone" type="xsd:string"/>
16                <xsd:element name="knows" type="xsd:string"/>
17                <xsd:element name="description" type="xsd:string"/>
18              </xsd:sequence>
19              <xsd:attribute name="person" type="xsd:string"/>
20              <xsd:attribute name="tags" type="ContactTagsType"/>
21            </xsd:complexType>
22          </xsd:element>
23        </xsd:sequence>
24      </xsd:complexType>
25    </xsd:element>
26  </xsd:schema>
  
```

Schemas

Creating schema from multiple documents



The screenshot shows an IDE window titled 'includeDemo.xml' with the following XML code:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <contacts
4      xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
5      xmlns='http://xml.netbeans.org/schema/included'
6      xsi:schemaLocation='http://xml.netbeans.org/schema/included includeDemo.xsd'>
7      <contact person="John" tags="xml">
8          <location>University</location>
9          <phone>0123456</phone>
10         <knows>3</knows>
11         <description>Developer</description>
12     </contact>
13
14 </contacts>
  
```

Below the code editor is an 'Output' window with two tabs: 'Retriever Output' and 'XML check'. The 'XML check' tab is active and displays the following text:

```

XML validation started.

Checking file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/includeDemo.xml...
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/includeDemo.xsd".
Referenced entity at "file:/Z:/LapTrinh/Servlet/XML/Day3XML/src/included.xsd".
XML validation finished.
  
```

Summary

- **Introduction**
- **XSD Elements and Attributes**
- **XSD Data Types**
- **XSD Facets**
- **XSD Indicators**
- **Extending XSD**

Q&A

Next Lecture

- **Cascading Style Sheet (CSS)**
- **Extensible Style Language (XSL)**