Звіт

Автор: Лунгол О., КІТ101.8б

Дата: 19.12.2019

Лабораторна робота №8

ОСНОВИ ВВЕДЕННЯ/ВИВЕДЕННЯ JAVA SE

**Мета.**

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

**Вимоги:**

1. Забезпечити можливість збереження і відновлення масива об’єктів рішення завдання [лабораторної роботи №7](https://oop-khpi.github.io/#task_07).
2. Забороняється використання [стандартного протокола серіалізації](https://docs.oracle.com/javase/8/docs/platform/serialization/spec/serialTOC.html).
3. Продемонструвати використання моделі [Long Term Persistence](https://docs.oracle.com/javase/tutorial/javabeans/advanced/longpersistence.html).
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**

**private** **long** NumberPassport;

**private** String specialty;

**private** String workingConditions;

**private** **int** marks;

**private** **int** payment;

**private** String data;

**private** String position;

**private** String education;

**private** String department;

**Ієрархія та структура класів**

**class** Container- клас в якому описуються та знаходяться контейнеры

**class** Lab08- головний клас

**class** HDR – клас в якому данні про відділ кадрів

**class** Functions- клас в якому описуються функції

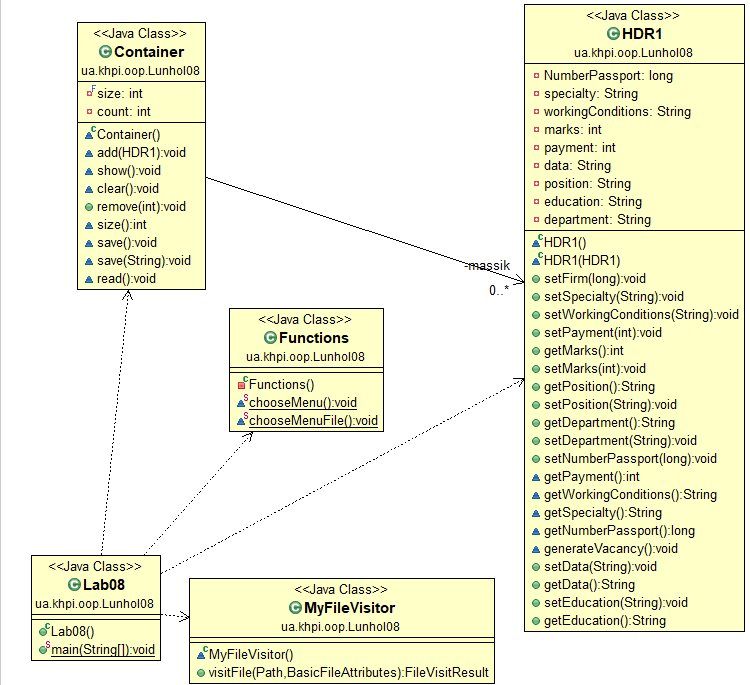


Рисунок 1 – Ієрархія класів

ТЕКСТ ПРОГРАМИ

Lab08.java

**package** ua.khpi.oop.Lunhol08;

**import** org.w3c.dom.Document;

**import** org.w3c.dom.Node;

**import** org.w3c.dom.NodeList;

**import** org.xml.sax.SAXException;

**import** javax.xml.parsers.DocumentBuilder;

**import** javax.xml.parsers.DocumentBuilderFactory;

**import** javax.xml.parsers.ParserConfigurationException;

**import** java.io.IOException;//выброс ошибок если что-то не получилось

**import** java.nio.file.\*;//для классов файл

**import** java.util.HashSet;

**import** java.util.Scanner;

**public** **class** Lab08 {

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

Container contains = **new** Container();

HDR1 rec1 = **new** HDR1();

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

String str = "";

String str3 = "";

**boolean** loop = **true**;

**while** (loop) {

Functions.*chooseMenu*();

**int** choose = scan.nextInt();

**switch** (choose) {

**case** 1:

rec1.generateVacancy();

contains.add(rec1);

System.***out***.println("Успешно!");

**break**;

**case** 2:

**if** (contains.size() == 0) {

System.***out***.println("Ошибка! Массив пустой!");

**break**;

} **else** {

contains.show();

**break**;

}

**case** 3:

**if** (contains.size() == 0) {

System.***out***.println("Ошибка! Массив пустой!");

**break**;

} **else** {

System.***out***.print("Введите позицию удаления от 1 до " + contains.size());

**int** index = scan.nextInt();

**if** (index > contains.size() || index < 1) {

System.***out***.println("Повторите ввод: ");

} **else** {

contains.remove(index);

System.***out***.println("Успешно!");

}

}

**break**;

**case** 4:

**if** (contains.size() == 0) {

System.***out***.println("Ошибка! Массив пустой!");

**break**;

} **else** {

contains.clear();

System.***out***.println("Успешно!");

**break**;

}

**case** 5:

DocumentBuilderFactory factory = DocumentBuilderFactory.*newDefaultInstance*();

**try** {

DocumentBuilder builder = factory.newDocumentBuilder();

Document document = builder.parse("test.xml");

Node root = document.getDocumentElement();

System.***out***.println("Список кандидатов: ");

System.***out***.println();

NodeList vacancies = root.getChildNodes();

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

Node vacancy = vacancies.item(i);

**if** (vacancy.getNodeType() != Node.***TEXT\_NODE***) {

NodeList bookProps = vacancy.getChildNodes();

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

Node bookProp = bookProps.item(j);

**if** (bookProp.getNodeType() != Node.***TEXT\_NODE***) {

System.***out***.println(bookProp.getNodeName() + ":" + bookProp.getChildNodes().item(0).getTextContent());

}

}

System.***out***.println("===========>>>>");

}

}

} **catch** (ParserConfigurationException | SAXException | IOException ex) {

ex.printStackTrace(System.***out***);

}

**break**;

**case** 6:

**boolean** loop2 = **true**;

**while** (loop2) {

Functions.*chooseMenuFile*();

**int** choose2 = scan.nextInt();

**switch** (choose2) {

**case** 1:

contains.save();

System.***out***.println("Успешно!");

**break**;

**case** 2:

Files.*walkFileTree*(Paths.*get*("."), **new** HashSet<FileVisitOption>(), 1, **new** MyFileVisitor());

**break**;

**case** 3:

Path path = Paths.*get*("save.txt").toAbsolutePath();

System.***out***.println(path.getParent());

**break**;

**case** 4:

**boolean** loop5 = **true**;

**while** (loop5) {

**int** cho = 0;

System.***out***.println("Введите путь к директории: ");

str = scan.next();

Files.*walkFileTree*(Paths.*get*(str), **new** HashSet<FileVisitOption>(), 1, **new** MyFileVisitor());

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

System.***out***.println("Вы желаете сохранить файл в эту директорию?");

System.***out***.println("1. - Да. 0. - Нет.");

**while** (**true**) {

cho = scan.nextInt();

**if** (cho == 1) {

str3 = str.concat("\\save.txt");

contains.save(str3);

loop5 = **false**;

**break**;

} **else** **if** (cho == 0) {

**break**;

} **else** {

System.***out***.println("Ошибка повторите ввод: ");

}

}

}

**break**;

**case** 5:

System.***out***.println("Вы вводили собственный путь директории?");

System.***out***.println("1 - Да. 0 - Нет.");

**boolean** loop4 = **true**;

**while** (loop4) {

**int** chosik = scan.nextInt();

**if** (chosik == 1) {

System.***out***.println("Введите новое имя файла: ");

String str1 = scan.next();

String str4 = str;

str4 = str4.concat("\\");

str4 = str4.concat(str1);

Files.*move*(Paths.*get*(str3), Paths.*get*(str4), StandardCopyOption.***REPLACE\_EXISTING***);

System.***out***.println("Успешно!");

loop4 = **false**;

} **else** **if** (chosik == 0) {

System.***out***.println("Выполните пункт 4");

loop4 = **false**;

} **else** {

System.***out***.println("Ошибка! Повторите ввод: ");

}

}

**break**;

**case** 6:

contains.read();

System.***out***.println("Успешно!");

**break**;

// case 7:

// XMLEncoder encoder = new XMLEncoder(new BufferedOutputStream(new FileOutputStream("trunk.xml")));

// encoder.writeObject(contains);

// encoder.close();

// break;

// case 8:

// XMLDecoder decoder = new XMLDecoder(new BufferedInputStream(new FileInputStream("trunk.xml")));

// Kontainer contic = (Kontainer) decoder.readObject();

// contic.show();

// decoder.close();

**case** 0:

System.***out***.println("Выход!");

loop2 = **false**;

**break**;

**default**:

System.***out***.println("Ошибка. Неверное число!");

**break**;

}

}

**break**;

**case** 0:

System.***out***.println("Спасибо за работу!");

loop = **false**;

**break**;

**default**:

System.***out***.println("Ошибка. Неверное число!");

**break**;

}

}

}

}

HDR.java

Container.java

**package** ua.khpi.oop.Lunhol07;

**import** java.util.Arrays;

**class** Container {

/\*\*

\*/

**private** **final** **int** size = 10;

/\*\*

\* Array contains all data.

\*/

**private** **int** count = 0;

**private** HDR[] massik = **new** HDR[size];

**void** add(**final** HDR temp){

**if** (count == massik.length) {

massik = Arrays.*copyOf*(massik, massik.length \* 2);

massik[count++] = temp;

} **else** {

massik[count++] = temp;

}

}

**void** show() {

**if** (count == 0) {

System.***out***.println("Ошибка! Массив пустой!");

} **else** {

System.***out***.println();

**for** (**int** i = 0; i < count; i++) {

System.***out***.println("Кандидат №" + (i+1));

System.***out***.println("Паспорт: " + massik[i].getNumberPassport());

System.***out***.println("Образование: " + massik[i].getEducation());

System.***out***.println("Оклад: " + massik[i].getPayment());

System.***out***.println("Дата трудоустройства: " + massik[i].getData());

System.***out***.println("Должность: " + massik[i].getPosition());

System.***out***.println("Отдел: " + massik[i].getDepartment());

System.***out***.println("Набор качеств: " + massik[i].getWorkingConditions());

System.***out***.println("Оценка кандидата: " + massik[i].getEducation());

}

}

}

**void** clear() {

**for** (**int** i = 0; i < count; i++) {

massik[i] = **null**;

}

count = 0;

}

**public** **void** remove(**int** index) {

**for** (**int** i = index; i < count; i++) {

massik[index++] = massik[i + 1];

}

count--;

}

}

\* First size for array.

РОБОТА ПРОГРАМИ

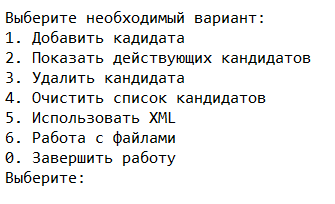


Рисунок 2- меню робочої програми

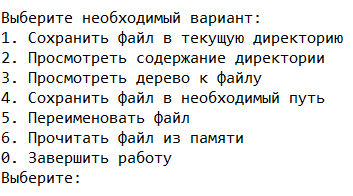


Рисунок 3- меню роботи з файлами

ВИСНОВОК

Оволоділи навичками управління введенням/виведенням даних з використанням класів платформи Java SE.