Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ



ЗВІТ

до лабораторної роботи №2

З дисципліни: «Кросплатформні засоби програмування»

На тему: «КЛАСИ ТА ПАКЕТИ»

Варіант 20

**Виконав:**

ст. групи КІ-306

Петрук О. С.

**Прийняв:**

доцент кафедри ЕОМ

Олексів М.В.

Львів – 2024

**Мета:** ознайомитися з процесом розробки класів та пакетів мовою Java.

**Завдання:**

1. Написати та налагодити програму на мові Java, що реалізує у вигляді класу предметну область згідно варіанту. Програма має задовольняти наступним вимогам:

• програма має розміщуватися в пакеті Група.Прізвище.Lab2;

• клас має містити мінімум 3 поля, що є об’єктами класів, які описують складові частини предметної області;

• клас має містити кілька конструкторів та мінімум 10 методів;

• для тестування і демонстрації роботи розробленого класу розробити клас-драйвер;

• методи класу мають вести протокол своєї діяльності, що записується у файл;

• розробити механізм коректного завершення роботи з файлом (не надіятися на метод finalize());

• програма має володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

2. Автоматично згенерувати документацію до розробленої програми.

3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.

4. Скласти звіт про виконану роботу з приведенням тексту програми, результату її

виконання та фрагменту згенерованої документації та завантажити його у ВНС.

5. Дати відповідь на контрольні запитання.

**Тема згідно варіанту №20 – « Взуття »**

**GitHub Repository:**

**Хід роботи**

Код програми:

***Shoes.java***

package KI\_306.Petruk.Lab2;

import java.io.\*;

import java.io.IOException;

public class Shoes {

private Color baseColor;

private Laces laces;

private Material material;

private Sole sole;

private PrintWriter fout;

private Size size; /\*\* \* Basic Constructor \* @throws FileNotFoundException \*/

public Shoes() throws FileNotFoundException {

baseColor = new Color();

laces = new Laces();

material = new Material();

sole = new Sole();

fout = new PrintWriter(new File("Log.txt"));

size = new Size(42);

}

/\*\* \* Constructor with parameters

\* \* @param color defines shoe color

\* \* @param lacesColor defines laces color

\* \* @param unlaced defines if lases are lased or not

\* \* @param baseMaterial defines which material is used

\* \* @param soleColor defines sole color

\* \* @param baseSize defines shoe size

\* \* @throws FileNotFoundException \*/

public Shoes(String color,String lacesColor, boolean unlaced,String baseMaterial, String soleColor, int baseSize) throws FileNotFoundException {

baseColor = new Color(color);

laces = new Laces(lacesColor,unlaced);

material = new Material(baseMaterial);

sole = new Sole(soleColor);

fout = new PrintWriter(new File("Log.txt"));

size = new Size(baseSize);

}

/\*\*

\* Метод для чищення взуття

\*/

/\*\* \* Method for unlacing shoes \*/

public void unlaceShoes(){

if (!laces.getUnlaced()) {

laces.setUnlaced(true);

fout.print("Shoes unlaced\n");

}

else {

fout.print("Shoes are already unlaced\n");

}

}

/\*\* \* Method for lacing shoes \*/

public void laceShoes(){

if (laces.getUnlaced()) {

laces.setUnlaced(false);

fout.print("Shoes laced\n");

}

else {

fout.print("Shoes are already laced\n");

}

}

/\*\* \* Method for cleaning shoes \*/

public void cleanShoes(){

fout.print("Shoes were cleaned\n");

}

/\*\* \* Method for increasing shoe size \*

\* @param plusSize defines how much shoe has to be expanded \*/

public void expandShoes(int plusSize){

size.setSize(size.getSize()+plusSize);

fout.print("Shoes size expanded to " +size.getSize()+"\n");

}

/\*\*

\* Method for shrinking shoe size

\* \*\* @param minusSize defines how much shoe has to shrink \*/

public void shrinkShoes(int minusSize){

size.setSize(size.getSize()-minusSize);

fout.print("Shoes size shrunk to " + size.getSize()+"\n");

}

/\*\* \* Method for repairing shoes \*/

public void repairShoes(){

fout.print("Shoes were repaired\n");

}

/\*\* \* Method for showing info about shoes \*/

public void shoeInfo(){

fout.print("------------------------------------------\n");

fout.print("Shoe size: " + size.getSize() + "\n");

fout.print("Shoe material: " + material.getMaterial() + "\n");

fout.print("Shoe color: " + baseColor.getColor() + "\n");

fout.print("Laces color: " + laces.getColor() + "\n");

fout.print("Sole color: " + sole.getColor() + "\n");

if (!laces.getUnlaced()){

fout.print("Shoes are laced\n");

}

else{

fout.print("Shoes are unlaced\n");

}

fout.print("------------------------------------------\n");

}

public void dispose(){

fout.flush();

fout.close();

}

/\*\* \* Method for changing color \* @param newColor defines new color of a shoe \*/

public void changeColor(String newColor) {

baseColor.setColor(newColor);

}

/\*\* \* Method for changing color of laces \* @param newLacesColor defines new color of laces \*/

public void changeLaces(String newLacesColor){

laces.setColor(newLacesColor);

fout.print("Laces color was changed into " + laces.getColor() + "\n");

}

/\*\* \* Method for putting shoes on \*/ public void putOnShoes(){

fout.print("Shoes were worn\n");

}

/\*\* \* Method for removing shoes \*/

public void takeOffShoes(){

fout.print("Shoes were taken off\n");

}

/\*\* \* Method for changing sole color \* @param newSoleColor defines new sole color \*/

public void changeSoleColor(String newSoleColor){

sole.setColor(newSoleColor);

fout.print("Sole color was changed into " + sole.getColor() + "\n");

}

}

/\*\* \* <code>class Color</code> represents color of shoes \* @author Oleksandr Petruk \* @version 1.0 \*/

class Color{

private String color;

/\*\* \* Constructor \*/

public Color(){

color="black";

}

/\*\* \* Constructor with parameters \* @param col is used to set color of a shoe \*/

public Color(String col){

color=col;

}

/\*\* \* Getter \* @return color of a shoe \*/

public String getColor(){

return color;

}

/\*\* \* Setter \* @param col sets color of a shoe \*/

public void setColor(String col){

color=col;

}

}

/\*\* \* <code>class Material</code> represents material of shoes \* @author Oleksandr Petruk \* @version 1.0 \*/

class Material{

private String material;

/\*\* \* Constructor \*/

public Material(){

material="leather";

}

/\*\* \* Constructor with parameters \* @param mat is used to set <code>material</code> of a shoe \*/

public Material(String mat){

material=mat;

}

/\*\* \* Setter \* @param mat sets <code>material</code> of a shoe \*/

public void setMaterial(String mat){

material=mat;

}

/\*\* \* Getter

\* @return material of a shoe \*/

public String getMaterial(){

return material;

}

}

/\*\* \* <code>class Laces</code> represents laces of shoes \* @author Oleksandr Petruk \*/

class Laces{

private String color;

private boolean unlaced;

/\*\* \* Constructor \*/

public Laces(){

color="black";

unlaced=true;

}

/\*\* \* Constructor with parameters \* @param col is used to set <code>color</code> of a shoe \* @param un is used to set <code>unlaced</code> of a shoe \*/

public Laces(String col, boolean un){

color=col;

unlaced=un;

}

/\*\* \* Setter \* @param col is used to set <code>color</code> of a shoe \*/

public void setColor(String col){

color=col;

}

/\*\* \* Getter \* @return <code>color</code> of a shoe \*/

public String getColor(){

return color;

}

/\*\* \* Setter

\* @param un is used to <code>unlaced</code> \*/

public void setUnlaced(boolean un){

unlaced=un;

}

/\*\* \* Getter \* @return <code>unlaced</code> \*/

public boolean getUnlaced(){

return unlaced;

}

}

/\*\* \* <code>class Sole</code> represents sole of shoes \* @author Oleksandr Petruk \* @version 1.0 \*/

class Sole{

private String color;

/\*\* \* Constructor \*/

public Sole(){

color="black";

}

/\*\* \* Constructor with parameters \* @param col is used to set <code>color</code> of a sole \*/

public Sole(String col){

color=col;

}

/\*\* \* Setter \* @param col sets color of a sole \*/

public void setColor(String col){

color=col;

}

/\*\* \* Getter \* @return color of a sole \*/

public String getColor(){

return color;

}

}

/\*\* \* <code>class Size</code> represents size of shoes \* @author Oleksandr Petruk \* @version 1.0 \*/

class Size{

private int size;

/\*\* \* Constructor with parameters \* @param s is used to set size \*/

public Size(int s){

size=s;

}

/\*\* \* Setter \* @param s sets size \*/

public void setSize(int s){

size=s;

}

/\*\* \* Getter \* @return size of a shoe \*/

public int getSize(){

return size;

}

}

***ShoesApp.java***

package KI\_306.Petruk.Lab2;

import java.io.\*;

public class ShoesApp {

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

KI\_306.Petruk.Lab2.Shoes shoes = new KI\_306.Petruk.Lab2.Shoes("black", "white", true, "fabric", "white", 42);

shoes.shoeInfo();

shoes.unlaceShoes();

shoes.repairShoes();

shoes.cleanShoes();

shoes.changeColor("red");

shoes.changeLaces("red");

shoes.changeSoleColor("black");

shoes.putOnShoes();

shoes.expandShoes(4);

shoes.shrinkShoes(2);

shoes.laceShoes();

shoes.shoeInfo();

shoes.takeOffShoes();

shoes.dispose();

}

}

Рис.1 Вивід логу у консоль

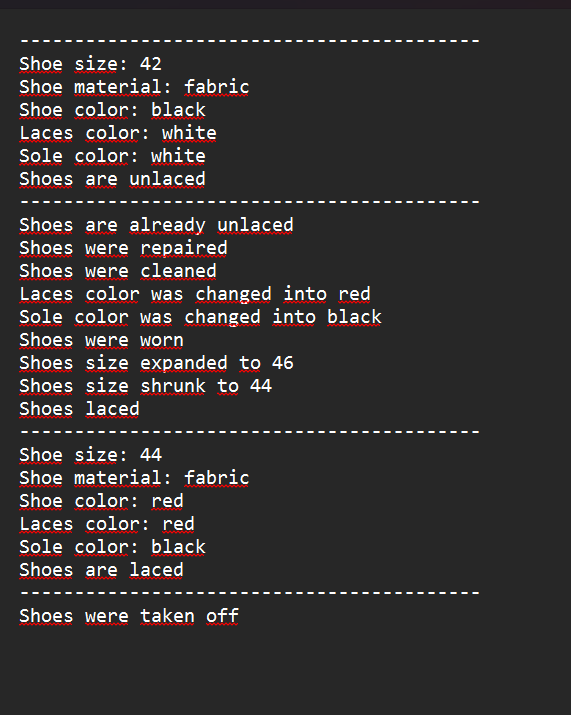


Рис.2 Вивід логу у текстовий файл

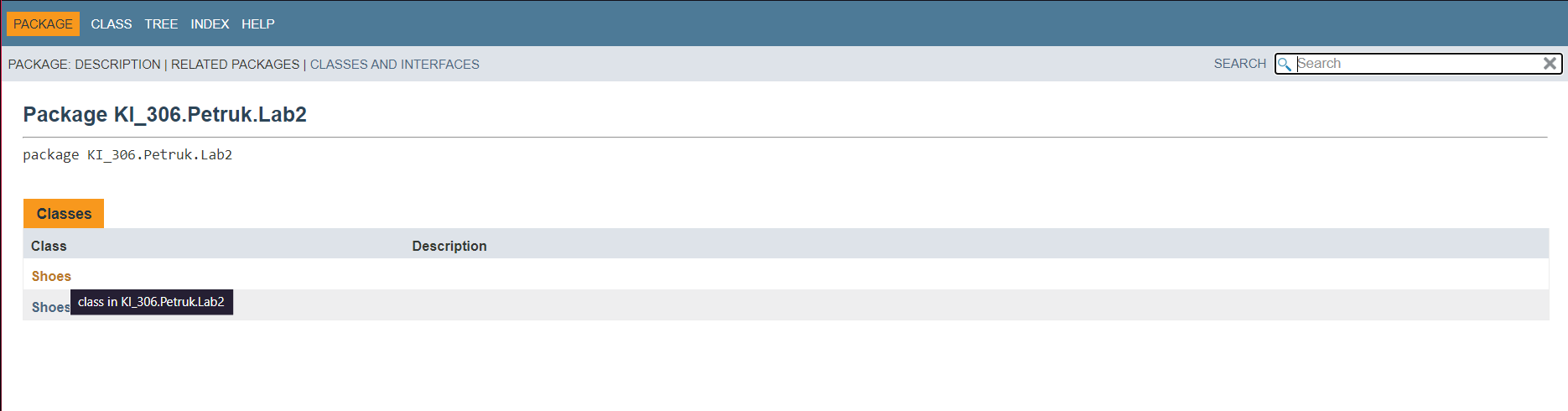


Рис.3.1 Фрагмент згенерованої документації

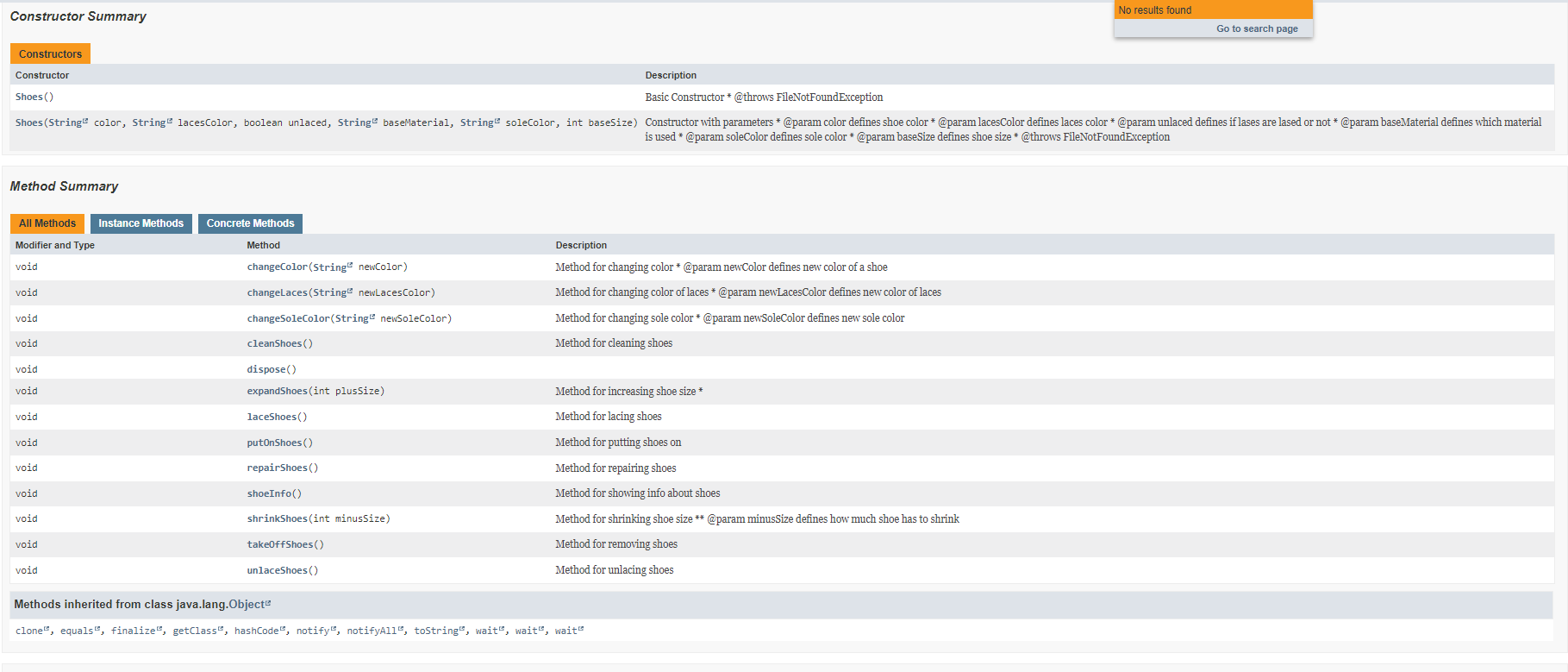


Рис.3.2 Фрагмент згенерованої документації

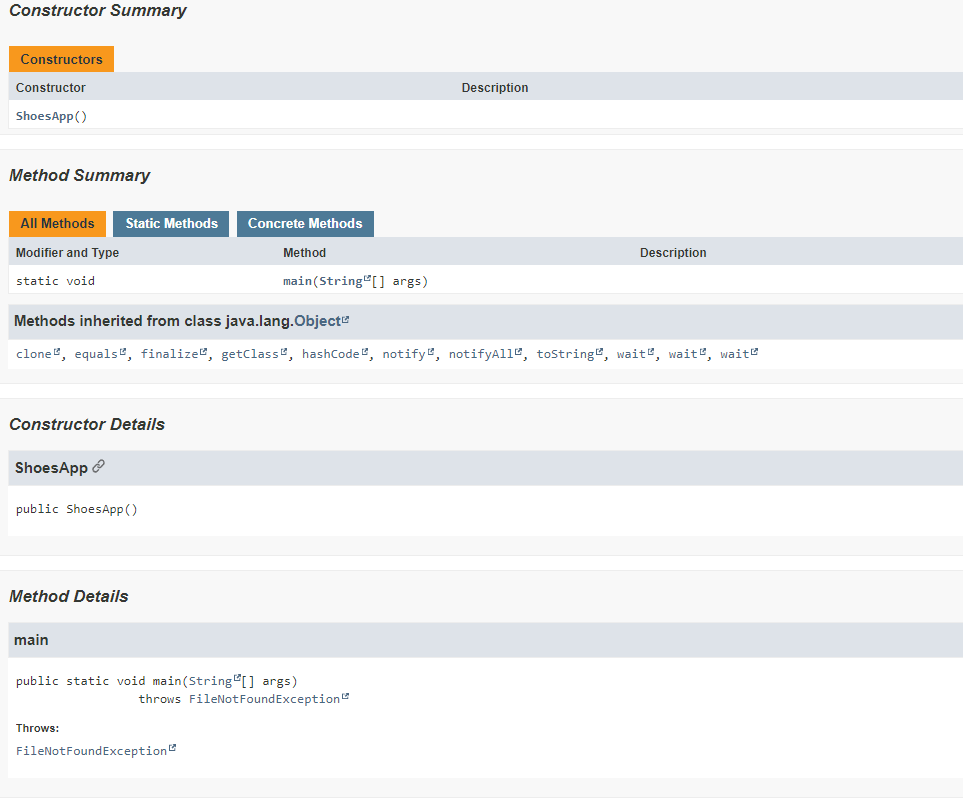


Рис.3.3 Фрагмент згенерованої документації

**Висновок:** На лабораторній роботі яознайомився з базовими конструкціями мови Java та оволодів навиками написання й автоматичного документування простих консольних програм мовою Java.