# *Національний технічний університет України*

# *«Київський політехнічний інститут»*

#### ***Факультет інформатики та обчислювальної техніки***

## Лабораторна робота №1

*з курсу "****Інженерія програмного забезпечення****"*

***Виконав:***

*Бедь А.М.*

***Група*** *ІО-12,*

***Номер варіанту*** *1202*

***Київ - 2013р.***

***Мета***

Отримання базових навичок з використання мови XML. Вивчення функціональних можливостей пакетів java.io та java.nio. Ознайомлення з механізмом підтримки стійкості даних. Отримання навичок з організації вводу/виводу та серіалізації.

***Завдання***

1. Вивчити структуру і функціональні можливості пакетів вводу/виводу java.io та java.nio. Володіти концепцією потоків вводу/виводу (байтових та символьних), механізмом використання буферів і каналів.

2. Ознайомитись з засобами підтримки стійкості даних. Вільно володіти механізмом серіалізації (інтерфейси Serializable та Externalizable)

3. Ознайомитись з механізмом виключень (Exceptions). Вміти перехоплювати виключення.

4. Створити клас CSVProcessor для роботи з CSV-файлами. У класі CSVProcessor реалізувати методи завантаження/вивантаження CSV-файлу в/з агрегований об’єкт класу ArrayList, в якому знаходяться рядки CSV-файлу. Реалізувати методи серіалізації/десеріалізації агрегованого об’єкту класу ArrayList в файлі serialized.dat.

5. Створити тестовий клас TestIO, в методі main() якого виконати наступні дії (за допомогою класу CSVProcessor і класів з пакетів java.io, java.nio):

А – перевірити чи існує файл serialized.dat, якщо існує – виконати десеріалізацію даних і перейти до пп. Д;

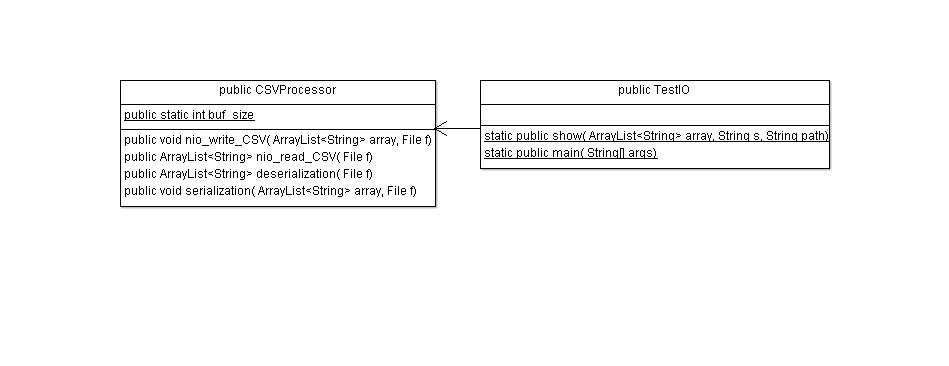
Б – ввести з консолі ім’я CSV-файлу;

В – перехоплюючи виключення вводу/виводу спробувати завантажити CSV-файл, в разі виникнення виключень повторювати пп. Б і В тричі після чого викинути виключення на рівень вище (перервавши виконання програми)

Г – виконати серіалізацію даних в файл serialized.dat;

Д – вивести на консоль дані з агрегованого об’єкту класу ArrayList.

***Діаграма класів***

******

***Лістинг коду***

package lab1;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.nio.ByteBuffer;

import java.nio.channels.FileChannel;

import java.nio.charset.Charset;

import java.util.ArrayList;

import java.util.StringTokenizer;

/\*\*

\* У класі CSVProcessor реалізувані методи завантаження/вивантаження CSV-файлу в/з агрегований об’єкт класу ArrayList,

\* в якому знаходяться рядки CSV-файлу. Реалізувати методи серіалізації/десеріалізації агрегованого об’єкту класу ArrayList

\* в файлі serialized.dat.

\* @author Aruy

\*/

public class **CSVProcessor** {

/\*\*

\* Розмір буфера

\*/

private static int buf\_size = 1024;

/\*\*

\* Метод серіалізує дані об'єкту array у файл

\* @param array об'єкт, що серіалізуватиметься

\* @param f об'єкт типу File, що містить шлях до файлу, в який серіалізується об'єкт

\*/

public void serialization (ArrayList<String> array, File f) {

ObjectOutputStream oos = null;

try {

oos = new ObjectOutputStream(new FileOutputStream(f));

oos.writeObject(array);

} catch (IOException e) {

e.printStackTrace();

} finally {

try {

if (oos != null)

oos.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* Метод десеріалізує дані з файлу і повертає у вигляді об'єкту типу ArrayList<String>

\* @param f об'єкт типу File, що містить шлях до файлу, в якому міститься серіалізований об'єкт

\* @return десеріалізований об'єкт типу ArrayList<String>

\*/

public ArrayList<String> deserialization (File f) {

ArrayList<String> array = null;

ObjectInputStream ois = null;

try {

ois = new ObjectInputStream(new FileInputStream(f));

array = (ArrayList<String>) ois.readObject();

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

} catch (ClassNotFoundException e) {

e.printStackTrace();

} finally {

try {

if (ois != null)

ois.close();

} catch (IOException e) {

e.printStackTrace();

}

}

return array;

}

/\*\*

\* Метод зчитує інформацію з CSV-файлу за допомогою потоків вводу пакету java.nio

\* @param f CSV-файл, звідки зчитуються дані

\* @return масив елементів типу String, що містяться в класі ArrayList

\* @throws IOException

\*/

public ArrayList<String> nio\_read\_CSV (File f) throws Exception {

ArrayList<String> array = new ArrayList<String>();

FileChannel in\_channel = null;

ByteBuffer buf;

in\_channel = new FileInputStream(f).getChannel();

buf = ByteBuffer.allocate(buf\_size);

while ((in\_channel.read(buf)) != -1 ) {

buf.flip();

for (int i = 0; i < buf.limit(); i++)

array.add((char)buf.get() + "");

buf.clear();

}

in\_channel.close();

return array;

}

/\*\*

\* Метод записує дані з заданого масиву типу ArrayList<String> в файл за допомогою потоків виводу пакету java.nio

\* @param array масив даних, що записуватимуться в файл

\* @param f об'єкт типу File, що містить шлях до файлу, куди записуватимуться дані

\*/

public void nio\_write\_CSV (ArrayList<String> array, File f) {

FileChannel out\_channel = null;

ByteBuffer buf;

try {

out\_channel = new FileOutputStream(f).getChannel();

buf = ByteBuffer.allocate(buf\_size);

for (int i = 0; i < array.size(); i++) {

buf = ByteBuffer.wrap(array.get(i).getBytes(Charset.forName("utf-8")));

i++;

buf.flip();

out\_channel.write(buf);

buf.clear();

}

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

} finally {

try {

if (out\_channel != null)

out\_channel.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

**public** **class** **TestIO** {

/\*\*

\* Виводить на екран дані об'єкту типу ArrayList<String>

\* **@param** array об'єкт типу ArrayList<String>

\*/

**public** **static** **void** show(ArrayList<String> array, String s, String path) {

System.*out*.println (" " + s + " \"" + path.toString() + "\" :");

**for** (**int** i = 0; i < array.size(); i++)

System.*out*.print(array.get(i));

System.*out*.println ();

}

**public** **static** **void** main(String[] args) **throws** Exception {

String ser\_path = "D:/serialized.dat";

File file = **null**;

File file\_ser = **new** File(ser\_path);

CSVProcessor csv = **new** CSVProcessor();

Scanner sc = **new** Scanner(System.*in*);

**byte** attempt = 3;

ArrayList<String> array = **null**;

**if** (file\_ser.exists()) {

ArrayList<String> ser = csv.deserialization(file\_ser);

*show*(ser, "Дані десеріалізовані з файлу", ser\_path);

} **else** {

**do** {

**try** {

System.*out*.print (" Введіть шлях до CSV-файлу, звідки будуть зчитуватись дані.\n");

file = **new** File(sc.nextLine()); // D:/in.csv

array = csv.nio\_read\_CSV(file);

} **catch** (Exception e) {

attempt--;

**if** (attempt == 0) {

System.*out*.print ("У вас закінчились спроби!\n");

**throw** **new** Exception();

}

System.*out*.println (e);

System.*out*.println ("Помилка!!! Залишилось спроб: " + attempt + "\n");

}

} **while** (!file.exists()) ;

csv.serialization(array, file\_ser);

*show*(array, "Дані зчитані з файлу", file.getAbsolutePath());

}

}

}

***JavaDoc***

## C

[****CSVProcessor****](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html) - Class in [lab1](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\package-summary.html)

У класі CSVProcessor реалізувані методи завантаження/вивантаження CSV-файлу в/з агрегований об’єкт класу ArrayList, в якому знаходяться рядки CSV-файлу.

[**CSVProcessor()**](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html#CSVProcessor()) - Constructor for class lab1.[CSVProcessor](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html)

## D

[**deserialization(File)**](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html#deserialization(java.io.File)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html)

Метод десеріалізує дані з файлу і повертає у вигляді об'єкту типу ArrayList

## L

[lab1](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\package-summary.html) - package lab1

## M

[**main(String[])**](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\TestIO.html#main(java.lang.String[])) - Static method in class lab1.[TestIO](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\TestIO.html)

## N

[**nio\_read\_CSV(File)**](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html#nio_read_CSV(java.io.File)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html)

Метод зчитує інформацію з CSV-файлу за допомогою потоків вводу пакету java.nio

[**nio\_write\_CSV(ArrayList<String>, File)**](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html#nio_write_CSV(java.util.ArrayList, java.io.File)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html)

Метод записує дані з заданого масиву типу ArrayList в файл за допомогою потоків виводу пакету java.nio

## S

[**serialization(ArrayList<String>, File)**](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html#serialization(java.util.ArrayList, java.io.File)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\CSVProcessor.html)

Метод серіалізує дані об'єкту array у файл

[**show(ArrayList<String>, String, String)**](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\TestIO.html#show(java.util.ArrayList, java.lang.String, java.lang.String)) - Static method in class lab1.[TestIO](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\TestIO.html)

Виводить на екран дані об'єкту типу ArrayList

## T

[****TestIO****](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\TestIO.html) - Class in [lab1](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\package-summary.html)

[**TestIO()**](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\TestIO.html#TestIO()) - Constructor for class lab1.[TestIO](file:///D:\--==--\Java\Worckbench\EngineeringSoftware\doc\lab1\TestIO.html)

***Висновок***

*Виконавши цю лабораторну роботу, я отримав базові навички з використання мови XML. Вивчив функціональні можливості пакетів java.io та java.nio. Ознайомився з механізмом підтримки стійкості даних. Отримав навички з організації вводу/виводу та серіалізації.*