МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М.Є. Жуковського “ХАІ”

Кафедра 603

Лабораторна робота № 5

**Робота з системою введення-виведення**

**у програмах Java**

з дисципліни “Програмування Інтернет”

ХАІ.603.631п.15В.050103.126337.ПЗ

Виконав студент гр.631п  Суща О.О.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (№ групи) (П.І.Б)

(підпис, дата)

Перевірив:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наукова ступінь, вчене звання)

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

(підпис, дата) (П.І.Б)

2015

Цель работы: получение практических навыков работы с входными и выходными потоками данных в программах на языке Java.

Изучаемый материал: система ввода-вывода Java: основные концепции. Пакет **java.io**: основные классы, интерфейсы и методы. Особенности применения классов байтовых и символьных потоков ввода-вывода в приложениях. Классы исключений пакета **java.io**.

**Постановка задачи**

Разработать и реализовать на языке Java приложение для демонстрации работы с потоками ввода-вывода. В разрабатываемом приложении показать применение классов и методов пакета **java.io** для работы с различными источниками и приемниками данных (клавиатура, файловая система, память), а также возможности сохранения и восстановления состояния объектов с помощью потоков ввода-вывода.

Примечание. В качестве основы можно использовать программные приложения, разработанные в предыдущих лабораторных работах.

**Ход работы**

1. В качестве основы было взято приложение из лабораторной работы №1. Исходная программа дополнена необходимым кодом, который поддерживает работу с системой ввода-вывода.

Main.java

…

import java.io.IOException;

. . .

**private** **static** String *pathRead* = "C:/Users/Alyona/workspace/FirstJava/read.txt";

**private** **static** String *pathWrite*="C:/Users/Alyona/workspace/FirstJava/write.txt";

. . .

str = InputOutputStream.read(pathRead);

. . .

InputOutputStream.write(pathWrite,rez);

InputOutputStream.java

**package** First;

**import** java.io.\*;

**public** **class** InputOutputStream {

**private** InputStream inputstream;

**private** OutputStream outputstream;

**private** String path;

**public** InputOutputStream(String path){

**this**.path = path;

}

/\*\*

\* StringBuilder - специальный объектдля построения строки

\* BufferedReader - объект для чтения файла в буфер

\* В цикле мы построчно считываем файл

\* Затем закрываем поток и передаём считанный текст

\*/

**public** **static** String read(String fileName) **throws** FileNotFoundException {

StringBuilder sb = **new** StringBuilder();

File file = **new** File(fileName);

**if** (!file.exists()){

**throw** **new** FileNotFoundException(file.getName());

}

**try**{

BufferedReader in = **new** BufferedReader(**new** FileReader(file.getAbsoluteFile()));

**try**{

String s;

**while** ((s = in.readLine()) != **null**){

sb.append(s);

}

}**finally**{

in.close();

}

}**catch** (IOException e){

**throw** **new** RuntimeException(e);

}

**return** sb.toString();

}

/\*\*

\* Сначала определяем файл.

\* Затем если такой не существует, - создаём.

\* PrintWriter обеспечивает запись в файл

\* В конце обязательно закрываем поток.

\*/

**public** **static** **void** write(String fileName, String text) {

File file = **new** File(fileName);

**try**{

**if** (!file.exists()){

file.createNewFile();

}

PrintWriter out = **new** PrintWriter(file.getAbsoluteFile());

**try**{

out.print(text);

}

**finally**{

out.close();

}

}

**catch** (IOException e){

**throw** **new** RuntimeException(e);

}

}

}

1. Приложение откомпилировано и протестировано.

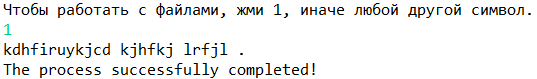


Рисунок 1 – Тестирование, работа с файлами

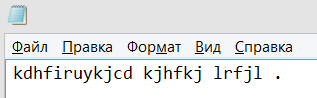
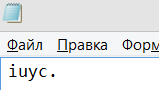
 

Рисунок 2 – Исходный файл Рисунок 3 – Результат

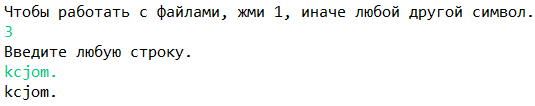


Рисунок 4 – Работа с потоками ввода-вывода

1. Документация Javadoc.

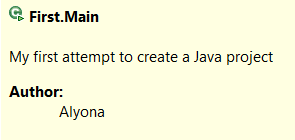


Рисунок 5 – Main.java

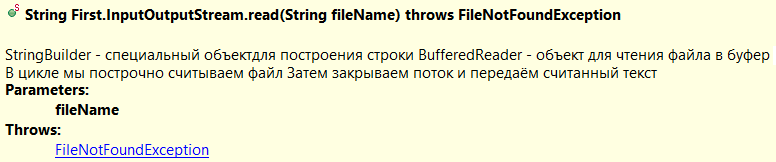


Рисунок 6 – InputOutputStream.java

Выводы: во время выполнения лабораторной работы были изучены теоретические сведения на тему систем ввода-вывода в языке Java. В качестве основы была взята программа из лабораторной работы №1. Исходный код дополнен всеми необходимыми компонентами и успешно протестирован. Также была получена документация Javadoc.

**Приложение А. Исходный код программы**

Main.java

**package** First;

**import** java.io.IOException;

**import** java.util.Scanner;

/\*\*

\* My first attempt to create a Java project

\* **@author** Alyona

\*

\*/

**public** **class** Main {

/\*\*

\* **@param** args

\*/

**private** **static** String *pathRead* = "C:/Users/Alyona/workspace/FirstJava/read.txt";

**private** **static** String *pathWrite*="C:/Users/Alyona/workspace/FirstJava/write.txt";

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

// **TODO** Auto-generated method stub

**try** {

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

String str;

System.***out***.println("Чтобы работать с файлами, жми 1, иначе любой другой символ.");

**int** vvod = Integer.*parseInt*(in.nextLine());

**if** (vvod==1){

str = InputOutputStream.*read*(*pathRead*);

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

}

**else** {

System.***out***.println("Введите любую строку.");

str = in.nextLine();

}

**char** last = str.charAt(str.length()-1);

**if**(last!='.') {

**throw** **new** CustomException("В конце строки должна быть точка!");

}

**int** pos=0, lastPos=0, firstPos=0;

String rez="";

**for** (;pos<str.length();pos++) {

lastPos=str.lastIndexOf(str.charAt(pos));

firstPos=str.indexOf(str.charAt(pos));

**if** (lastPos==firstPos)

rez+=str.charAt(pos);

}

**if** (vvod==1){

InputOutputStream.*write*(*pathWrite*,rez);

System.***out***.println("The process successfully completed!");

}

**else**{

System.***out***.print(rez);

}

in.close();

}

**catch**(Exception ex){

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

}

}

}

CustomException.java

**package** First;

**public** **class** CustomException **extends** Exception {

**public** CustomException(String msg){

**super**("Invalid input: "+msg);

}

}

InputOutputStream.java

**package** First;

**import** java.io.\*;

**public** **class** InputOutputStream {

**private** InputStream inputstream;

**private** OutputStream outputstream;

**private** String path;

**public** InputOutputStream(String path){

**this**.path = path;

}

/\*\*

\* StringBuilder - специальный объектдля построения строки

\* BufferedReader - объект для чтения файла в буфер

\* В цикле мы построчно считываем файл

\* Затем закрываем поток и передаём считанный текст

\*/

**public** **static** String read(String fileName) **throws** FileNotFoundException {

StringBuilder sb = **new** StringBuilder();

File file = **new** File(fileName);

**if** (!file.exists()){

**throw** **new** FileNotFoundException(file.getName());

}

**try**{

BufferedReader in = **new** BufferedReader(**new** FileReader(file.getAbsoluteFile()));

**try**{

String s;

**while** ((s = in.readLine()) != **null**){

sb.append(s);

}

}**finally**{

in.close();

}

}**catch** (IOException e){

**throw** **new** RuntimeException(e);

}

**return** sb.toString();

}

/\*\*

\* Сначала определяем файл.

\* Затем если такой не существует, - создаём.

\* PrintWriter обеспечивает запись в файл

\* В конце обязательно закрываем поток.

\*/

**public** **static** **void** write(String fileName, String text) {

File file = **new** File(fileName);

**try**{

**if** (!file.exists()){

file.createNewFile();

}

PrintWriter out = **new** PrintWriter(file.getAbsoluteFile());

**try**{

out.print(text);

}

**finally**{

out.close();

}

}

**catch** (IOException e){

**throw** **new** RuntimeException(e);

}

}

}

Классы Transport, Bus, Tram и Trolley остались без изменения.