**JEGYZŐKÖNYV**

**Adatkezelés XML környezetben**

Féléves feladat

**Facebookhoz hasonló közösségi médiás nyilvántartása**

**Készítette:** Flaskó Lilian Laura

**Neptun kód:** GCNS8S

**Gyakorlatvezető:** Dr. Bednarik László

**Képzés:** Programtervező informatikus

Tartalomjegyzék

[1. Feladat leírása 3](#_Toc149337347)

[1a.) Az egyedek tulajdonságai 3](#_Toc149337348)

[1b.) A feladat ER Modellje 5](#_Toc149337349)

[1c.) Az egyedek közötti kapcsolatok 5](#_Toc149337350)

[2. Az ER modell konvertálása XDM modellé 6](#_Toc149337351)

[3. XML dokumentum készítése 6](#_Toc149337352)

[4. XMLSchema készítése XML dokumentum alapján 6](#_Toc149337353)

[5. DOM program készítése JAVA környezetben 18](#_Toc149337354)

[5a.) DOM adatolvasás 18](#_Toc149337355)

[5b.) DOM adatlekérdezés 33](#_Toc149337356)

[5c.) DOM adatmódosítás 40](#_Toc149337357)

[5d.) DOM adatírás 48](#_Toc149337358)

# 1. Feladat leírása

A féléves beadandóm tematikája egy Facebookhoz hasonló közösségi média nyilvántartás létrehozása, ahol XML-ben tartanánk nyilván a felhasználókat, érdeklődési körüket, postjaikat és csoportjaikat. Az XML dokumentumban tudunk beolvasni, írni, lekérdezni és módosítani.

## 1a.) Az egyedek tulajdonságai

* **User Entity**

Attribútumok:

* + UserID (Primary Key)
  + Name
  + Email
  + UserTheme
* **Post Entity**

Attribútumok:

* + PostID (Primary Key)
  + Text
  + Date
  + LikesNumber
* **Image Entity**

Attribútumok:

* + ImageID (Primary Key)
  + ImgName
  + ImgSize
  + Resolution
* **Comment Entity**

Attribútumok:

* + CommentID (Primary Key)
  + ComText
  + ComDate
  + LikesNum
* **Group Entity**

Attribútumok:

* + GroupID (Primary Key)
  + Tulajdonságok:
  + GroupName
  + Description
  + MembersNum
* **SignedUp Entity**

Attribútumok:

* + GroupID (Foreign Key)
  + UserID (Foreign Key)
  + JoinDate

## 1b.) A feladat ER Modellje

A képen diagram, rajz, vázlat, Tervrajz látható

Automatikusan generált leírás

## 1c.) Az egyedek közötti kapcsolatok

* Felhasználó - Poszt (1:n)
  + Egy felhasználó több posztot készíthet, de egy poszt csak egy felhasználóhoz tartozhat.
  + Kapcsolat: Egy Felhasználó több Poszttal kapcsolatban van.
* Felhasználó - Komment (1:n)
  + Egy felhasználó több kommentet írhat, de egy komment csak egy felhasználóhoz tartozhat.
  + Kapcsolat: Egy Felhasználó több Kommenttel kapcsolatban van.
* Poszt - Kép (1:1)
  + Egy poszthoz egy kép tartozik, és egy képhez csak egy poszt kapcsolódik.
  + Kapcsolat: Egy Poszthoz egy Kép tartozik.
* Felhasználó - Csoport (n:m)
  + Egy felhasználó több csoportban lehet, és egy csoport több felhasználót tartalmazhat.
  + Kapcsolat: Egy Felhasználó több Csoporttal kapcsolatban van, és egy Csoport több Felhasználót tartalmazhat.
* Poszt - Komment (1:n)
  + Egy poszthoz több komment tartozhat, de egy komment csak egy poszthoz kapcsolódhat.
  + Kapcsolat: Egy Poszthoz több Komment tartozik.

# 2. Az ER modell konvertálása XDM modellé

XDM modellnél háromféle jelölés használunk: ellipszist, rombuszt és téglalapot. Az ellipszis jelöli az elemeket, minden egyedből elem lesz, illetve a tulajdonságokból is. A rombusz jelöli az attribútumokat, amelyek a kulcs tulajdonságokból keletkeznek. A téglalap jelöli a szöveget, amely majd az XML dokumentumban fog megjelenni. Azoknak az elemeknek, amelyek többször is előfordulhatnak, a jelölése dupla ellipszissel történik. Az idegenkulcsok és a kulcsok közötti kapcsolatot szaggatott vonalas nyíllal jelöljük.

**XDM modell:**

A képen vázlat, fehér látható

Automatikusan generált leírás

# 3. XML dokumentum készítése

Az XDM modell alapján készítettem el az XML dokumentumot. Legelőször a gyökérelemmel kezdtem, amelynek a **„Facebook”** nevet adtam. Ezek után a gyermekelemeiből eltérő módon hoztam létre példányokat.

**XML dokumentum forráskódja:**

<?xml version="1.0"?>  
<facebook xmlns="file://C:/Users/flask/IdeaProjects/DOMParseGCNS8S/src/Sources/scheme\_GCNS8S.xsd">  
  
 <User UserID="1" CommentID="301" PostID="101">  
 <Name>John Doe</Name>  
 <Email>john@example.com</Email>  
 <UserTheme>Theme1</UserTheme>  
 <Address>  
 <ZipCode>3600</ZipCode>  
 <City>Budapest</City>  
 <StreetAddress>Main Street 6</StreetAddress>  
 </Address>  
 </User>  
 <User UserID="2" CommentID="302" PostID="102">  
 <Name>Jane Smith</Name>  
 <Email>jane@example.com</Email>  
 <UserTheme>Theme2</UserTheme>  
 <Address>  
 <ZipCode>7400</ZipCode>  
 <City>New York</City>  
 <StreetAddress>First Avenue 4</StreetAddress>  
 </Address>  
 </User>  
 <User UserID="3" CommentID="303" PostID="103">  
 <Name>Bob Johnson</Name>  
 <Email>bob@example.com</Email>  
 <UserTheme>Theme3</UserTheme>  
 <Address>  
 <ZipCode>8900</ZipCode>  
 <City>London</City>  
 <StreetAddress>Oxford street 5</StreetAddress>  
 </Address>  
 </User>  
  
  
 <Post PostID="101" CommentID="301" ImageID="201">  
 <Text>This is the first post.</Text>  
 <Date>2023-01-01</Date>  
 <LikesNumber>5</LikesNumber>  
 </Post>  
 <Post PostID="102" CommentID="302" ImageID="202">  
 <Text>Another post here.</Text>  
 <Date>2023-02-01</Date>  
 <LikesNumber>8</LikesNumber>  
 </Post>  
 <Post PostID="103" CommentID="303" ImageID="203">  
 <Text>Yet another post.</Text>  
 <Date>2023-03-01</Date>  
 <LikesNumber>12</LikesNumber>  
 </Post>  
  
  
 <Image ImageID="201">  
 <ImgName>image1.jpg</ImgName>  
 <ImgSize>1024</ImgSize>  
 <Resolution>1920x1080</Resolution>  
 </Image>  
 <Image ImageID="202">  
 <ImgName>image2.jpg</ImgName>  
 <ImgSize>2048</ImgSize>  
 <Resolution>2560x1440</Resolution>  
 </Image>  
 <Image ImageID="203">  
 <ImgName>image3.jpg</ImgName>  
 <ImgSize>4096</ImgSize>  
 <Resolution>3840x2160</Resolution>  
 </Image>  
  
 <Comment CommentID="301" PostID="101" UserID="1">  
 <ComText>Comment 1</ComText>  
 <ComDate>2023-01-05</ComDate>  
 <LikesNum>3</LikesNum>  
 </Comment>  
 <Comment CommentID="302" PostID="102" UserID="2">  
 <ComText>Comment 2</ComText>  
 <ComDate>2023-02-10</ComDate>  
 <LikesNum>1</LikesNum>  
 </Comment>  
 <Comment CommentID="303" PostID="103" UserID="3">  
 <ComText>Comment 3</ComText>  
 <ComDate>2023-03-15</ComDate>  
 <LikesNum>7</LikesNum>  
 </Comment>  
  
 <Group GroupID="401">  
 <GroupName>Developers</GroupName>  
 <Description>Programming enthusiasts</Description>  
 <MembersNum>10</MembersNum>  
 </Group>  
 <Group GroupID="402">  
 <GroupName>Photographers</GroupName>  
 <Description>Passionate about photography</Description>  
 <MembersNum>8</MembersNum>  
 </Group>  
 <Group GroupID="403">  
 <GroupName>Travelers</GroupName>  
 <Description>Exploring the world</Description>  
 <MembersNum>15</MembersNum>  
 </Group>  
  
 <SignedUp GroupID="401" UserID="1">  
 <JoinDate>2023-01-01</JoinDate>  
 </SignedUp>  
 <SignedUp GroupID="402" UserID="2">  
 <JoinDate>2023-02-01</JoinDate>  
 </SignedUp>  
 <SignedUp GroupID="403" UserID="3">  
 <JoinDate>2023-03-01</JoinDate>  
 </SignedUp>  
</facebook>

# 4. XMLSchema készítése XML dokumentum alapján

Az XML dokumentumhoz készíteni kellett egy validációt elősegítő sémát. Először kigyűjtöttem az egyszerű típusokat, majd meghatároztam a saját típusokat. Összesen 4 egyszerű saját típust hoztam létre. Például a dátumhoz készítettem egy olyan saját típust, amely regext tartalmaz. Ezek után elkészítettem a komplex típusokat minden elemre, majd elsődleges- és idegenkulcsokat hoztam létre.

**XMLSchema forráskódja:**

<?xml version="1.0" encoding="UTF-8"?>  
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">  
  
 <!-- Egyszerű típusok kigyűjtése -->  
 <!-- User -->  
 <xs:element name="Name" type="xs:string" />  
 <xs:element name="UserTheme" type="xs:string" />  
 <xs:element name="Email" type="EmailType" />  
  
 <!-- Post -->  
 <xs:element name="Text" type="xs:string" />  
 <xs:element name="LikesNumber" type="xs:integer" />  
 <xs:element name="Date" type="DateType" />  
  
 <!-- Comment -->  
 <xs:element name="ComText" type="xs:string" />  
 <xs:element name="LikesNum" type="xs:integer" />  
 <xs:element name="ComDate" type="DateType" />  
  
 <!-- Image -->  
 <xs:element name="ImgName" type="xs:string" />  
 <xs:element name="ImgSize" type="xs:integer" />  
 <xs:element name="Resolution" type="ResolutionType" />  
  
 <!-- Group -->  
 <xs:element name="GroupName" type="xs:string" />  
 <xs:element name="Description" type="xs:string" />  
 <xs:element name="MembersNum" type="xs:integer" />  
  
 <!-- SignedUp -->  
 <xs:element name="JoinDate" type="DateType" />  
  
 <!-- Egyszerű saját típusok meghatározása -->  
 <xs:simpleType name="DateType">  
 <xs:restriction base="xs:string">  
 <xs:pattern value="[12]\d{3}-(0[1-9]|1[0-2])-(0[1-9]|[12]\d|3[01])"/>  
 </xs:restriction>  
 </xs:simpleType>  
  
 <xs:simpleType name="EmailType">  
 <xs:restriction base="xs:string">  
 <xs:pattern value="[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}"/>  
 </xs:restriction>  
 </xs:simpleType>  
  
 <xs:simpleType name="ResolutionType">  
 <xs:restriction base="xs:string">  
 <xs:pattern value="\d{3,5}x\d{3,5}"/>  
 </xs:restriction>  
 </xs:simpleType>  
  
 <xs:simpleType name="StreetAddressType">  
 <xs:restriction base="xs:string">  
 <xs:pattern value="[a-zA-Z0-9\s]+ \d+"/>  
 </xs:restriction>  
 </xs:simpleType>  
  
 <!-- Komplex típusokhoz saját típus meghatározása -->  
  
  
 <xs:complexType name="UserType">  
 <xs:sequence>  
 <xs:element ref="Name" />  
 <xs:element ref="UserTheme" minOccurs="0" maxOccurs="20"/>  
 <xs:element ref="Email" />  
 <xs:element name="Address">  
 <xs:complexType>  
 <xs:sequence>  
 <xs:element name="ZipCode" type="xs:integer"/>  
 <xs:element name="City" type="xs:string"/>  
 <xs:element name="StreetAddress" type="StreetAddressType"/>  
 </xs:sequence>  
 </xs:complexType>  
 </xs:element>  
 </xs:sequence>  
 <xs:attribute name="UserID" type="xs:integer" use="required" />  
 <xs:attribute name="PostID" type="xs:integer" />  
 <xs:attribute name="CommentID" type="xs:integer" />  
 </xs:complexType>  
  
 <xs:complexType name="PostType">  
 <xs:sequence>  
 <xs:element ref="Text" />  
 <xs:element ref="Date" />  
 <xs:element ref="LikesNumber" />  
 </xs:sequence>  
 <xs:attribute name="PostID" type="xs:integer" use="required" />  
 <xs:attribute name="ImageID" type="xs:integer" />  
 <xs:attribute name="CommentID" type="xs:integer" />  
 </xs:complexType>  
  
 <xs:complexType name="ImageType">  
 <xs:sequence>  
 <xs:element ref="ImgName" />  
 <xs:element ref="ImgSize" />  
 <xs:element ref="Resolution" />  
 </xs:sequence>  
 <xs:attribute name="ImageID" type="xs:integer" use="required" />  
 </xs:complexType>  
  
 <xs:complexType name="CommentType">  
 <xs:sequence>  
 <xs:element ref="ComText" />  
 <xs:element ref="ComDate" />  
 <xs:element ref="LikesNum" />  
 </xs:sequence>  
 <xs:attribute name="CommentID" type="xs:integer" use="required" />  
 <xs:attribute name="PostID" type="xs:integer" />  
 <xs:attribute name="UserID" type="xs:integer" />  
 </xs:complexType>  
  
 <xs:complexType name="GroupType">  
 <xs:sequence>  
 <xs:element ref="GroupName" />  
 <xs:element ref="Description" />  
 <xs:element ref="MembersNum" />  
 </xs:sequence>  
 <xs:attribute name="GroupID" type="xs:integer" use="required" />  
 </xs:complexType>  
  
 <xs:complexType name="SignedUpType">  
 <xs:sequence>  
 <xs:element ref="JoinDate" />  
 </xs:sequence>  
 <xs:attribute name="GroupID" type="xs:integer" />  
 <xs:attribute name="UserID" type="xs:integer" />  
 </xs:complexType>  
  
 <!-- Gyökérelem -->  
 <xs:element name="facebook">  
  
 <xs:complexType>  
 <xs:sequence>  
 <xs:element name="User" type="UserType" minOccurs="0" maxOccurs="unbounded"/>  
 <xs:element name="Post" type="PostType" minOccurs="0" maxOccurs="unbounded"/>  
 <xs:element name="Image" type="ImageType" minOccurs="0" maxOccurs="unbounded"/>  
 <xs:element name="Comment" type="CommentType" minOccurs="0" maxOccurs="unbounded"/>  
 <xs:element name="Group" type="GroupType" minOccurs="0" maxOccurs="unbounded"/>  
 <xs:element name="SignedUp" type="SignedUpType" minOccurs="0" maxOccurs="unbounded"/>  
 </xs:sequence>  
 </xs:complexType>  
 <!--Elsődleges kulcsok-->  
 <xs:unique name="UserIDKey">  
 <xs:selector xpath="User"/>  
 <xs:field xpath="@UserID"/>  
 </xs:unique>  
  
 <xs:unique name="PostIDKey">  
 <xs:selector xpath="Post"/>  
 <xs:field xpath="@PostID"/>  
 </xs:unique>  
  
 <xs:unique name="ImageIDKey">  
 <xs:selector xpath="Image"/>  
 <xs:field xpath="@ImageID"/>  
 </xs:unique>  
  
 <xs:unique name="CommentIDKey">  
 <xs:selector xpath="Comment"/>  
 <xs:field xpath="@CommentID"/>  
 </xs:unique>  
  
 <xs:unique name="GroupIDKey">  
 <xs:selector xpath="Group"/>  
 <xs:field xpath="@GroupID"/>  
 </xs:unique>  
 <xs:unique name="SignedUpKey">  
 <xs:selector xpath="SignedUp"/>  
 <xs:field xpath="@GroupID"/>  
 <xs:field xpath="@UserID"/>  
 </xs:unique>  
 <!--Idegen kulcsok-->  
 <xs:keyref name="User\_Post" refer="UserIDKey">  
 <xs:selector xpath="Post"/>  
 <xs:field xpath="@PostID"/>  
 </xs:keyref>  
  
 <xs:keyref name="User\_Comment" refer="UserIDKey">  
 <xs:selector xpath="Comment"/>  
 <xs:field xpath="@CommentID"/>  
 </xs:keyref>  
  
 <xs:keyref name="Post\_Image" refer="PostIDKey">  
 <xs:selector xpath="Image"/>  
 <xs:field xpath="@ImageID"/>  
 </xs:keyref>  
  
 <xs:keyref name="Post\_Comment" refer="PostIDKey">  
 <xs:selector xpath="Comment"/>  
 <xs:field xpath="@CommentID"/>  
 </xs:keyref>  
  
  
 <xs:keyref name="Comment\_Post" refer="CommentIDKey">  
 <xs:selector xpath="Post"/>  
 <xs:field xpath="@PostID"/>  
 </xs:keyref>  
  
 <xs:keyref name="Comment\_User" refer="CommentIDKey">  
 <xs:selector xpath="User"/>  
 <xs:field xpath="@UserID"/>  
 </xs:keyref>  
  
 <xs:keyref name="SignedUp\_Group" refer="SignedUpKey">  
 <xs:selector xpath="Group"/>  
 <xs:field xpath="@GroupID"/>  
 </xs:keyref>  
  
 <xs:keyref name="SignedUp\_USer" refer="SignedUpKey">  
 <xs:selector xpath="User"/>  
 <xs:field xpath="@UserID"/>  
 </xs:keyref>  
</xs:element>  
</xs:schema>

# 5. DOM program készítése JAVA környezetben

A DOM programokat JAVA-ban készítettem el, ahogy a feladatkiírásban is szerepelt. Az alábbi programokat a következő alfejezetekben fogom részletesebben taglalni.

## 5a.) DOM adatolvasás

**’** **DomReadGCNS8S’ osztály:** Ez az osztály egy XML fájl elemzésére szolgál a DOM (Document Object Model) technológiát használva. Az osztály több metódust tartalmaz az XML adatok strukturált feldolgozásához.

main metódus:

A program belépési pontja. Egyszerűen meghívja a readXMLDocument metódust.

readXMLDocument metódus:

A readXMLFile metódust használja az XML fájl beolvasásához.

Az XML dokumentum gyökérelemét (facebook) olvassa be.

Egy for ciklusban végigmegy az összes gyerek elemen, és az elem típusa alapján feldolgozza azokat.

A switch utasítás segítségével kezeli az egyes elem típusokat, mint pl. User, Post, Image, stb.

Az elemek attribútumait és gyerek elemeit olvassa be, majd kiírja azokat a konzolra.

readXMLFile metódus:

Az XML fájl beolvasását végzi el a DocumentBuilder és DocumentBuilderFactory segítségével.

**DomReadGCNS8S.java forráskódja:**

package hu.domparse.GCNS8S;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import org.w3c.dom.NodeList;  
  
  
public class DomReadGCNS8S {  
  
 public static void main(String[] args) {  
  
 *readXMLDocument*();  
  
 }  
  
 private static void readXMLDocument() {  
 try {  
 // XML fájl beolvasása  
 Document document = *readXMLFile*();  
  
 Element facebookElement = document.getDocumentElement();  
  
 // User, Post és Image elemek beolvasása egyetlen for ciklusban  
 NodeList nodeList = facebookElement.getChildNodes();  
 for (int i = 0; i < nodeList.getLength(); i++) {  
 if (nodeList.item(i) instanceof Element element) {  
  
 // User elem beolvasása  
 switch (element.getNodeName()) {  
 case "User" -> {  
 String userID = element.getAttribute("UserID");  
 String name = element.getElementsByTagName("Name").item(0).getTextContent();  
 String email = element.getElementsByTagName("Email").item(0).getTextContent();  
  
  
  
 System.*out*.println("User ID: " + userID);  
 System.*out*.println("Name: " + name);  
 System.*out*.println("Email: " + email);  
  
 System.*out*.println();  
 }  
  
 // Post elem beolvasása  
 case "Post" -> {  
 String postID = element.getAttribute("PostID");  
 String text = element.getElementsByTagName("Text").item(0).getTextContent();  
 String date = element.getElementsByTagName("Date").item(0).getTextContent();  
  
  
  
 System.*out*.println("Post ID: " + postID);  
 System.*out*.println("Text: " + text);  
 System.*out*.println("Date: " + date);  
  
 System.*out*.println();  
 }  
  
 // Image elem beolvasása  
 case "Image" -> {  
 String imageID = element.getAttribute("ImageID");  
 String imgName = element.getElementsByTagName("ImgName").item(0).getTextContent();  
 String imgSize = element.getElementsByTagName("ImgSize").item(0).getTextContent();  
 String resolution = element.getElementsByTagName("Resolution").item(0).getTextContent();  
  
  
  
 System.*out*.println("Image ID: " + imageID);  
 System.*out*.println("Image Name: " + imgName);  
 System.*out*.println("Image Size: " + imgSize);  
 System.*out*.println("Resolution: " + resolution);  
  
 System.*out*.println();  
 }  
 // Comment elem beolvasása  
 case "Comment" -> {  
 String commentID = element.getAttribute("CommentID");  
 String comText = element.getElementsByTagName("ComText").item(0).getTextContent();  
 String comDate = element.getElementsByTagName("ComDate").item(0).getTextContent();  
  
  
  
 System.*out*.println("Comment ID: " + commentID);  
 System.*out*.println("Comment Text: " + comText);  
 System.*out*.println("Comment Date: " + comDate);  
  
 System.*out*.println();  
 }  
 // Group elem beolvasása  
 case "Group" -> {  
 String groupID = element.getAttribute("GroupID");  
 String groupName = element.getElementsByTagName("GroupName").item(0).getTextContent();  
 String description = element.getElementsByTagName("Description").item(0).getTextContent();  
 String membersNum = element.getElementsByTagName("MembersNum").item(0).getTextContent();  
  
  
  
 System.*out*.println("Group ID: " + groupID);  
 System.*out*.println("Group Name: " + groupName);  
 System.*out*.println("Description: " + description);  
 System.*out*.println("Members Number: " + membersNum);  
  
 System.*out*.println();  
 }  
  
 // SignedUp elem beolvasása  
 case "SignUP" -> {  
 String joinDate = element.getElementsByTagName("JoinDate").item(0).getTextContent();  
 String userID = element.getAttribute("UserID");  
 String groupID = element.getAttribute("GroupID");  
  
  
  
 System.*out*.println("Join Date: " + joinDate);  
 System.*out*.println("User ID: " + userID);  
 System.*out*.println("Group ID: " + groupID);  
  
 System.*out*.println();  
 }  
 }  
  
  
 }  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 }  
  
 private static Document readXMLFile() throws Exception {  
 // XML fájl beolvasása  
 DocumentBuilderFactory factory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder builder = factory.newDocumentBuilder();  
 return builder.parse("./facebookGCNS8S.xml");  
 }  
}

**Output:**

C:\Users\flask\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Users\flask\Desktop\IntelliJ IDEA Community Edition 2023.3.1\lib\idea\_rt.jar=4679:C:\Users\flask\Desktop\IntelliJ IDEA Community Edition 2023.3.1\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\flask\IdeaProjects\DOMParseGCNS8S\out\production\DOMParseGCNS8S;C:\Users\flask\IdeaProjects\DOMParseGCNS8S\lib\dom-2.3.0-jaxb-1.0.6.jar hu.domparse.GCNS8S.DomReadGCNS8S

## User ID: 1

## Name: John Doe

## Email: john@example.com

## User ID: 2

## Name: Jane Smith

## Email: jane@example.com

## User ID: 3

## Name: Bob Johnson

## Email: bob@example.com

## Post ID: 101

## Text: This is the first post.

## Date: 2023-01-01

## Post ID: 102

## Text: Another post here.

## Date: 2023-02-01

## Post ID: 103

## Text: Yet another post.

## Date: 2023-03-01

## Image ID: 201

## Image Name: image1.jpg

## Image Size: 1024

## Resolution: 1920x1080

## Image ID: 202

## Image Name: image2.jpg

## Image Size: 2048

## Resolution: 2560x1440

## Image ID: 203

## Image Name: image3.jpg

## Image Size: 4096

## Resolution: 3840x2160

## Comment ID: 301

## Comment Text: Comment 1

## Comment Date: 2023-01-05

## Comment ID: 302

## Comment Text: Comment 2

## Comment Date: 2023-02-10

## Comment ID: 303

## Comment Text: Comment 3

## Comment Date: 2023-03-15

## Group ID: 401

## Group Name: Developers

## Description: Programming enthusiasts

## Members Number: 10

## Group ID: 402

## Group Name: Photographers

## Description: Passionate about photography

## Members Number: 8

## Group ID: 403

## Group Name: Travelers

## Description: Exploring the world

## Members Number: 15

## Process finished with exit code 0

## 5b.) DOM adatlekérdezés

**’** **DomQueryGCNS8S’ osztály:** Ez az osztály olyan metódusokat tartalmaz, amelyek egy XML fájl tartalmát képes beolvasni és feldolgozni DOM segítségével, ráadásul lekérdezéseket is képesek vagyunk végrehajtani.

main: A fő metódus, amely inicializálja a lekérdezéseket és meghívja azokat.

queryUsers: Az összes felhasználó adatainak kiírása.

queryPostsForUser: Egy adott felhasználóhoz tartozó posztok lekérdezése.

queryImagesForPost: Egy adott poszthoz tartozó képek lekérdezése.

queryCommentsWithLikes: Azoknak a kommenteknek a lekérdezése, amelyeknek a "LikesNum" értéke nagyobb mint 1.

queryGroupWithMostMembers: A legtöbb taggal rendelkező csoport lekérdezése.

**Segédmetódusok**:

findGroupWithMostMembers: A legtöbb taggal rendelkező csoport kiválasztása.

getChildValue: Az egyes elemek gyermekének értékét adja vissza.

**readXMLFile metódus:**

Az XML fájl beolvasásáért felelős. A fájl útvonalát a kódban rögzítve használja fel.

**DomQueryGCNS8S.java forráskódja:**

package hu.domparse.GCNS8S;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import org.w3c.dom.NodeList;  
import java.util.HashMap;  
import java.util.Map;  
  
public class DomQueryGCNS8S {  
 public static void main(String[] args) {  
 try {  
 // Fájl beolvasása és feldolgozása  
 Document document = *readXMLFile*();  
  
 // Példa lekérdezések hívása  
 *queryUsers*(document);  
 *queryPostsForUser*(document, "1");  
 *queryPostsForUser*(document, "2");  
 *queryImagesForPost*(document, 2);  
 *queryImagesForPost*(document, 1);  
 *queryCommentsWithLikes*(document);  
 *queryGroupWithMostMembers*(document);  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
 // Példa: Összes User kiírása  
 private static void queryUsers(Document document) {  
 System.*out*.println("### Users ###");  
 NodeList userList = document.getElementsByTagName("User");  
 for (int i = 0; i < userList.getLength(); i++) {  
 Element user = (Element) userList.item(i);  
 String userID = user.getAttribute("UserID");  
 String name = user.getElementsByTagName("Name").item(0).getTextContent();  
 String email = user.getElementsByTagName("Email").item(0).getTextContent();  
 System.*out*.println("UserID: " + userID + ", Name: " + name + ", Email: " + email);  
 }  
 System.*out*.println();  
 }  
 private static void queryPostsForUser(Document document, String userID) {  
 System.*out*.println("### Posts for User " + userID + " ###");  
 NodeList postList = document.getElementsByTagName("Post");  
 for (int i = 0; i < postList.getLength(); i++) {  
 Element post = (Element) postList.item(i);  
 String postUserID = post.getAttribute("UserID");  
 if (userID.equals(postUserID)) {  
 String postID = post.getAttribute("PostID");  
 String text = *getChildValue*(post, "Text");  
 String date = *getChildValue*(post, "Date");  
 String likesNumber = *getChildValue*(post, "LikesNumber");  
 System.*out*.println("PostID: " + postID + ", Text: " + text + ", Date: " + date + ", LikesNumber: " + likesNumber);  
 }  
 }  
 System.*out*.println();  
 }  
 private static void queryImagesForPost(Document document, int postId) {  
 System.*out*.println("### Images for PostID " + postId + " ###");  
 NodeList postList = document.getElementsByTagName("Post");  
 for (int i = 0; i < postList.getLength(); i++) {  
 Element post = (Element) postList.item(i);  
 int postID = Integer.*parseInt*(post.getAttribute("PostID"));  
  
 // Megfelelő PostID-hez tartozó képek lekérdezése  
 if (postID == postId) {  
 NodeList imageList = post.getElementsByTagName("Image");  
 for (int j = 0; j < imageList.getLength(); j++) {  
 Element image = (Element) imageList.item(j);  
 String imgName = *getChildValue*(image, "ImgName");  
 int imgSize = Integer.*parseInt*(*getChildValue*(image, "ImgSize"));  
  
 System.*out*.println("ImgName: " + imgName + ", ImgSize: " + imgSize);  
 }  
 }  
 }  
 System.*out*.println();  
 }  
 // Példa: Kommentek lekérdezése Like-ok számával  
 private static void queryCommentsWithLikes(Document document) {  
 System.*out*.println("### Comments with Likes > 1 ###");  
 NodeList commentList = document.getElementsByTagName("Comment");  
 for (int i = 0; i < commentList.getLength(); i++) {  
 Element comment = (Element) commentList.item(i);  
 int likesNum = Integer.*parseInt*(*getChildValue*(comment, "LikesNum"));  
  
 // Megfelelő Like számmal rendelkező kommentek lekérdezése  
 if (likesNum > 1) {  
 int commentID = Integer.*parseInt*(comment.getAttribute("CommentID"));  
 String comText = *getChildValue*(comment, "ComText");  
 int userID = Integer.*parseInt*(comment.getAttribute("UserID"));  
  
 System.*out*.println("CommentID: " + commentID + ", ComText: " + comText +  
 ", LikesNum: " + likesNum + ", UserID: " + userID);  
 }  
 }  
 System.*out*.println();  
 }  
  
 // Példa: A legtöbb taggal rendelkező csoport lekérdezése  
 private static void queryGroupWithMostMembers(Document document) {  
 NodeList groupList = document.getElementsByTagName("Group");  
  
 // Map a csoportok és tagjaik számának tárolásához  
 Map<String, Integer> groupMembersCount = new HashMap<>();  
  
 // Csoportok bejárása és tagok számának számolása  
 for (int i = 0; i < groupList.getLength(); i++) {  
 Element groupElement = (Element) groupList.item(i);  
 String groupName = groupElement.getElementsByTagName("GroupName").item(0).getTextContent();  
 int membersNum = Integer.*parseInt*(groupElement.getElementsByTagName("MembersNum").item(0).getTextContent());  
  
 groupMembersCount.put(groupName, membersNum);  
 }  
  
 // Legtöbb taggal rendelkező csoport kiválasztása  
 String groupWithMostMembers = *findGroupWithMostMembers*(groupMembersCount);  
  
 // Eredmény kiíratása  
 if (groupWithMostMembers != null) {  
 System.*out*.println("### Group with most members: " + groupWithMostMembers + " ###");  
 } else {  
 System.*out*.println("### There is no clear answer. Multiple groups have the same maximum number of members. ###");  
 }  
 }  
  
 // Legtöbb taggal rendelkező csoport kiválasztása  
 private static String findGroupWithMostMembers(Map<String, Integer> groupMembersCount) {  
 String groupWithMostMembers = null;  
 int maxMembers = -1;  
  
 for (Map.Entry<String, Integer> entry : groupMembersCount.entrySet()) {  
 if (entry.getValue() > maxMembers) {  
 groupWithMostMembers = entry.getKey();  
 maxMembers = entry.getValue();  
 } else if (entry.getValue() == maxMembers) {  
 // Több csoport is azonos maximális tagszámmal  
 groupWithMostMembers = null;  
 }  
 }  
  
 return groupWithMostMembers;  
 }  
  
  
 private static String getChildValue(Element parent, String childName) {  
 Element child = (Element) parent.getElementsByTagName(childName).item(0);  
 return child.getTextContent();  
 }  
 // XML fájl beolvasása  
 private static Document readXMLFile() throws Exception {  
 DocumentBuilderFactory factory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder builder = factory.newDocumentBuilder();  
 return builder.parse("C:\\Users\\flask\\IdeaProjects\\DOMParseGCNS8S\\src\\Sources\\facebookGCNS8S.xml");  
 }  
 }

**Output:**

C:\Users\flask\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Users\flask\Desktop\IntelliJ IDEA Community Edition 2023.3.1\lib\idea\_rt.jar=46592:C:\Users\flask\Desktop\IntelliJ IDEA Community Edition 2023.3.1\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\flask\IdeaProjects\DOMParseGCNS8S\out\production\DOMParseGCNS8S;C:\Users\flask\IdeaProjects\DOMParseGCNS8S\lib\dom-2.3.0-jaxb-1.0.6.jar hu.domparse.GCNS8S.DomQueryGCNS8S

### Users ###

UserID: 1, Name: John Doe, Email: john@example.com

UserID: 2, Name: Jane Smith, Email: jane@example.com

UserID: 3, Name: Bob Johnson, Email: bob@example.com

### Posts for User 1 ###

### Posts for User 2 ###

### Images for PostID 2 ###

### Images for PostID 1 ###

### Comments with Likes > 1 ###

CommentID: 301, ComText: Comment 1, LikesNum: 3, UserID: 1

CommentID: 303, ComText: Comment 3, LikesNum: 7, UserID: 3

### Group with most members: Travelers ###

Process finished with exit code 0

## 5c.) DOM adatmódosítás

**’** **DomModifyGCNS8S’ osztály:** Ez az osztály végzi el az XML fájlban a módosításokat a DOM használatával.

main: A fő metódus, amely inicializálja a módosításokat és meghívja azokat.

modifyElementValue: Egy adott elem értékének módosítása a dokumentumban. Megkeresi az elemet a megadott attribútum értékek alapján és módosítja a cél elem értékét.

saveXMLFile: A módosított dokumentum mentése XML fájlba.

**modifyElementValue metódus:**

Paraméterek:

Document document: Az XML dokumentum, amelyben a módosításokat végezzük.

String elementName: Az elem neve, amelyet módosítani szeretnénk.

String attributeName: Az attribútum neve, amely alapján az elemet azonosítjuk.

String attributeValue: Az attribútum értéke, amely alapján az elemet azonosítjuk.

String targetElementName: A célelem neve, amelynek az értékét módosítani szeretnénk.

String newValue: Az új érték, amelyre módosítjuk a célelem értékét.

**saveXMLFile metódus:**

Paraméterek:

Document document: A módosított XML dokumentum, amelyet menteni szeretnénk.

A metódus a DOM API segítségével menti a módosított dokumentumot egy új fájlba.

**readXMLFile metódus:**

Visszatérési érték: Document.

Az XML fájl beolvasásáért felelős metódus, amely visszaadja az olvasott dokumentumot.

**DomModifyGCNS8S.java forráskódja:**

package hu.domparse.GCNS8S;  
import java.io.FileOutputStream;  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.transform.Transformer;  
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;  
  
public class DomModifyGCNS8S {  
  
  
  
  
 public static void main(String[] args) {  
 try {  
 // Fájl beolvasása és feldolgozása  
 Document document = *readXMLFile*();  
  
 // Példa módosítások hívása  
 *modifyElementValue*(document, "User", "UserID", "1", "Name", "ModifiedLili");  
 *modifyElementValue*(document, "Group", "GroupID", "2", "Description", " Lili modified the description");  
 *modifyElementValue*(document, "Post", "PostID", "3", "Text", "Modified by Lili");  
 *modifyElementValue*(document, "Comment", "CommentID", "1", "ComText", "Modified by Lili");  
 *modifyElementValue*(document, "Image", "ImageID", "1", "ImageSize", "500x600");  
  
 // Módosított dokumentum mentése  
 *saveXMLFile*(document);  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
 // Példa: Adott elem értékének módosítása  
 private static void modifyElementValue(Document document, String elementName, String attributeName,  
 String attributeValue, String targetElementName, String newValue) {  
 NodeList nodeList = document.getElementsByTagName(elementName);  
 for (int i = 0; i < nodeList.getLength(); i++) {  
 Element element = (Element) nodeList.item(i);  
 if (element.getAttribute(attributeName).equals(attributeValue)) {  
 NodeList targetNodeList = element.getElementsByTagName(targetElementName);  
 if (targetNodeList.getLength() > 0) {  
 Node targetNode = targetNodeList.item(0);  
 targetNode.setTextContent(newValue);  
 }  
 }  
 }  
 }  
  
 // XML dokumentum mentése fájlba  
 private static void saveXMLFile(Document document) throws Exception {  
 TransformerFactory transformerFactory = TransformerFactory.*newInstance*();  
 Transformer transformer = transformerFactory.newTransformer();  
 DOMSource source = new DOMSource(document);  
 StreamResult result = new StreamResult(new FileOutputStream("C:\\Users\\flask\\IdeaProjects\\DOMParseGCNS8S\\src\\Sources\\facebookGCNS8S.xml"));  
 transformer.transform(source, result);  
 System.*out*.println("Módosított XML fájl elmentve: " + "./facebookGCNS8S.xml");  
 }  
  
 // XML fájl beolvasása  
 private static Document readXMLFile() throws Exception {  
 DocumentBuilderFactory factory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder builder = factory.newDocumentBuilder();  
 return builder.parse("C:\\Users\\flask\\IdeaProjects\\DOMParseGCNS8S\\src\\Sources\\facebookGCNS8S.xml");  
 }  
 }

**Output:**

**C:\Users\flask\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Users\flask\Desktop\IntelliJ IDEA Community Edition 2023.3.1\lib\idea\_rt.jar=47740:C:\Users\flask\Desktop\IntelliJ IDEA Community Edition 2023.3.1\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\flask\IdeaProjects\DOMParseGCNS8S\out\production\DOMParseGCNS8S;C:\Users\flask\IdeaProjects\DOMParseGCNS8S\lib\dom-2.3.0-jaxb-1.0.6.jar hu.domparse.GCNS8S.DomModifyGCNS8S**

**Módosított XML fájl elmentve: ./facebookGCNS8S.xml**

**Process finished with exit code 0**

## 5d.) DOM adatírás

**’** **DOMWriteGCNS8S’ osztály:** Ez az osztály valósítja meg az XML dokumentum létrehozását és abban való írását.

**main metódus:**

Inicializálja a DOM környezetet. Létrehozza az XML dokumentumot, majd hozzáadja az alapvető elemeket és adatokat. Kiírja az XML-t a konzolra. Elmenti az XML dokumentumot egy fájlba.

**addUserData metódus:**

Hozzáad egy "User" elemet az XML dokumentumhoz a megadott adatokkal. Az adatokat attribútumok és gyermek elemek formájában adja hozzá az "User" elemhez.

**addPostData metódus:**

Hozzáad egy "Post" elemet az XML dokumentumhoz a megadott adatokkal. Az adatokat attribútumok és gyermek elemek formájában adja hozzá a "Post" elemhez.

**addImageData metódus:**

Hozzáad egy "Image" elemet az XML dokumentumhoz a megadott adatokkal. Az adatokat attribútumok és gyermek elemek formájában adja hozzá az "Image" elemhez.

**addCommentData metódus:**

Hozzáad egy "Comment" elemet az XML dokumentumhoz a megadott adatokkal. Az adatokat attribútumok és gyermek elemek formájában adja hozzá a "Comment" elemhez.

**addGroupData metódus:**

Hozzáad egy "Group" elemet az XML dokumentumhoz a megadott adatokkal. Az adatokat attribútumok és gyermek elemek formájában adja hozzá a "Group" elemhez.

**addSignedUpData metódus:**

Hozzáad egy "SignedUp" elemet az XML dokumentumhoz a megadott adatokkal. Az adatokat attribútumok és gyermek elemek formájában adja hozzá a "SignedUp" elemhez.

**createElement metódus:**

Hozzáad egy gyermek elemet a szülő elemhez a megadott névvel és szöveges tartalommal.

**printXMLToConsole metódus:**

Kiírja az XML dokumentumot a konzolra.

**saveXMLToFile metódus:**

Elmenti az XML dokumentumot egy megadott fájlba.

**DOMWriteGCNS8S.java forráskódja:**

package hu.domparse.GCNS8S;  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.transform.Transformer;  
import javax.xml.transform.TransformerFactory;  
import javax.xml.transform.dom.DOMSource;  
import javax.xml.transform.stream.StreamResult;  
import javax.xml.transform.\*;  
import java.io.File;  
  
  
public class DOMWriteGCNS8S {  
  
 public static void main(String[] args) {  
 try {  
 DocumentBuilderFactory docFactory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder docBuilder = docFactory.newDocumentBuilder();  
 Document doc = docBuilder.newDocument();  
  
 // Root element  
 Element rootElement = doc.createElement("facebook");  
 doc.appendChild(rootElement);  
  
 // User elements  
 *addUserData*(doc, rootElement, "1", "John Doe", "john@example.com", "Theme1", "3600", "Budapest", "Main Street 6");  
 *addUserData*(doc, rootElement, "2", "Jane Smith", "jane@example.com", "Theme2", "7400", "New York", "First Avenue 4");  
 *addUserData*(doc, rootElement, "3", "Bob Johnson", "bob@example.com", "Theme3", "8900", "London", "Oxford street 5");  
  
 // Post elements  
 *addPostData*(doc, rootElement, "101", "This is the first post.", "2023-01-01", "5", "201");  
 *addPostData*(doc, rootElement, "102", "Another post here.", "2023-02-01", "8", "202");  
 *addPostData*(doc, rootElement, "103", "Yet another post.", "2023-03-01", "12", "203");  
  
 // Image elements  
 *addImageData*(doc, rootElement, "201", "image1.jpg", "1024", "1920x1080");  
 *addImageData*(doc, rootElement, "202", "image2.jpg", "2048", "2560x1440");  
 *addImageData*(doc, rootElement, "203", "image3.jpg", "4096", "3840x2160");  
  
 // Comment elements  
 *addCommentData*(doc, rootElement, "301", "Comment 1", "2023-01-05", "3", "101", "1");  
 *addCommentData*(doc, rootElement, "302", "Comment 2", "2023-02-10", "1", "102", "2");  
 *addCommentData*(doc, rootElement, "303", "Comment 3", "2023-03-15", "7", "103", "3");  
  
 // Group elements  
 *addGroupData*(doc, rootElement, "401", "Developers", "Programming enthusiasts", "10");  
 *addGroupData*(doc, rootElement, "402", "Photographers", "Passionate about photography", "8");  
 *addGroupData*(doc, rootElement, "403", "Travelers", "Exploring the world", "15");  
  
 // SignedUp elements  
 *addSignedUpData*(doc, rootElement, "2023-01-01", "1", "401");  
 *addSignedUpData*(doc, rootElement, "2023-02-01", "2", "402");  
 *addSignedUpData*(doc, rootElement, "2023-03-01", "3", "403");  
  
 // Print the XML to console  
 *printXMLToConsole*(doc);  
  
 // Save the document to a file  
 *saveXMLToFile*(doc, "C:/Users/flask/Desktop/xml/Beadanadó/xmlGCNS8S1.xml");  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
 // Function to add User data  
 private static void addUserData(Document doc, Element rootElement, String userID, String name, String email, String userTheme, String zipCode, String city, String streetAddress) {  
 Element userElement = doc.createElement("User");  
 userElement.setAttribute("UserID", userID);  
 userElement.setAttribute("CommentID", "301");  
 userElement.setAttribute("PostID", "101");  
  
 *createElement*(doc, userElement, "Name", name);  
 *createElement*(doc, userElement, "Email", email);  
 *createElement*(doc, userElement, "UserTheme", userTheme);  
  
 Element addressElement = doc.createElement("Address");  
 *createElement*(doc, addressElement, "ZipCode", zipCode);  
 *createElement*(doc, addressElement, "City", city);  
 *createElement*(doc, addressElement, "StreetAddress", streetAddress);  
  
 userElement.appendChild(addressElement);  
 rootElement.appendChild(userElement);  
 }  
  
 // Function to add Post data  
 private static void addPostData(Document doc, Element rootElement, String postID, String text, String date, String likesNumber, String imageID) {  
 Element postElement = doc.createElement("Post");  
 postElement.setAttribute("PostID", postID);  
 postElement.setAttribute("CommentID", "301");  
 postElement.setAttribute("ImageID", imageID);  
  
 *createElement*(doc, postElement, "Text", text);  
 *createElement*(doc, postElement, "Date", date);  
 *createElement*(doc, postElement, "LikesNumber", likesNumber);  
  
 rootElement.appendChild(postElement);  
 }  
  
 // Function to add Image data  
 private static void addImageData(Document doc, Element rootElement, String imageID, String imgName, String imgSize, String resolution) {  
 Element imageElement = doc.createElement("Image");  
 imageElement.setAttribute("ImageID", imageID);  
  
 *createElement*(doc, imageElement, "ImgName", imgName);  
 *createElement*(doc, imageElement, "ImgSize", imgSize);  
 *createElement*(doc, imageElement, "Resolution", resolution);  
  
 rootElement.appendChild(imageElement);  
 }  
  
 // Function to add Comment data  
 private static void addCommentData(Document doc, Element rootElement, String commentID, String comText, String comDate, String likesNum, String postID, String userID) {  
 Element commentElement = doc.createElement("Comment");  
 commentElement.setAttribute("CommentID", commentID);  
 commentElement.setAttribute("PostID", postID);  
 commentElement.setAttribute("UserID", userID);  
  
 *createElement*(doc, commentElement, "ComText", comText);  
 *createElement*(doc, commentElement, "ComDate", comDate);  
 *createElement*(doc, commentElement, "LikesNum", likesNum);  
  
 rootElement.appendChild(commentElement);  
 }  
  
 // Function to add Group data  
 private static void addGroupData(Document doc, Element rootElement, String groupID, String groupName, String description, String membersNum) {  
 Element groupElement = doc.createElement("Group");  
 groupElement.setAttribute("GroupID", groupID);  
  
 *createElement*(doc, groupElement, "GroupName", groupName);  
 *createElement*(doc, groupElement, "Description", description);  
 *createElement*(doc, groupElement, "MembersNum", membersNum);  
  
 rootElement.appendChild(groupElement);  
 }  
  
 // Function to add SignedUp data  
 private static void addSignedUpData(Document doc, Element rootElement, String joinDate, String userID, String groupID) {  
 Element signedUpElement = doc.createElement("SignedUp");  
 signedUpElement.setAttribute("GroupID", groupID);  
 signedUpElement.setAttribute("UserID", userID);  
  
 *createElement*(doc, signedUpElement, "JoinDate", joinDate);  
  
 rootElement.appendChild(signedUpElement);  
 }  
  
 // Function to create and append an element with text content  
 private static void createElement(Document doc, Element parentElement, String elementName, String textContent) {  
 Element element = doc.createElement(elementName);  
 element.appendChild(doc.createTextNode(textContent));  
 parentElement.appendChild(element);  
 }  
  
 // Function to print the XML to console  
 private static void printXMLToConsole(Document doc) {  
 try {  
 TransformerFactory transformerFactory = TransformerFactory.*newInstance*();  
 Transformer transformer = transformerFactory.newTransformer();  
 transformer.setOutputProperty(OutputKeys.*INDENT*, "yes");  
  
 DOMSource source = new DOMSource(doc);  
 StreamResult consoleResult = new StreamResult(System.*out*);  
 transformer.transform(source, consoleResult);  
  
 } catch (TransformerException e) {  
 e.printStackTrace();  
 }  
 }  
  
 // Function to save the XML to a file  
 private static void saveXMLToFile(Document doc, String fileName) {  
 try {  
 TransformerFactory transformerFactory = TransformerFactory.*newInstance*();  
 Transformer transformer = transformerFactory.newTransformer();  
 transformer.setOutputProperty(OutputKeys.*INDENT*, "yes");  
  
 DOMSource source = new DOMSource(doc);  
 StreamResult fileResult = new StreamResult(new File(fileName));  
 transformer.transform(source, fileResult);  
  
 } catch (TransformerException e) {  
 e.printStackTrace();  
 }  
 }  
}

**Output:**

C:\Users\flask\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Users\flask\Desktop\IntelliJ IDEA Community Edition 2023.3.1\lib\idea\_rt.jar=48394:C:\Users\flask\Desktop\IntelliJ IDEA Community Edition 2023.3.1\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\flask\IdeaProjects\DOMParseGCNS8S\out\production\DOMParseGCNS8S;C:\Users\flask\IdeaProjects\DOMParseGCNS8S\lib\dom-2.3.0-jaxb-1.0.6.jar hu.domparse.GCNS8S.DOMWriteGCNS8S

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<facebook>

<User CommentID="301" PostID="101" UserID="1">

<Name>John Doe</Name>

<Email>john@example.com</Email>

<UserTheme>Theme1</UserTheme>

<Address>

<ZipCode>3600</ZipCode>

<City>Budapest</City>

<StreetAddress>Main Street 6</StreetAddress>

</Address>

</User>

<User CommentID="301" PostID="101" UserID="2">

<Name>Jane Smith</Name>

<Email>jane@example.com</Email>

<UserTheme>Theme2</UserTheme>

<Address>

<ZipCode>7400</ZipCode>

<City>New York</City>

<StreetAddress>First Avenue 4</StreetAddress>

</Address>

</User>

<User CommentID="301" PostID="101" UserID="3">

<Name>Bob Johnson</Name>

<Email>bob@example.com</Email>

<UserTheme>Theme3</UserTheme>

<Address>

<ZipCode>8900</ZipCode>

<City>London</City>

<StreetAddress>Oxford street 5</StreetAddress>

</Address>

</User>

<Post CommentID="301" ImageID="201" PostID="101">

<Text>This is the first post.</Text>

<Date>2023-01-01</Date>

<LikesNumber>5</LikesNumber>

</Post>

<Post CommentID="301" ImageID="202" PostID="102">

<Text>Another post here.</Text>

<Date>2023-02-01</Date>

<LikesNumber>8</LikesNumber>

</Post>

<Post CommentID="301" ImageID="203" PostID="103">

<Text>Yet another post.</Text>

<Date>2023-03-01</Date>

<LikesNumber>12</LikesNumber>

</Post>

<Image ImageID="201">

<ImgName>image1.jpg</ImgName>

<ImgSize>1024</ImgSize>

<Resolution>1920x1080</Resolution>

</Image>

<Image ImageID="202">

<ImgName>image2.jpg</ImgName>

<ImgSize>2048</ImgSize>

<Resolution>2560x1440</Resolution>

</Image>

<Image ImageID="203">

<ImgName>image3.jpg</ImgName>

<ImgSize>4096</ImgSize>

<Resolution>3840x2160</Resolution>

</Image>

<Comment CommentID="301" PostID="101" UserID="1">

<ComText>Comment 1</ComText>

<ComDate>2023-01-05</ComDate>

<LikesNum>3</LikesNum>

</Comment>

<Comment CommentID="302" PostID="102" UserID="2">

<ComText>Comment 2</ComText>

<ComDate>2023-02-10</ComDate>

<LikesNum>1</LikesNum>

</Comment>

<Comment CommentID="303" PostID="103" UserID="3">

<ComText>Comment 3</ComText>

<ComDate>2023-03-15</ComDate>

<LikesNum>7</LikesNum>

</Comment>

<Group GroupID="401">

<GroupName>Developers</GroupName>

<Description>Programming enthusiasts</Description>

<MembersNum>10</MembersNum>

</Group>

<Group GroupID="402">

<GroupName>Photographers</GroupName>

<Description>Passionate about photography</Description>

<MembersNum>8</MembersNum>

</Group>

<Group GroupID="403">

<GroupName>Travelers</GroupName>

<Description>Exploring the world</Description>

<MembersNum>15</MembersNum>

</Group>

<SignedUp GroupID="401" UserID="1">

<JoinDate>2023-01-01</JoinDate>

</SignedUp>

<SignedUp GroupID="402" UserID="2">

<JoinDate>2023-02-01</JoinDate>

</SignedUp>

<SignedUp GroupID="403" UserID="3">

<JoinDate>2023-03-01</JoinDate>

</SignedUp>

</facebook>

Process finished with exit code 0