

Jegyzőkönyv

Adatkezelés XML környezetben

Féléves feladat

Pizzéria adatbázis rendszer

Készítette: Csonka Patrik

Neptunkód: CMU4ZN

Dátum: 2024.12.07

Tartalomjegyzék

1. A feladat leírása
2. Az adatbázisban szereplő adatok
3. Kapcsolatok leírása
4. Az ER model és XDM model
5. Pizza XML schema leírás
6. Java programok
 - 6.1 DOMRead
 - 6.2 DOMQuery
 - 6.3 DOMWrite
 - 6.4 DOMModify

1. A feladat leírása

Számos pizzéria adatait és működési információit tároló XML dokumentum létrehozása XML Schema validálással, majd ezen állományok különböző nemű manipulálása és olvasása Java DOM programokkal. Maga az ötlet egy pizzéria adatbázis logikai modellezéséből származik, amelyet az XML beadandó érdekében tartalmilag és szerkezetileg adaptáltam az XML feldolgozási követelményekhez. A pizzériához tartozó egyedeket (pl. ügyfelek, rendelések, dolgozók, pizzák, hozzávalók) részletesen definiáltam, és külön figyelmet fordítottam az adatok változatosságára, mint például többértékű elemek (pl. pizzák feltétei) és különböző adattípusok (pl. árak, időpontok). Az adatkezelés Java DOM programokkal valósul meg, amelyek magukban foglalják az adatok olvasását, lekérdezését, módosítását és új adatok generálását.

2. Az adatbázisban szereplő adatok

Személyzet

A pizzériában dolgozó személyek (pl. szakács, felszolgáló, sofőr, stb.).

- **Személyzet ID:** (szem_id), 3 jegyű egész szám. Az egyedi azonosítójuk. **KULCS.**
- **Pozíció:** Szöveg. A dolgozó pozíciója (szakács, futár, konyhás stb.).
- **Munkaterület:** Szöveg. Az a terület, ahol a dolgozó munkát végez (konyha, kiszállítás, vendégtér stb.).
- **Családi állapot:** Többértékű mező. A dolgozó családi állapota.

Alkalmazott

A pizzériában dolgozó alkalmazottak általános adatai.

- **Életkor:** (életkor), 3 jegyű egész szám. Az alkalmazott életkora.
- **Név:** (név), szöveg. Az alkalmazott neve.
- **Alkalmazotti ID:** (alk_id), 3 jegyű egész szám. Az alkalmazott egyedi azonosítója. **KULCS.**
- **Lakhely:**

- **Irányítószám:** (irsz), 4 jegyű egész.
- **Város:** Szöveg. Az alkalmazott lakhelyének városa.
- **Utca:** Szöveg. Az alkalmazott lakhelyének utca neve.
- **Házszám:** (házszám), 4 jegyű egész.

Konyha

A pizzéria konyhájának adatai.

- **Kapacitás:** (kapacitás), számadat. Az adott konyha által egyszerre készíthető pizzák száma.
- **Konyha ID:** (konyha_id), 1 jegyű nagybetű. Az adott konyha egyedi azonosítója. **KULCS.**
- **Konyha neve:** Szöveg. Az adott konyha neve.
- **Berendezések:** Többértékű elem. Az adott konyha berendezései (pl. kemence, dagasztógép stb.).

Rendelések

- **Rendelés ID:** (rend_id), szöveg + szám. A rendelés azonosítója (pl. rend001). **KULCS.**
- **Mennyiség:** (mennyiség), egész szám. Az adott rendelés darabszáma.
- **Megjegyzés:** Szöveg. Az ügyfél által írt megjegyzés.

Pizzák

- **Pizza ID:** (pizza_id), betű + szám. A pizza egyedi azonosítója (pl. P01). **KULCS.**
- **Név:** Szöveg. A pizza neve.
- **Ár:** (ár), számadat. A pizza ára.
- **Feltétek:** Többértékű elem. A pizza összetevői (pl. szalámi, sajt, paradicsom).

```

<!--Customer instances-->

<customer customer_id="01">
  <username>Pekseg</username>
  <password>alma123</password>
  <nickname>Pek</nickname>
  <adress>
    <city>Galgamácsa</city>
    <street>Petőfi Utca 27.</street>
    <zip_code>2183</zip_code>
  </adress>
</customer>

<customer customer_id="02">
  <username>pisti</username>
  <password>helokapistike</password>
  <nickname>pityesz</nickname>
  <adress>
    <city>Veresegyház</city>
    <street>Kálvin tér 7.</street>
    <zip_code>2112</zip_code>
  </adress>
</customer>

<customer customer_id="03">
  <username>eliza1212</username>
  <password>liza2000</password>
  <nickname>lizi</nickname>
  <adress>
    <city>Aszód</city>
    <street>Falujárók útja 5.</street>
    <zip_code>2170</zip_code>
  </adress>
</customer>

<!--Orders instances-->

<order order_id="01" customer_id="02">
  <order_number>12</order_number>
  <price>12000</price>
  <quantity>2</quantity>
  <note>Légyszi ne rakjatok rá sajtot</note>
</order>

<order order_id="02" customer_id="01">
  <order_number>15</order_number>
  <price>3000</price>
  <quantity>1</quantity>
  <note>Extra csípősen kérem</note>

```

```

</order>

<order order_id="03" customer_id="03">
  <order_number>20</order_number>
  <price>30000</price>
  <quantity>10</quantity>
  <note>Holnap DU 3-ra kérem</note>
</order>

<!--Employee instances-->

<employee employee_id="01">
  <username>Kiskakas</employee>
  <password>utalomafonokom</employee>
  <experience>2</experience>
  <nickname>nagykakas</nickname>
</employee>

<employee employee_id="02">
  <username>KisBéla</employee>
  <password>szeretemafofonokom</employee>
  <experience>10</experience>
  <nickname>Béci</nickname>
</employee>

<employee employee_id="03">
  <username>HorváthDeborah</employee>
  <password>debike902</employee>
  <experience>5</experience>
  <nickname>Debi</nickname>
</employee>

<!--Order handling switchboard instances-->

<order_handling order_id="01" employee_id="02">
  <time>17:45</time>
</order_handling>

  <order_handling order_id="02" employee_id="01">
    <time>17:20</time>
  </order_handling>

  <order_handling order_id="03" employee_id="03">
    <time>18:00</time>
  </order_handling>

<!--Pizza instances-->

<pizza pizza_id="01">

```

```

    <name>Diablo</name>
    <price>2000</price>
    <topping>Szalámi</topping>
    <topping>Habanero Paprika</topping>
    <topping>Mozzarella sajt</topping>
</pizza>

<pizza pizza_id="02">
    <name>SonGoKu</name>
    <price>1800</price>
    <topping>Sonka</topping>
    <topping>Gomba</topping>
    <topping>Kukorica</topping>
</pizza>

<pizza pizza_id="03">
    <name>Margareta</name>
    <price>1000</price>
    <topping>Paradicsomszós</topping>
    <topping>Bazsalikom</topping>
    <topping>Mozzarella sajt</topping>
</pizza>

<!--Ingredients instances-->

<ingredients ingredients_id="01">
    <name>Paradicsom</name>
    <price>100</price>
    <origin>Magyarország</origin>
</ingredients>

<ingredients ingredients_id="02">
    <name>Sonka</name>
    <price>200</price>
    <origin>Olaszország</origin>
</ingredients>

<ingredients ingredients_id="03">
    <name>Sajt</name>
    <price>500</price>
    <origin>Olaszország</origin>
</ingredients>

```

3.Kapcsolatok leírása

Rendelés-Vásárló

- **Kapcsolat típusa:** 1:N kapcsolat.
- Egy vásárló több rendelést is leadhat.
- Minden rendelés egy adott vásárlóhoz tartozik.

Alkalmazott-Rendelés

- **Kapcsolat típusa:** N:M kapcsolat.
- Egy rendelést egy alkalmazott kezel (pl. kiszállító), de egy alkalmazott több rendelést is kezelhet.

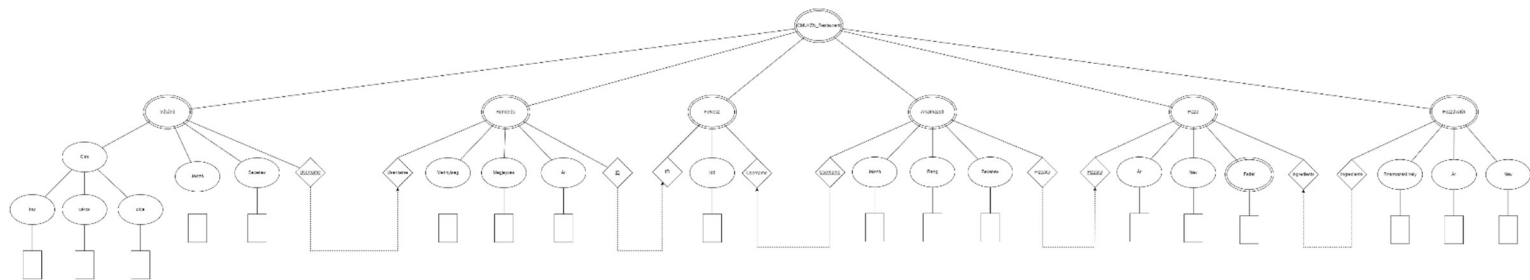
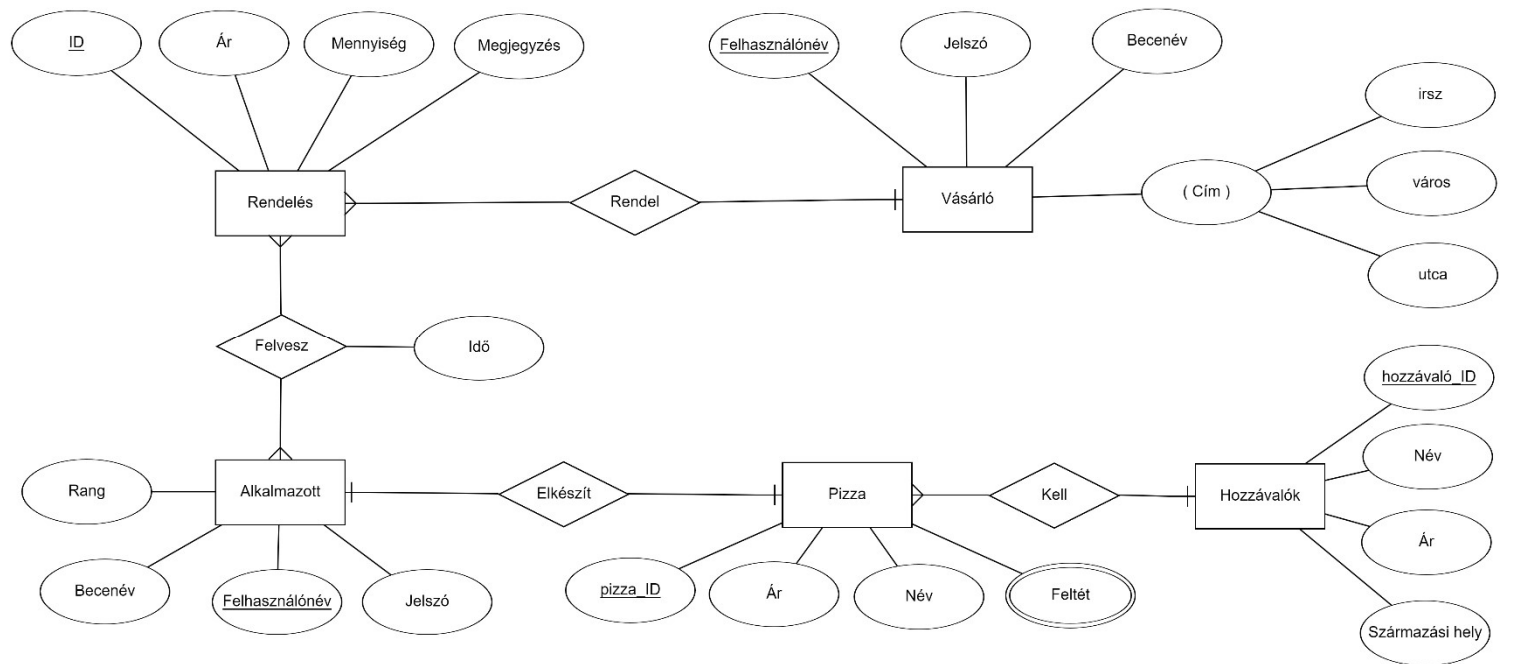
Alkalmazott-Pizza

- **Kapcsolat típusa:** 1:1 kapcsolat.
- Egy alkalmazott csak egy pizzát készíthet.

Pizza-Hozzávalók

- **Kapcsolat típusa:** 1:N kapcsolat.
- Egy pizzához több hozzávaló szükséges, és egy hozzávalót több pizza is felhasználhat.

4. Az ER model és XDM model



5. Pizza XML Schema leírás

- **Gyökérelem (`pizzeria_nyilvantartas`):**
 - A teljes pizzéria adatbázist tartalmazza.
 - Hierarchikusan szervezett főbb elemek: `vasarlok`, `rendelesek`, `alkalmazottak`, `pizzak`, és `hozzavalok`.
- **Entitások és kapcsolatok:**
 - **Vásárlók (`vasarlok`):**
 - Tartalmazza az egyes vásárlók adatait, mint például felhasználónév, jelszó, becenév és cím (irányítószám, város, utca, házszám).
 - A vásárlók egyedi azonosítóval rendelkeznek (`id` attribútum).
 - **Rendelések (`rendelesek`):**
 - Kapcsolódik a vásárlóhoz (`vasarlo_id`) és az alkalmazottakhoz (`alkalmazott_id`).
 - Minden rendeléshez tartozik egy vagy több pizza azonosító (`pizza_id`) és mennyiség.
 - **Alkalmazottak (`alkalmazottak`):**
 - Alkalmazotti adatok, például név, felhasználónév, jelszó, becenév és rang.
 - Minden alkalmazottnak van egy egyedi azonosítója (`id` attribútum).
 - **Pizzák (`pizzak`):**
 - Definiált pizzák neve, ára és feltétek listája.
 - **Hozzávalók (`hozzavalok`):**
 - Minden hozzávaló egyedi azonosítóval (`id`) rendelkezik, és tartalmazza a nevét, árát és származási helyét.

```

<!-- Root element -->
<xs:element name="CMU4ZN_Resaurant">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="customer" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="username" type="xs:string"/>
            <xs:element name="password" type="xs:string"/>
            <xs:element name="nickname" type="xs:string"/>
            <xs:element name="adress">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="city"
type="xs:string"/>
                  <xs:element name="street"
type="xs:string"/>
                  <xs:element name="zip_code"
type="xs:integer"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
          <xs:attribute name="customer_id" type="xs:string"
use="required"/>
        </xs:complexType>
        <xs:key name="CustomerPrimaryKey">
          <xs:selector xpath="."/>
          <xs:field xpath="@customer_id"/>
        </xs:key>
      </xs:element>

      <xs:element name="order" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="order_number"
type="xs:integer"/>
            <xs:element name="price" type="xs:decimal"/>
            <xs:element name="quantity" type="xs:integer"/>
            <xs:element name="note" type="xs:string"/>
          </xs:sequence>
          <xs:attribute name="order_id" type="xs:string"
use="required"/>
          <xs:attribute name="customer_id" type="xs:string"
use="required"/>
        </xs:complexType>
        <xs:key name="OrderPrimaryKey">
          <xs:selector xpath="."/>
          <xs:field xpath="@order_id"/>

```

```

        </xs:key>
        <xs:keyref name="OrderCustomerForeignKey"
refer="CustomerPrimaryKey">
            <xs:selector xpath="."/>
            <xs:field xpath="@customer_id"/>
        </xs:keyref>
    </xs:element>

    <xs:element name="employee" maxOccurs="unbounded">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="username" type="xs:string"/>
                <xs:element name="password" type="xs:string"/>
                <xs:element name="experience" type="xs:integer"/>
                <xs:element name="nickname" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="employee_id" type="xs:string"
use="required"/>
        </xs:complexType>
        <xs:key name="EmployeePrimaryKey">
            <xs:selector xpath="."/>
            <xs:field xpath="@employee_id"/>
        </xs:key>
    </xs:element>

    <xs:element name="order_handling" maxOccurs="unbounded">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="time" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="order_id" type="xs:string"
use="required"/>
            <xs:attribute name="employee_id" type="xs:string"
use="required"/>
        </xs:complexType>
        <xs:keyref name="OrderHandlingOrderForeignKey"
refer="OrderPrimaryKey">
            <xs:selector xpath="."/>
            <xs:field xpath="@order_id"/>
        </xs:keyref>
        <xs:keyref name="OrderHandlingEmployeeForeignKey"
refer="EmployeePrimaryKey">
            <xs:selector xpath="."/>
            <xs:field xpath="@employee_id"/>
        </xs:keyref>
    </xs:element>

    <xs:element name="pizza" maxOccurs="unbounded">
        <xs:complexType>

```

```

        <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="price" type="xs:decimal"/>
            <xs:element name="topping" type="xs:string"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="pizza_id" type="xs:string"
use="required"/>
    </xs:complexType>
    <xs:key name="PizzaPrimaryKey">
        <xs:selector xpath="."/>
        <xs:field xpath="@pizza_id"/>
    </xs:key>
</xs:element>

    <xs:element name="ingredients" maxOccurs="unbounded">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="name" type="xs:string"/>
                <xs:element name="price" type="xs:decimal"/>
                <xs:element name="origin" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="ingredients_id" type="xs:string"
use="required"/>
        </xs:complexType>
        <xs:key name="IngredientsPrimaryKey">
            <xs:selector xpath="."/>
            <xs:field xpath="@ingredients_id"/>
        </xs:key>
    </xs:element>

</xs:sequence>
</xs:complexType>
</xs:element>

```

6. A Java programok

6.1 DOMRead

A **DOMReadCMU4ZN.java** egy Java program, amely a DOM API-t használva olvassa be az XML fájlt (XMLCMU4ZN.xml), és annak teljes tartalmát hierarchikusan kiírja a konzolra. A program iterál az XML főbb elemein (customer, order, employee stb.), kiolvassa az attribútumaikat és gyermekelemeiket, majd azokat érthető formában jeleníti meg. Támogatja az összetett elemek, például címek vagy többször előforduló mezők (pl. pizzák feltétei) feldolgozását is. Célja, hogy az XML szerkezetét és tartalmát áttekinthetően megjelenítse, miközben az adatokat olvasásra előkészíti más programok számára.

Az alábbi példa bemutatja, hogy például a customer egyedeket hogyan olvassuk be:

```
// Customer elemek
NodeList customerList = doc.getElementsByTagName("customer");
System.out.println("\n--- Customers ---");
for (int i = 0; i < customerList.getLength(); i++) {
    Node node = customerList.item(i);

    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element element = (Element) node;
        System.out.println("Customer ID: " +
element.getAttribute("customer_id"));
        System.out.println("Username: " +
element.getElementsByTagName("username").item(0).getTextContent());
        System.out.println("Password: " +
element.getElementsByTagName("password").item(0).getTextContent());
        System.out.println("Nickname: " +
element.getElementsByTagName("nickname").item(0).getTextContent());
        Element address = (Element)
element.getElementsByTagName("address").item(0);
        System.out.println("Address: " +
address.getElementsByTagName("city").item(0).getTextContent() + ", "
+
address.getElementsByTagName("street").item(0).getTextContent() + ", "
+
address.getElementsByTagName("zip_code").item(0).getTextContent());
    }
}
```

6.2 DOMQuery

A program célja, hogy az XML tartalmából meghatározott elemeket vagy attribútumokat keressen ki, és azokat a konzolra írja. A lekérdezések nem használják az XPath-t, hanem DOM metódusokkal, például a `getElementsByTagName` vagy a `getAttribute` segítségével történnek.

A program például kereshet egy adott vásárlót azonosító (`customer_id`) alapján, vagy kilistázhathja az összes pizzát és azok árait. Támogatja az összetettebb kereséseket is, például olyan rendeléseket, amelyek tartalmazzák a "csípős" szót a megjegyzésben. Minden lekérdezés eredményét érthető formában írja ki a konzolra.

Itt egy példa az egyik lekérdezésre, ahol azt szeretnénk megtudni, hogy ki rendelt úgy pizzát, hogy a megjegyzésben szerepeltette a „csípős” szót:

```
// Lekérdezés 3: Az összes rendelés, ahol a megjegyzés tartalmazza a "csípős" szót
System.out.println("\nQuery 3: Orders with a note containing 'csípős'");
NodeList orderList = doc.getElementsByTagName("order");
for (int i = 0; i < orderList.getLength(); i++) {
    Node node = orderList.item(i);

    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element element = (Element) node;
        String note = element.getElementsByTagName("note").item(0).getTextContent();
        if (note.contains("csípős")) {
            System.out.println("Order ID: " + element.getAttribute("order_id"));
            System.out.println("Note: " + note);
        }
    }
}
```

6.3 DOMWrite

A program célja, hogy egy hierarchikusan felépített XML dokumentumot építsen fel programozott módon, majd azt egy fájlba (XMLCMU4ZN1.xml) mentse. Ezzel párhuzamosan a létrehozott XML fa struktúráját konzolra is kiírja.

A program először felépíti az XML gyökérelemét, majd hozzáadja a főbb entitásokat, például vásárlókat, rendeléseket, pizzákat és hozzávalókat. Minden elemhez attribútumokat és alárendelt elemeket is társít, például címadatokat vagy pizzák feltéteit. Az XML mentését a Transformer osztály végzi, amely a DOM fához tartozó adatokat megfelelően formázott XML fájlba exportálja.

Én rekurzívan oldottam meg a beolvasást, itt a módja:

```
// Rekurzív metódus az XML fa bejárására és kiírására

private static void printElement(Element element, int indent) {
    // Behúzás az aktuális elem szintjének megfelelően
    for (int i = 0; i < indent; i++) System.out.print(" ");

    // Elem neve
    System.out.print("<" + element.getTagName());

    // Attribútumok kiírása
    NamedNodeMap attributes = element.getAttributes();
    for (int i = 0; i < attributes.getLength(); i++) {
        Node attribute = attributes.item(i);
        System.out.print(" " + attribute.getNodeName() + "=\"" +
attribute.getNodeValue() + "\"");
    }
    System.out.println(">");

    // Gyerek elemek feldolgozása
    NodeList children = element.getChildNodes();
    for (int i = 0; i < children.getLength(); i++) {
        Node node = children.item(i);

        if (node.getNodeType() == Node.ELEMENT_NODE) {
            printElement((Element) node, indent + 1);
        } else if (node.getNodeType() == Node.TEXT_NODE) {
            // Szöveges tartalom kiírása (ha nem üres)
            String textContent = node.getTextContent().trim();
            if (!textContent.isEmpty()) {
                for (int j = 0; j <= indent; j++) System.out.print(" ");
                System.out.println(textContent);
            }
        }
    }
}
```


6.4 DOMModify

A program célja, hogy az XML dokumentumban bizonyos elemek vagy attribútumok értékeit programozottan megváltoztassa, és az eredményt egy új fájlba mentse.

A program például módosíthat egy vásárló felhasználónevét az id attribútuma alapján, frissítheti egy rendelés megjegyzését, vagy megváltoztathatja egy pizza árát. Támogatja új elemek hozzáadását is, például egy új feltétet adhat hozzá egy pizzához. Az XML fájl módosítása után a program a frissített tartalmat egy új fájlba (XMLCMU4ZN_modified.xml) menti a Transformer osztály segítségével.

Itt pedig egy példa lenne, ami előálhat a való életben is. A felhasználó értesített bennünket, hogy mégse csípősen kéri a pizzáját, ezért megváltoztatjuk azt:

```
// Módosítás 3: Egy rendelés megjegyzésének hozzáadása
    NodeList orderList = doc.getElementsByTagName("order");
    for (int i = 0; i < orderList.getLength(); i++) {
        Node node = orderList.item(i);

        if (node.getNodeType() == Node.ELEMENT_NODE) {
            Element element = (Element) node;
            if (element.getAttribute("order_id").equals("02")) {
                element.getElementsByTagName("note").item(0).setTextContent("Frissítve: mégse csípősen kérem!");
                System.out.println("Order 02's note updated to 'Frissítve: mégse csípősen kérem!'");
            }
        }
    }
}
```