Лабораторна робота №8

**ОСНОВИ ВВЕДЕННЯ / ВИВЕДЕННЯ JAVA SE**

**Мета:** Оволодіння навичками управління введенням/виведенням даних з використанням класів платформи Java SE.

**ВИМОГИ**

**Розробник:**

- Веремчук Дарина Анатоліївна;

- КІТ-119д;

- Варіант №5.

**Загальне завдання:**

1) Забезпечити можливість збереження і відновлення масива об'єктів рішення завдання лабораторної роботи №7.

2) Забороняється використання стандартного протокола серіалізації.

3) Продемонструвати використання моделі Long Term Persistence.

4) Забезпечити діалог з користувачем у вигляді простого текстового меню.

5) При збереженні та відновленні даних забезпечити діалоговий режим вибору директорії з відображенням вмісту і можливістю переміщення по підкаталогах.

**ОПИС ПРОГРАМИ**

**Опис змінних:**

BuyersGuide[] TestStore; // набір торгівельних точок

Scanner in; // для отримання даних з клавіатури

String choice; // зберігання вибору користувача

**Ієрархія та структура класів:**

**class** Veremchuk08 – точка входу в програму;

**class** UI – реалізація діалогового режиму.

**ТЕКСТ ПРОГРАМИ**

Текст класу **Veremchuk08**:

package ua.oop.khpi.veremchuk08;

import ua.oop.khpi.veremchuk07.BuyersGuide;

import java.io.IOException;

import java.util.Scanner;

public class Veremchuk08 {

private Veremchuk08(){}

/\*\*

\* An entry point, the main method

\*

\* @param args - arguments

\*/

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

BuyersGuide [] stores = new BuyersGuide[0];

Scanner in = new Scanner(System.in);

String choice;

do {

UI.mainMenu();

choice = UI.getChoice();

switch (choice) {

case "1":

stores = UI.addStore(stores);

break;

case "2":

System.out.print("Введите индекс"

+ " удаляемой торговой точки: ");

int index = in.nextInt();

stores = UI.dropStore(stores, index);

break;

case "3":

UI.printInfo(stores);

System.out.println();

break;

case "4":

UI.saveToFile(stores);

break;

case "5":

try {

BuyersGuide[] newStores = UI.loadFromFile();

UI.printInfo(newStores);

} catch (IOException e) {

System.out.println(e.toString());

}

break;

case "0":

System.out.println("Выход...");

break;

default:

System.out.println("Введите номер"

+ " одного из пунктов!\n");

}

} while (!choice.equals("0"));

}

}

Текст класу **UI**:

package ua.oop.khpi.veremchuk08;

import ua.oop.khpi.veremchuk07.BuyersGuide;

import java.beans.XMLDecoder;

import java.beans.XMLEncoder;

import java.io.BufferedReader;

import java.io.File;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.BufferedInputStream;

import java.io.BufferedOutputStream;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class UI {

private UI() {

}

/\*\*

\* Reading data from keyboard

\*/

private static BufferedReader buffer = new BufferedReader(

new InputStreamReader(System.in));

/\*\*

\* Main menu

\*/

public static void mainMenu() {

System.out.println();

System.out.println("1. Добавление торговой точки.");

System.out.println("2. Удаление торговой точки.");

System.out.println("3. Вывод информации о всех торговых точках.");

System.out.println("4. Сохранить в файл.");

System.out.println("5. Загрузить из файла.");

System.out.println("0. Выход.");

System.out.print("Введите ваш ответ сюда: ");

}

/\*\*

\* Get answer in dialog menu

\*

\* @return - answer of user

\* @throws IOException when reading is wrong

\*/

public static String getChoice() throws IOException {

return buffer.readLine();

}

/\*\*

\* Добавление мероприятия.

\*

\* @param stores - old list of stores

\* @return new list of stores

\* @throws IOException when reading is wrong

\*/

public static BuyersGuide[] addStore(final BuyersGuide[] stores)

throws IOException{

BuyersGuide[] newStores = new BuyersGuide[stores.length + 1];

System.arraycopy(stores, 0, newStores, 0, stores.length);

newStores[stores.length] = BuyersGuide.generate();

return newStores;

}

/\*\*

\* Deleting a store.

\*

\* @param stores - old list of stores

\* @param pos - position of the store we'll delete

\* @return new list of stores

\*/

public static BuyersGuide[] dropStore(final BuyersGuide[] stores,

final int pos) {

if (pos >= stores.length) {

System.out.println("Error. Out of bounds.");

return stores;

} else {

BuyersGuide[] newStores = new BuyersGuide[stores.length - 1];

int i = 0;

for (int j = 0; j < stores.length; j++) {

if (j != pos) {

newStores[i] = stores[j];

i++;

}

}

return newStores;

}

}

/\*\*

\* Showing information about all of stores.

\* @param stores - list of stores

\*/

public static void printInfo(final BuyersGuide[] stores) {

for (int i = 0; i < stores.length; i++) {

System.out.format("%nТОРГОВАЯ ТОЧКА №%d:%n", i + 1);

System.out.println(stores[i].toString());

}

}

/\*\*

\* Saving an object to file

\* @param stores - list of stores

\* @throws IOException when reading is wrong

\*/

public static void saveToFile(final BuyersGuide[] stores)

throws IOException {

final String semi = ";";

Pattern pattern = Pattern.compile(semi);

Matcher matcher;

String path;

StringBuilder direct = new StringBuilder();

System.out.print("Введите имя директории: ");

while (true) {

System.out.print(direct.toString());

path = buffer.readLine();

direct.append(path).append("\\");

matcher = pattern.matcher(direct.toString());

if (matcher.find()) {

break;

}

File directory = new File(direct.toString());

File[] list = directory.listFiles();

if (list == null) {

System.out.println("Неверное"

+ " имя директории!");

if (direct.length() != 0) {

direct.delete(

direct.length() - path.length() - 1,

direct.length());

}

continue;

}

System.out.println("---------------");

for (File it : list) {

if (it.isDirectory()) {

System.out.print(it.getName());

System.out.println(" (...)");

continue;

}

System.out.println(it.getName());

}

System.out.println("---------------");

}

String currentDir = direct.toString();

currentDir = currentDir.replaceAll(semi, "");

FileOutputStream fos = new FileOutputStream(

currentDir + "\\Encoded.xml");

XMLEncoder xmlEncoder = new XMLEncoder(new BufferedOutputStream(fos));

xmlEncoder.writeObject(stores);

xmlEncoder.close();

}

/\*\*

\* Load an object from file

\* @return array of objects

\* @throws IOException when reading is wrong

\*/

public static BuyersGuide[] loadFromFile() throws IOException {

System.out.print("Введите имя директории: ");

String dirToExtract = buffer.readLine();

FileInputStream fis = new FileInputStream(dirToExtract);

XMLDecoder xmlDecoder = new XMLDecoder(new BufferedInputStream(fis));

BuyersGuide[] getStores = (BuyersGuide[]) xmlDecoder.readObject();

xmlDecoder.close();

return getStores;

}

}

**РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ**

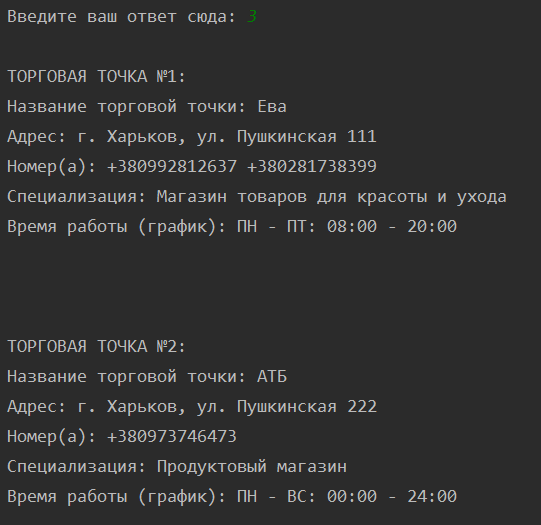
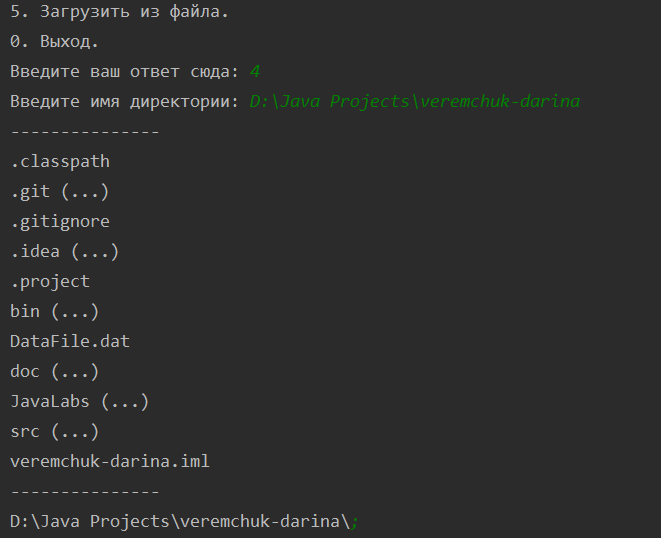
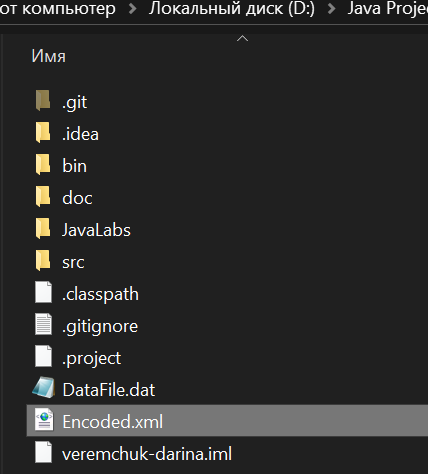


Рисунок 8.1 – Виведення інформації про торгівельні точки

а) б)

Рисунок 8.2 - Виконання зберігання об’єкта у файл

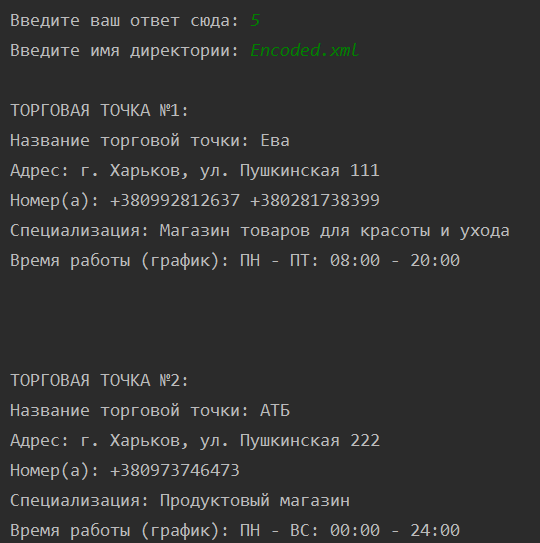


Рисунок 8.3 – Виконання зчитування з файлу

**ВАРІАНТИ ВИКОРИСТАННЯ**

Програма може бути використана для створення та збереження інформації про торгівельні точки. Реалізовано додавання та видалення торгівельних точок.

**ВИСНОВКИ**

Під час лабораторної роботи, набула практичних навичок щодо використання моделі Long Term Persistence, яка являє собою зберігання об’єктів в XML-файл та їх зчитування.