

Jegyzőkönyv

Adatkezelés XML környezetben

Féléves feladat

Készítette: Sergyán László

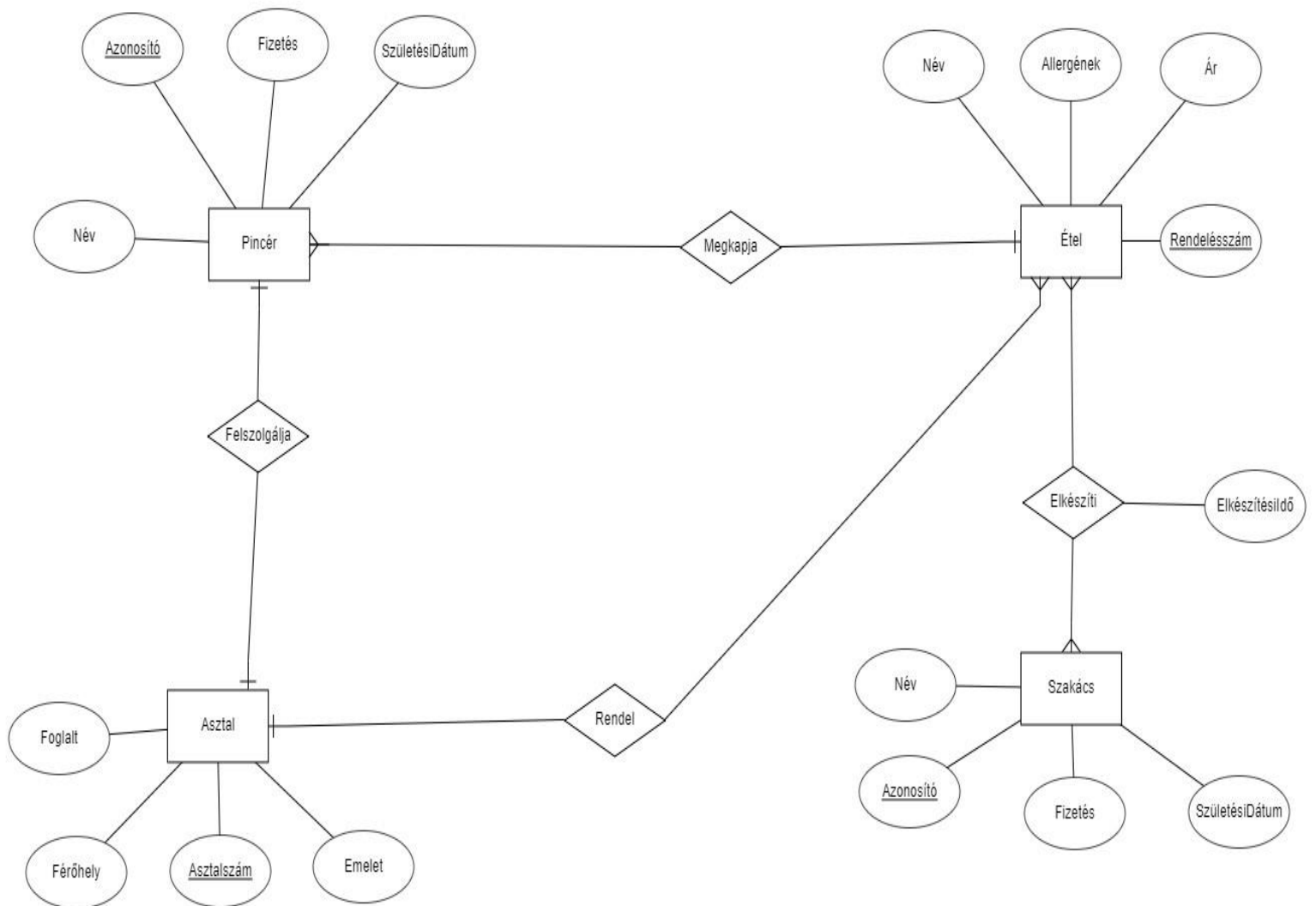
Neptunkód: JDBXZ1

A feladat leírása

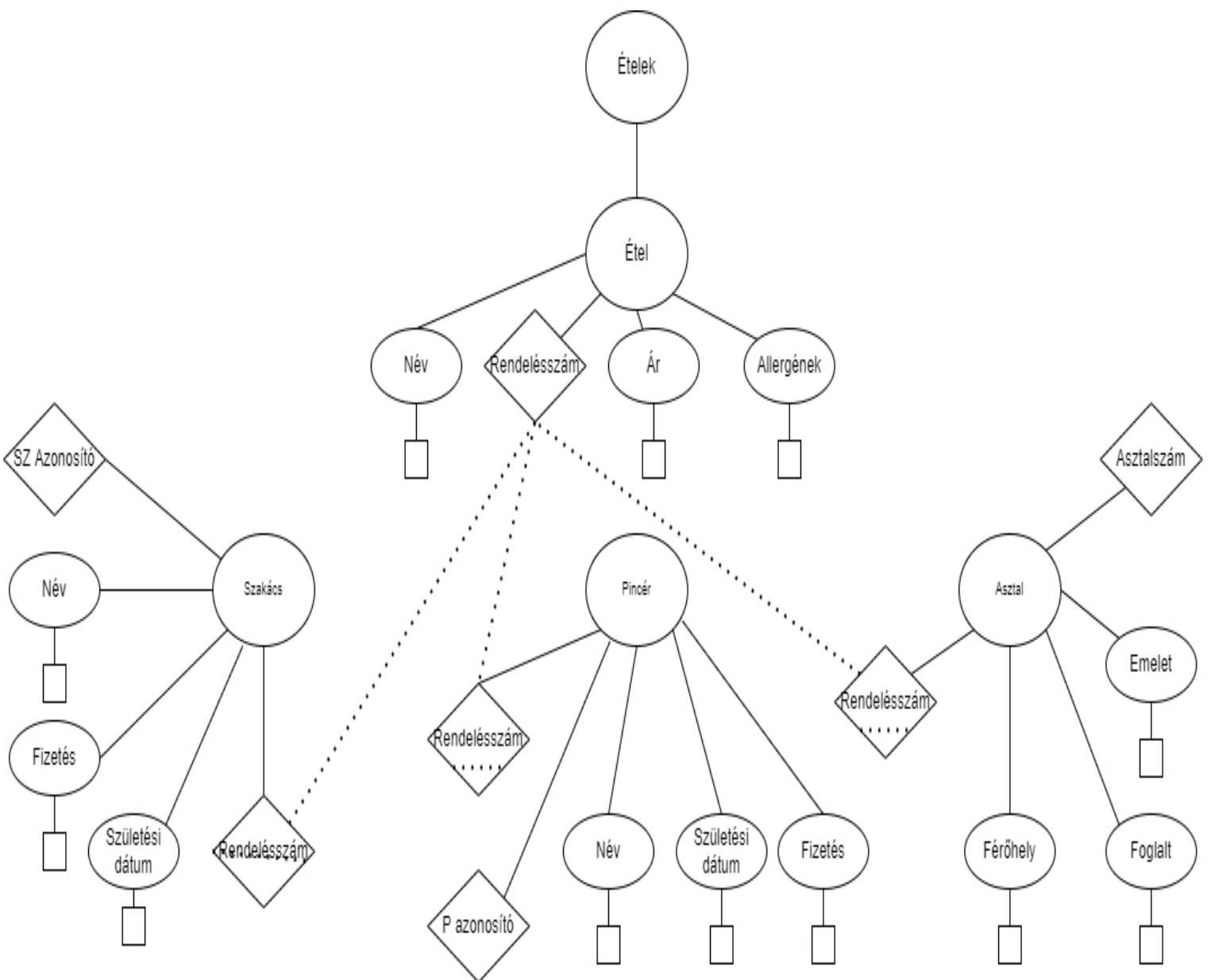
A feladat egy ER modell elkészítése volt, amely alapján egy XDM modellből létrehozunk egy XML dokumentumot, ebből pedig egy valid XSD sémát. A feladatban az általam választott elképzelés egy étterem működését veszi alapul, azon belül pedig egy megrendelt étel áll a középpontba, az XML fájl tartalmaz információt magáról az ételről, az ezt elkészítő szakácsról, a pincérről és az asztalról, ahova az étel végül kerül.

Az XML file elkészítése után a feladat második része egy Java program írása, ami lehetővé teszi a felhasználónak, hogy az XML fileból adatokat olvasson ki, módosítson és különböző lekérdezéseket hajthasson végre XPath segítségével.

Az adatbázis ER-modellje.



Az adatbázis konvertálása XDM modellre:



XML Dokumentum készítése:

```
<?xml version="1.0" encoding="UTF-8"?>
<etelek xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="XMLSchemaJDBXZ1.xsd">
```

```
  <etel rendeleesszam="150">
    <nev>Húsleves</nev>
    <allergenek>glutén</allergenek>
    <ar>950</ar>

    <szakacs SZ_azonosito="SZ_1" rendeleesszam="150">
      <nev>Szakács Péter</nev>
      <fizetes>265000</fizetes>
      <szuletesiDatum>1995.01.28</szuletesiDatum>
    </szakacs>

    <pincer P_azonosito="P_1" rendeleesszam="150">
      <nev>Pincér Petra</nev>
      <fizetes>190000</fizetes>
      <szuletesiDatum>2000.11.07</szuletesiDatum>
    </pincer>

    <asztal asztalszam="4" rendeleesszam="150">
      <foglalt>Igen</foglalt>
      <ferohely>6</ferohely>
      <emelet>2</emelet>
    </asztal>
  </etel>
```

```
  <etel rendeleesszam="152">
    <nev>Carbonara spagetti</nev>
    <allergenek>glutén</allergenek>
    <allergenek>tojás</allergenek>
    <ar>1300</ar>

    <szakacs SZ_azonosito="SZ_2" rendeleesszam="152">
      <nev>Tészta Tamás</nev>
      <fizetes>220000</fizetes>
      <szuletesiDatum>2001.08.11</szuletesiDatum>
    </szakacs>

    <pincer P_azonosito="P_2" rendeleesszam="152">
      <nev>Asztal András</nev>
      <fizetes>165000</fizetes>
      <szuletesiDatum>1998.04.21</szuletesiDatum>
    </pincer>

    <asztal asztalszam="5" rendeleesszam="152">
      <foglalt>Nem</foglalt>
```

```
        <ferohely>2</ferohely>
        <emelet>2</emelet>
    </asztal>
</etel>

<etel rendeleesszam="97">
    <nev>Halászlé</nev>
    <allergenek>hal</allergenek>
    <ar>1100</ar>

    <szakacs SZ_azonosito="SZ_3" rendeleesszam="97">
        <nev>Leves Levente</nev>
        <fizetes>290000</fizetes>
        <szuletesiDatum>1985.11.02</szuletesiDatum>
    </szakacs>

    <pincer P_azonosito="P_3" rendeleesszam="97">
        <nev>Szék Sára</nev>
        <fizetes>175000</fizetes>
        <szuletesiDatum>1983.07.23</szuletesiDatum>
    </pincer>

    <asztal asztalszam="6" rendeleesszam="97">
        <foglalt>Igen</foglalt>
        <ferohely>4</ferohely>
        <emelet>1</emelet>
    </asztal>
</etel>
</etelek>
```

Az XML alapján XSD készítése:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="etelek">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="etel">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="nev" type="xs:string" />
              <xs:element maxOccurs="unbounded" name="allergenek" type="xs:string"
/>

              <xs:element name="ar" type="xs:integer" />
              <xs:element name="szakacs">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="nev" type="xs:string" />
                    <xs:element name="fizetes" type="xs:integer" />
                    <xs:element name="szuletesiDatum" type="xs:string" />
                  </xs:sequence>
                  <xs:attribute name="SZ_azonosito" type="xs:string" use="required"
/>

                  <xs:attribute name="rendelesszam" type="xs:integer"
use="required" />
                </xs:complexType>
              </xs:element>
              <xs:element name="pincer">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="nev" type="xs:string" />
                    <xs:element name="fizetes" type="xs:integer" />
                    <xs:element name="szuletesiDatum" type="xs:string" />
                  </xs:sequence>
                  <xs:attribute name="P_azonosito" type="xs:string" use="required"
/>

                  <xs:attribute name="rendelesszam" type="xs:integer"
use="required" />
                </xs:complexType>
              </xs:element>
              <xs:element name="asztal">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="foglalt" type="xs:string" />
                    <xs:element name="ferohely" type="xs:integer" />
                    <xs:element name="emelet" type="xs:integer" />
                  </xs:sequence>
                  <xs:attribute name="asztalszam" type="xs:integer" use="required"
/>

                  <xs:attribute name="rendelesszam" type="xs:integer"
use="required" />
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:attribute name="rendelesszam" type="xs:integer" use="required" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```
        </xs:sequence>
    </xs:complexType>
    <xs:key name="rendelesszam">
        <xs:selector xpath="etelek/etel"></xs:selector>
        <xs:field xpath="@rendelesszam"></xs:field>
    </xs:key>
    <xs:key name="SZ_azonosito">
        <xs:selector xpath="etelek/etel/szakacs"></xs:selector>
        <xs:field xpath="@SZ_azonosito"></xs:field>
    </xs:key>
    <xs:key name="P_azonosito">
        <xs:selector xpath="etelek/etel/pincer"></xs:selector>
        <xs:field xpath="@P_azonosito"></xs:field>
    </xs:key>
    <xs:keyref name="rendelesszamREF" refer="rendelesszam">
        <xs:selector xpath="etelek/etel/szakacs"/>
        <xs:field xpath="@rendelesszamREF"/>
    </xs:keyref>
    <xs:keyref name="rendelesszamREF1" refer="rendelesszam">
        <xs:selector xpath="etelek/etel/pincer"/>
        <xs:field xpath="@rendelesszamREF1"/>
    </xs:keyref>
</xs:element>
</xs:schema>
```


DOM program készítése:

-Adatolvasás: DOMReadJDBXZ1.java

```
package hu.domparse.jdbxz1;

import javax.xml.parsers.*;
import java.io.*; import
org.w3c.dom.*; import
org.xml.sax.*;

public class DOMReadJDBXZ1 {

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

        File xmlfile = new File("src/hu/domparse/jdbxz1/XMLJDBXZ1.xml"); //File objektum létrehozása

        //DocumentBuilderFactory példányosítása, majd ezt meghívva DocumentBuilder létrehozása
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dbuilder = factory.newDocumentBuilder();

        //A Documentum példányosítása amibe parseoljuk a fájlunkat.
        Document doc = dbuilder.parse(xmlfile);

        //normalizálás
        doc.getDocumentElement().normalize();

        //Gyökérelem megjelenítése
        System.out.println("Gyökérelem: " + doc.getDocumentElement().getNodeName());

        NodeList nList = doc.getElementsByTagName("étel");
        System.out.println("-----");
        //For ciklusban végigiterálunk az elementeken.
        for(int i =0; i< nList.getLength(); i++) {

            //A nodelistekből egyesével node-okat készítünk
            Node nNode = nList.item(i);

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

                //A nodeokat elementté konvertáljuk
                Element elem = (Element) nNode;
```

```

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

//Étel stringgé konvertálása
Node node1 = elem.getElementsByTagName("nev").item(0);
String nev = node1.getTextContent();
Node node2 = elem.getElementsByTagName("allergenek").item(0);
String allergenek = node2.getTextContent();
Node node3 = elem.getElementsByTagName("ar").item(0);
String ar = node3.getTextContent();

//Majd az étel stringként való kiírása
System.out.println("Étel rendelésszám: "+rendelesszam);
System.out.println("Étel neve : "+nev);
System.out.println("Allergének: "+allergenek);
System.out.println("Étel ára : "+ar +"\n");

//Metódusok meghívása
System.out.println("Szakács: ");

listSzakacs(doc, rendelesszam);
System.out.println("-----");

System.out.println("Pincér: ");
listPincer(doc,rendelesszam);
System.out.println("-----");

System.out.println("Asztal: ");
listAsztal(doc,rendelesszam);

System.out.println("-----"); //Olvashatóság érdekében
    }
}

}

public static void listSzakacs(Document doc,String rendelesszam) {
    NodeList nList = doc.getElementsByTagName("szakacs"); //Most a szakács elemeket nézzük
    meg
    for(int i =0; i<nList.getLength();i++) {
        Node nNode = nList.item(i);
        if(nNode.getNodeType() == Node.ELEMENT_NODE) {

```

```

        Element elem = (Element) nNode;
        if(elem.getAttribute("rendelesszam").toString().equals(rendelesszam)) {
//Megkeressük azt a szakácsot, akihez az étel rendelésszáma tartozik

            //Az ételhez hasonlóan stringgé alakítjuk

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

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

            Node node3 =
elem.getElementsByTagName("szuletesiDatum").item(0);

            String szuletesiDatum = node3.getTextContent();

            //Majd kiírjuk

            System.out.println("Szakács neve: " +nev);

            System.out.println("Fizetése: "+fizetes);

            System.out.println("Születési dátum: "+szuletesiDatum);

        }
    }
}
}
}

```

```

public static void listPincer(Document doc,String rendelesszam) {
    NodeList nList = doc.getElementsByTagName("pincer");          for(int
    i =0; i<nList.getLength();i++) {
        Node nNode = nList.item(i);
        if(nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;

            if(elem.getAttribute("rendelesszam").toString().equals(rendelesszam)) {
//Megkeressük a pincért akihez a rendelésszám tartozik

                //Stringgé alakítjuk

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

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

                Node node3 =
elem.getElementsByTagName("szuletesiDatum").item(0);

```

```

        String szuletesiDatum = node3.getTextContent();

        //Majd kiírjuk
        System.out.println("Pincér neve: " +nev);

        System.out.println("Fizetése: "+fizetes);

        System.out.println("Születési dátum: "+szuletesiDatum);

    }

}

}

}

```

```

public static void listAsztal(Document doc,String rendeleesszam) {
    NodeList nList = doc.getElementsByTagName("asztal");          for(int
    i =0; i<nList.getLength();i++) {
        Node nNode = nList.item(i);
        if(nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;

            if(elem.getAttribute("rendelesszam").toString().equals(rendelesszam)) {
//Megkeressük az asztalt ahova a rendelés készül

                //Stringgé alakítjuk

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

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

                String foglalt = node1.getTextContent();

                Node node2 = elem.getElementsByTagName("ferohely").item(0);

                String ferohely = node2.getTextContent();

                Node node3 = elem.getElementsByTagName("emelet").item(0);

                String emelet = node3.getTextContent();

                //Kiírjuk

                System.out.println("Asztal: " +asztalszam);

                System.out.println("Foglalt: "+foglalt);

                System.out.println("Ferohely: "+ferohely);

                System.out.println("Emelet: "+emelet);

            }

        }

    }

}

```

```
}  
}
```

Adatmódosítás:

DOMModifyJDBXZ1.java

```
package hu.domparse.jdbxz1;  
  
import java.io.*; import  
java.util.Scanner; import  
javax.xml.parsers.*; import  
javax.xml.transform.*; import  
javax.xml.transform.dom.*; import  
javax.xml.transform.stream.*;  
import org.w3c.dom.*; import  
org.xml.sax.*;  
  
public class DOMModifyJDBXZ1 {  
    public static void main(String[] args) throws SAXException, IOException,  
ParserConfigurationException, TransformerConfigurationException, TransformerException {  
        //Itt nincs hibakezelés, a throws miatt kihagyható a try-catch  
  
        //File objektum létrehozása az XML fájlból  
        File xmlfile = new File("src/hu/domparse/jdbxz1/XMLJDBXZ1.xml");  
  
        //Létrehozzuk az output fileunk ami a módosítást tartalmazza.  
        File modifiedXmlFile = new  
File("src/hu/domparse/jdbxz1/XMLJDBXZ1MODIFIED.xml");  
  
        //Scanner billentyűzet input olvasására.  
        Scanner sc = new Scanner(System.in);  
  
        //DocumentBuilderFactory példányosítása, majd ezt meghívva DocumentBuilder  
létrehozása  
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();  
        DocumentBuilder dBuilder = factory.newDocumentBuilder();  
  
        //XML fájl beolvasása majd normalizálása  
        Document doc = dBuilder.parse(xmlfile);  
        doc.getDocumentElement().normalize();  
  
        //Lista létrehozása az etel nevu elemekből  
        NodeList nList = doc.getElementsByTagName("etel");
```

```

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

            //A lista itemeit Nodeokban tároljuk

            Node nNode = nList.item(i);

            //A nodeot elementté konvertáljuk

            Element elem = (Element) nNode;

            //A nodeból kiszedjük azt az elemet amit módosítani kívánunk, most az árat.

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

            //Ezt stringként eltároljuk

            String ar = node1.getTextContent();

            //Hogy megjeleníthessük

            System.out.println("Az étel jelenlegi ára: " + ar + "FT\n");

            //Bekérjük az új árat szintén stringként

            System.out.println("Az étel új ára: \n");

            String modifiedAr = sc.next();

            //És ezt állítjuk be a nodeba új árként

            node1.setTextContent(modifiedAr);

        }

        //A scannert bezárjuk

        sc.close();

        //TransformerFactory objektummal egy transformer objektumot hozunk létre.

        TransformerFactory transformerFactory = TransformerFactory.newInstance();

        Transformer transformer = transformerFactory.newTransformer();

        //Megadjuk a forrásfájlt

        DOMSource source = new DOMSource(doc);

        //Majd "átalakítjuk" egy transformer result fileá.

        StreamResult result = new StreamResult(modifiedXmlFile);

        transformer.transform(source, result);

    }

}

```

Adatlekérdezés:

DOMQueryJDBXZ1.java

```
package hu.domparse.jdbxz1;

import javax.xml.parsers.*; import
javax.xml.xpath.*;

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 java.io.*;

public class DOMQueryJDBXZ1 {

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

            //DocumentBuilderFactory példányosítása, majd ezt meghívva
            DocumentBuilder létrehozása

            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();

            //Dokumentum objektum létrehozása az XML fájlunkból, majd normalizálás

            Document doc = builder.parse(new
            File("src/hu/domparse/jdbxz1/XMLJDBXZ1.xml"));

            doc.getDocumentElement().normalize();

            //XPath létrehozása hasonlóan

            XPath xPath = XPathFactory.newInstance().newXPath();

            //Lekérdezés stringként, 1000Ft feletti ételek

            String query = "etelek/etel[ar>1000]";

            //lista létrehozása, a lekérdezésnek megfelelő elemekkel
```

```

        NodeList nList=(NodeList)
XPath.compile(query).evaluate(doc,XPathConstants.NODESET);

        //for ciklus a nodeokon való iterációhoz.

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

            Node node=nList.item(i);

            //étel kiírása

            System.out.println(node.getNodeName());

            //Ha element típusú és "étel neve"

if(node.getNodeType() ==Node.ELEMENT_NODE &&
node.getNodeName().equals("étel")) {

                //Akkor a nodeot elementté konvertáljuk

                Element elem = (Element) node;

                //Majd ennek az elementnek kiírjuk az adatait, a rendelésszám
attribútum, így azt getAttribute-al

                System.out.println("Rendelés szám :
"+elem.getAttribute("rendelesszam"));

                System.out.println("Étel neve: "
+elem.getElementsByTagName("nev").item(0).getTextContent());

                System.out.println("Allergének:
"+elem.getElementsByTagName("allergenek").item(0).getTextContent());

                System.out.println("Ár:
"+elem.getElementsByTagName("ar").item(0).getTextContent());

            }

        }

        //Try blokk vége, "hibakezelés",hiba esetén tájékoztatjuk a felhasználót a fellépő
hibákról

        }catch(ParserConfigurationException | SAXException | XPathException | IOException
e ) {

            e.printStackTrace();

        }

    }

}

```