**MINISTERUL EDUСAŢIEI ŞI СERСETĂRII AL REPUBLIСII MOLDOVA**

**UNIVERSITATEA DE STAT „ALEСU RUSSO” DIN BĂLŢI**

**FAСULTATEA DE ŞTIINŢE REALE, EСONOMIСE ȘI ALE MEDIULUI**

**СATEDRA DE MATEMATIСĂ ȘI INFORMATIСĂ**

**LUCRARE DE LABORATOR NR. 1**

**„XML”**

**Curs „Platforma JAVA Enterprise II”**

**Coordonator:**

**Olga CERBU**

dr.,conf.univ

\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura)

**Autor:**

Studentul grupei AW21M

**Dumitru BAZAOСHI**

\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura)

**BĂLȚI, 2023**

**CUPRINS**

[Scop. Condiția problemei 2](#_Toc147419466)

[Ce este XML? 2](#_Toc147419467)

[Cum se citește fișierul XML în Java? 2](#_Toc147419468)

[Java DOM Parser 2](#_Toc147419469)

[Parsarea datelor. Ce este un XML parser? 3](#_Toc147419470)

[Pași pentru a citi fișierul XML în Java 3](#_Toc147419471)

[Cod XML 4](#_Toc147419472)

[Cod Java 5](#_Toc147419473)

[Rezultate Consolă 7](#_Toc147419474)

[Concluzii 8](#_Toc147419475)

[Referințe bibliografice 8](#_Toc147419476)

Scop. Condiția problemei

Scopul lucrării de laborator este de a învăța și a implementa citirea și procesarea datelor dintr-un fișier XML folosind limbajul JAVA. Astfel întelegerea structurii XML, studierea tehnologiilor disponibile pentru procesarea unui XML cum ar fi DOM, SAX etc., citirea unui fișier XML cu DOM parser, interpretarea rezultatelor și formularea concluziilor.

Ce este XML?

Extensible Markup Language (XML) ce permite definirea și stocarea datelor într-un mod partajat, ierarhic. XML acceptă schimbul de informații între sisteme informatice, cum ar fi site-uri web, baze de date și aplicații terțe. Regulile predefinite facilitează transmiterea datelor ca fișiere XML prin orice rețea, deoarece destinatarul poate folosi regulile pentru a citi datele cu acuratețe și eficient. În XML noi singuri putem crea etichete care vor corespunde tematicii documentului nostru.

Cum se citește fișierul XML în Java?

Citirea fișierului XML în Java este mult diferită de citirea altor fișiere precum .docx și .txt, deoarece fișierul XML conține date între etichete. Java oferă multe modalități de a analiza un fișier XML. Există doi analizatori în Java care analizează un fișier XML:

* Java DOM Parser
* Java SAX Parser

Java DOM Parser

API-ul DOM oferă clasele pentru a citi și scrie un fișier XML. Putem crea, șterge, modifica și rearanja nodul folosind API-ul DOM. Parserul DOM analizează întregul fișier XML și creează un obiect DOM în memorie. Modelează un fișier XML într-o structură arborescentă pentru parcurgere și manipulare ușoară. În DOM, totul într-un fișier XML este un nod. Nodul reprezintă o componentă a unui fișier XML. Analizatorul DOM este lent în proces și ocupă multă memorie atunci când încarcă un fișier XML în memorie.

Trebuie să fi urmat procesul pentru a citi un fișier XML în Java:

* **Instanțiați fișierul XML**: analizatorul DOM încarcă fișierul XML în memorie și consideră fiecare etichetă ca un element.
* **Obține nodul rădăcină**: Clasa Document oferă metoda **getDocumentElement()** pentru a obține nodul rădăcină și elementul fișierului XML.
* **Obține toate nodurile**: metoda **getElementByTagName()** preia toate numele etichetei specifice din fișierul XML. Unde tipul **ELEMENT\_NODE** se referă la un nod non-text care are sub-elemente. Dacă trebuie să accesăm toate nodurile de la început, inclusiv nodul rădăcină, putem apela recursiv metoda **getChildElement()**.
* **Obține nod după valoare text**: putem folosi metoda **getElementByTextValue()** pentru a căuta un nod după valoarea sa.
* **Obține valoarea nodului după atribut**: Dacă dorim să căutăm un nod după valoarea unui anumit atribut, putem folosi metoda **getElementByTagName()** împreună cu metoda **getAttribute()**.

Parsarea datelor. Ce este un XML parser?

Un parser XML (eXtensible Markup Language) este un software care poate procesa sau citi documente XML pentru a extrage date din ele. Analizatorii XML verifică, de asemenea, sintaxa sau regulile unui fișier XML și îl pot valida pe o anumită schemă XML. Deoarece XML este un limbaj de marcare strict, analizatorii nu vor procesa fișierul dacă există erori de validare sau de sintaxă. De exemplu, analizatorul XML va genera erori dacă una dintre următoarele condiții este adevărată:

* Lipsește eticheta de închidere sau de încheiere
* Valorile atributelor nu conțin ghilimele
* Condiția schemei nu este îndeplinită

Aplicațiile software folosesc parsere XML pentru a converti fișierele XML în tipuri de date native. În acest fel, se pot concentra pe logica aplicației fără a fi nevoie să intre în detaliile XML-ului în sine.

Pași pentru a citi fișierul XML în Java

1. Creați un **proiect Java** simplu.
2. Creați un fișier de clasă și furnizați un nume de fișier de clasă. Am creat fișierul de clasă cu numele **ReadXMLFileExample1**
3. Scrieți următorul cod
4. Descărcați fișierul **dom-2.3.0-jaxb-1.0.6.jar** : Faceți clic aici (<https://www.javatpoint.com/how-to-read-xml-file-in-java> )
5. Creați un folder lib în proiect
6. Copiați fișierul **dom-2.3.0-jaxb-1.0.6.jar** și inserați-l în folderul **lib**
7. Setați calea clasei: aceți clic dreapta pe proiect->Build Path->Configurare Build Path->Add External JAR->Selectați fișierul JAR->faceți clic pe butonul Open->Apply and Close
8. Creați un fișier XML . Am creat un fișier XML cu numele XMLFile.xml și scriem următoarele date în el
9. Rulați proiectul.

Cod XML

*Crearea fișierului XML și codul acestuia:*

<?xml version="1.0"?>

<class>

<student>

<id>101</id>

<firstname>Olga</firstname>

<lastname>Criclivaia</lastname>

<subject>Math</subject>

<marks>9</marks>

</student>

<student>

<id>102</id>

<firstname>Valentina</firstname>

<lastname>Advahova</lastname>

<subject>Chemistry</subject>

<marks>8</marks>

</student>

<student>

<id>103</id>

<firstname>Alina</firstname>

<lastname>Gutul</lastname>

<subject>English</subject>

<marks>10</marks>

</student>

<student>

<id>104</id>

<firstname>Nadejda</firstname>

<lastname>Ianciuc</lastname>

<subject>Physics</subject>

<marks>7</marks>

</student>

</class>

Cod Java

*Scrierea codului in fișierul java (daca sunt careva erori nu uitam să importam clasele necesare):*

import org.w3c.dom.Document;

import org.w3c.dom.Element;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import java.io.File;

public class ReadXMLFileExample1 {

public static void main(String argv[])

{

try

{

//crearea unui constructor al clasei de fișiere și analizarea unui fișier XML

File file = new File("XMLFile.xml");

// Crearea unei instanțe a generatorului de documente

DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

// Crearea unei instanțe a constructorului pentru a analiza fișierul XML specificat

DocumentBuilder db = dbf.newDocumentBuilder();

Document doc = db.parse(file);

doc.getDocumentElement().normalize();

// Afișarea numelui elementului rădăcină

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

// Obținerea unei liste de noduri pentru elementul "student"

NodeList nodeList = doc.getElementsByTagName("student");

// Iterarea prin lista de noduri

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

{

Node node = nodeList.item(itr);

System.out.println("\nNode Name :" + node.getNodeName());

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

{

// Conversia nodului la un element

Element eElement = (Element) node;

// Afișarea detaliilor despre student

System.out.println("Student id: "+ eElement.getElementsByTagName("id").item(0).getTextContent());

System.out.println("First Name: "+ eElement.getElementsByTagName("firstname").item(0).getTextContent());

System.out.println("Last Name: "+ eElement.getElementsByTagName("lastname").item(0).getTextContent());

System.out.println("Subject: "+ eElement.getElementsByTagName("subject").item(0).getTextContent());

System.out.println("Marks: "+ eElement.getElementsByTagName("marks").item(0).getTextContent());

}

}

}

catch (Exception e)

{

// Afișarea oricăror excepții care pot apărea

e.printStackTrace();

}

}

}

Rezultate Consolă

*Rezultatele obținute în Consolă sunt:*

Root element: class

Node Name :student

Student id: 101

First Name: Olga

Last Name: Criclivaia

Subject: Math

Marks: 9

Node Name :student

Student id: 102

First Name: Valentina

Last Name: Advahova

Subject: Chemistry

Marks: 8

Node Name :student

Student id: 103

First Name: Alina

Last Name: Gutul

Subject: English

Marks: 10

Node Name :student

Student id: 104

First Name: Nadejda

Last Name: Ianciuc

Subject: Physics

Marks: 7

Concluzii

În concluzie, în cadrul acestei lucrări de laborator am explorat modul în care se parsează fișiere XML în limbajul de programare Java. Am înțeles că XML (Extensible Markup Language) este un format eficient și ușor de utilizat pentru stocarea și transmiterea datelor într-un mod structurat. Am observat că generatorul XML poate genera erori dacă structura fișierului XML nu este corectă sau nu respectă regulile de marcări. Este esențial să asigurăm validitatea fișierelor XML pentru a evita erori în timpul parsării.

Utilizarea DOM parser ne permite să accesăm ușor la elementele și atributele din documentul XML și să le procesăm conform necesităților noastre.

Referințe bibliografice

1. *What is XML?* [online] [vizitat 03.10.2023]. Disponibil: <https://aws.amazon.com/ru/what-is/xml/>
2. How to read XML file in Java [online] [vizitat 03.10.2023]. Disponibil: <https://www.javatpoint.com/how-to-read-xml-file-in-java>