Adatkezelés XML-ben

Féléves feladat

Jegyzőkönyv

Készítette:

Hegedűs Attila László
Mérnök-informatikus
levelező
D2OVJ9

Miskolc, 2022

A feladat bemutatása

Ebben a feladatban az ezen félév során Adatkezelés XML-ben tantárgy óráin elsajátított módszerek kerülnek bemutatásra.

A feladat elkészítéséhez rendelkezni kell egy helyesen formált XML fájlal és a hozzá tartozó sémával. Ehhez elsőként el kell készíteni a megoldás ER-modelljét, melyet át kell alakítani XDM-re, majd ezek alapján létrehozni az XML-t. Az elkészült XML fájlhoz ezután Java nyelven programot kell készíteni, ami képes lesz az XML fájl beolvasására, szűrésére és módosítására, a W3C DOM könyvtárak segítségével.

A megoldásom bemutatása

A feladatomban egy adatbázist mutatok be, mely bejegyzéseiben különböző zeneiparral kapcsolatos egyedek szerepelnek. A gyökér neve Adatok, ezen belül a bejegyzések:

Egyedek és tulajdonságaik:

1. Zenész (ID, Név (Vezetéknév, Keresztnév), Születési dátum)
2. Zenekar (ID, Név, Műfaj, Alakult)
3. Kiadó (ID, Név, Telephely, Email)
4. Tulajdonos (ID, Név (Vezetéknév, Keresztnév), Telefonszám, Email)
5. Hangszer (ID, Név (Gyártó, Típus), Osztály)

Kapcsolatok:

- 1:1: Tulajdonos – tulajdona – Kiadó
- 1:N: Kiadó – alkalmazza – Zenekart
- 1:N: Zenész – tagja – Zenekarnak
- N:M: Zenész – játszik -Hangszeren, Tulajdonság: Mióta

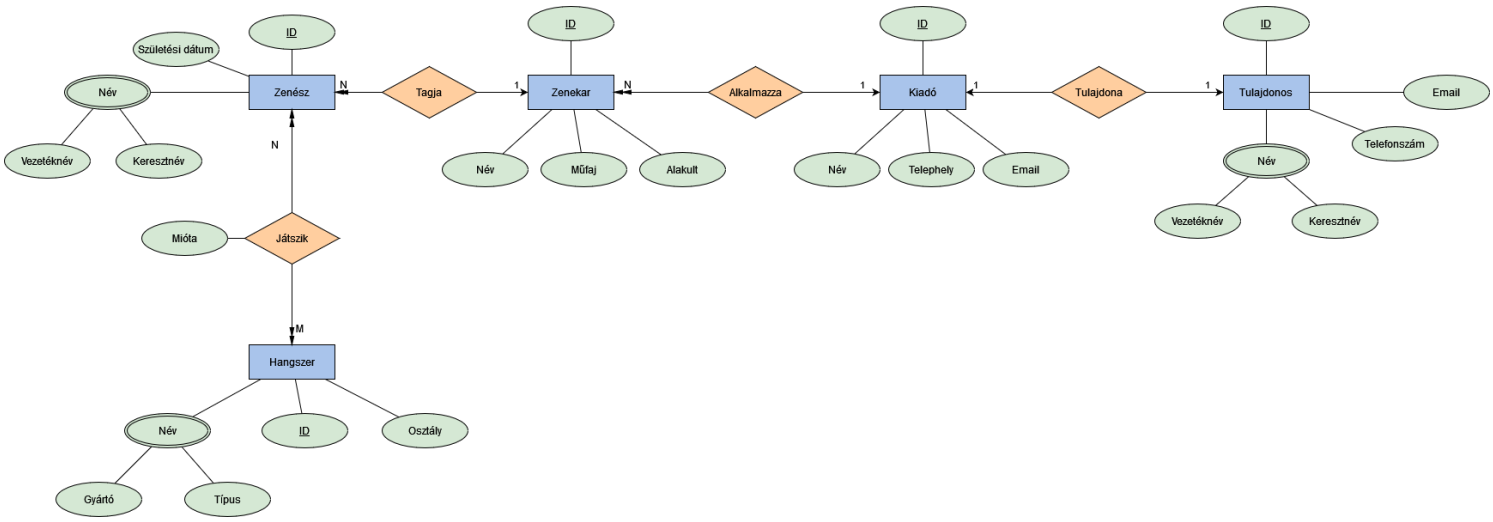
Github:

https://github.com/ati72/d2ovj9_XMLGyak/tree/main/XMLTaskD2ovj9

1. a) Feladat: ER-modell

Az ER-modellemet a draw.io internetes ábrarajzoló alkalmazás segítségével készítettem el. Az ábra elkészítése során alkalmaztam a hozzá tartozó formai megkötéseket, ezen felül színekkel is elláttam a modellt, hogy még szemléletesebb legyen.

Az egyedek világoskék négyzetekként vannak ábrázolva, a tulajdonságok világoszöld ellipszisekként. Az egyszerű tulajdonságok szimpla, az összetettek dupla körvonalakkal lettek megrajzolva. A kulcs tulajdonságok neve alá van húzva. A kapcsolatok világos narancssárga rombuszokban szerepelnek, a számosságuk nyilakkal és feliratban is fel lett tüntetve.

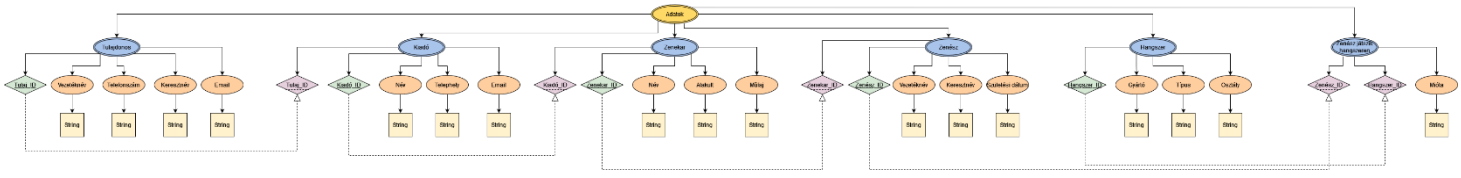


1. b) Feladat: XDM-modell

Az XDM-modellt az ER-modellből kellett átkonvertálni. Az egyedekből elemek lesznek, a tulajdonságokból gyerekelemek, a kulcsokból attribútumok stb. Az XDM-ben már feltüntetve lesznek olyan elemek is amelyek az ER-ben nem szerepelnek, ilyenek az idegen kulcsok, a gyökér elem, az elemek adattípusai.

Az ábrát az ER-hez hasonlóan a draw.io alkalmazással készítettem el. Sárga ellipszis a gyökérelem, világoskék ellipszisek az elemek, narancssárga ellipszisek a gyerekelemek, világos sárga négyzetek az adattípusok. A kulcsok világoszöld rombuszok, aláhúzott nevekkel, az idegen kulcsok lila rombuszok, szaggatott vonallal aláhúzott nevekkel. A kulcsokat és a rájuk referáló idegen kulcsokat szaggatott vonalú nyilak kapcsolják össze. Az elemek leszármazását nyilak jelölik.

Mindkét ábra megtalálható a feladathoz tartozó github repositoryban, nagyobb felbontásban.



1. c) Feladat: XML dokumentum

A modellek elkészítése után létrehoztam az általuk felvázolt XML dokumentumot.

A dokumentum létrehozásához a VSCode szövegszerkesztőt használtam. 6 többszörös előfordulást tartalmazó elemet hoztam létre a gyökérelemen belül, mindegyikből legalább 3 példány készült.

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

<adatok xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="XSDD2ovj9.xsd">
    <!-- Tulajok -->
    <tulajdonos ID="tul-01">
        <vezeteknev>Major</vezeteknev>
        <keresztnev>Anna</keresztnev>
        <telefonszam>06201234567</telefonszam>
        <email>majoranna@gmail.com</email>
    </tulajdonos>
    <tulajdonos ID="tul-02">
        <vezeteknev>Feles</vezeteknev>
        <keresztnev>Elek</keresztnev>
        <telefonszam>06309877654</telefonszam>
        <email>feles@elek.hu</email>
    </tulajdonos>
    <tulajdonos ID="tul-03">
        <vezeteknev>Ultra</vezeteknev>
        <keresztnev>Viola</keresztnev>
        <telefonszam>06709844565</telefonszam>
        <email>ultrav@gmail.com</email>
    </tulajdonos>
    <!-- Kiadók -->
    <kiado ID="kia-01" tulaj_ID="tul-01">
        <nev>Major Records</nev>
        <telephely>Eger, Széchenyi út 20</telephely>
        <email>majorrecords@gmail.com</email>
    </kiado>
    <kiado ID="kia-02" tulaj_ID="tul-02">
        <nev>Feles Music</nev>
        <telephely>Budapest, Rákóczi út 32</telephely>
        <email>feles_music@gmail.com</email>
    </kiado>
    <kiado ID="kia-03" tulaj_ID="tul-03">
        <nev>Ultra Sound</nev>
        <telephely>Székesfehérvár, Kárpát út 13</telephely>
        <email>ultrasound@hotmail.com</email>
    </kiado>
    <!-- Zenekarok -->
    <zenekar ID="zk-01" kiado_ID='kia-01'>
        <nev>Admin</nev>
        <alakult>1999</alakult>
        <mufaj>Rock</mufaj>
    </zenekar>
    <zenekar ID="zk-02" kiado_ID='kia-02'>
        <nev>Quake</nev>
        <alakult>2009</alakult>
        <mufaj>Punk</mufaj>
    </zenekar>
    <zenekar ID="zk-03" kiado_ID='kia-03'>
```

```xml
    <zenekar ID="zk-03" kiado_ID='kia-03'>
        <nev>Algoritmus</nev>
        <alakult>1986</alakult>
        <mufaj>Alternatív</mufaj>
    </zenekar>
    <!-- Zeneszek -->
    <zenesz ID="zen-01" zenekar_ID="zk-01">
        <vezeteknev>Para</vezeteknev>
        <keresztnev>Zita</keresztnev>
        <szuletett>1980</szuletett>
    </zenesz>
    <zenesz ID="zen-02" zenekar_ID="zk-02">
        <vezeteknev>Git</vezeteknev>
        <keresztnev>Áron</keresztnev>
        <szuletett>1990</szuletett>
    </zenesz>
    <zenesz ID="zen-03" zenekar_ID="zk-03">
        <vezeteknev>Techno</vezeteknev>
        <keresztnev>Kolos</keresztnev>
        <szuletett>1969</szuletett>
    </zenesz>
    <!-- Hangszerek -->
    <hangszer ID="hsz-01">
        <gyarto>Roland</gyarto>
        <tipus>FP-30X</tipus>
        <osztaly>Billentyűs</osztaly>
    </hangszer>
    <hangszer ID="hsz-02">
        <gyarto>Fender</gyarto>
        <tipus>Stratocaster</tipus>
        <osztaly>Gitár</osztaly>
    </hangszer>
    <hangszer ID="hsz-03">
        <gyarto>Pearl</gyarto>
        <tipus>Roadshow</tipus>
        <osztaly>Dob</osztaly>
    </hangszer>
    <!-- Hangszerismeret -->
    <hangszerismeret hangszer_ID="hsz-01" zenesz_ID="zen-01">
        <ideje>5</ideje>
    </hangszerismeret>
    <hangszerismeret hangszer_ID="hsz-02" zenesz_ID="zen-02">
        <ideje>3</ideje>
    </hangszerismeret>
    <hangszerismeret hangszer_ID="hsz-03" zenesz_ID="zen-03">
        <ideje>20</ideje>
    </hangszerismeret>
</adatok>
```

1. d) Feladat: XML Schema

Az XML elkészítését követően elkészítettem a hozzá tartozó séma fájlt. Az XML validálásához egy weboldalt alkalmaztam https://www.liquid-technologies.com/online-xsd-validator, ennek a használata során szimplán be kell másolni a két dokumentumot az adott beviteli mezőkbe és a validálás gombra kattintva megtörténik a folyamat.

Szerkezetileg a séma fájlom először leírja a gyökérelemben lévő elemeket complexType-ként, sequence-ben megadva a saját típusaimat. Ezután definiálom a kulcsokat és idegen kulcsokat. Végül a saját típusaim lettek elkészítve complexType-ként.

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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <xs:element name="adatok">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="tulajdonos" type="tulaj_type" minOccurs="1" maxOccurs="unbounded"/>
                <xs:element name="kiado" type="kiado_type" minOccurs="1" maxOccurs="unbounded"/>
                <xs:element name="zenekar" type="zenekar_type" minOccurs="1" maxOccurs="unbounded"/>
                <xs:element name="zenesz" type="zenesz_type" minOccurs="1" maxOccurs="unbounded"/>
                <xs:element name="hangszer" type="hangszer_type" minOccurs="1" maxOccurs="unbounded"/>
                <xs:element name="hangszerismeret" type="hangszerismeret_type" minOccurs="1" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>

        <!-- Kulcsok -->
        <!-- Tulaj -->
        <xs:key name="tulaj_ID">
            <xs:selector xpath=".//tulajdonos" />
            <xs:field xpath="@ID" />
        </xs:key>
        <xs:keyref name="tulaj_ID_ref" refer="tulaj_ID">
            <xs:selector xpath=".//kiado" />
            <xs:field xpath="@tulaj_ID" />
        </xs:keyref>
        <!-- Kiado -->
        <xs:key name="kiado_ID">
            <xs:selector xpath=".//kiado" />
            <xs:field xpath="@ID" />
        </xs:key>
        <xs:keyref name="kiado_ID_ref" refer="kiado_ID">
            <xs:selector xpath=".//zenekar" />
            <xs:field xpath="@kiado_ID" />
        </xs:keyref>
        <!-- Zenekar -->
        <xs:key name="zenekar_ID">
            <xs:selector xpath=".//zenekar" />
            <xs:field xpath="@ID" />
        </xs:key>
        <xs:keyref name="zenekar_ID_ref" refer="zenekar_ID">
            <xs:selector xpath=".//zenesz" />
            <xs:field xpath="@zenekar_ID" />
        </xs:keyref>
        <!-- Zenesz -->
        <xs:key name="zenesz_ID">
            <xs:selector xpath=".//zenesz" />
            <xs:field xpath="@ID" />
        </xs:key>
```

```xml
        <xs:keyref name="zenesz_ID_ref" refer="zenesz_ID">
            <xs:selector xpath=".//hangszerismeret" />
            <xs:field xpath="@zenesz_ID" />
        </xs:keyref>
        <!-- Hangszer -->
        <xs:key name="hangszer_ID">
            <xs:selector xpath=".//hangszer" />
            <xs:field xpath="@ID" />
        </xs:key>
        <xs:keyref name="hangszer_ID_ref" refer="hangszer_ID">
            <xs:selector xpath=".//hangszerismeret" />
            <xs:field xpath="@hangszer_ID" />
        </xs:keyref>
    </xs:element>


    <!-- Sajat tipusok -->
    <!-- Tulaj -->
    <xs:complexType name="tulaj_type">
        <xs:sequence>
            <xs:element name="vezeteknev" type="xs:string"/>
            <xs:element name="keresztnev" type="xs:string"/>
            <xs:element name="telefonszam" type="xs:string"/>
            <xs:element name="email" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:string" use="required"/>
    </xs:complexType>
    <!-- Kiado -->
    <xs:complexType name="kiado_type">
        <xs:sequence>
            <xs:element name="nev" type="xs:string"/>
            <xs:element name="telephely" type="xs:string"/>
            <xs:element name="email" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:string" use="required"/>
        <xs:attribute name="tulaj_ID" type="xs:string" use="required"/>
    </xs:complexType>
    <!-- Zenekar -->
    <xs:complexType name="zenekar_type">
        <xs:sequence>
            <xs:element name="nev" type="xs:string"/>
            <xs:element name="alakult" type="xs:string"/>
            <xs:element name="mufaj" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:string" use="required"/>
        <xs:attribute name="kiado_ID" type="xs:string" use="required"/>
    </xs:complexType>
        <!-- Zenesz -->
        <xs:complexType name="zenesz_type">
            <xs:sequence>
                <xs:element name="vezeteknev" type="xs:string"/>
                <xs:element name="keresztnev" type="xs:string"/>
                <xs:element name="szuletett" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="ID" type="xs:string" use="required"/>
            <xs:attribute name="zenekar_ID" type="xs:string" use="required"/>
        </xs:complexType>
        <!-- Hangszer -->
        <xs:complexType name="hangszer_type">
            <xs:sequence>
                <xs:element name="gyarto" type="xs:string"/>
                <xs:element name="tipus" type="xs:string"/>
                <xs:element name="osztaly" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="ID" type="xs:string" use="required"/>
        </xs:complexType>
        <!-- Hangszerismeret -->
        <xs:complexType name="hangszerismeret_type">
            <xs:sequence>
                <xs:element name="ideje" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="hangszer_ID" type="xs:string" use="required"/>
            <xs:attribute name="zenesz_ID" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:schema>
```

2. a) Feladat: Java – XML beolvasás

A fájlt beolvasó program Java nyelven íródik és felhasználja a W3C DOM könyvtárát a feladat teljesítéséhez.

A dokumentum beolvasásához szükségünk van a fájl-ra, melyet a File osztályból példányosítunk az elérési útvonalának megadásával. Példányosítanunk kell továbbá egy DocumentBuilderFactory-t a newInstance() metódusának segítségével, egy DocumentBuilder-t a DocumentBuilderFactorynk példányából a newDocumentBuilder metódussal, egy Document-et a DocumentBuilder példányunk parse metódusával, melynek paramétere az előbb megadott fájl. Ezután ezen a Document példányon végezhetjük el a beolvasást.

A beolvasáshoz egy NodeList-et hozunk létre, a getElementsByTagName() függvény segítségével. A függvény paraméterében meg kell adnunk az xml fájlban elérni kívánt elemek nevét stringként. A NodeList-be kerülnek az adott nevű node-ok, ezután a listát for ciklussal bejárva getAttribute() függvénnyel az elem adott nevű attribútumát tudjuk elérni, a getTextContent() metódussal pedig az elemek tartalmát. Ezeket változókba mentem, majd System.out.println() függvénnyel íratom ki a kimenetre. Az elemek megszámlálására egy elementCount változót implementáltam.

```java
package hu.domparse.d2ovj9;

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 DomReadD2ovj9 {

    public static void printRoot(Document doc) {
        //Gyökér kiírás
        System.out.println("Root: " + doc.getDocumentElement().getNodeName());
        System.out.println("==========================================");
    }

    public static int printOwners(Document doc, int elementCount) {
        //Tulajdonosok kiírása
        NodeList nListTulaj = doc.getElementsByTagName("tulajdonos");
        for(int i = 0; i < nListTulaj.getLength(); i++) {
            Node nNode = nListTulaj.item(i);
            System.out.println(elementCount + ". Element: " + nNode.getNodeName());
            if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                String uid = elem.getAttribute("ID");

                Node node1 = elem.getElementsByTagName("vezeteknev").item(0);
```

```
35                String fname = node1.getTextContent();
36
37                Node node2 = elem.getElementsByTagName("keresztnev").item(0);
38                String lname = node2.getTextContent();
39
40                Node node3 = elem.getElementsByTagName("telefonszam").item(0);
41                String phoneNumber = node3.getTextContent();
42
43                Node node4 = elem.getElementsByTagName("email").item(0);
44                String email = node4.getTextContent();
45
46                System.out.println("ID: "+ uid);
47                System.out.println("Vezetéknév: " + fname);
48                System.out.println("Keresztnév: " + lname);
49                System.out.println("Telefonszám: " + phoneNumber);
50                System.out.println("Email cím: " + email);
51                System.out.println("============================================");
52                elementCount++;
53            }
54        }
55        return elementCount;
56    }
57
58    public static int printLabels(Document doc, int elementCount) {
59        //Kiadók kiírása
60        NodeList nListKiado = doc.getElementsByTagName("kiado");
61        for(int i = 0; i < nListKiado.getLength(); i++) {
62            Node nNode = nListKiado.item(i);
63            System.out.println(elementCount + ". Element: " + nNode.getNodeName());
64            if(nNode.getNodeType() == Node.ELEMENT_NODE) {
65                Element elem = (Element) nNode;
66                String uid = elem.getAttribute("ID");
67                String fkeyOwner = elem.getAttribute("tulaj_ID");
68
```

```
69                Node node1 = elem.getElementsByTagName("nev").item(0);
70                String name = node1.getTextContent();
71
72                Node node2 = elem.getElementsByTagName("telephely").item(0);
73                String location = node2.getTextContent();
74
75                Node node3 = elem.getElementsByTagName("email").item(0);
76                String email = node3.getTextContent();
77
78                System.out.println("ID: "+ uid);
79                System.out.println("Tulajdonos ID: " + fkeyOwner);
80                System.out.println("Név: " + name);
81                System.out.println("Telephely: " + location);
82                System.out.println("Email cím: " + email);
83                System.out.println("============================================");
84                elementCount++;
85            }
86        }
87        return elementCount;
88    }
89
90    public static int printBands(Document doc, int elementCount) {
91        //Zenekarok kiírása
92        NodeList nListZenekar = doc.getElementsByTagName("zenekar");
93        for(int i = 0; i < nListZenekar.getLength(); i++) {
94            Node nNode = nListZenekar.item(i);
95            System.out.println(elementCount + ". Element: " + nNode.getNodeName());
96            if(nNode.getNodeType() == Node.ELEMENT_NODE) {
97                Element elem = (Element) nNode;
98                String uid = elem.getAttribute("ID");
99                String fkeyKiado = elem.getAttribute("kiado_ID");
100
101               Node node1 = elem.getElementsByTagName("nev").item(0);
102               String name = node1.getTextContent();
```

```
102                 String name = node1.getTextContent();
103
104                 Node node2 = elem.getElementsByTagName("alakult").item(0);
105                 String est = node2.getTextContent();
106
107                 Node node3 = elem.getElementsByTagName("mufaj").item(0);
108                 String genre = node3.getTextContent();
109
110                 System.out.println("ID: "+ uid);
111                 System.out.println("Kiadó ID: " + fkeyKiado);
112                 System.out.println("Név: " + name);
113                 System.out.println("Alakult: " + est);
114                 System.out.println("Műfaj: " + genre);
115                 System.out.println("=============================================");
116                 elementCount++;
117             }
118         }
119         return elementCount;
120     }
121
122     public static int printMusicians(Document doc, int elementCount) {
123         //Zenészek kiírása
124         NodeList nListZenesz = doc.getElementsByTagName("zenesz");
125         for(int i = 0; i < nListZenesz.getLength(); i++) {
126             Node nNode = nListZenesz.item(i);
127             System.out.println(elementCount + ". Element: " + nNode.getNodeName());
128             if(nNode.getNodeType() == Node.ELEMENT_NODE) {
129                 Element elem = (Element) nNode;
130                 String uid = elem.getAttribute("ID");
131                 String fkeyZenekar = elem.getAttribute("zenekar_ID");
132
133                 Node node1 = elem.getElementsByTagName("vezeteknev").item(0);
134                 String fname = node1.getTextContent();
135
```

```
135
136                 Node node2 = elem.getElementsByTagName("keresztnev").item(0);
137                 String lname = node2.getTextContent();
138
139                 Node node3 = elem.getElementsByTagName("szuletett").item(0);
140                 String born = node3.getTextContent();
141
142                 System.out.println("ID: "+ uid);
143                 System.out.println("Zenekar ID: " + fkeyZenekar);
144                 System.out.println("Vezetéknév: " + fname);
145                 System.out.println("Keresztnév: " + lname);
146                 System.out.println("Született: " + born);
147                 System.out.println("================================================");
148                 elementCount++;
149             }
150         }
151         return elementCount;
152     }
153
154     public static int printInstruments(Document doc, int elementCount) {
155         //Hangszerek kiírása
156         NodeList nListHangszer = doc.getElementsByTagName("hangszer");
157         for(int i = 0; i < nListHangszer.getLength(); i++) {
158             Node nNode = nListHangszer.item(i);
159             System.out.println(elementCount + ". Element: " + nNode.getNodeName());
160             if(nNode.getNodeType() == Node.ELEMENT_NODE) {
161                 Element elem = (Element) nNode;
162                 String uid = elem.getAttribute("ID");
163
164                 Node node1 = elem.getElementsByTagName("gyarto").item(0);
165                 String manufacturer = node1.getTextContent();
166
167                 Node node2 = elem.getElementsByTagName("tipus").item(0);
168                 String type = node2.getTextContent();
```

```
169
170                 Node node3 = elem.getElementsByTagName("osztaly").item(0);
171                 String instrumentClass = node3.getTextContent();
172
173                 System.out.println("ID: "+ uid);
174                 System.out.println("Gyártó: " + manufacturer);
175                 System.out.println("Típus: " + type);
176                 System.out.println("Osztály: " + instrumentClass);
177                 System.out.println("================================================");
178                 elementCount++;
179             }
180         }
181         return elementCount;
182     }
183
184     public static int printInstrumentKnowledge(Document doc, int elementCount) {
185         //Hangszerismeretek kiírása
186         NodeList nListHangszerismeret = doc.getElementsByTagName("hangszerismeret");
187         for(int i = 0; i < nListHangszerismeret.getLength(); i++) {
188             Node nNode = nListHangszerismeret.item(i);
189             System.out.println(elementCount + ". Element: " + nNode.getNodeName());
190             if(nNode.getNodeType() == Node.ELEMENT_NODE) {
191                 Element elem = (Element) nNode;
192                 String instrumentId = elem.getAttribute("hangszer_ID");
193                 String fkeyMusician = elem.getAttribute("zenesz_ID");
194
195                 Node node1 = elem.getElementsByTagName("ideje").item(0);
196                 String since = node1.getTextContent();
197
198                 System.out.println("Hangszer ID: "+ instrumentId);
199                 System.out.println("Zenész ID: " + fkeyMusician);
200                 System.out.println("Tapasztalat: " + since + " év");
201                 System.out.println("================================================");
```

```
201                 System.out.println("=============================================");
202                 elementCount++;
203             }
204         }
205         return elementCount;
206     }
207 
208     public static void printXml(Document doc, int elementCount) {
209         printRoot(doc);
210         elementCount = printOwners(doc, elementCount);
211         elementCount = printLabels(doc, elementCount);
212         elementCount = printBands(doc, elementCount);
213         elementCount = printMusicians(doc, elementCount);
214         elementCount = printInstruments(doc, elementCount);
215         elementCount = printInstrumentKnowledge(doc, elementCount);
216     }
217 
218     public static Document createDocument() throws ParserConfigurationException, SAXException, IOException {
219         //Fájl beolvasás
220         File xmlData = new File("XMLD2ovj9.xml");
221         DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
222         DocumentBuilder builder = dbf.newDocumentBuilder();
223         Document doc = builder.parse(xmlData);
224         doc.getDocumentElement().normalize();
225 
226         return doc;
227     }
228 
229     public static void main(String[] args) {
230 
231         try {
232 
233             Document doc = createDocument();
234             int elementCount = 1;
```

```
234             int elementCount = 1;
235             printXml(doc, elementCount);
236 
237         } catch (Exception e) {
238             System.out.println(e);
239         }
240     }
241 }
242 
```

2. b) Feladat: Java – XML szűrés

A szűréshez az előbbi feladat módszerét használtam fel, pluszban hozzáadva egy bemeneti prompt-ot mely opcióként adja, hogy melyik elemekre szeretnénk szűrni.

Az opció megadása után a megfelelő kiíratő metódusok kerülnek meghívásra. Hozzáadásra került még két opció, melyek XPath segítségével szűrnek, tulajdonos ID-je alapján megkapjuk az email-címét, illetve zenekar nevét megadva megkapjuk a műfaját. Az opciók kiválasztásához ellenőrzött bemeneti függvényt készítettem.

```java
package hu.domparse.d2ovj9;

import java.io.File;
import java.io.IOException;

import java.util.Scanner;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathConstants;
import javax.xml.xpath.XPathExpression;
import javax.xml.xpath.XPathExpressionException;
import javax.xml.xpath.XPathFactory;

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 DOMQueryD2ovj9 {

    public static void printOwners(Document doc, int elementCount) {
        NodeList nListTulaj = doc.getElementsByTagName("tulajdonos");
        for(int i = 0; i < nListTulaj.getLength(); i++) {
            Node nNode = nListTulaj.item(i);
            System.out.println(elementCount + ". Element: " + nNode.getNodeName());
            if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                String uid = elem.getAttribute("ID");

                Node node1 = elem.getElementsByTagName("vezeteknev").item(0);
                String fname = node1.getTextContent();

                Node node2 = elem.getElementsByTagName("keresztnev").item(0);
                String lname = node2.getTextContent();

                Node node3 = elem.getElementsByTagName("telefonszam").item(0);
                String phoneNumber = node3.getTextContent();

                Node node4 = elem.getElementsByTagName("email").item(0);
                String email = node4.getTextContent();

                System.out.println("ID: "+ uid);
                System.out.println("Vezetéknév: " + fname);
                System.out.println("Keresztnév: " + lname);
                System.out.println("Telefonszám: " + phoneNumber);
                System.out.println("Email cím: " + email);
                System.out.println("==========================================");
                elementCount++;
```

```
            }
        }
    }

    public static void printLabels(Document doc, int elementCount) {
        NodeList nListKiado = doc.getElementsByTagName("kiado");
        for(int i = 0; i < nListKiado.getLength(); i++) {
            Node nNode = nListKiado.item(i);
            System.out.println(elementCount + ". Element: " + nNode.getNodeName());
            if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                String uid = elem.getAttribute("ID");
                String fkeyOwner = elem.getAttribute("tulaj_ID");

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

                Node node2 = elem.getElementsByTagName("telephely").item(0);
                String location = node2.getTextContent();

                Node node3 = elem.getElementsByTagName("email").item(0);
                String email = node3.getTextContent();

                System.out.println("ID: "+ uid);
                System.out.println("Tulajdonos ID: " + fkeyOwner);
                System.out.println("Név: " + name);
                System.out.println("Telephely: " + location);
                System.out.println("Email cím: " + email);
                System.out.println("==========================================");
                elementCount++;
            }
        }

    }

    public static void printMusicians(Document doc, int elementCount) {
        NodeList nListZenesz = doc.getElementsByTagName("zenesz");
        for(int i = 0; i < nListZenesz.getLength(); i++) {
            Node nNode = nListZenesz.item(i);
            System.out.println(elementCount + ". Element: " + nNode.getNodeName());
            if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                String uid = elem.getAttribute("ID");
                String fkeyBand = elem.getAttribute("zenekar_ID");

                Node node1 = elem.getElementsByTagName("vezeteknev").item(0);
                String fname = node1.getTextContent();

                Node node2 = elem.getElementsByTagName("keresztnev").item(0);
                String lname = node2.getTextContent();

                Node node3 = elem.getElementsByTagName("szuletett").item(0);
                String born = node3.getTextContent();
```

```
                String born = node3.getTextContent();

                System.out.println("ID: "+ uid);
                System.out.println("Zenekar ID: " + fkeyBand);
                System.out.println("Vezetéknév: " + fname);
                System.out.println("Keresztnév: " + lname);
                System.out.println("Született: " + born);
                System.out.println("=============================================");
                elementCount++;
            }
        }
    }

    public static void getEmail(Document doc ,String id) throws XPathExpressionException {
        //XPath keresés
                XPathFactory xpathFactory = XPathFactory.newInstance();
                XPath xpath = xpathFactory.newXPath();
                XPathExpression expr = xpath.compile("//tulajdonos[@ID='"+ id +"']/email/text()");
                Object result = expr.evaluate(doc, XPathConstants.NODESET);
                NodeList nodes = (NodeList) result;
                for (int i = 0; i < nodes.getLength(); i++) {
                  System.out.println(nodes.item(i).getNodeValue());
                }

    }

    public static void getGenre(Document doc, String band) throws XPathExpressionException {
        XPathFactory xpathFactory = XPathFactory.newInstance();
        XPath xpath = xpathFactory.newXPath();
        XPathExpression expr = xpath.compile("//zenekar[nev='"+ band +"']/mufaj/text()");
        Object result = expr.evaluate(doc, XPathConstants.NODESET);
        NodeList nodes = (NodeList) result;
        for (int i = 0; i < nodes.getLength(); i++) {
          System.out.println(nodes.item(i).getNodeValue());
        }
    }

    public static String getQueryOption(Scanner scan) {
        System.out.println("Szűrés a következőre: \n1.Tulaj\n2.Kiadó\n3.Zenész\n4.Tulajdonos e-mail címe\n5.Zenekar műfaja\n0.Kilépés");
        String option =  Integer.toString(readInt(scan, 0, 5));
        return option;
    }

    public static String getQueryID(Scanner scan) {
        System.out.println("Adja meg a tulajdonos ID-jét");
        scan.nextLine();    //A readInt nextInt-je miatt kell!
        String id = scan.nextLine();
        return id;
    }

    public static String getBandName(Scanner scan) {
        System.out.println("Adja meg a zenekar nevét");
        scan.nextLine();    //A readInt nextInt-je miatt kell!
        String id = scan.nextLine();
```

```
160         return id;
161     }
162
163     public static Document createDocument() throws ParserConfigurationException, SAXException, IOException {
164         //Fájl beolvasás
165         File xmlData = new File("XMLD2ovj9.xml");
166         DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
167         DocumentBuilder builder = dbf.newDocumentBuilder();
168         Document doc = builder.parse(xmlData);
169         doc.getDocumentElement().normalize();
170
171         return doc;
172     }
173
174     public static int readInt(Scanner scan, int minLimit, int maxLimit) {
175
176         int number;
177         do {
178             System.out.println("Adj meg egy opciót " + minLimit + " és " + maxLimit + " között!");
179             while (!scan.hasNextInt()) {
180                 System.out.println("Ilyen opció nincs! Próbálkozz újra!");
181                 scan.next();
182             }
183             number = scan.nextInt();
184         } while (number < minLimit || number > maxLimit);
185
186         return number;
187     }
188
189     public static void main(String[] args) {
190         try {
191
192         Document doc = createDocument();
193         int elementCount = 1;
194
195         //Szûrési opció beolvasás
196         Scanner scan = new Scanner(System.in);
197         String option = getQueryOption(scan);
198
199         if(option.equals("1")) {
200             printOwners(doc, elementCount);
201         } else if(option.equals("2")) {
202             printLabels(doc, elementCount);
203         } else if(option.equals("3")) {
204             printMusicians(doc, elementCount);
205         } else if(option.equals("4")) {
206             String id = getQueryID(scan);
207             getEmail(doc, id);
208         } else if(option.equals("5")) {
209             String band = getBandName(scan);
210             getGenre(doc, band);
211         } else if (option.equals("0")) {
212             System.exit(1);
213             scan.close();
```

```
211         } else if (option.equals("0")) {
212             System.exit(1);
213             scan.close();
214         }
215         scan.close();
216     } catch (Exception e){
217         System.out.println(e);
218     }
219
220 }
221 }
```

2. c) Feladat: Java – XML módosítás

Az XML módosítására olyan függvényeket implementáltam, melyek új tulajdonost hoznak létre, letörölnek egy tulajdonos elemet, illetve módosítanak egy létező tulajdonos elemet.

Új tulajdonos elem hozzáadásához a createElement() metódust használtam, ugyanígy készült két gyermek ehhez az elemhez, keresztnév és vezetéknév megadására. Az elemeket a setTextContent() függvénnyel tölthetjük meg tartalommal. Ezután a gyerekelemeket fel kell fűznünk a szülő elemekre, először a tulajdonosra azon gyermekeit, majd a tulajdonost a gyökér elemre. A tulajdonoshoz a setAttribute() függvénnyel rendeltem azonosítót.

Tulajdonos törléséhez NodeList-be gyűjtöttem a tulajdonos elemeket, majd ezen végigiterálva kerestem ki a törlendő elem ID-jét. Ha a függvény megtalálta az adott id-vel rendelkező elemet a getParentNode() függvénnyel megkeresi a szülő elemét és a removeChild() függvénnyel törli ki azt.

Tulajdonos nevének módosításához szintén bejárok egy NodeListet egy megadott ID-t keresve, majd ha ezt megtaláltam a módosítandó elemeket a getElementsByTagName() függvény segítségével eltárolom változókban. A változókra a setTextContent() metódus meghívásával állítom át az elemek tartalmát.

A módosított XML dokumentumot ezután egy új XML fájlba írom ki. Ehhez a Java TransformerFactory-jét implementálom, létrehozva egy Transformert. A DOMSource-t példányosítva megadom az XML DOM-et. A StreamResultot felhasználva megadom a fájlt, amelybe írni szeretnék, ezután a Transformer példányom hívja a transform() függvényt, melynek paraméterében megadom az XML DOM-et és az új fájlt melybe azt beleírja.

```java
package hu.domparse.d2ovj9;
import java.io.File;
import java.io.IOException;
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.TransformerConfigurationException;
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.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomModifyD2ovj9 {

    public static Document createDocument() throws ParserConfigurationException, SAXException, IOException {
        //Fájl beolvasás
        File xmlData = new File("XMLD2ovj9_edit.xml");
        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = dbf.newDocumentBuilder();
        Document doc = builder.parse(xmlData);
        doc.getDocumentElement().normalize();

        return doc;
    }

    public static void executeEdit(Document doc) throws TransformerException, TransformerConfigurationException {
        //XML átírás
        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        DOMSource source = new DOMSource(doc);
        StreamResult newFile = new StreamResult(new File("modositott_xml.xml"));
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        transformer.transform(source, newFile);
    }

    public static void editOwnerName(Document doc) throws TransformerConfigurationException, TransformerException {
        //Tulaj vezetéknév átírás
        NodeList nodeList = doc.getElementsByTagName("tulajdonos");
        for(int i = 0; i < nodeList.getLength(); i++) {
            Node nNode = nodeList.item(i);
            Element element = (Element) nNode;
            if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                if(element.getAttribute("ID").equals("edit-me")) {
                    Node fName = element.getElementsByTagName("vezeteknev").item(0);
                    Node lName = element.getElementsByTagName("keresztnev").item(0);
                    fName.setTextContent("Hegedus");
```

```
                    lName.setTextContent("Attila");
                }
            }
        }
        doc.normalize();
    }

    public static void deleteOwner(Document doc) {
        //Tulaj törlése
        NodeList nodeList = doc.getElementsByTagName("tulajdonos");
         for(int i = 0; i < nodeList.getLength(); i++) {
            Node nNode = nodeList.item(i);
            Element element = (Element) nNode;
            if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                if(element.getAttribute("ID").equals("delete-me")) {
                    element.getParentNode().removeChild(element);
                }
            }
         }
    }

    public static void addOwner(Document doc) {
        //Tulaj hozzáadása
        Element root = doc.getDocumentElement();
        Element newOwner = doc.createElement("tulajdonos");
        newOwner.setAttribute("ID", "UJ-TULAJ");
        Element fName = doc.createElement("vezeteknev");
        fName.setTextContent("Miklós");
        Element lName = doc.createElement("keresztnev");
        lName.setTextContent("Béla");
        newOwner.appendChild(fName);
        newOwner.appendChild(lName);
        root.appendChild(newOwner);
    }

    public static void main(String[] args) {

        try {

            Document doc = createDocument();
            editOwnerName(doc);
            deleteOwner(doc);
            addOwner(doc);
            executeEdit(doc);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```