JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Készítette: Nagy László

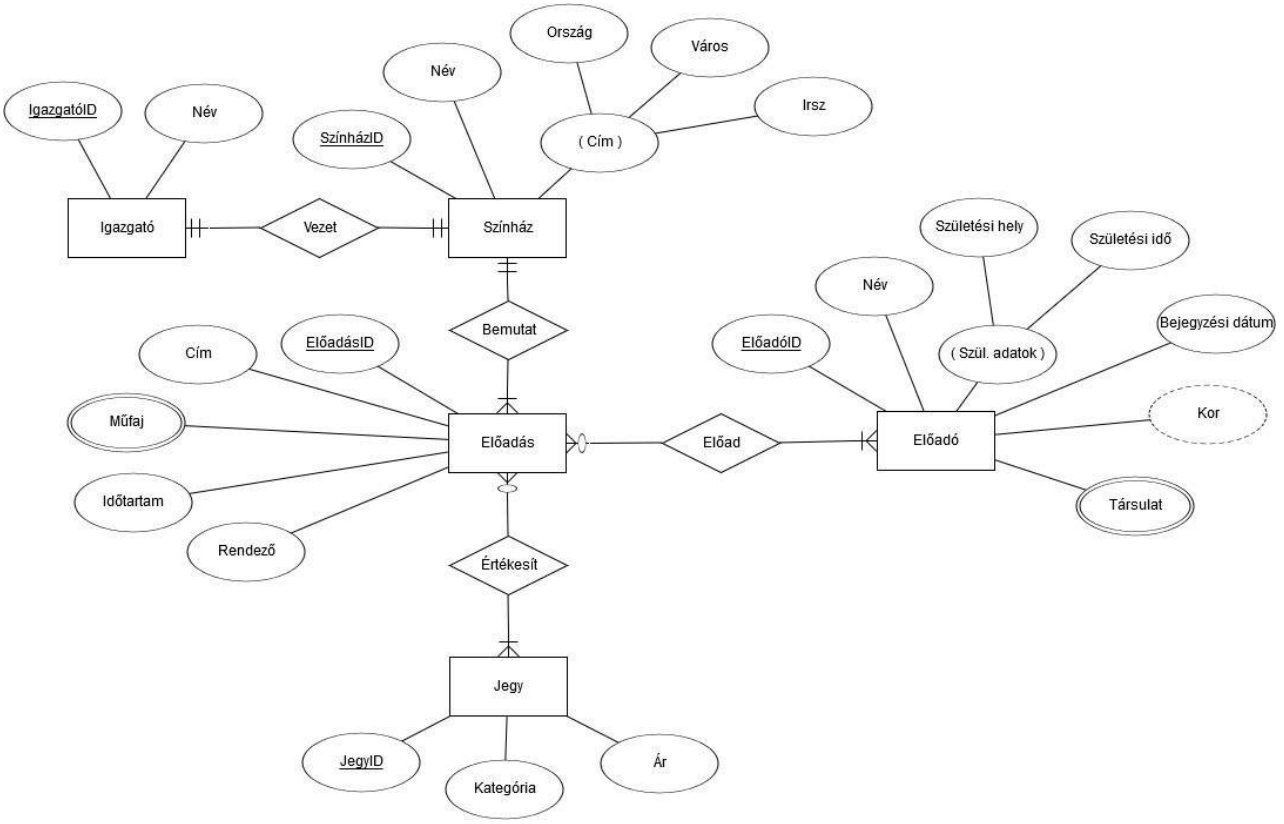
Neptunkód: LWI9Z1

A feladat leírása:

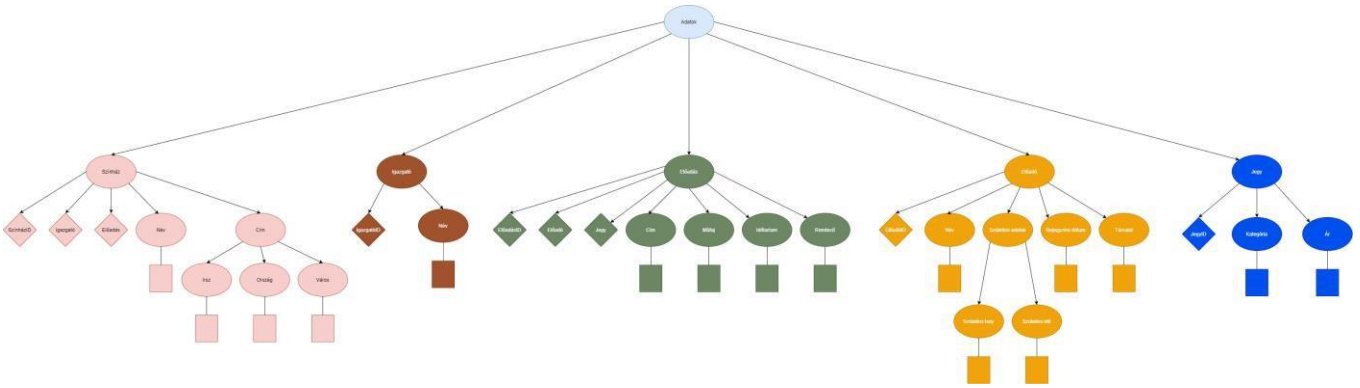
A beadandó témája egy olyan adatbázis, amely színházak kezelésére szolgál. Egy olyan adatbázist hozok létre, amiben a színház adatain kívül nyilván lesznek tartva a színház igazgatójának az adatai, a színház előadásai, valamint előadói és az előadásokra szóló jegyek. Tartalmazza azt is, hogy az egyes előadók melyik előadásokban szerepelnek, valamint minden előadáshoz tartozni fog egy, vagy akár több jegyfajta is.

1. feladat

1a) Az adatbázis ER modell:



1b) Az adatbázis konvertálása XDM modellre:



1c) Az XDM modell alapján XML dokumentum készítése:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<data xmlns="data" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="XMLSchemaLWI9Z1.xsd">

    <theatre id="theatre1" principalid="principal1" showid="show1">
        <name>
            Miskolci Nemzeti Színház
        </name>
        <address>
            <zipcode>
                3525
            </zipcode>
            <country>
                Magyarország
            </country>
            <city>
                Miskolc
            </city>
        </address>
    </theatre>

    <principal id="principal1">
        <name>
            Béres Attila
        </name>
    </principal>

    <show id="show1" actorid="actor1" ticketid="ticket1">
        <title>
            Cirkuszhercegnő
        </title>
        <genre>
            operett
        </genre>
        <length>
            185 perc
        </length>
        <director>
            Szabó Máté
        </director>
    </show>
    <show id="show2" actorid="actor2" ticketid="ticket2">
        <title>
            A velencei kalmár
        </title>
        <genre>
            vígjáték
        </genre>
        <length>
            200 perc
        </length>
```

```
        <director>
            Mohácsi János
        </director>
    </show>
    <show id="show3" actorid="actor3" ticketid="ticket3">
        <title>
            Hegedűs a háztetőn
        </title>
        <genre>
            musical
        </genre>
        <length>
            185 perc
        </length>
        <director>
            Béres Attila
        </director>
    </show>

    <actor id="actor1">
        <name>
            Papp Endre
        </name>
        <datedata>
            <dateplace>
                Debrecen
            </dateplace>
            <datetime>
                1993-10-03
            </datetime>
        </datedata>
        <dramagroup>
            színművész
        </dramagroup>
    </actor>
    <actor id="actor2">
        <name>
            Rózsa Krisztián
        </name>
        <datedata>
            <dateplace>
                Vásárosnamény
            </dateplace>
            <datetime>
                1991-07-26
            </datetime>
        </datedata>
        <dramagroup>
            színművész
        </dramagroup>
    </actor>
    <actor id="actor3">
        <name>
```

```
            Prohászka Fanni
        </name>
        <datedata>
            <dateplace>
                Budapest
            </dateplace>
            <datetime>
                1993-07-23
            </datetime>
        </datedata>
        <dramagroup>
            színművésznő
        </dramagroup>
    </actor>

    <ticket id="ticket1">
        <category>
            A
        </category>
        <price>
            2400
        </price>
    </ticket>
    <ticket id="ticket2">
        <category>
            B
        </category>
        <price>
            2700
        </price>
    </ticket>
    <ticket id="ticket3">
        <category>
            C
        </category>
        <price>
            3000
        </price>
    </ticket>

</data>
```

1d) Az XML dokumentum alapján XMLSchema készítése:

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" targe
tNamespace="data" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="data">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded" minOccurs="0">
        <xs:element name="theatre">
```

```
      <xs:complexType>
        <xs:sequence>
          <xs:element type="xs:string" name="name"/>
          <xs:element name="address">
            <xs:complexType>
              <xs:sequence>
                <xs:element type="xs:float" name="zipcode"/>
                <xs:element type="xs:string" name="country"/>
                <xs:element type="xs:string" name="city"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
        <xs:attribute type="xs:string" name="id"/>
        <xs:attribute type="xs:string" name="principalid"/>
        <xs:attribute type="xs:string" name="showid"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="principal">
      <xs:complexType>
        <xs:sequence>
          <xs:element type="xs:string" name="name"/>
        </xs:sequence>
        <xs:attribute type="xs:string" name="id"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="show" maxOccurs="unbounded" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element type="xs:string" name="title"/>
          <xs:element type="xs:string" name="genre"/>
          <xs:element type="xs:string" name="length"/>
          <xs:element type="xs:string" name="director"/>
        </xs:sequence>
        <xs:attribute type="xs:string" name="id" use="optional"/>
        <xs:attribute type="xs:string" name="actorid" use="optional"/>
        <xs:attribute type="xs:string" name="ticketid" use="optional"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="actor">
      <xs:complexType>
        <xs:sequence>
          <xs:element type="xs:string" name="name"/>
          <xs:element name="datedata">
            <xs:complexType>
              <xs:sequence>
                <xs:element type="xs:string" name="dateplace"/>
                <xs:element type="xs:date" name="datetime"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element type="xs:string" name="dramagroup"/>
        </xs:sequence>
```

```
            <xs:attribute type="xs:string" name="id" use="optional"/>
          </xs:complexType>
        </xs:element>
        <xs:element name="ticket" maxOccurs="unbounded" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element type="xs:string" name="category"/>
              <xs:element type="xs:float" name="price"/>
            </xs:sequence>
            <xs:attribute type="xs:string" name="id" use="optional"/>
          </xs:complexType>
        </xs:element>
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

2 feladat

A feladat egy DOM program készítése az XML dokumentum adatainak adminisztrálása alapján:

2a) adatolvasás

```
package hu.domparse.LWI9Z1;

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;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.TransformerException;

import java.io.File;
import java.io.IOException;
import java.util.Scanner;

public class DOMReadLWI9Z1 {

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

		try {
			File xmlFile = new File("src/hu/domparse/LWI9Z1/XMLlwi9z1.xml");
			DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
			DocumentBuilder dBuilder = factory.newDocumentBuilder();
			Document doc = dBuilder.parse(xmlFile);
```

```
                        doc.getDocumentElement().normalize();
                        System.out.println("Nagy LĂˇszlĂł LWI9Z1 XML fÃ©lÃ©ves
feladat");
                        Action(doc);
                }catch (ParserConfigurationException pce) {
                   pce.printStackTrace();
                } catch (TransformerException tfe) {
                 tfe.printStackTrace();
                } catch (IOException ioe) {
                 ioe.printStackTrace();
                } catch (SAXException sae) {
                 sae.printStackTrace();
                }
        }

        public static void Action(Document doc) throws TransformerException {
                System.out.println("\nOlvasni vagy modosĂ-tani szeretne?");
                System.out.println("1 - olvasĂˇs");
                System.out.println("2 - mĂłdosĂ-tĂˇs");
                int action = ReadCategory();

                switch (action) {
                case 1:
                        Read(doc);
                        break;
                case 2:
                        Update(doc);
                        break;
                default:
                        Action(doc);
                        break;
                }

        }

        //Kategória beolvasása
        public static int ReadCategory() {
                Scanner scan = new Scanner(System.in);
                System.out.print("\nAdja meg a sorszĂˇmot:");
                int readCategory = scan.nextInt();
                return readCategory;
        }

        //Módosítani kívánt adat sorszámának bekérése
        public static void Update(Document doc) throws TransformerException {
                System.out.println("\nXML MĂłdosĂ-tĂˇs\n");
                System.out.println("KÃ©rem adja meg mit szeretne mĂłdosĂ-tani");
                System.out.println("1 - szĂ-nhĂˇz\n2 - igazgatĂł\n3 - elĹ‘adĂˇs\n4 -
elĹ‘adĂł\n5 - jegy");
                int category = 0;
                category = ReadCategory();
                ShowElementUpdates(category, doc);
        }

        //Olvasni kívánt adatok sorszámának bekérése
```

```
	public static void Read(Document doc) {
		System.out.println("\nXML OlvasÃ˘s\n");
		System.out.println("KÃ©rem adja meg mit szeretne olvasni");
		System.out.println("1 - szÃ-nhÃ˘z\n2 - igazgatÃł\n3 - elĹ‘adÃ˘s\n4 -
elĹ‘adÃł\n5 - jegy");
		int category = 0;
		category = ReadCategory();
		ShowCategoryElements(category, doc);
	}

	public static void ShowCategoryElements(int category, Document doc) {
		switch (category) {
		case 1:
			ReadTheatre(doc);
			break;
		case 2:
			ReadPrincipal(doc);
			break;
		case 3:
			ReadShow(doc);
			break;
		case 4:
			ReadActor(doc);
			break;
		case 5:
			ReadTicket(doc);
			break;
		default:
			int newCategory = ReadCategory();
			ShowCategoryElements(newCategory, doc);
			break;
		}
	}

	public static void ShowElementUpdates(int category, Document doc) throws
TransformerException {
		switch (category) {
		case 1:
			DOMModifyLWI9Z1.UpdateTheatre(doc);
			break;
		case 2:
			DOMModifyLWI9Z1.UpdatePrincipal(doc);
			break;
		case 3:
			DOMModifyLWI9Z1.UpdateShow(doc);
			break;
		case 4:
			DOMModifyLWI9Z1.UpdateActor(doc);
			break;
		case 5:
			DOMModifyLWI9Z1.UpdateTicket(doc);
			break;
		default:
			int newCategory = ReadCategory();
			ShowElementUpdates(newCategory, doc);
```

```
                        break;
                }
        }

        public static void ReadTheatre(Document doc) {

                NodeList nList = doc.getElementsByTagName("theatre");

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

                        Node nNode = nList.item(i);

                        if (nNode.getNodeType() == Node.ELEMENT_NODE) {

                                Element element = (Element) nNode;

                                String theatreid = element.getAttribute("id");
                                String principalid = element.getAttribute("principalid");
                                String showid = element.getAttribute("showid");

                                Node node1 =
element.getElementsByTagName("name").item(0);
                                String name = node1.getTextContent();

                                String zipcode = "";
                                String country = "";
                                String city = "";

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

                                        Node nnode1 =
element.getElementsByTagName("zipcode").item(0);
                                        Node cnode1 = null;
                                        country = cnode1.getTextContent();

                                        Node nnode2 =
element.getElementsByTagName("country").item(0);
                                        Node cnode2 = null;
                                        country = cnode2.getTextContent();

                                        Node cnode3 =
element.getElementsByTagName("city").item(0);
                                        city = cnode3.getTextContent();

                                }

                                System.out.println("SzÁ-nhÁˇz id:" + theatreid + "\tNev: " +
name + "\tIrsz: " + zipcode + "\tOrszÁˇg: " + country + "\tVÁˇros: " + city);
                        }
                }

        }

        public static void ReadPrincipal(Document doc) {
```

```
		NodeList nList = doc.getElementsByTagName("principal");

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

			Node nNode = nList.item(i);

			if (nNode.getNodeType() == Node.ELEMENT_NODE) {

				Element element = (Element) nNode;

				String principalid = element.getAttribute("id");

				Node node1 =
element.getElementsByTagName("name").item(0);
				String name = node1.getTextContent();

				System.out.println("IgazgatĂł id:" + principalid + "\tNev: " +
name);
			}
		}

	}

	public static void ReadShow(Document doc) {

		NodeList nList = doc.getElementsByTagName("show");

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

			Node nNode = nList.item(i);

			NodeList cList = nList.item(i).getChildNodes();

			if (nNode.getNodeType() == Node.ELEMENT_NODE) {

				Element element = (Element) nNode;

				String showid = element.getAttribute("id");

				Node node1 = element.getElementsByTagName("title").item(0);
				String title = node1.getTextContent();

				Node node2 =
element.getElementsByTagName("genre").item(0);
				String genre = node2.getTextContent();

				Node node3 =
element.getElementsByTagName("length").item(0);
				String length = node3.getTextContent();

				Node node4 =
element.getElementsByTagName("director").item(0);
				String director = node4.getTextContent();
```

```
                        System.out.println("ElĹ‘adĂ˝s id:\t" + showid + "\tCĂ-m:\t" +
title + "\tMĹ±faj:\t" + genre + "\tIdĹ‘tartam:\t" + length + "\tRendezĹ‘:\t" + director);
                    }
                }
            }

            public static void ReadActor(Document doc) {

                NodeList nList = doc.getElementsByTagName("actor");

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

                    Node nNode = nList.item(i);

                    NodeList cList = nList.item(i).getChildNodes();

                    if (nNode.getNodeType() == Node.ELEMENT_NODE) {

                        Element element = (Element) nNode;

                        String actorid = element.getAttribute("id");

                        Node node1 =
element.getElementsByTagName("name").item(0);
                        String name = node1.getTextContent();

                        Node node2 =
element.getElementsByTagName("dramagroup").item(0);
                        String dramagroup = node2.getTextContent();

                        String dateplace = "";
                        String datetime = "";

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

                            Node cnode1 =
element.getElementsByTagName("dateplace").item(0);
                            dateplace = cnode1.getTextContent();

                            Node cnode2 =
element.getElementsByTagName("datetime").item(0);
                            datetime = cnode2.getTextContent();

                        }

                        System.out.println("ElĹ‘adĂł id:" + actorid + "\tNĂ©v: " + name
+ "\tTĂ˝rsulat: " + dramagroup
                                + "\tSzĂĽletĂ©si hely: " + dateplace +
"\tSzĂĽletĂ©si idĹ‘: " + datetime);
                    }
                }
            }

            public static void ReadTicket(Document doc) {
```

```
		NodeList nList = doc.getElementsByTagName("ticket");

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

			Node nNode = nList.item(i);

			if (nNode.getNodeType() == Node.ELEMENT_NODE) {

				Element element = (Element) nNode;

				String ticketid = element.getAttribute("id");

				Node node1 =
element.getElementsByTagName("category").item(0);
				String category = node1.getTextContent();

				Node node2 = element.getElementsByTagName("price").item(0);
				String price = node2.getTextContent();

				System.out.println("Jegy id:" + ticketid + "\tKategĂłria: " +
category + "\tĂ�r: " + price);
			}
		}

	}

}
```

2b) adatmódosítás

```
package hu.domparse.LWI9Z1;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

import java.io.File;
import java.util.Scanner;

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;

public class DOMModifyLWI9Z1 {

	public static String ReadId() {
		Scanner sc = new Scanner(System.in);
		System.out.print("\nid:");
		String id = sc.nextLine();
		return id;
	}
```

```
    public static void CreateXML(Document doc) throws TransformerException {
    TransformerFactory transformerFactory = TransformerFactory.newInstance();
    Transformer transformer = transformerFactory.newTransformer();
    DOMSource source = new DOMSource(doc);
    StreamResult result = new StreamResult(new
File("src/hu/domparse/LWI9Z1/XMLlwi9z1.updated.xml"));
    transformer.transform(source, result);
    }

    //Bekérjük a módosítani kívánt színház id-ját
    public static void UpdateTheatre(Document doc) throws TransformerException {

        System.out.println("\nMelyik színházat szeretnÃ© mÃdosÃ-tani?\n");

        DOMReadLWI9Z1.ReadTheatre(doc);

        String id = ReadId();

        Scanner sc = new Scanner(System.in);
        System.out.print("SzÃ-nhÃˇz neve: ");
        String name = sc.nextLine();
        System.out.print("SzÃ-nhÃˇz irÃˇnyÃ-tÃ³szÃˇma: ");
        String zipcode = sc.nextLine();
        System.out.print("OrszÃˇga: ");
        String country = sc.nextLine();
        System.out.print("VÃˇrosa: ");
        String city = sc.nextLine();

        UpdateTheatreById(doc, id, name, zipcode, country, city);
    }

    //Módosítjuk a kívánt színházat
    public static void UpdateTheatreById(Document doc, String id, String name, String
zipcode, String country, String city) throws TransformerException {

        NodeList nList = doc.getElementsByTagName("theatre");

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

            Node nNode = nList.item(i);

            if (nNode.getNodeType() == Node.ELEMENT_NODE) {

                Element element = (Element) nNode;

                String theatreid = element.getAttribute("id");

                if (theatreid.equals(id)) {

                    String principalid = element.getAttribute("principalid");
                    String showid = element.getAttribute("showid");

                    Node node1 =
element.getElementsByTagName("name").item(0);
                    node1.setTextContent(name);
```

```
                                        Node node2 =
element.getElementsByTagName("zipcode").item(0);
                                        node2.setTextContent(zipcode);

                                        Node node3 =
element.getElementsByTagName("country").item(0);
                                        node3.setTextContent(country);

                                        Node node4 =
element.getElementsByTagName("city").item(0);
                                        node4.setTextContent(city);

                                        System.out.println("SzĂ-nhĂˇz ID:" + theatreid +
"\tIgazgatĂł ID:" + principalid + "\tElĹ‘adĂˇs ID:"
                                                        + showid + "\tNĂ©v: " +
node1.getTextContent() + "\tIrĂˇnyĂ-tĂłszĂˇm: " + node2.getTextContent()
                                                        + "\tOrszĂˇg: " + node3.getTextContent()
+ "\tVĂˇros:" + node4.getTextContent());

                                        System.out.println("\nSikeres mĂłdosĂ-tĂˇs
tĂ¶rtĂ©nt!\n");

                                }
                        }
                }

                CreateXML(doc);
        }

        public static void UpdatePrincipal(Document doc) throws TransformerException {

                System.out.println("\nMelyik igazgatĂłt szeretnĂ© mĂłdosĂ-tani?\n");

                DOMReadLWI9Z1.ReadPrincipal(doc);

                String id = ReadId();

                Scanner sc = new Scanner(System.in);
                System.out.print("NĂ©v: ");
                String name = sc.nextLine();

                UpdatePrincipalById(doc, id, name);

        }

        public static void UpdatePrincipalById(Document doc, String id, String name) throws
TransformerException {

                NodeList nList = doc.getElementsByTagName("principal");

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

                        Node nNode = nList.item(i);
```

```
                        if (nNode.getNodeType() == Node.ELEMENT_NODE) {

                                Element element = (Element) nNode;

                                String principalid = element.getAttribute("id");

                                if (principalid.equals(id)) {

                                        Node node1 =
element.getElementsByTagName("name").item(0);
                                        node1.setTextContent(name);

                                        System.out.println("IgazgatĂł ID:" + principalid +
"\tNĂ©v: " + node1.getTextContent());

                                        System.out.println("\nSikeres mĂłdosĂ-tĂ˙s
tÅ¶rtÃ©nt!\n");

                                }
                        }
                }

                CreateXML(doc);
        }

        public static void UpdateShow(Document doc) throws TransformerException {

                System.out.println("\nMelyik elĹ‘adĂ˙st szeretnÃ© mĂłdosĂ-tani?\n");

                DOMReadLWI9Z1.ReadShow(doc);

                String id = ReadId();

                Scanner sc = new Scanner(System.in);
                System.out.print("CĂ-m: ");
                String title = sc.nextLine();
                System.out.print("MĹ±faj: ");
                String genre = sc.nextLine();
                System.out.print("IdĹ‘tartam: ");
                String length = sc.nextLine();
                System.out.print("RendezĹ‘: ");
                String director = sc.nextLine();

                UpdateShowById(doc, id, title, genre, length, director);

        }

        public static void UpdateShowById(Document doc, String id, String title, String genre,
String length, String director) throws TransformerException {

                NodeList nList = doc.getElementsByTagName("show");

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

                        Node nNode = nList.item(i);
```

```
                        NodeList cList = nList.item(i).getChildNodes();

                        if (nNode.getNodeType() == Node.ELEMENT_NODE) {

                                Element element = (Element) nNode;

                                String showid = element.getAttribute("id");

                                if (showid.equals(id)) {

                                        Node node1 =
element.getElementsByTagName("title").item(0);
                                        node1.setTextContent(title);

                                        Node node2 =
element.getElementsByTagName("genre").item(0);
                                        node2.setTextContent(genre);

                                        Node node3 =
element.getElementsByTagName("length").item(0);
                                        node3.setTextContent(length);

                                        Node node4 =
element.getElementsByTagName("director").item(0);
                                        node4.setTextContent(director);

                                        System.out.println("ElĹ‘adĂˇs ID:" + id + "\tCĂ-m: " +
node1.getTextContent() + "\tMĹ±faj: " + node2.getTextContent()
                                                        + "\tIdĹ‘tartam: " +
node3.getTextContent() + "\tRendezĹ‘: " + node4.getTextContent());

                                        System.out.println("\nSikeres mĂłdosĂ-tĂˇs
tĂ¶rtÃ©nt!\n");

                                }
                        }
                }

                CreateXML(doc);
        }

        public static void UpdateActor(Document doc) throws TransformerException {

                System.out.println("\nMelyik elĹ‘adĂłt szeretnĂ© mĂłdosĂ-tani?\n");

                DOMReadLWI9Z1.ReadActor(doc);

                String id = ReadId();

                Scanner sc = new Scanner(System.in);
                System.out.print("NÃ©v: ");
                String name = sc.nextLine();
                System.out.print("SzĂĽletÃ©si hely: ");
                String dateplace = sc.nextLine();
```

```
                System.out.print("SzĂĽletĂ©si idĹ‘: ");
                String datetime = sc.nextLine();
                System.out.print("TĂ˝rsulat: ");
                String dramagroup = sc.nextLine();

                UpdateActorById(doc, id, name, dateplace, datetime, dramagroup);

        }

        public static void UpdateActorById(Document doc, String id, String name, String
dateplace, String datetime, String dramagroup) throws TransformerException {

                NodeList nList = doc.getElementsByTagName("actor");

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

                        Node nNode = nList.item(i);

                        NodeList cList = nList.item(i).getChildNodes();

                        if (nNode.getNodeType() == Node.ELEMENT_NODE) {

                                Element element = (Element) nNode;

                                String actorid = element.getAttribute("id");

                                if (actorid.equals(id)) {

                                        Node node1 =
element.getElementsByTagName("name").item(0);
                                        node1.setTextContent(name);

                                        Node node2 =
element.getElementsByTagName("dramagroup").item(0);
                                        node2.setTextContent(dramagroup);

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

                                                Node cnode1 =
element.getElementsByTagName("dateplace").item(0);
                                                cnode1.setTextContent(dateplace);

                                                Node cnode2 =
element.getElementsByTagName("datetime").item(0);
                                                cnode2.setTextContent(datetime);

                                        }

                                        System.out.println("ElĹ‘adĂł ID:" + id + "\tNĂ©v: " +
node1.getTextContent() + "\tTĂ˝rsulat: " + node2.getTextContent()
                                                        + "\tSzĂĽletĂ©si adatok: " + dateplace +
"." + datetime);

                                        System.out.println("\nSikeres mĂłdosĂ-tĂ˝s
tĂ¶rtĂ©nt!\n");
```

```
                        }
                    }
                }

                CreateXML(doc);
        }

        public static void UpdateTicket(Document doc) throws TransformerException {

                System.out.println("\nMelyik jegyet szeretnÄ© mÄłdosÄ-tani?\n");

                DOMReadLWI9Z1.ReadTicket(doc);

                String id = ReadId();

                Scanner sc = new Scanner(System.in);
                System.out.print("KategÄłria: ");
                String category = sc.nextLine();
                System.out.print("Ä�r: ");
                String price = sc.nextLine();

                UpdateTicketById(doc, id, category, price);

        }

        public static void UpdateTicketById(Document doc, String id, String category, String
price) throws TransformerException {

                NodeList nList = doc.getElementsByTagName("ticket");

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

                        Node nNode = nList.item(i);

                        if (nNode.getNodeType() == Node.ELEMENT_NODE) {

                                Element element = (Element) nNode;

                                String ticketid = element.getAttribute("id");

                                if (ticketid.equals(id)) {

                                        Node node1 =
element.getElementsByTagName("category").item(0);
                                        node1.setTextContent(category);

                                        Node node2 =
element.getElementsByTagName("price").item(0);
                                        node2.setTextContent(price);

                                        System.out.println("Ticket ID:" + ticketid +
"\tKategÄłria: " + node1.getTextContent() + "\tÄ�r: " + node2.getTextContent());
```

```
                                        System.out.println("\nSikeres mĂłdosĂ-tĂˇs
tĂ¶rtĂ©nt!\n");

                                    }
                                }
                            }

                            CreateXML(doc);
                        }

}
```