Jegyzőkönyv

Adatkezelés XML környezetben

Féléves feladat

Készítette: **Valaska Gergő**

Nepunkód: **AQBMIV**

Gyakorlat: **szerda** 10-12

Gyakorlatvezető: Dr. Bednarik László

A témaválsztásom egy férfi fodrászatra esett. Fontos megjegyezni hogy éppen ezért csak férfi fodrászok és vendégek fognak megjelenni az adatok között. Ugyanakkor a későbbi bővítés és univerzitás miatt meghagytam az opciót, hogy hölgyeket is fel lehessen venni az adatbázisba. Ezért hagytam fent a nem megadásának opcióját. A valósághoz hasonló de attól eltérő lehet ez a projekt, ugynakkor a lehető legnagyobb hasonlóságra törekedtem. A fodrászokat a Tulajdonos alkalmazza és tőlük kapják a fizetésüket. A teljes keresetet a borravaló és a fizetés összege határozza meg. Utóbbi csak opcionális, azért mert vannak elégedetlen és illetlen ügyfelek. A fodrászok üzletekben dolgoznak amit a Tulajdonos felügyel. Az üzletnek vannak beszállítói, akik eltérő készletek pótlásáért felelnek. Az átvételt a dátum feltüntetésével jelöljük.





XML KÓD:

<?xml version=*"1.0"* encoding=*"UTF-8"* standalone=*"no"*?>

<adatbazis xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xsi:noNamespaceSchemaLocation=*"XMLSchemaAQBMIV.xsd"*>

<ugyfel id=*"Ugy0"*>

<nev>Tóth Ferenc</nev>

<kor>17</kor>

<nem>Ferfi</nem>

</ugyfel>

<ugyfel id=*"Ugy1"*>

<nev>Kiss Greta</nev>

<kor>29</kor>

<nem>Nő</nem>

</ugyfel>

<ugyfel id=*"Ugy2"*>

<nev>Hasas Csaba</nev>

<kor>23</kor>

<nem>Ferfi</nem>

</ugyfel>

<uzlet id=*"Uzi0"*>

<ferohely>145</ferohely>

<nyitvatartas>8-16</nyitvatartas>

<cim>

<varos>Miskolc</varos>

<utca>Hősök tere</utca>

</cim>

</uzlet>

<fodrasz id=*"F0"*>

<nev>Gál József</nev>

<kor>22</kor>

<kereset>250000</kereset>

</fodrasz>

<fodrasz id=*"F1"*>

<nev>Szabó Dániel</nev>

<kor>24</kor>

<kereset>450000</kereset>

</fodrasz>

<fodrasz id=*"F2"*>

<nev>Cziko Tivadar</nev>

<kor>33</kor>

<kereset>198000</kereset>

</fodrasz>

<tulajdonos id=*"T0"*>

<nev>Kemény Tamás</nev>

<kor>24</kor>

<profit>12300000</profit>

</tulajdonos>

</adatbazis>

XML SCHEMA:

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

<xs:schema xmlns:xs=*"http://www.w3.org/2001/XMLSchema"* xmlns:tns=*"http://www.example.org/XMLSchemaFOQH34"* elementFormDefault=*"qualified"*>

<xs:element name=*"adatbazis"*>

<xs:complexType>

<xs:sequence>

<xs:element name=*"ugyfel"* type=*"UgyfelTipus"* minOccurs=*"0"* maxOccurs=*"unbounded"*/>

<xs:element name=*"uzlet"* type=*"UzletTipus"* maxOccurs=*"unbounded"*/>

<xs:element name=*"fodrasz"* type=*"FodraszTipus"* maxOccurs=*"unbounded"*/>

<xs:element name=*"tulajdonos"* type=*"TulajdonosTipus"* maxOccurs=*"unbounded"*/>

</xs:sequence>

</xs:complexType>

<xs:key name=*"UgyfelKulcs"*>

<xs:selector xpath=*"ugyfel"*/>

<xs:field xpath=*"@id"* />

</xs:key>

<xs:key name=*"UzletKulcs"*>

<xs:selector xpath=*"uzlet"*/>

<xs:field xpath=*"@id"* />

</xs:key>

<xs:key name=*"FodraszKulcs"*>

<xs:selector xpath=*"fodrasz"*/>

<xs:field xpath=*"@id"* />

</xs:key>

<xs:key name=*"TulajdonosKulcs"*>

<xs:selector xpath=*"tulajdonos"*/>

<xs:field xpath=*"@id"* />

</xs:key>

</xs:element>

<xs:simpleType name=*"nev"*>

<xs:restriction base=*"xs:string"*/>

</xs:simpleType>

<xs:simpleType name=*"kor"*>

<xs:restriction base=*"xs:positiveInteger"*/>

</xs:simpleType>

<xs:simpleType name=*"nem"*>

<xs:restriction base=*"xs:string"*/>

</xs:simpleType>

<xs:simpleType name=*"UgyID"*>

<xs:restriction base=*"xs:positiveInteger"*/>

</xs:simpleType>

<xs:complexType name=*"UgyfelTipus"*>

<xs:sequence>

<xs:element name=*"nev"* type=*"nev"*/>

<xs:element name=*"kor"* type=*"kor"*/>

<xs:element name=*"nem"* type=*"nem"*/>

</xs:sequence>

<xs:attribute name=*"id"* type=*"xs:ID"* use=*"required"* />

</xs:complexType>

<xs:simpleType name=*"ferohely"*>

<xs:restriction base=*"xs:positiveInteger"*/>

</xs:simpleType>

<xs:simpleType name=*"nyitvatartas"*>

<xs:restriction base=*"xs:string"*/>

</xs:simpleType>

<xs:complexType name=*"UzletTipus"*>

<xs:sequence>

<xs:element name=*"ferohely"* type=*"ferohely"*/>

<xs:element name=*"nyitvatartas"* type=*"nyitvatartas"*/>

<xs:element name=*"cim"* type=*"cimTipus"*/>

</xs:sequence>

<xs:attribute name=*"id"* type=*"xs:ID"* use=*"required"* />

</xs:complexType>

<xs:simpleType name=*"varos"*>

<xs:restriction base=*"xs:string"*/>

</xs:simpleType>

<xs:simpleType name=*"utca"*>

<xs:restriction base=*"xs:string"*/>

</xs:simpleType>

<xs:complexType name=*"cimTipus"*>

<xs:sequence>

<xs:element name=*"varos"* type=*"varos"*/>

<xs:element name=*"utca"* type=*"utca"*/>

</xs:sequence>

</xs:complexType>

<xs:simpleType name=*"kereset"*>

<xs:restriction base=*"xs:string"*/>

</xs:simpleType>

<xs:complexType name=*"FodraszTipus"*>

<xs:sequence>

<xs:element name=*"nev"* type=*"nev"*/>

<xs:element name=*"kor"* type=*"kor"*/>

<xs:element name=*"kereset"* type=*"kereset"*/>

</xs:sequence>

<xs:attribute name=*"id"* type=*"xs:ID"* use=*"required"* />

</xs:complexType>

<xs:simpleType name=*"profit"*>

<xs:restriction base=*"xs:positiveInteger"*/>

</xs:simpleType>

<xs:complexType name=*"TulajdonosTipus"*>

<xs:sequence>

<xs:element name=*"nev"* type=*"nev"*/>

<xs:element name=*"kor"* type=*"kor"*/>

<xs:element name=*"profit"* type=*"profit"*/>

</xs:sequence>

<xs:attribute name=*"id"* type=*"xs:ID"* use=*"required"* />

</xs:complexType>

</xs:schema>

DOMREAD:

package hu.domparse.AQBMIV;

import java.io.File;

import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;

import org.w3c.dom.Element;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;

public class DomReadAQBMIV {

public static void main(String argv[]) throws SAXException, IOException, ParserConfigurationException {

File xmlFile = new File("XMLAQBMIV.xml");

//A documnetBuilderFactory-ból megkapjuk a documentBuildert

//A documentbuilder tartalmazza az aAPI-t

//a parse() metódus elemzi az xml fájlt a dokument.

//DOM - fa épites

DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = factory.newDocumentBuilder();

Document doc =dBuilder.parse(xmlFile);

doc.getDocumentElement().normalize();

//A dokumentum normalizalasa segit a helyes eredmenyek elereseben

System.out.println("Root element: "+doc.getDocumentElement().getNodeName());

String[] tagNames= {"ugyfel" ,"uzlet", "tulajdonos", "fodrasz"};

for(String tagName :tagNames) {

NodeList nList = doc.getElementsByTagName(tagName);

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

Node nNode = nList.item(i);

System.out.println("\nCurrent Element: " + nNode.getNodeName()); //egyedek elérése

if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

Element elem =(Element) nNode;

//Azonositó kiirása

String id = elem.getAttribute("id");

System.out.println(" ID: " +id);

//Tulajdonságok kiirása

String nodeContent="";

NodeList childNodes =elem.getChildNodes();

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

if(childNodes.item(j).getTextContent().trim() !="") {

nodeContent =normalizeText(childNodes.item(j).getTextContent().trim());

System.out.println(" "+childNodes.item(j).getNodeName()+": "+nodeContent);

}

}

}

System.out.println();

}

}

}

private static String normalizeText(String text) {

text=text.replaceAll("\\n", ", ");

text=text.replaceAll("\\s+", " ");

return text;

}

}

DOMQUERY:

package hu.domparse.AQBMIV;

import java.io.File;

import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;

import org.w3c.dom.Element;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;

public class DomQueryAQBMIV {

public static void main(String[] args) {

try {

File xmlFile = new File("XMLAQBMIV.xml");

DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = factory.newDocumentBuilder();

Document doc =dBuilder.parse(xmlFile);

doc.getDocumentElement().normalize();

String feladat="Jól kereső fodrászok (205000 ft felett):";

System.out.println(feladat +"\n\n");

Lekerdezes1(doc);

}

catch(ParserConfigurationException | IOException | SAXException ex){

System.out.println("Some error happened:\n"+ex.getMessage());

ex.printStackTrace();

}

}

private static String normalizeText(String text) {

text=text.replaceAll("\\n", ", ");

text=text.replaceAll("\\s+", " ");

return text;

}

private static void Lekerdezes1(Document doc) {

NodeList fodraszok = doc.getElementsByTagName("fodrasz");

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

Element fodrasz =(Element)fodraszok.item(i);

NodeList childNodes = fodrasz.getChildNodes();

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

Node childNode = childNodes.item(j);

if(childNode.getNodeName().equals("kereset")) {

if(Integer.parseInt(childNode.getTextContent())>=205000) {

printElement(fodrasz);

}

}

}

}

}

private static void printElement(Element elem) {

String id = elem.getAttribute("id");

System.out.println(" ID: "+ id);

String nodeContent="";

NodeList childNodes = elem.getChildNodes();

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

if(childNodes.item(j).getTextContent().trim()!="") {

nodeContent = normalizeText(childNodes.item(j).getTextContent().trim());

System.out.println(childNodes.item(j).getNodeName()+": "+nodeContent);

}

}

}

}

DOMMODIFY:

package hu.domparse.AQBMIV;

import java.io.File;

import java.io.IOException;

import java.io.UnsupportedEncodingException;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import javax.xml.transform.OutputKeys;

import javax.xml.transform.Transformer;

import javax.xml.transform.TransformerException;

import javax.xml.transform.TransformerFactory;

import javax.xml.transform.dom.DOMSource;

import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Document;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;

public class DomModifyAQBMIV {

public static void main(String[] args) {

try {

File xmlFile = new File("XMLAQBMIV.xml");

DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = factory.newDocumentBuilder();

Document doc =dBuilder.parse(xmlFile);

doc.getDocumentElement().normalize();

//DOM fává alakítom

//A tulajdonos profit módositása

NodeList nodes =doc.getElementsByTagName("tulajdonos");

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

Node node = nodes.item(i);

if(node.getNodeType()== Node.ELEMENT\_NODE) {

if(node.getAttributes().getNamedItem("id").getTextContent().equals("T0")) {

NodeList childNodes = node.getChildNodes();

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

Node childNode = childNodes.item(j);

if(childNode.getNodeName().equals("profit")) {

childNode.setTextContent("0");

}

}

}

}

}

//ügyfél név módosítás

nodes =doc.getElementsByTagName("ugyfel");

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

Node node = nodes.item(i);

if(node.getNodeType()== Node.ELEMENT\_NODE) {

if(node.getAttributes().getNamedItem("id").getTextContent().equals("Ugy1")) {

NodeList childNodes = node.getChildNodes();

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

Node childNode = childNodes.item(j);

if(childNode.getNodeName().equals("nev")) {

childNode.setTextContent("Kiss Gertrud");

}

}

}

}

}

File file =new File("XMLAQBMIV.xml");

writeXml(doc,file);

}

catch(ParserConfigurationException | IOException | SAXException | TransformerException ex){

System.out.println("Some error happened:\n"+ex.getMessage());

ex.printStackTrace();

}

}

//fájl felülirasa

private static void writeXml(Document doc, File output) throws TransformerException,UnsupportedEncodingException{

TransformerFactory transformerFactory= TransformerFactory.newInstance();

Transformer transf =transformerFactory.newTransformer();

transf.setOutputProperty(OutputKeys.ENCODING, "UTF-8");

transf.setOutputProperty(OutputKeys.INDENT, "yes");

transf.setOutputProperty("{http://xml.apache.org/xslt}indent-amount", "2");

DOMSource source = new DOMSource(doc);

StreamResult console = new StreamResult(System.out);

StreamResult file = new StreamResult(output);

transf.transform(source, console);

transf.transform(source, file);

}

}