**JEGYZŐKÖNYV**

Adatkezelés XML környezetben  
Féléves feladat

*Gépkocsi nyilvántartás*

Somody Máté  
IGRYHL  
2023

Tartalomjegyzék

[1. Témakör leírása 3](#_Toc120716771)

[1.1. Egyedek, attribútumok, kapcsolatok 3](#_Toc120716772)

[1.2. GitHub repository elérése 4](#_Toc120716773)

[2. Első feladat 5](#_Toc120716774)

[2.1. Az adatbázis ER modell 5](#_Toc120716775)

[2.2. Az adatbázis konvertálása XDM modellre 5](#_Toc120716776)

[2.3. Az XDM modell alapján XML dokumentum készítése 7](#_Toc120716777)

[2.4. Az XML dokumentum alapján XMLSchema készítése 10](#_Toc120716778)

[3. Második feladat 15](#_Toc120716779)

[3.1. Adatolvasás 15](#_Toc120716780)

[3.2. Adatmódosítás 20](#_Toc120716781)

[3.3. Adatlekérdezés 23](#_Toc120716782)

3.4. Adatírás.................................................................................................... 26

# 1. Témakör leírása

Az általam választott témakör: Személygépjárművek és tulajdonságaik. Ez a modell 5 egyeddel rendelkezik, melyek mindegyike reguláris elem, vagyis a modell nem rendelkezik gyengén típusos komponenssel. Ha a kapcsolatokat figyeljük meg, akkor megkülönböztethetünk identifikáló, és nem identifikáló kapcsolatokat.

## 1.1. Egyedek, attribútumok, kapcsolatok

Az alábbi kapcsolatok egyike sem identifikáló:

* Motor-Személygépjármű (HAS\_A, egy a többhöz)
* Gyártó-Személygépjármű (HAS\_A, egy a többhöz)
* Karosszéria-Személygépjármű (HAS\_A, egy az egyhez)
* Tulajdonos-Személygépjármű (HAS\_A, egy a többhöz)

Személygépjármű

* Rendszám (elsődleges kulcs), két betű, két betű, három szám
* alváz száma (idegen kulcs)
* tulaj\_jogsi (idegen kulcs)
* motor\_száma (idegen kulcs)
* gyártás\_sorszám (idegen kulcs)
* Kor (a Gyártási év mezőből származtatott)
* CASCO azonosító (opcionális) – min előfordulás 0

Tulajdonos

* Jogosítvány száma (elsődleges kulcs), két betű, hat szám
* Név (összetett), elemei: Vezetéknév, Keresztnév

Motor:

* Motorszám (elsődleges kulcs), 14 betű vagy szám
* üzemanyag (lehet benzin, dízel, hibrid vagy elektromos)
* Lóerő
* Nyomaték

Karosszéria:

* Alvázszám (elsődleges kulcs), 17 betű vagy szám
* Felület (matt, részleges, teljes fényezés)
* Szín (többértékű)
* Matricák (van vagy nincs)

Gyártás:

* GyártásID (elsődleges kulcs), 2 szám 1 betű
* márka
* modell
* Gyártási év (valós évszám kell legyen)

## 1.2. GitHub repository elérése

<https://github.com/SomodyMate/IGRYHL_XMLGyak/>

# 2. Első feladat

## 2.1. Az adatbázis ER modell

## A képen képernyőkép, diagram, kör, tervezés látható Automatikusan generált leírás2.2. Az adatbázis konvertálása XDM modellre

Az ER modellben szereplő egyedeket ellipszissel,   
a tulajdonságokat ellipszissel és téglalappal,   
az elsődleges kulcsokat rombusszal ábrázoljuk.   
A többszörös előfordulást dupla vonallal jelöljük.   
Az adatbázis root eleme a *gepjarmu* nevet kapja.

*A képen képernyőkép, fekete, sor, művészet látható

Automatikusan generált leírás*

## 2.3. Az XDM modell alapján XML dokumentum készítése

Az XDM modell alapján késztettem el az XML dokumentumot, kezdve a root elementtel. Figyeltem arra, hogy a gyerekelemekből legalább 3 legyen, mert úgy szemléletes, illetve az opcionális elementből valahol van, valahol nincs, a többértékűnél pedig előfordul egy, kettő illetve 3 is egy parenten belül.

Forráskód A képen szöveg, képernyőkép, szoftver látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, menü látható

Automatikusan generált leírás

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

Forráskód vége

## 2.4. Az XML dokumentum alapján XMLSchema készítése

Az XML dokumentum validálásához saját sémát késztettem, melyben próbáltam kreatív lenni, validációs megkötésnek használtam saját típusokat és olyat is, amit az xsd nyelv tartalmaz alapvetően (például xs:gYear). A séma szerkezetét azalábbiak szerint alakítottam: egyszerű elemek, amikre később referálok, saját típusok, amikkel a témakör sajátos megkötéseit érvényesítem, felépítés, elsődleges kulcsok majd végül idegen kulcsok.

Forráskód

A képen szöveg, képernyőkép, menü látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, menü, Betűtípus látható

Automatikusan generált leírás

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, menü látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, Betűtípus, menü látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, menü látható

Automatikusan generált leírás

Forráskód vége

# 3. Második feladat

## 3.1. Adatolvasás

A teljes XML dokumentum feldolgozásához létrehoztam egy osztályszintű metódust, amit kétszer hívok meg a kódban, eredményét egyszer a konzolra írjuk ki, egyszer pedig szöveges dokumentumba. A *Feldolgozas()* metóduson belül először inicializáljuk a beolvasandó fájlt, majd a parsert és kiírjuk a root elementet. Ezután követve az XML dokumentum struktúráját, minden eltárolt adatot megjelenítünk. Az opcionális elemnél nincset írunk, ha nem található abban a parent elementben, míg a többszörös előfordulásoknál for ciklussal számozzuk és kiírjuk az összes értéket.

Forráskód:

package hu.domparse.IGRYHL;

import java.io.File;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.PrintStream;

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 DomReadIGRYHL {

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

// Adatok kiirasa console-ra

*Feldolgozas*();

// Adatok mentése TXT-be

PrintStream out = new PrintStream(new FileOutputStream("XML\_Parse\_output.txt"));

System.*setOut*(out);

*Feldolgozas*();

}

private static void Feldolgozas() throws ParserConfigurationException, SAXException, IOException {

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

DocumentBuilderFactory factory = DocumentBuilderFactory.*newInstance*();

DocumentBuilder dBuilder = factory.newDocumentBuilder();

Document doc = dBuilder.parse(xmlFile);

doc.getDocumentElement().normalize();

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

// szemelygepjarmuvek kiirasa

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

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

Node nNode = nList.item(i);

System.***out***.println("\nCurrent element: " + nNode.getNodeName());

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

Element elem = (Element) nNode;

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

Node node1 = elem.getElementsByTagName("tulaj\_jogsi").item(0);

String tj = node1.getTextContent();

Node node2 = elem.getElementsByTagName("motor\_szama").item(0);

String msz = node2.getTextContent();

Node node3 = elem.getElementsByTagName("alvaz\_szama").item(0);

String asz = node3.getTextContent();

Node node4 = elem.getElementsByTagName("gyartas\_sorszam").item(0);

String gys = node4.getTextContent();

String ca = "nincs";

Node node5 = elem.getElementsByTagName("casco\_azon").item(0);

if (node5 != null) {

ca = node5.getTextContent();

}

System.***out***.println("Rendszam: " + rendszam);

System.***out***.println("Tulaj jogsi: " + tj);

System.***out***.println("Motor szama: " + msz);

System.***out***.println("Alvaz szama: " + asz);

System.***out***.println("Gyartasi sorszam: " + gys);

System.***out***.println("Casco azonosito: " + ca);

}

}

// tulajdonosok kiirasa

nList = doc.getElementsByTagName("tulajdonos");

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

Node nNode = nList.item(i);

System.***out***.println("\nCurrent element: " + nNode.getNodeName());

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

Element elem = (Element) nNode;

String jogsi = elem.getAttribute("jogositvany\_szama");

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

String vn = node1.getTextContent();

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

String kn = node2.getTextContent();

System.***out***.println("Jogositvany szama: " + jogsi);

System.***out***.println("Vezeteknev: " + vn);

System.***out***.println("Keresztnev: " + kn);

}

}

// motorok kiirasa

nList = doc.getElementsByTagName("motor");

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

Node nNode = nList.item(i);

System.***out***.println("\nCurrent element: " + nNode.getNodeName());

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

Element elem = (Element) nNode;

String msz = elem.getAttribute("motorszam");

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

String ua = node1.getTextContent();

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

String le = node2.getTextContent();

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

String ny = node3.getTextContent();

System.***out***.println("Motorszam: " + msz);

System.***out***.println("Uzemamyag: " + ua);

System.***out***.println("Loero: " + le);

System.***out***.println("Nyomatek: " + ny);

}

}

// karosszeriak kiirasa

nList = doc.getElementsByTagName("karosszeria");

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

Node nNode = nList.item(i);

System.***out***.println("\nCurrent element: " + nNode.getNodeName());

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

Element elem = (Element) nNode;

String asz = elem.getAttribute("alvazszam");

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

String ft = node1.getTextContent();

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

String ma = node2.getTextContent();

System.***out***.println("Alvazszam: " + asz);

System.***out***.println("Felulet: " + ft);

System.***out***.println("Matrica: " + ma);

for (int a = 0; a < elem.getElementsByTagName("szin").getLength(); a++) {

Node node3 = elem.getElementsByTagName("szin").item(a);

String sz = node3.getTextContent();

System.***out***.println("Szin #" + (a + 1) + ": " + sz);

}

}

}

// gyártási infók kiirasa

nList = doc.getElementsByTagName("gyartas");

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

Node nNode = nList.item(i);

System.***out***.println("\nCurrent element: " + nNode.getNodeName());

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

Element elem = (Element) nNode;

String gyi = elem.getAttribute("gyartasID");

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

String mk = node1.getTextContent();

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

String me = node2.getTextContent();

Node node3 = elem.getElementsByTagName("gyartasi\_ev").item(0);

String gye = node3.getTextContent();

System.***out***.println("GyartasID: " + gyi);

System.***out***.println("Marka: " + mk);

System.***out***.println("Modell: " + me);

System.***out***.println("Gyartasi ev: " + gye);

}

}

return;

}

}

Kimenet:

**A képen szöveg, képernyőkép, menü, Betűtípus látható

Automatikusan generált leírásA képen szöveg, menü, képernyőkép, Betűtípus látható

Automatikusan generált leírás**

## 3.2. Adatmódosítás

Az adatmódosítás során cél, hogy az eredeti XML dokumentumból olvassunk be, de egy másikban (XMLIGRYHLModify.xml) tároljuk el a módosításokat, így nem vész el az első fájlunk. Az alábbi 5 változtatást végezzük el:

* A második személygépjármű elem rendszámát átírjuk
* Az első tulajdonos jogosítvány számát átírjuk
* Az összes karosszérián beállítjuk, hogy van matrica
* A benzines motorokat hibridre állítjuk
* A *gyartasi\_ev* elemnevet *evjaratra* módosítjuk

Forráskód:

package hu.domparse.IGRYHL;

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.Transformer;

import javax.xml.transform.TransformerException;

import javax.xml.transform.TransformerFactory;

import javax.xml.transform.dom.DOMSource;

import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Document;

import org.w3c.dom.NamedNodeMap;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;

public class DomModifyIGRYHL {

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

File inputFile = new File("XMLIGRYHL.xml");

DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.*newInstance*();

DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBuilder();

Document doc = documentBuilder.parse(inputFile);

doc.getDocumentElement().normalize();

// második személygépjármű attribútumának módosítása

Node jarmu = doc.getElementsByTagName("szemelygepjarmu").item(1);

NamedNodeMap attr = jarmu.getAttributes();

Node nodeAttr = attr.getNamedItem("rendszam");

nodeAttr.setTextContent("MM-AB-123");

System.***out***.println("szemelygepjarmu attributum modositva");

// első tulajdonos attribútumának módosítása

Node tulaj = doc.getElementsByTagName("tulajdonos").item(0);

attr = tulaj.getAttributes();

nodeAttr = attr.getNamedItem("jogositvany\_szama");

nodeAttr.setTextContent("FR346544");

System.***out***.println("tulajdonos attributum modositva");

// összes matrica beállitása van-ra

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

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

Node node = nodes.item(i);

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

NodeList childNodes = node.getChildNodes();

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

Node childNode = childNodes.item(j);

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

childNode.setTextContent("van");

}

}

}

}

System.***out***.println("matrica modositasok kesz");

// összes benzines motor átállitása hibridre

nodes = doc.getElementsByTagName("motor");

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

Node node = nodes.item(i);

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

NodeList childNodes = node.getChildNodes();

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

Node childNode = childNodes.item(j);

if (childNode.getNodeName().equals("uzemanyag") && childNode.getTextContent().equals("benzin")) {

childNode.setTextContent("hibrid");

}

}

}

}

System.***out***.println("benzin motorok atallitva hibridre");

// gyartasi\_ev element atnevezese evjaratra

nodes = doc.getElementsByTagName("gyartasi\_ev");

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

doc.renameNode(nodes.item(i), null, "evjarat");

}

System.***out***.println("gyartasi\_ev atnevezeve evjaratra");

// modositott xml dokumentum elmentese

*writeXml*(doc, new File("XMLIGRYHL2.xml"));

}

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

Transformer transformer = TransformerFactory.*newInstance*().newTransformer();

DOMSource source = new DOMSource(doc);

StreamResult file = new StreamResult(output);

transformer.transform(source, file);

}

}

Forráskód vége

## 3.3. Adatlekérdezés

Az adatlekérdezés során rendezett keretek között nyerünk ki információt az adatbázisunkból, viszont, hogy az eredmény letisztult legyen, a forráskódban a lekérdezések ki vannak kommentelve. Ezek a lekérdezések az alábbiak:

* Az adatbázisban szereplő összes tulajdonos és minden adatuk
* Az adatbázisban 10B azonosítóval szereplő gyártási infók
* Az adatbázis utolsó motor rekordja
* Az adatbázisban található összes olyan karosszéria, melynek elsődleges színe a fehér
* Az adatbázis összes olyan gyártási infója, ami 2008 utáni

Forráskód:

package hu.domparse.IGRYHL;

import java.io.IOException;

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.XPathExpressionException;

import javax.xml.xpath.XPathFactory;

import org.w3c.dom.Document;

import org.w3c.dom.NodeList;

import org.w3c.dom.Node;

import org.w3c.dom.Element;

import org.xml.sax.SAXException;

public class DomQueryIGRYHL {

public static void main(String[] args) {

try {

DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.*newInstance*();

DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBuilder();

Document document = documentBuilder.parse("XMLIGRYHL.xml");

document.getDocumentElement().normalize();

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

//Lekerdezesek

String expression = "";

//expression = "/gepjarmu/tulajdonos";

//expression = "/gepjarmu/gyartas[@gyartasID='10B']";

//expression = "/gepjarmu/motor[last()]";

//expression = "/gepjarmu/karosszeria[szin='fehér']";

expression = "/gepjarmu/gyartas[gyartasi\_ev>2008]";

NodeList nodeList = (NodeList) xPath.compile(expression).evaluate(document, XPathConstants.***NODESET***);

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

Node node = nodeList.item(i);

System.***out***.println("\nAktuális elem: " + node.getNodeName());

if (node.getNodeType() == Node.***ELEMENT\_NODE*** && node.getNodeName().equals("szemelygepjarmu")) {

Element element = (Element) node;

System.***out***.println("Rendszám: " + element.getAttribute("rendszam"));

System.***out***.println("Tulaj jogsi: " + element.getElementsByTagName("tulaj\_jogsi").item(0).getTextContent());

System.***out***.println("Alváz száma: " + element.getElementsByTagName("alvaz\_szama").item(0).getTextContent());

System.***out***.println("Motor száma: " + element.getElementsByTagName("motor\_szama").item(0).getTextContent());

System.***out***.println("Gyártási szám: " + element.getElementsByTagName("gyartas\_sorszam").item(0).getTextContent());

}

if (node.getNodeType() == Node.***ELEMENT\_NODE*** && node.getNodeName().equals("tulajdonos")) {

Element element = (Element) node;

System.***out***.println("Jogositvány száma: " + element.getAttribute("jogositvany\_szama"));

System.***out***.println("Keresztnév: " + element.getElementsByTagName("keresztnev").item(0).getTextContent());

System.***out***.println("Vezetéknév: " + element.getElementsByTagName("vezeteknev").item(0).getTextContent());

}

if (node.getNodeType() == Node.***ELEMENT\_NODE*** && node.getNodeName().equals("motor")) {

Element element = (Element) node;

System.***out***.println("Motor száma: " + element.getAttribute("motorszam"));

System.***out***.println("Üzemanyag: " + element.getElementsByTagName("uzemanyag").item(0).getTextContent());

System.***out***.println("Lóerő: " + element.getElementsByTagName("loero").item(0).getTextContent());

System.***out***.println("Nyomaték: " + element.getElementsByTagName("nyomatek").item(0).getTextContent());

}

if (node.getNodeType() == Node.***ELEMENT\_NODE*** && node.getNodeName().equals("karosszeria")) {

Element element = (Element) node;

System.***out***.println("Alvázszám: " + element.getAttribute("alvazszam"));

System.***out***.println("Felület: " + element.getElementsByTagName("felulet").item(0).getTextContent());

System.***out***.println("Matrica: " + element.getElementsByTagName("matrica").item(0).getTextContent());

System.***out***.println("Elsődleges szin: " + element.getElementsByTagName("szin").item(0).getTextContent());

}

if (node.getNodeType() == Node.***ELEMENT\_NODE*** && node.getNodeName().equals("gyartas")) {

Element element = (Element) node;

System.***out***.println("ID: " + element.getAttribute("gyartasID"));

System.***out***.println("Márka: " + element.getElementsByTagName("marka").item(0).getTextContent());

System.***out***.println("Modell: " + element.getElementsByTagName("modell").item(0).getTextContent());

System.***out***.println("Gyártási év: " + element.getElementsByTagName("gyartasi\_ev").item(0).getTextContent());

}

}

} catch (ParserConfigurationException e) {

e.printStackTrace();

} catch (SAXException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

} catch (XPathExpressionException e) {

e.printStackTrace();

}

}

}

Forráskód vége

Lekérdezések eredménye (kimenet):

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírásA képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírásA képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírásA képen szöveg, Betűtípus, képernyőkép látható

Automatikusan generált leírás

## 3.4. Adatírás

Az adatírás feladatban egy olyan DOM API programra volt szükség amely az XMLIGRYHL.xml fájlomat beolvassa, majd fa struktúra formájában kiíratja a konzolra és elmenti egy XMLIGRYHL1.xml nevezetű új fájlba.