**INTERNATIONAL UNIVERSITY**

VIETNAM NATIONAL UNIVERSITY HCMC

**School of Computer Science & Engineer**

**Report Lab 7**

Course: WEB APPLICATION DEVELOPMENT

Lab Instructor: Assoc. Prof.Nguyen Van Sinh

Lab Instructor: T.K.Minh

Group: ITIT22IU01

Name: Chau Thanh Phat

ID: ITITIU21135

**Lab 7: XML**

1. **Introduction:**

* Tools & Techniques: NetBeans IDE, JDK21, Glassfish
* Language: Java
* GitHub: [Link](https://github.com/thanhphat18/WebApplicationDevelopmentLab-HCMIU.git)
* Demonstration:

1. **Code Implementation:**

***Exercise 1:***

Book.xml

<root>

<book>

<isbn>

0470114878

</isbn>

<title>

Beginning XML, 4th Edition (Programmer to Programmer)

</title>

<author>

David Hunter, Jeff Rafter, Joe Fawcett, and Eric van Dist

</author>

<publisher>

Wrox

</publisher>

<publicationdate>

May 21, 2007

</publicationdate>

<price>

26.39

</price>

</book>

<book>

<isbn>

0596007647

</isbn>

<title>

XML in a Nutshell, Third Edition

</title>

<author>

Elliotte Rusty Harold and W. Scott Means

</author>

<publisher>

O'Reilly Media, Inc.

</publisher>

<publicationdate>

September 2004

</publicationdate>

<price>

26.37

</price>

</book>

<book>

<isbn>

0596004206

</isbn>

<title>

Learning XML, Second Edition

</title>

<author>

Erik Ray

</author>

<publisher>

O'Reilly Media, Inc.

</publisher>

<publicationdate>

September 22, 2003

</publicationdate>

<price>

26.37

</price>

</book>

<book>

<isbn>

0130655678

</isbn>

<title>

Definitive XML Schema (The Charles F. Goldfarb Definitive XML Series)

</title>

<author>

Priscilla Walmsley

</author>

<publisher>

Prentice Hall PTR

</publisher>

<publicationdate>

December 17, 2001

</publicationdate>

<price>

33.38

</price>

</book>

</root>

Book.dtd

*<!--- Put your DTDDoc comment here. -->*

<!ELEMENT root (book)\*>

*<!--- Put your DTDDoc comment here. -->*

<!ELEMENT book (isbn|title|author|publisher|publicationdate|price)\*>

*<!--- Put your DTDDoc comment here. -->*

<!ELEMENT isbn (#PCDATA)>

*<!--- Put your DTDDoc comment here. -->*

<!ELEMENT title (#PCDATA)>

*<!--- Put your DTDDoc comment here. -->*

<!ELEMENT author (#PCDATA)>

*<!--- Put your DTDDoc comment here. -->*

<!ELEMENT publisher (#PCDATA)>

*<!--- Put your DTDDoc comment here. -->*

<!ELEMENT publicationdate (#PCDATA)>

*<!--- Put your DTDDoc comment here. -->*

<!ELEMENT price (#PCDATA)>

***Exercise 2:***

Index.jsp

<%--

Document : index

Created on : May 24, 2024, 6:36:48 PM

Author : thanhphatchau

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE *html*>

<%@page import="org.w3c.dom.\*, javax.xml.parsers.\*" %>

<%

DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder docBuilder = docFactory.newDocumentBuilder();

Document doc = docBuilder.parse("/Users/thanhphatchau/Library/CloudStorage/OneDrive-VietNamNationalUniversity-HCMINTERNATIONALUNIVERSITY/Study Docs/recent semester/Web Application Development/lab/lab7/exercise2/WebClass.xml");

%>

<%!

public boolean isTextNode(Node n){

return n.getNodeName().equals("#text"); }

%>

<html>

<head><title>Parsing of xml using DOM Parser</title></head>

<body>

<h2>

<font *color*='red'>Student of Web Class</font>

</h2>

<table border="2">

<tr>

<th>Name of Student</th>

<th>ID Number</th>

<th>Date of Birth</th>

<th>City</th>

</tr>

<%

Element element = doc.getDocumentElement();

NodeList personNodes = element.getChildNodes();

for (int i=0; i<personNodes.getLength(); *i++){*

*Node* *stu* = personNodes.item(i); *if* *(isTextNode(stu))*

*continue;*

*NodeList* *NameDOBCity* = stu.getChildNodes();

*%*>

<tr>

<%

for (int j=0; j<NameDOBCity.getLength(); *j++* *){*

*Node* *node* = NameDOBCity.item(j);

*if* *(* *isTextNode(node))*

*continue;*

*%*>

<td><%= node.getFirstChild().getNodeValue() %></td>

<%}%>

</tr>

<%}%>

</table>

</body>

</html>

***Exercise 3:***

DOMServlet.java

*/\**

*\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license*

*\* Click nbfs://nbhost/SystemFileSystem/Templates/JSP\_Servlet/Servlet.java to edit this template*

*\*/*

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;

import org.w3c.dom.NamedNodeMap;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;

*/\*\**

*\**

*\* @author thanhphatchau*

*\*/*

@WebServlet(urlPatterns = {"/DOMServlet"})

public class **DOMServlet** extends **HttpServlet** {

*/\*\**

*\* Processes requests for both HTTP <code>GET</code> and <code>POST</code>*

*\* methods.*

*\**

*\* @param request servlet request*

*\* @param response servlet response*

*\* @throws* **ServletException** *if a servlet-specific error occurs*

*\* @throws* **IOException** *if an I/O error occurs*

*\*/*

protected void **processRequest**(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.**setContentType**("text/html;charset=UTF-8");

try (PrintWriter out = response.**getWriter**()) {

*/\* TODO output your page here. You may use following sample code. \*/*

out.**println**("<!DOCTYPE html>");

out.**println**("<html>");

out.**println**("<head>");

out.**println**("<title>Servlet DOMServlet</title>");

out.**println**("</head>");

out.**println**("<body>");

out.**println**("<h1><center>List of Students in Web Class </center></h1>");

out.**println**("<center><table border=1 cellpadding=0 bgcolor=#FFFFFF></center>");

out.**println**("<tr><td><b>Name</b></td> <td><b>ID</b></td> <td><b>DATE</b></td> <td><b>CITY</b></td> </tr>");

DocumentBuilderFactory factory = DocumentBuilderFactory.**newInstance**();

*// Turn on namespace support*

factory.**setNamespaceAware**(true);

*// Create a JAXP document builder*

DocumentBuilder parser = factory.**newDocumentBuilder**();

*// Read the entire document into memory*

Document document = parser.**parse**("/Users/thanhphatchau/Library/CloudStorage/OneDrive-VietNamNationalUniversity-HCMINTERNATIONALUNIVERSITY/Study Docs/recent semester/Web Application Development/lab/lab7/exercise3/WebClass.xml");

*// Obtain the root node of the tree*

Node booklist = document.**getDocumentElement**();

NodeList books = booklist.**getChildNodes**();

int nBooks = books.**getLength**();

for (int i = 0; i < nBooks; i++) {

Node book = books.**item**(i);

if (book.**getNodeType**() != Node.TEXT\_NODE) {

out.**println**("<tr>"); **printBook**(book, out);

out.**println**("</tr>");

} }

out.**println**("</body>");

out.**println**("</html>");

} catch (ParserConfigurationException ex) {

Logger.**getLogger**(DOMServlet.class.**getName**()).**log**(Level.SEVERE, null, ex);

} catch (SAXException ex) {

Logger.**getLogger**(DOMServlet.class.**getName**()).**log**(Level.SEVERE, null, ex);

}

}

*// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">*

*/\*\**

*\* Handles the HTTP <code>GET</code> method.*

*\**

*\* @param request servlet request*

*\* @param response servlet response*

*\* @throws* **ServletException** *if a servlet-specific error occurs*

*\* @throws* **IOException** *if an I/O error occurs*

*\*/*

@Override

protected void **doGet**(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

**processRequest**(request, response);

}

*/\*\**

*\* Handles the HTTP <code>POST</code> method.*

*\**

*\* @param request servlet request*

*\* @param response servlet response*

*\* @throws* **ServletException** *if a servlet-specific error occurs*

*\* @throws* **IOException** *if an I/O error occurs*

*\*/*

@Override

protected void **doPost**(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

**processRequest**(request, response);

}

*/\*\**

*\* Returns a short description of the servlet.*

*\**

*\* @return a String containing servlet description*

*\*/*

@Override

public String **getServletInfo**() {

return "Short description";

}*// </editor-fold>*

private void **printBook**(Node book, PrintWriter out) {

NamedNodeMap attributes = book.**getAttributes**();

if (attributes != null) {

NodeList childNodes = book.**getChildNodes**(); String name = "";

String id = "";

String date = "";

String city = "";

for (int i = 0; i < childNodes.**getLength**(); i++) {

Node child = childNodes.**item**(i);

String nodeName = child.**getLocalName**();

if (nodeName != null) {

switch (nodeName) {

case "name":

{

NodeList children = child.**getChildNodes**();

Node dateNode = children.**item**(0);

if (dateNode.**getNodeType**() == Node.TEXT\_NODE) {

name = dateNode.**getNodeValue**(); }

break;

}

case "idNum":

{

NodeList children = child.**getChildNodes**();

Node dateNode = children.**item**(0);

if (dateNode.**getNodeType**() == Node.TEXT\_NODE) {

id = dateNode.**getNodeValue**();

} break;

}

case "date-of-birth":

{

NodeList children = child.**getChildNodes**();

Node priceNode = children.**item**(0);

if (priceNode.**getNodeType**() == Node.TEXT\_NODE) {

date = priceNode.**getNodeValue**(); }

break;

}

case "city":

{

NodeList children = child.**getChildNodes**();

Node priceNode = children.**item**(0);

if (priceNode.**getNodeType**() == Node.TEXT\_NODE) {

city = priceNode.**getNodeValue**(); }

break;

}

default:

break;

}

}

}

out.**print**("<td>" + name + "</td>" + "<td>" + id + "</td>" + "<td>" + date + "</td>" + "<td>" + city + "</td>");

}

}*//end method*

}

***Exercise 4:***

DOMServlet.java

*/\**

*\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license*

*\* Click nbfs://nbhost/SystemFileSystem/Templates/JSP\_Servlet/Servlet.java to edit this template*

*\*/*

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;

import org.w3c.dom.NamedNodeMap;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;

*/\*\**

*\**

*\* @author thanhphatchau*

*\*/*

@WebServlet(urlPatterns = {"/DOMServlet"})

public class **DOMServlet** extends **HttpServlet** {

*/\*\**

*\* Processes requests for both HTTP <code>GET</code> and <code>POST</code>*

*\* methods.*

*\**

*\* @param request servlet request*

*\* @param response servlet response*

*\* @throws* **ServletException** *if a servlet-specific error occurs*

*\* @throws* **IOException** *if an I/O error occurs*

*\*/*

protected void **processRequest**(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.**setContentType**("text/html;charset=UTF-8");

try (PrintWriter out = response.**getWriter**()) {

*/\* TODO output your page here. You may use following sample code. \*/*

out.**println**("<!DOCTYPE html>");

out.**println**("<html>");

out.**println**("<head>");

out.**println**("<title>Servlet DOMServlet</title>");

out.**println**("</head>");

out.**println**("<body>");

out.**println**("<h1><center>List of Books</center></h1>");

out.**println**("<center><table border=1 cellpadding=0 bgcolor=#FFFFFF></center>");

out.**println**("<tr><td><b>ISBN-10</b></td> <td><b>TITLE</b></td> <td><b>AUTHOR</b></td> <td><b>PUBLISHER</b></td><td><b>DATE</b></td><td><b>PRICE</b></td> </tr>");

DocumentBuilderFactory factory = DocumentBuilderFactory.**newInstance**();

*// Turn on namespace support*

factory.**setNamespaceAware**(true);

*// Create a JAXP document builder*

DocumentBuilder parser = factory.**newDocumentBuilder**();

*// Read the entire document into memory*

Document document = parser.**parse**("/Users/thanhphatchau/Library/CloudStorage/OneDrive-VietNamNationalUniversity-HCMINTERNATIONALUNIVERSITY/Study Docs/recent semester/Web Application Development/lab/lab7/exercise4/book.xml");

*// Obtain the root node of the tree*

Node booklist = document.**getDocumentElement**();

NodeList books = booklist.**getChildNodes**();

int nBooks = books.**getLength**();

for (int i = 0; i < nBooks; i++) {

Node book = books.**item**(i);

if (book.**getNodeType**() != Node.TEXT\_NODE) {

out.**println**("<tr>"); **printBook**(book, out); out.**println**("</tr>");

} }

out.**println**("</body>");

out.**println**("</html>");

} catch (ParserConfigurationException ex) {

Logger.**getLogger**(DOMServlet.class.**getName**()).**log**(Level.SEVERE, null, ex);

} catch (SAXException ex) {

Logger.**getLogger**(DOMServlet.class.**getName**()).**log**(Level.SEVERE, null, ex);

}

}

*// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">*

*/\*\**

*\* Handles the HTTP <code>GET</code> method.*

*\**

*\* @param request servlet request*

*\* @param response servlet response*

*\* @throws* **ServletException** *if a servlet-specific error occurs*

*\* @throws* **IOException** *if an I/O error occurs*

*\*/*

@Override

protected void **doGet**(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

**processRequest**(request, response);

}

*/\*\**

*\* Handles the HTTP <code>POST</code> method.*

*\**

*\* @param request servlet request*

*\* @param response servlet response*

*\* @throws* **ServletException** *if a servlet-specific error occurs*

*\* @throws* **IOException** *if an I/O error occurs*

*\*/*

@Override

protected void **doPost**(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

**processRequest**(request, response);

}

*/\*\**

*\* Returns a short description of the servlet.*

*\**

*\* @return a String containing servlet description*

*\*/*

@Override

public String **getServletInfo**() {

return "Short description";

}*// </editor-fold>*

private void **printBook**(Node book, PrintWriter out) {

NamedNodeMap attributes = book.**getAttributes**();

if(attributes != null){

NodeList childNodes = book.**getChildNodes**();

String isbn = "";

String title="";

String author="";

String publisher="";

String publicationdate="";

String price="";

for(int i =0; i< childNodes.**getLength**();i++){

Node child = childNodes.**item**(i);

String nodeName = child.**getLocalName**();

if(nodeName != null){

switch(nodeName){

case"isbn":

{

NodeList children = child.**getChildNodes**();

Node dateNode = children.**item**(0);

if (dateNode.**getNodeType**() == Node.TEXT\_NODE) {

isbn = dateNode.**getNodeValue**(); }

break;

}

case"title":

{

NodeList children = child.**getChildNodes**();

Node dateNode = children.**item**(0);

if (dateNode.**getNodeType**() == Node.TEXT\_NODE) {

title = dateNode.**getNodeValue**(); }

break;

}

case"author":

{

NodeList children = child.**getChildNodes**();

Node dateNode = children.**item**(0);

if (dateNode.**getNodeType**() == Node.TEXT\_NODE) {

author = dateNode.**getNodeValue**(); }

break;

}

case"publicationdate":

{

NodeList children = child.**getChildNodes**();

Node dateNode = children.**item**(0);

if (dateNode.**getNodeType**() == Node.TEXT\_NODE) {

publicationdate = dateNode.**getNodeValue**(); }

break;

}

case"publisher":

{

NodeList children = child.**getChildNodes**();

Node dateNode = children.**item**(0);

if (dateNode.**getNodeType**() == Node.TEXT\_NODE) {

publisher = dateNode.**getNodeValue**(); }

break;

}

case"price":

{

NodeList children = child.**getChildNodes**();

Node dateNode = children.**item**(0);

if (dateNode.**getNodeType**() == Node.TEXT\_NODE) {

price = dateNode.**getNodeValue**(); }

break;

}

default:

break;

}*//end switch*

}*//end if*

}

out.**print**("<td>"+isbn+"</td>"+"<td>"+title+"</td><td>"+author+"</td><td>"+publisher+"</td><td>"+publicationdate+"</td><td>"+price+"</td>");

}*//end if*

}

}

***Exercise 5:***

Exercise5.java

*/\**

*\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license*

*\*/*

package com.mycompany.exercise5;

import java.io.\*;

import org.xml.sax.\*;

import javax.xml.parsers.SAXParserFactory;

import javax.xml.parsers.ParserConfigurationException; import javax.xml.parsers.SAXParser;

*/\*\**

*\**

*\* @author thanhphatchau*

*\*/*

public class **Exercise5** extends **HandlerBase** {

protected static final String XML\_FILE\_NAME = "/Users/thanhphatchau/Library/CloudStorage/OneDrive-VietNamNationalUniversity-HCMINTERNATIONALUNIVERSITY/Study Docs/recent semester/Web Application Development/lab/lab7/exercise5/WebClass.xml";

public static void **main**(String[] args) {

*// Use the default (non-validating) parser*

SAXParserFactory factory = SAXParserFactory.**newInstance**();

try{

*// Set up output stream*

out = new **OutputStreamWriter**(System.out, "UTF8");

*// Parse the input*

SAXParser saxParser = factory.**newSAXParser**(); saxParser.**parse**(new **File**(XML\_FILE\_NAME), new **Exercise5**());

}catch(Throwable t){

t.**printStackTrace**();

}*//end try*

System.**exit**(0);

}

static private Writer out;

*//===========================================================*

*// Methods in SAX DocumentHandler*

*//===========================================================*

public void **startDocument**() throws SAXException {

**showData**("<?xml version='1.0' encoding='UTF-8'?>"); **newLine**();

}

public void **endDocument**() throws SAXException {

try {

**newLine**();

out.**flush**();

} catch (IOException e) {

throw new **SAXException**("I/O error", e); }

}

public void **startElement**(String name, AttributeList attrs) throws SAXException{

**showData**("<" + name); if (attrs != null) {

for (int i = 0; i < attrs.**getLength**(); i++) {

**showData**(" ");

**showData**(attrs.**getName**(i) + "=\"" + attrs.**getValue**(i) +"\"");

}

}

**showData**(">");

}

public void **endElement**(String name) throws SAXException{

**showData**("</" + name + ">");

}

public void **characters**(char buf[], int offset, int len) throws SAXException{

String s = new **String**(buf, offset, len);

**showData**(s);

}

private void **showData**(String s) throws SAXException{

try {

out.**write**(s);

out.**flush**();

} catch (IOException e) {

throw new **SAXException**("I/O error", e);

}

}

*// Start a new line*

private void **newLine**() throws SAXException{

String lineEnd = System.**getProperty**("line.separator");

try {

out.**write**(lineEnd);

} catch (IOException e) {

throw new **SAXException**("I/O error", e);

}

}

}

***Exercise 6:***

Transferxml.xsl

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

<xsl:stylesheet *version*="1.0" *xmlns:xsl*="http://www.w3.org/1999/XSL/Transform">

<xsl:output *method*="html" *indent*="yes"/>

*<!-- Template for the root element -->*

<xsl:template *match*="/">

<html>

<head>

<title>WebBook Collection</title>

</head>

<body>

<h1>WebBook Collection</h1>

<table *border*="1">

<tr>

<th>Title</th>

<th>ISBN</th>

<th>Author</th>

<th>Publisher</th>

<th>Publication Date</th>

<th>Price</th>

</tr>

<xsl:for-each *select*="WebBook/book">

<tr>

<td><xsl:value-of *select*="title"/></td>

<td><xsl:value-of *select*="isbn"/></td>

<td><xsl:value-of *select*="author"/></td>

<td><xsl:value-of *select*="publisher"/></td>

<td><xsl:value-of *select*="publicationdate"/></td>

<td><xsl:value-of *select*="price"/></td>

</tr>

</xsl:for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>