PRACTICAL NO. 10

Aim:

Creating basic XML file for storing basic information of student.

Theory:

10.1 XML File

XML is a file extension for an Extensible Markup Language(XML) file format used to create common information formats and share both the format and the data on the World Wide Web, intranets, and elsewhere using standard ASCII text.

XML is similar to HTML. Both XML and HTML contain markup symbols to describe the contents of a page or file. HTML, however, describes the content of a Web page (mainly text and graphic images) only in terms of how it is to be displayed and interacted with. For example, the letter "p" placed within markup tags starts a new paragraph.

XML describes the content in terms of what data is being described. For example, the word "phoneme" placed within markup tags could indicate that the data that followed was a phone number. An XML file can be processed purely as data by a program or it can be stored with similar data on another computer or it can be displayed, like an HTML file. For example, depending on how the application in the receiving computer wanted to handle the phone number, it could be stored, displayed, or dialed.

XML is considered extensible because, unlike HTML, the markup symbols are unlimited and self-defining. XML is a simpler and easier-to-use subset of the Standard Generalized Markup Language (SGML) standard for how to create a document structure. It is expected that HTML and XML will be used together in many Web applications. XML markup, for example, may appear within an HTML page.

XML Introduction

A markup language is a mechanism to identify structures in a document. XML (Extensible Markup language) is a markup language for documents containing structured information. As a meta-language, XML is about to describe data and its structure; while HTML is about to display the data.XML is not to replace HTML, but compliment it by allowing writers to create and format their own document markups. For HTML, The tags used to mark up HTML documents and the structure of HTML documents are predefined. For XML, there is no predefined tag set, so there is no any preconceived semantics. Writers are provided a facility to define tags and the structural relationships. That is the extensible feature of XML.

To be valid, XML documents should be well-formed. That means XML documents follow some strict rules. For example, every opening tag in XML documents must have a matching closing tag. While in HTML, pairing is optional for some of the tags, and only the opening tag is required.

Before providing some examples of setting up XML documents, lets take a look of some rules for composing XML documents.

XML Tree Structure

XML document are formed as element tree.

An XML tree start at a root element and branches from the root to child elements.

The terms parent, child and sibling are used to describe the relationship between elements.

Parent have children. Every children have parent. Sibling are children on same level (brothers and sisters)

```
<root>
<child>
        <subchild>
</subchild>
</child>
</root>
```

In above example <root> is parent element

Pre-defined Entity reference in XML

<	<	less than
>	>	greater than
&	&	ampersand
&apos	‘	apostrophe
&quot	“	quotation mark

The XML specification does not use the term “character entity” or “character entity reference”. The XML specification defines five “predefined entities” representing special characters and requires that all XML processor honor them The entities can be explicitly declared in a DTD as well, but if this is done, the replacement text must be the same as the built-in-definitions. XML also allows other named entities of any size to be defined on a per-document basis.

Some Rules

1. Closing Tags An element in XML must have a closing tag.
For example:

```
<firstname>James</firstname>
```

2. Overlapping Elements An element must be nested properly by closing all child elements before closing the parent elements.
For example:

```
<student>
<firstname>James</firstname>
<lastname>Smith</lastname>
</student>
```

It would be incorrectly nested if the student element was closed before forename element.

3. Single Root Element An XML document only can have a single root element. All other elements should be within this root element.
For example:

```
<mec>
<student>
<firstname>James</firstname>
<lastname>Smith</lastname>
</student>
</mec>
```

As above, mec is the root element, and student is its descendant.

4. Case Sensitive XML is case sensitive, and care should be taken to ensure that Opening and closing tags are in the same case. An element name of "MEC" is not the same as "mec".

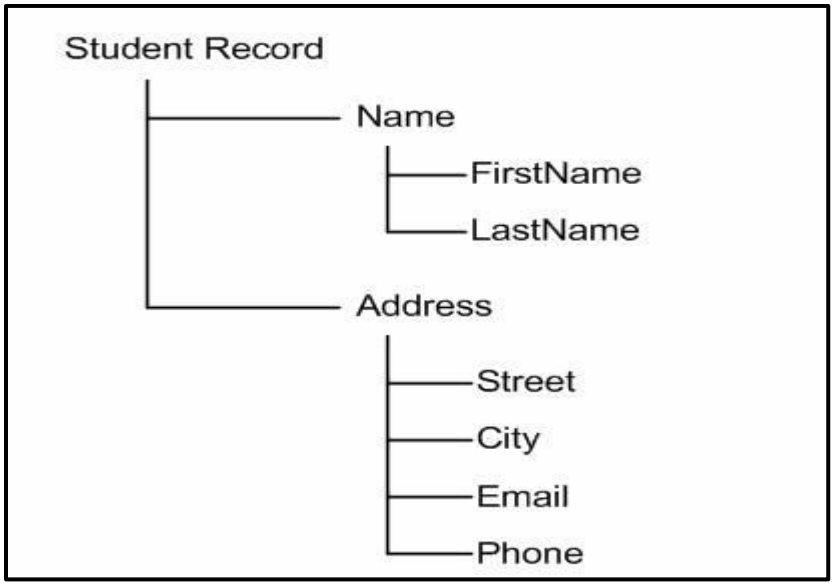
5. Quoted Attribute Values In XML, attributes must be specified a value, and the values must be in quotes, regardless of the data type. Single and double quotes are both acceptable.
For example:
Correct: <input type="checkbox" checked="checked"/>
Wrong: <input type="checkbox" checked>

6. Naming Conventions In XML, since there is no any pre-defined tag, there are no reserved words needed to be avoided when naming elements. The following are some simple rules when naming an element.

a. Elements must start with a character or an underscore but not a number or punctuation. After the first character, numbers, hyphens and periods are allowed.
b. Names must not contain spaces.
c. A colon is reserved for namespaces

Tutorial Examples

1. Create a XML file
   Now we use XML to create a student record database for the student information management. Each student record includes students name and address. The name has two parts: First name and last name. The address has four parts: Street, City, Email and Phone Number. So the structure of the student record can be presented as:

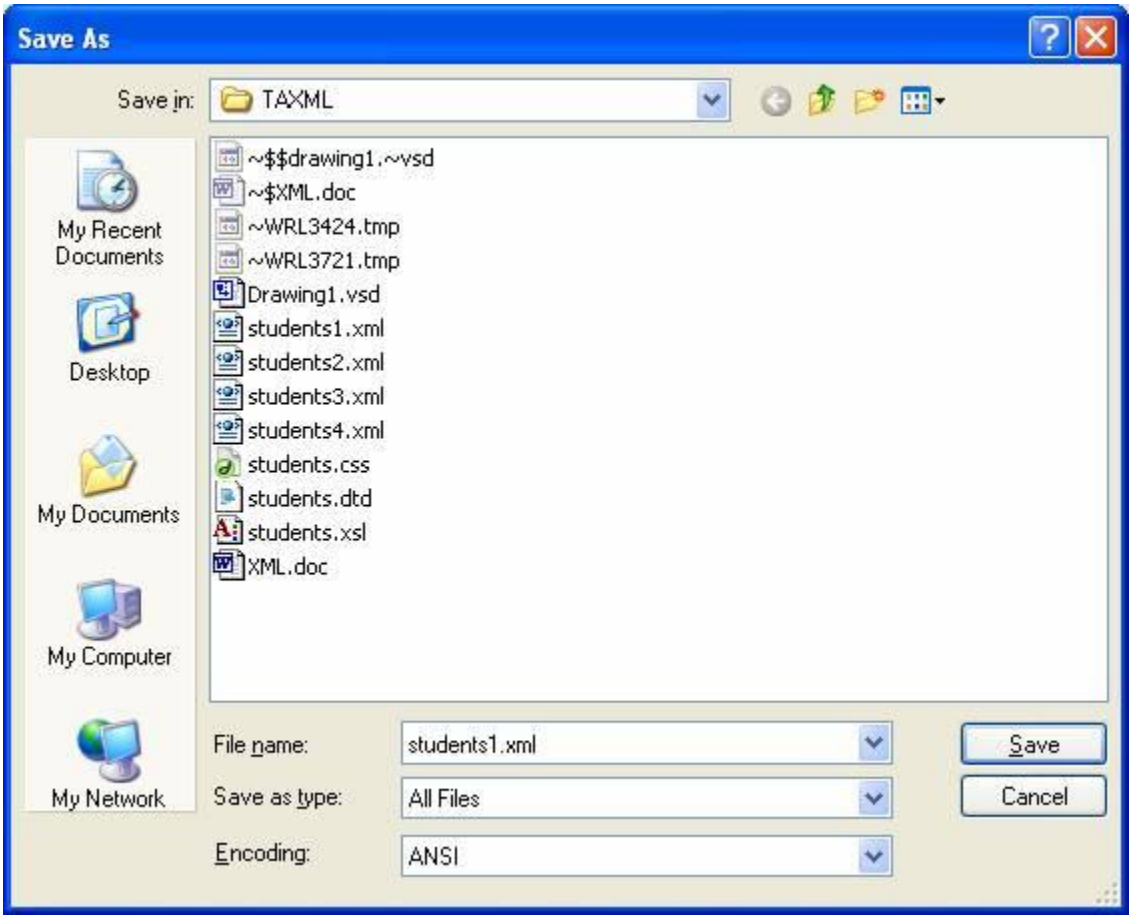


Using notepad.exe as our editor tool, we can create the xml file students1.xml. In this file, we input two students information.

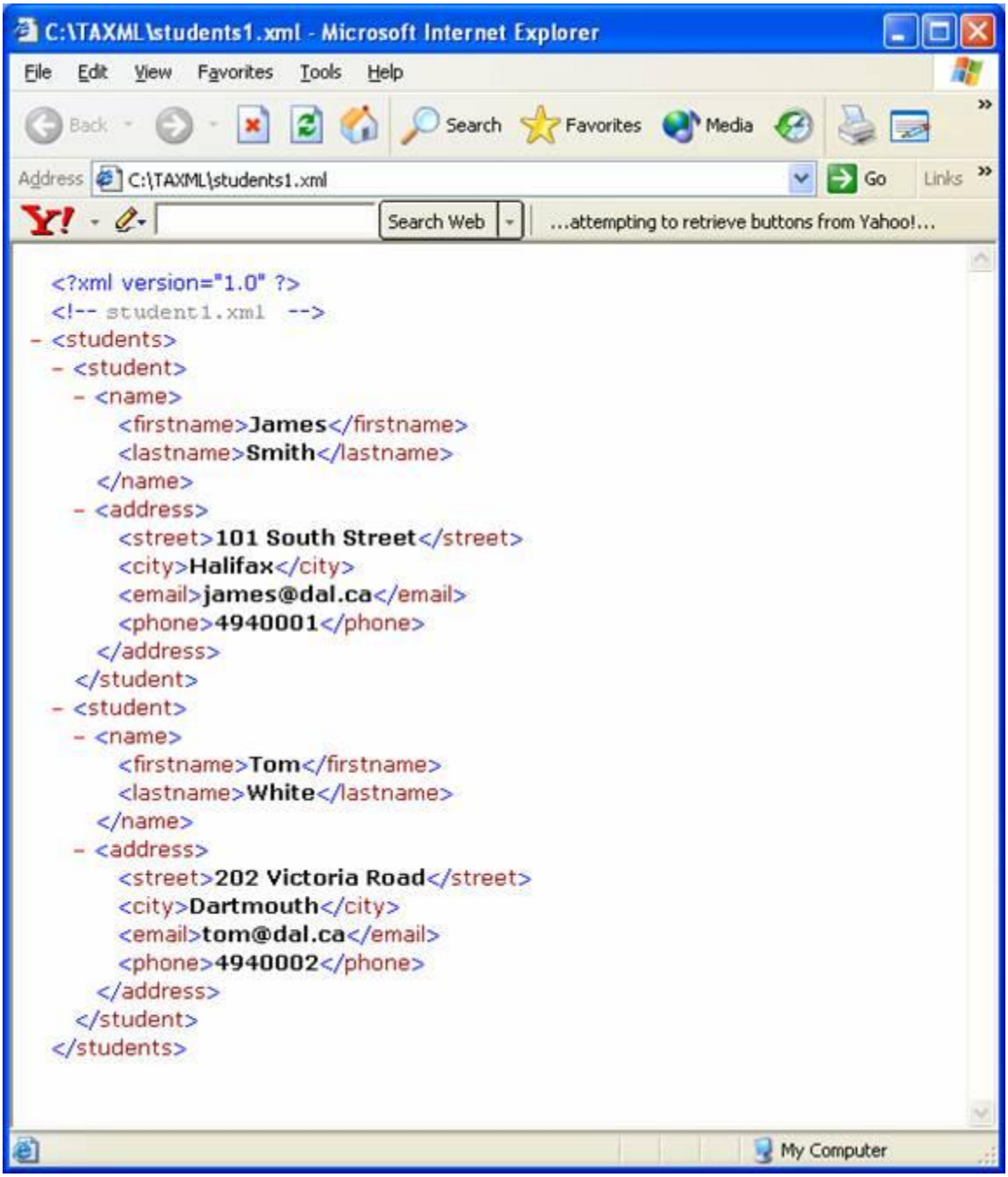
```
<?xml version = "1.0"?>
<!--student1.xml-->
<students>
<student>
<name>
<firstname> James </firstname>
<lastname> Smith </lastname>
</name>
<address>
<street> 101 South Street</street>
<city> Halifax </city>
<email> james@dal.ca </email>
<phone> 4940001 </phone>
</address>
</student>

<student>
<name>
<firstname> Tom </firstname>
<lastname> White </lastname>
</name>
<address>
<street> 202 Victoria Road </street>
<city> Dartmouth </city>
<email> tom@dal.ca</email>
<phone> 4940002 </phone>
</address>
</student>
</students>
```

When you save this file, you should use the Save As menu of the notepad.exe. You must change the save As Type: to All Files. And the suffix of file name must be .xml Just as below figure shows:



When the file is finished, we can use IE to browse it.



Until now, the process for create a xml file is completed.

2.Create a DTD file for the XML file A DTD (Document Type Definition) is a set of rules that defines what tags appear in a XML document, how and where these tags appear, what the relationship the tags have with each other. Be brief, a DTD defines the legal elements and their structure in an XML document.

When an XML document is processed, it is compared within the DTD to be sure the structure is valid and all tags are used in the proper manner. Independent developers can agree to use a common DTD for exchanging XML data, which makes data be shared easily.

We create the students.dtd file as below:

```
<?xml version = "1.0"?>
<!--students.dtd-a document type definition for the students.xml-->
<!ELEMENT students (student+)>
<!ELEMENT student (name,address)>
<!ELEMENT name (firstname,lastname)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT address (street,city,email,phone)>
<!ELEMENT street (#PCDATA)>
<!ELEMENT city (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
```

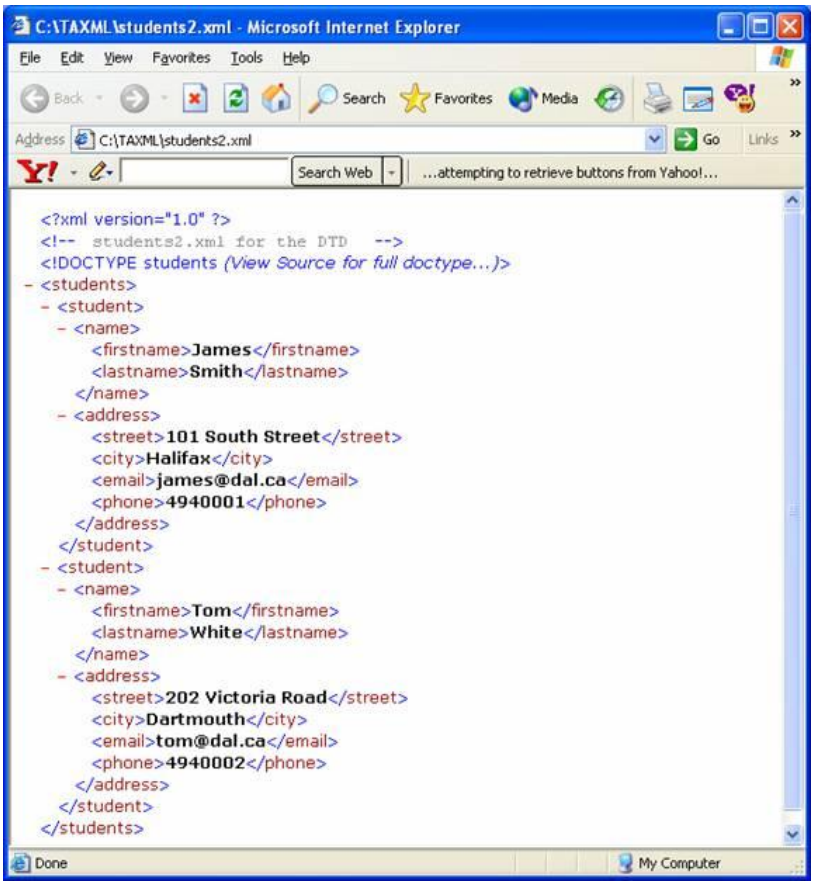
To use the DTD file, we must add this code into the XML file.

```
<!DOCTYPE students SYSTEM "students.dtd">
```

Students2.xml is completed XML file.

```
<?xml version = "1.0"?>
<!-- students2.xml for the DTD -->
<!DOCTYPE students SYSTEM "students.dtd">
<students>
<student>
<name>
<firstname> James </firstname>
<lastname> Smith </lastname>
</name>
<address>
<street> 101 South Street</street>
<city> Halifax </city>
<email> james@dal.ca </email>
<phone> 4940001 </phone>
</address>
</student>
```

```
<student>
<name>
<firstname> Tom </firstname>
<lastname> White </lastname>
</name>
<address>
<street> 202 Victoria Road </street>
<city> Dartmouth </city>
<email> tom@dal.ca</email>
<phone> 4940002 </phone>
</address>
</student>
</students>
```



3.Display the XML file in HTML by CSS and XSLT

Browsers are not particularly good at formatting XML. It is true that most of the time XML will be used to define data instead of displaying it. But for viewing, we have to format the XML data. CSS and XSL could be used to do this.

CSS

CSS (Cascading Style Sheets) is a markup language with the purpose of styling online content. It is used extensively for formatting standard HTML pages and also used to format XML documents. CSS can define how XML tags are displayed, allowing them to be presented in different ways.

```
<!--students.css- a style sheet for the students.xml document-->
student{display: block; margin-top: 15px; color: blue;}
name{display: block; margin-left:40px;margin-top: 30pt;color:red}
firstname{font-size: 28pt;}
lastname{font-size: 28pt;}
address{display: block; margin-left:40px;color:green}
street{display: block;font-size: 18pt;}
city{display: block;font-size: 18pt;}
email{display: block;font-size: 18pt;color:blue}
phone{display: block;font-size: 18pt;}
```

To use the CSS file, the XML must add the following sentence:

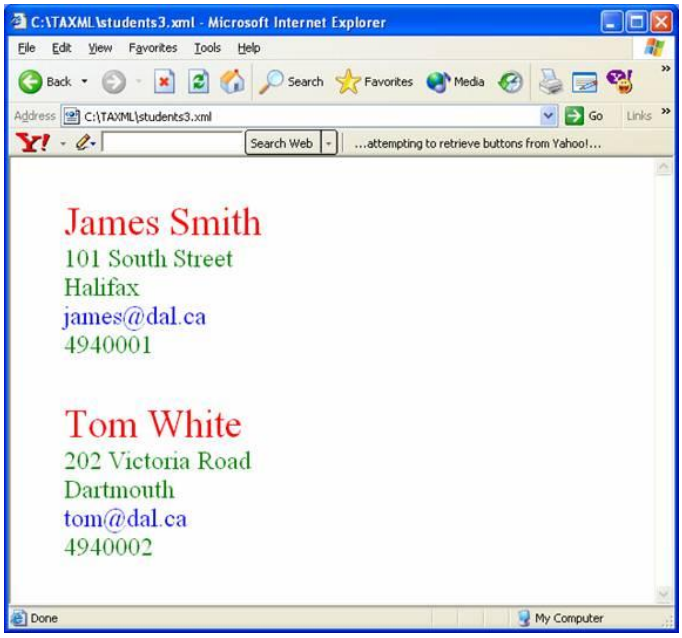
```
<?xml-stylesheet type="text/css" href= "students.css"?>
```

complete student3.xml is:

```
<?xml version = "1.0"?>
<!-- students3.xml for the CSS -->
<?xml-stylesheet type="text/css" href= "students.css"?>
<students>
<student>
<name>
<firstname> James </firstname>
<lastname> Smith </lastname>
</name>
<address>
<street> 101 South Street</street>
<city> Halifax </city>
<email> james@dal.ca </email>
<phone> 4940001 </phone>
</address>
</student>
```

```
<student>
<name>
<firstname> Tom </firstname>
<lastname> White </lastname>
</name>
<address>
<street> 202 Victoria Road </street>
<city> Dartmouth </city>
<email> tom@dal.ca</email>
<phone> 4940002 </phone>
</address>
</student>
</students>
```

Open the students3.xml by IE, we will see:



XSLT

To use the CSS file, the XML must add the following sentence:XSL (eXstensibleStylesheet Language) is XSL (eXstensibleStylesheet Language) is a new language developed to format XML documents. With the appearance of XML, in many cases the original data, not the HTML representation of it, will be exchanged. This gives the users a richer data-set to work with and data transformations are necessary. To perform these transformations, XSL could be used. XSLT is used to format a XML file into a HTML document. It maps the XML elements to XSL templates. So XSLT provides a powerful for XML.

```
<?xml version ="1.0"?>
<!--students.xsl-->
<xsl:stylesheet version = "1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://www.w3.org/TR/xhtml1/strict">
<xsl:template match = "/">
<h2> Student Record </h2>
<xsl:for-each select="students/student">
<br/>
<span style="font-weight:bold;color:red">FirstName: </span>
<xsl:value-of select="name/firstname" />
<span style="font-weight:bold;color:red">LastName: </span>
<xsl:value-of select="name/lastname" /><br/>
<span style="font-weight:bold;color:green"> Street: </span>
<xsl:value-of select="address/street" /><br/>
<span style="font-weight:bold;color:green"> City: </span>
<xsl:value-of select="address/city" /><br/>
<span style="font-weight:bold;color:blue"> Email: </span>
<xsl:value-of select="address/email" /><br/>
<span style="font-weight:bold;color:green"> Phone: </span>
<xsl:value-of select="address/phone" /><br/>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

To use the XSLT file, we must add the following sentence into XML file.

```
<?xml-stylesheet type = "text/xsl" href = "students.xsl"?>
```

The complete file students4.xml is :

```
<?xml version = "1.0"?>
<!--student4.xml for XSLT-->
<?xml-stylesheet type = "text/xsl" href = "students.xsl"?>
<students>
<student>
<name>
<firstname> James </firstname>
<lastname> Smith </lastname>
</name>
<address>
<street> 101 South Street</street>
<city> Halifax </city>
<email> james@dal.ca </email>
<phone> 4940001 </phone>
```

```
</address>
</student>

<student>
<name>
<firstname> Tom </firstname>
<lastname> White </lastname>
</name>
<address>
<street> 202 Victoria Road </street>
<city> Dartmouth </city>
<email> tom@dal.ca</email>
<phone> 4940002 </phone>
</address>
</student>
</students>
```

Now open the students4.xml by IE, we will get the following result: