Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних технологій, автоматики та метрології

Кафедра ЕОМ



**Звіт**

**До лабораторної роботи №4**

# З дисципліни: «Кросплатформні засоби програмування»

На тему «Спадкування та інтерфейси»

# Варіант №3

Виконав: ст. гр. КІ-34

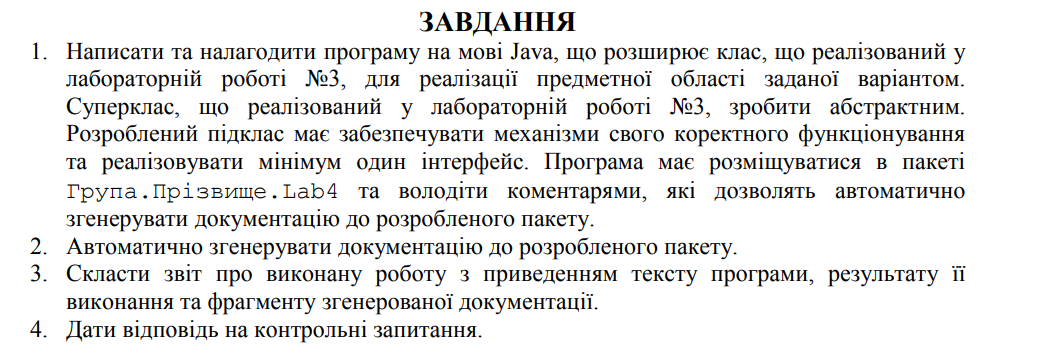
Гугель Роман

Прийняв:

Іванов Ю.С.

Львів – 2022

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





**Виконання:**

**Лістинг програми:**

package ki34.Huhel.lab4;  
  
import java.io.File;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
*/\*\*  
 \* Class Logger. Was created to log information, errors and warnings. Also there was implemented Singelton  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public class Logger  
{  
 private static Logger *logger*;  
 private final String fileName;  
  
 protected final String infoFlag = new String("[INFO] ");  
 protected final String errorFlag = new String("[ERROR] ");  
 protected final String warningFlag = new String("[WARNING] ");  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *fileName  
 \*/* private Logger(String fileName)  
 {  
 this.fileName = fileName;  
 File loggerFile = null;  
 FileWriter fout = null;  
 try  
 {  
 loggerFile = new File(fileName);  
 fout = new FileWriter(loggerFile, true);  
 SimpleDateFormat formatter= new SimpleDateFormat("yyyy-MM-dd 'at' HH:mm:ss z");  
 Date date = new Date(System.*currentTimeMillis*());  
 fout.write("[" + formatter.format(date) + "] " + "Logger start to work\n");  
 }  
 catch (IOException e)  
 {  
 System.*err*.println("Something wrong with log file" + e.getMessage());  
 System.*exit*(1);  
 }  
 finally  
 {  
 try  
 {  
 fout.flush();  
 fout.close();  
 }  
 catch (IOException e)  
 {  
 System.*out*.println(e.getMessage());  
 }  
 }  
 }  
  
 */\*\*  
 \* Method to do logging  
 \** ***@param*** *massege  
 \*/* public void log(String massege)  
 {  
 File loggerFile = null;  
 FileWriter fout = null;  
 try  
 {  
 loggerFile = new File(this.fileName);  
 fout = new FileWriter(loggerFile, true);  
 SimpleDateFormat formatter= new SimpleDateFormat("yyyy-MM-dd 'at' HH:mm:ss z");  
 Date date = new Date(System.*currentTimeMillis*());  
 fout.write("[" + formatter.format(date) + "] " + massege + "\n");  
 }  
 catch (IOException e)  
 {  
 System.*err*.println("Something wrong with log file" + e.getMessage());  
 System.*exit*(1);  
 }  
 finally  
 {  
 try  
 {  
 fout.flush();  
 fout.close();  
 }  
 catch (IOException | NullPointerException e)  
 {  
 System.*out*.println(e.getMessage());  
 }  
 }  
 }  
  
 */\*\*  
 \* Singleton implementation  
 \** ***@param*** *fileName  
 \** ***@return*** *\*/* public static Logger getLogger(String fileName)  
 {  
 if (*logger* == null)  
 {  
 *logger* = new Logger(fileName);  
 }  
 return *logger*;  
 }  
  
 */\*\*  
 \* Getter for logger  
 \** ***@return*** *logger  
 \*/* public static Logger getLogger()  
 {  
 return *logger*;  
 }  
  
}

package ki34.Huhel.lab4;  
  
*/\*\*  
 \* Class leg  
 \** ***@author*** *Roman  
 \** ***@version*** *1.0  
 \*/*public class Leg  
{  
 private String color;  
 private double length;  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *color  
 \** ***@param*** *length  
 \*/* public Leg(String color, double length) {  
 this.color = color;  
 this.length = length;  
 }  
  
 */\*\*  
 \* Getter for color  
 \** ***@return*** *color  
 \*/* public String getColor() {  
 return color;  
 }  
  
 */\*\*  
 \* Setter for color  
 \** ***@param*** *color  
 \*/* public void setColor(String color) {  
 this.color = color;  
 }  
  
 */\*\*  
 \* Getter for length  
 \** ***@return*** *length  
 \*/* public double getLength() {  
 return length;  
 }  
  
 */\*\*  
 \* Setter for length  
 \** ***@param*** *length  
 \*/* public void setLength(double length) {  
 this.length = length;  
 }  
  
 @Override  
 public String toString() {  
 return "Leg{ " +  
 "color = '" + color + '\'' +  
 ", length = " + length +  
 '}';  
 }  
}

package ki34.Huhel.lab4;  
  
*/\*\*  
 \* Class Head  
 \** ***@author*** *Roman  
 \** ***@version*** *1.0  
 \*/*public class Head  
{  
 private String color;  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *color  
 \*/* public Head(String color) {  
 this.color = color;  
 }  
  
 */\*\*  
 \* Getter for color  
 \** ***@return*** *color  
 \*/* public String getColor() {  
 return color;  
 }  
  
 */\*\*  
 \* Setter for color  
 \** ***@param*** *color  
 \*/* public void setColor(String color) {  
 this.color = color;  
 }  
  
 @Override  
 public String toString() {  
 return "Head{ " +  
 "color = '" + color + '\'' +  
 '}';  
 }  
}

package ki34.Huhel.lab4;  
  
public class ExperimentalDog extends Dog implements Bite  
{  
 */\*\*  
 \* Constructor  
 \*  
 \** ***@param*** *head  
 \** ***@param*** *body  
 \** ***@param*** *frontRightLeg  
 \** ***@param*** *backRightLeg  
 \** ***@param*** *frontLeftLeg  
 \** ***@param*** *backLeftLeg  
 \** ***@param*** *name  
 \** ***@param*** *breed  
 \*/* public ExperimentalDog(Head head  
 , Body body  
 , Leg frontRightLeg  
 , Leg backRightLeg  
 , Leg frontLeftLeg  
 , Leg backLeftLeg  
 , String name  
 , String breed)  
 {  
 super(head, body, frontRightLeg, backRightLeg, frontLeftLeg, backLeftLeg, name, breed);  
 logger.log(logger.infoFlag + "ExperimentalDog constructor called");  
 }  
  
 @Override  
 public void ToBite(String str) {  
 System.*out*.println("Experimental dog bite " + str);  
 }  
  
 @Override  
 public void Respond() {  
 System.*out*.println("I am experimental dog. My name is " + name + " and my breed is " + breed);  
 Bark();  
 logger.log(logger.infoFlag + "ExperimentalDog Respond method was called");  
 }  
  
 @Override  
 public void Bark() {  
 System.*out*.println("Gav-gav-gav rafff-rafff-raffff");  
 logger.log(logger.infoFlag + "ExperimentalDog bark method was called");  
 }  
  
 @Override  
 public String toString() {  
 return "ExperimentalDog: \n" +  
 "name='" + name + '\'' + "\n" +  
 "breed='" + breed + '\'' + "\n" +  
 "head=" + head + "\n" +  
 "body=" + body + "\n" +  
 "frontRightLeg=" + frontRightLeg + "\n" +  
 "backRightLeg=" + backRightLeg + "\n" +  
 "frontLeftLeg=" + frontLeftLeg + "\n" +  
 "backLeftLeg=" + backLeftLeg +"\n";  
 }  
}

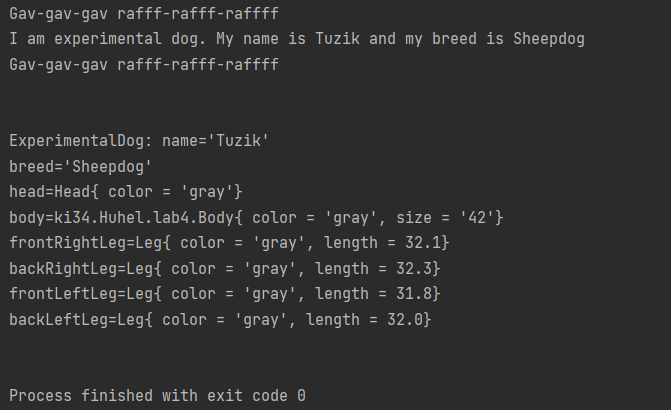
package ki34.Huhel.lab4;  
  
*/\*\*  
 \* Class Dog  
 \** ***@author*** *Roman  
 \** ***@version*** *1.0  
 \*/*public abstract class Dog  
{  
 protected Head head;  
 protected Body body;  
 protected Leg frontRightLeg;  
 protected Leg backRightLeg;  
 protected Leg frontLeftLeg;  
 protected Leg backLeftLeg;  
 protected String name;  
 protected String breed;  
 protected Logger logger = Logger.*getLogger*("logs.txt");  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *head  
 \** ***@param*** *body  
 \** ***@param*** *frontRightLeg  
 \** ***@param*** *backRightLeg  
 \** ***@param*** *frontLeftLeg  
 \** ***@param*** *backLeftLeg  
 \** ***@param*** *name  
 \** ***@param*** *breed  
 \*/* public Dog(Head head, Body body,  
 Leg frontRightLeg, Leg backRightLeg,  
 Leg frontLeftLeg, Leg backLeftLeg,  
 String name, String breed) {  
 this.head = head;  
 this.body = body;  
 this.frontRightLeg = frontRightLeg;  
 this.backRightLeg = backRightLeg;  
 this.frontLeftLeg = frontLeftLeg;  
 this.backLeftLeg = backLeftLeg;  
 this.name = name;  
 this.breed = breed;  
 logger.log(logger.infoFlag + "Dog constructor called");  
 }  
  
 */\*\*  
 \* Method Respond  
 \*/* public abstract void Respond();  
  
 */\*\*  
 \* Method Bark  
 \*/* public abstract void Bark();  
  
 @Override  
 public String toString() {  
 return "Dog: \n" +  
 "name = '" + name + '\'' + "\n" +  
 "breed = '" + breed + '\'' + "\n" +  
 "head = " + head + "\n" +  
 "body = " + body + "\n" +  
 "frontRightLeg = " + frontRightLeg + "\n" +  
 "backRightLeg = " + backRightLeg + "\n" +  
 "frontLeftLeg = " + frontLeftLeg + "\n" +  
 "backLeftLeg = " + backLeftLeg;  
 }  
  
 public Head getHead() {  
 return head;  
 }  
  
 public void setHead(Head head) {  
 this.head = head;  
 }  
  
 public Body getBody() {  
 return body;  
 }  
  
 public void setBody(Body body) {  
 this.body = body;  
 }  
  
 public Leg getFrontRightLeg() {  
 return frontRightLeg;  
 }  
  
 public void setFrontRightLeg(Leg frontRightLeg) {  
 this.frontRightLeg = frontRightLeg;  
 }  
  
 public Leg getBackRightLeg() {  
 return backRightLeg;  
 }  
  
 public void setBackRightLeg(Leg backRightLeg) {  
 this.backRightLeg = backRightLeg;  
 }  
  
 public Leg getFrontLeftLeg() {  
 return frontLeftLeg;  
 }  
  
 public void setFrontLeftLeg(Leg frontLeftLeg) {  
 this.frontLeftLeg = frontLeftLeg;  
 }  
  
 public Leg getBackLeftLeg() {  
 return backLeftLeg;  
 }  
  
 public void setBackLeftLeg(Leg backLeftLeg) {  
 this.backLeftLeg = backLeftLeg;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getBreed() {  
 return breed;  
 }  
  
 public void setBreed(String breed) {  
 this.breed = breed;  
 }  
}

package ki34.Huhel.lab4;  
  
*/\*\*  
 \* Class ki34.Huhel.lab4.Body  
 \** ***@author*** *Roman  
 \** ***@version*** *1.0  
 \*/*public class Body  
{  
 private String color;  
 private String size;  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *color  
 \** ***@param*** *size  
 \*/* public Body(String color, String size) {  
 this.color = color;  
 this.size = size;  
 }  
  
 */\*\*  
 \* Getter for color  
 \** ***@return*** *color  
 \*/* public String getColor() {  
 return color;  
 }  
  
 */\*\*  
 \* Setter for color  
 \** ***@param*** *color  
 \*/* public void setColor(String color) {  
 this.color = color;  
 }  
  
 */\*\*  
 \* Getter for size  
 \** ***@return*** *\*/* public String getSize() {  
 return size;  
 }  
  
 */\*\*  
 \* Setter for size  
 \** ***@param*** *size  
 \*/* public void setSize(String size) {  
 this.size = size;  
 }  
  
 @Override  
 public String toString() {  
 return "ki34.Huhel.lab4.Body{ " +  
 "color = '" + color + '\'' +  
 ", size = '" + size + '\'' +  
 '}';  
 }  
}

package ki34.Huhel.lab4;  
  
public interface Bite  
{  
 void ToBite(String str);  
}

package ki34.Huhel.lab4;  
  
public class Main {  
 public static void main(String[] args) {  
 ExperimentalDog experimentalDog = new ExperimentalDog(new Head("gray")  
 , new Body("gray", "42")  
 , new Leg("gray", 32.1)  
 , new Leg("gray", 32.3)  
 , new Leg("gray", 31.8)  
 , new Leg("gray", 32.0)  
 , "Tuzik", "Sheepdog");  
  
 experimentalDog.Bark();  
 experimentalDog.Respond();  
 System.*out*.println("\n\n" + experimentalDog);  
 }  
}

**Результати:**

****

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