Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ

Звіт

до лабораторної роботи № 4

з дисципліни: «Кросплатформні засоби програмування»

«СПАДКУВАННЯ ТА ІНТЕРФЕЙСИ»

Варіант №22

Виконав:
ст.гр. КІ-34
Трач В.І.
Прийняв:
Іванов Ю.С.

Львів 2022

Мета роботи: ознайомитися з спадкуванням та інтерфейсами у мові Java.

Завдання:

1. Написати та налагодити програму на мові Java, що розширює клас, що реалізований у лабораторній роботі №3, для реалізації предметної області заданої варіантом. Суперклас, що реалізований у лабораторній роботі №3, зробити абстрактним. Розроблений підклас має забезпечувати механізми свого коректного функціонування та реалізовувати мінімум один інтерфейс. Програма має розміщуватися в пакеті Група.Прізвище.Lab4 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.
2. Автоматично згенерувати документацію до розробленого пакету.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації.
4. Дати відповідь на контрольні запитання.

Варіант : 22

Штурмова гвинтівка

Код програми :

AssaultRiffleApp.Java

```java
/**
 * KI34.Trach.Lab4 package
 */

package KI34.Trach.Lab4;

import java.io.FileNotFoundException;

/**
 * AssaultRiffle Application class implements main method for AssaultRiffle
 class possibilities demonstration
 * @author Vladyslav
 * @version 1.0
 */

public class AssaultRifleApp {

    public static void main(String[] args) throws FileNotFoundException {
        AssaultRifle AR1 = new AssaultRifle();
        AR1.setAutomatic();
        AR1.shoot(13);
        System.out.println(AR1.weight());


    }

    public AssaultRifleApp() {
    }
}
```

AssaultRiffle.Java

```java
package KI34.Trach.Lab4;

import java.io.FileNotFoundException;

//Оголошую  клас AssaultRifle
public class AssaultRifle extends Avtomat implements Weighed {

    //Конструктор класу AssaultRifle
    public AssaultRifle() throws FileNotFoundException {
        barrel = new Barrel();
        firemode  = Barrel.Firemods.Locked;
        magazine = new Magazine();
        scope = new Scope();
        stock = new Stock();

        stock.setStockHeavy();
    }


    @Override // Метод, що визначає вагу ШГ
    public double weight() {
        if(stock.stockType == Stock.typeOfStock.Heavy){
            weight = 6;
        } else if (stock.stockType == Stock.typeOfStock.Light) {
            weight = 4;
        }
        weight += magazine.currentMagazineCapacity * 0.025;
        return weight;
    }
```

```
}

//Оголошую  клас Stock
class Stock {
    typeOfStock stockType;
    enum typeOfStock {Heavy, Normal, Light};

    //Конструктор класу AssaultRifle
    public Stock (){
        stockType = typeOfStock.Normal;
    }


    //Метод, що встановлює важкий приклад
    public void setStockHeavy(){
        stockType = typeOfStock.Heavy;
    }

    //Метод, що встановлює нормальний приклад

    public void setStockNormal(){
        stockType = typeOfStock.Normal;
    }

    //Метод, що встановлює легкий приклад

    public void setStockLight(){
        stockType = typeOfStock.Light;
    }



}
```

Avtomat.Java

```
package KI34.Trach.Lab4;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintWriter;

//Оголошую абстрактний клас Avtomat
public abstract class Avtomat implements Weighed
{
    public Barrel barrel;
    public Barrel.Firemods firemode;
    public Magazine magazine;
    public Scope scope;
    public Stock stock;
    public double weight = 5;
    public PrintWriter fout;



//Конструктор класу Avtomat
    public Avtomat () throws FileNotFoundException {
        barrel = new Barrel();
        firemode  = Barrel.Firemods.Locked;
        magazine = new Magazine();
        scope = new Scope();

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

```
    //Метод, що дозволяє автомату стріляти
    public void shoot (int numberOfShots) {
        if (barrel.currentBarrelResource > numberOfShots &&
magazine.currentMagazineCapacity > numberOfShots && numberOfShots > 0 &&
barrel.getFiremod() != Barrel.Firemods.Locked){
            barrel.currentBarrelResource -= numberOfShots;
            magazine.currentMagazineCapacity -= numberOfShots;
            scope.currentAccuracy -= numberOfShots * 0.02321;

            fout.println("AR has shooted " + numberOfShots + " times");

        } else if(magazine.currentMagazineCapacity < numberOfShots) {
            System.out.println("Error. There`s no enough bullets in
magazine.\n");
        }else if(barrel.currentBarrelResource < numberOfShots) {
            System.out.println("Error. There`s no left barrel resource.\n");
        } else if(firemode == Barrel.Firemods.Locked) {
            System.out.println("Error. Fire mod is locked.\n");
        }
    }

    //Метод, що виводить інформацію про автомат
    public void showInfo () {
        System.out.println("\nBarrel resource is : " +
barrel.currentBarrelResource + " rounds\nIt is " +
magazine.currentMagazineCapacity + " bullets left in the magazine.\n" + "Gun
accuracy : " + scope.currentAccuracy + "%.\nFiremode : " + barrel.getFiremod() +
"\n\n");

        fout.println("The information about AR was outputed");
    }

    //Метод, що проводить повне обслуговування автомату
    public void fullService () {
        barrel.setFiremodLocked();
        barrel.cleanBarrel();
        magazine.reload();
        scope.reconfigurate();

        fout.println("AR has been serviced");
    }

    //Метод, що переводить режим вогню в автоматичний
    public void setAutomatic () {
        barrel.setFiremodAutomatic();

        fout.println("AR has been set to automatic");
    }

    //Метод, що переводить режим вогню в напів-автоматичний
    public void setSemiAutomatic () {
        barrel.setFiremodSemiautomatic();

        fout.println("AR has been set to semi-automatic");
    }

    //Метод, що переводить режим вогню на запобіжник
    public void setLocked () {
        barrel.setFiremodLocked();

        fout.println("AR has been set to locked");
    }

    //Метод, що виводить точність прицілу
    public void getTotalAcc (){
        System.out.println(scope.totalAcc);
```

```
    }

    //Метод, що закриває файл
    public void closeFile (){
        fout.close();
    }


    @Override // Метод, що визначає вагу автомата
    public double weight() {
        weight += magazine.currentMagazineCapacity * 0.025;
        return weight;
    }
}

//Оголошую  клас Barrel
class Barrel {
    public final int barrelResource = 10000;
    public int currentBarrelResource;


    //Визначення всіх можливих режимів вогню
    enum Firemods {Automatic, Semiautomatic, Locked}
    private Firemods firemod;

    //Конструктор класу Barrel
    public Barrel()
    {
        currentBarrelResource = barrelResource;
        firemod = Firemods.Locked;
    }

    //Метод, що переводить режим вогню в автоматичний
    public void setFiremodLocked() {
        firemod = Firemods.Locked;
    }

    //Метод, що переводить режим вогню в напів-автоматичний
    public void setFiremodSemiautomatic() {
        firemod = Firemods.Semiautomatic;
    }

    //Метод, що переводить режим вогню на запобіжник
    public void setFiremodAutomatic() {
        firemod = Firemods.Automatic;
    }

    //Метод, що видає інформацію про поточний режим вогню
    public Firemods getFiremod() {
        return firemod;
    }

    //Метод, який очищує ствол
    public void cleanBarrel() {
        currentBarrelResource = 10000;
        System.out.println("Gun is cleaned");
    }
}

//Оголошую  клас Magazine
class Magazine {
    public final int magazineCapacity = 30;
    public int currentMagazineCapacity;

    //Конструктор класу Magazine
    public Magazine()
```

```
    {
        currentMagazineCapacity = magazineCapacity;
    }

    //Метод, що перезаряджає автомат
    public void reload() {
        currentMagazineCapacity = 30;
        System.out.println("Gun is reloaded");
    }

}
//Оголошую  клас Scope
class Scope {
    public static double idealAccuracy = 100;
    public double currentAccuracy;
    public static double totalAcc = 0;


    //Конструктор класу Scope
    public Scope()
    {
        currentAccuracy = idealAccuracy;
    }


    //Метод з додаткового заваддня при здачі 3 лаби
    public Scope(double acc)
    {
        currentAccuracy = acc;
        if (acc > 50)
        {
            totalAcc += acc;
        }
    }

    //Метод, який переналаштовує приціл
    public void reconfigurate() {
        currentAccuracy = 100;
        System.out.println("Scope is reconfigured");
    }

}
```

Weighed.Java

```
package KI34.Trach.Lab4;

//Оголошую інтерфейс Weighed
public interface Weighed {
    double weight();
}
```

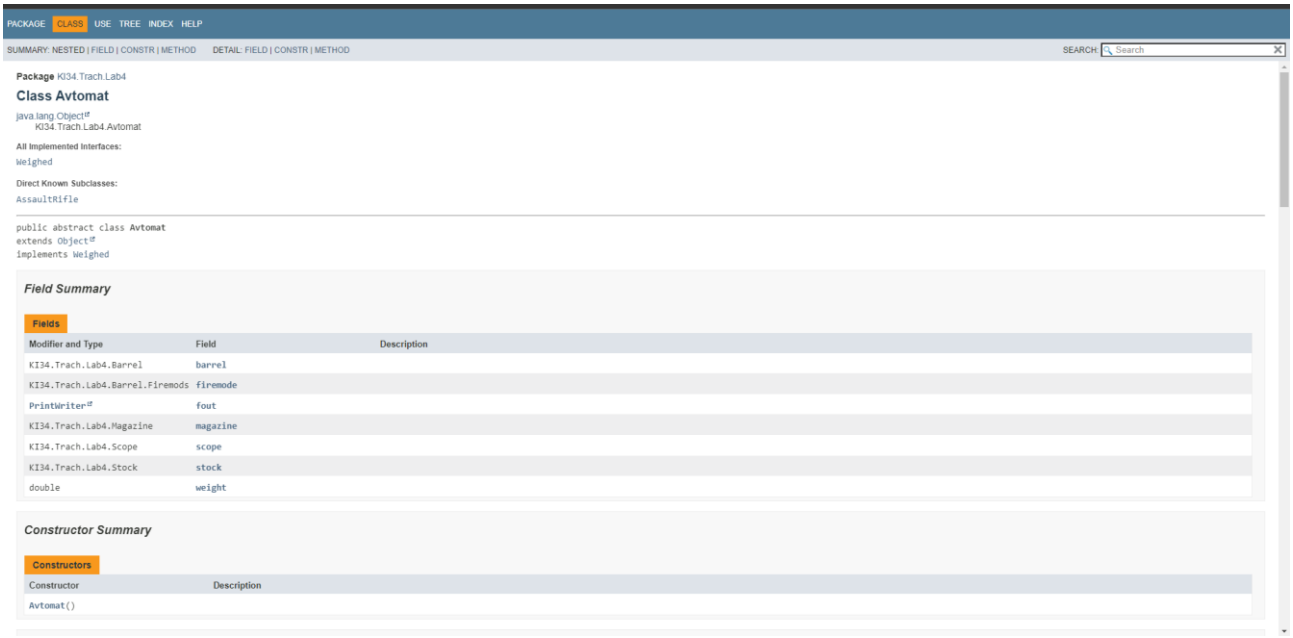
Результат виконання програми:

Вивід в консоль :

```
"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe"
6.425

Process finished with exit code 0
```

Фрагмент документації



PACKAGE CLASS USE TREE INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD SEARCH: Search

Package KI34.Trach.Lab4

Class Avtomat

java.lang.Object
KI34.Trach.Lab4.Avtomat

All Implemented Interfaces:
Weighed

Direct Known Subclasses:
AssaultRifle

public abstract class **Avtomat**
extends Object
implements Weighed

Field Summary

Fields

Modifier and Type	Field	Description
KI34.Trach.Lab4.Barrel	barrel	
KI34.Trach.Lab4.Barrel.Firemods	firemode	
PrintWriter	fout	
KI34.Trach.Lab4.Magazine	magazine	
KI34.Trach.Lab4.Scope	scope	
KI34.Trach.Lab4.Stock	stock	
double	weight	

Constructor Summary

Constructors

Constructor	Description
Avtomat()	

Висновок : Я ознайомився з спадкуванням та інтерфейсами у мові Java..