Міністерство освіти і науки України

Національний університет "Львівська політехніка"

Кафедра ЕОМ



Звіт

з лабораторної роботи №4

з дисципліни: «Кросплатформенні засоби програмування»

на тему: «Спадкування та інтерфейси»

Виконав: ст.гр. КІ-34

Лис Б. Л.

Прийняв:

викл. каф. ЕОМ

Іванов Ю. С.

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.

Лістинг програми:

Файл MonitorApp.java

```
package KI34.Lys.Lab4;
    * @param args param
    * @throws FileNotFoundException
   public static void main(String[] args) throws FileNotFoundException {
       object.changeMatrix("IPS");
       object.changeHz(75);
       object.getResource();
       object.dispose();
```

}

Файл Monitor.java

```
@throws FileNotFoundException param
     * # @param name <code>name</code> Name of monitor
     * @param diagonal <code>diagonal</code> Monitor diagonal
     * @param matrix <code>matrix</code> Monitor matrix
     * @param resolution <code>resolution</code> Monitor resolution
     * @param hz <code>hz</code> Monitor refresh rate
     * Cparam BrightDisplay <code>BrightDisplay</code> The brightness of the
     * @param monitorFormat <code>monitorFormat</code> Monitor screen format
    public Monitor (String name, double diagonal, String matrix, String
FileNotFoundException {
        this.matrix = matrix;
```

```
* @return monitor`s refresh rate
public String getResolution() {
public double getDiagonal() {
public String getMatrix() {
public double getBrightDisplay() {
* @param name <code>name</code> monitor`s name
```

```
* @param diagonal <code>diagonal</code> monitor`s diagonal
  @param matrix <code>matrix</code> monitor`s matrix type
public void setMatrix(String matrix) {
 * @param brightDisplay <code>brightDisplay</code> monitor`s display
public void setBrightDisplay(double brightDisplay) {
public void off onMonitor(Status comand) {
public void getSettings() {
```

```
public void brightDisplayMinus(int change) {
    public void changeMatrix(String mat) {
    public void changeHz(int hz) {
    public void changeName(String name) {
name);
```

```
public void changeMonitorFormat(String monitorFormat) {
public void dispose()
```

Файл TouchScreen.java

```
/**
    * lab 4 package
    */

package KI34.Lys.Lab4;

import java.io.FileNotFoundException;

/**
    * Class <code>TouchScreen</code> implements PresForce
    * * @author Lys Bohdan
    * * @version 1.0
    **/

public class TouchScreen extends Monitor implements PresForce {
    private int resource;
    private final double presForceConst = 2.5;
    private double presForce;
```

```
* @throws FileNotFoundException param
    public TouchScreen() throws FileNotFoundException {
     * @param name <code>name</code> Name of monitor
     * @param diagonal <code>diagonal </code> Monitor diagonal
     * @param matrix <code>matrix</code> Monitor matrix
     * @param monitorFormat <code>monitorFormat</code> Monitor screen format
      @param resource <code>resource</code> Touch monitor resource
      @param presForce <code>presForce</code> Optimal pressing force
      @throws FileNotFoundException
String monitorFormat) throws FileNotFoundException{
monitorFormat);
    public void screenResource(int touches) {
```

```
@Override
public void aboveTheNorm(int touches) {
    resource = resource - touches * error;
}
```

Файл PresForce.java

```
/**
  * lab 4 package
  */

package KI34.Lys.Lab4;

/**
  * Interface <code>PresForce</code> extends Resource
  * * @author Lys Bohdan
  * * @version 1.0
  **/

// оголошуемо інтерфейс PresForce, що успадковує інтерфейс Resource
public interface PresForce extends Resource{
  void aboveTheNorm (int touches); // прототип методу
  int error = 3; // константа
}
```

Файл Resource.java

```
/**
  * lab 4 package
  */

package KI34.Lys.Lab4;

/**

  * Interface <code>Resource</code>
     * * @author Lys Bohdan
     * * @version 1.0
     **/

// оголошуемо інтерфейс Resource
public interface Resource {
    void screenResource (int touches); // прототип методу
}
```

Файл Status.java

```
/**
  * lab 4 package
  */
package KI34.Lys.Lab4;

/**
  * Method simulates off/on
  */
public enum Status {
    ON, OFF
}
```

Результат виконання програми:

```
C:\Users\mrpet\.jdks\openjdk-18.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\
Name - MSI
Diagonal - 24.0
Resolution - 1920 x 1080
Hz - 144.0
Matrix - LED
Bright - 100.0
Selected matrix - LED
Matrix changed - IPS
Selected Hz - 75
Hz changed - 75
Selected resolution - 1080x720
Resolution changed - 1080x720
Monitor format changed - 4:3
Max resolution for this format is 1440x1080.
Name - MSI
Diagonal - 24.0
Resolution - 1080x720
Hz - 75.0
Matrix - IPS
Bright - 62.0
Touch monitor resource - 998266
Process finished with exit code 0
```

```
Log.txt: Блокнот
```

Файл Редагування Формат Вигляд Довідка

Монітор включено.

Налаштування монітора виведенно.

Яскравість монітора збільшенна.

Яскравість монітора зменшенна.

Матриця монітора зміненна з LED на IPS

Частота оновлення монітура зміненна з 144.0 на 75

Розширення монітура зміненно з 1920 x 1080 на 1080x720

Формат монітора зміненно на - 4:3

Налаштування монітора виведенно.

Ресурс сенсорного монітора зменшенно на 1734

Ресурс сенсорного монітора виведенно.

Монітор виключено.

PACKAGE: DESCRIPTION | RELATED PACKAGES | CLASSES AND INTERFACES

Package KI34.Lys.Lab4

package KI34.Lys.Lab4

All Classes and Interfaces	Interfaces	Classes	Enum Classes
Class	ı	Description	
Monitor	(Class Monito	r implements monitor * @author Lys Bohdan * @version 1.0
MonitorApp	1	Monitor Appl	ication class implements main method for Monitor class possibilities demonstration
PresForce	1	Interface Pre	sForce extends Resource * @author Lys Bohdan * @version 1.0
Resource	1	Interface Res	ource * @author Lys Bohdan * @version 1.0
Status	1	Method simul	lates off/on
TouchScreen	(Class TouchS	creen implements PresForce * @author Lys Bohdan * @version 1.0

Згенерована документація

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

Package Kl34.Lys.Lab4

Class Monitor

java.lang.Object[®] Kl34.Lys.Lab4.Monitor

Direct Known Subclasses:

TouchScreen

public abstract class Monitor extends Object[®]

Class Monitor implements monitor * @author Lys Bohdan * @version 1.0

Field Summary

Fields			
Modifier and Type	Field	Description	
PrintWriter [®]	myWrite		
Status	status		

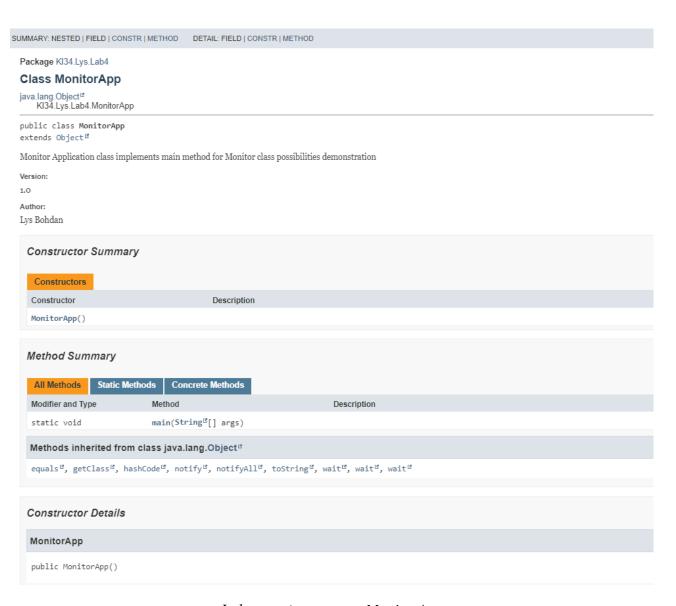
Constructor Summary

Constructors

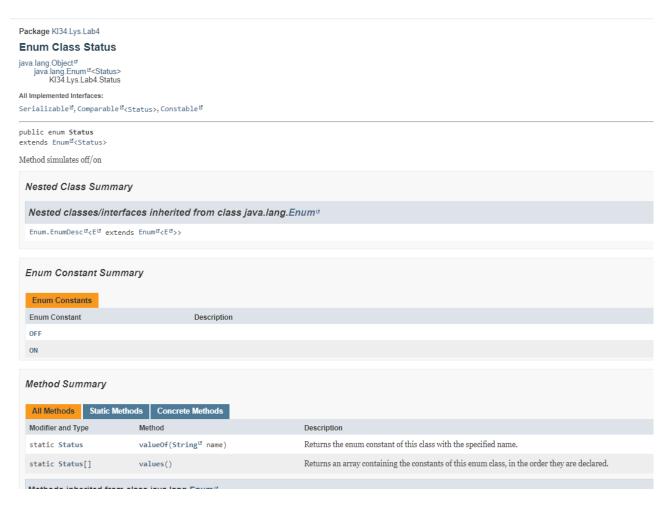
Constructor Description Constructor Monitor(String® name, double diagonal, String® matrix, String® resolution, double hz, double BrightDisplay, String® monitorFormat) Constructor

Method Summary		
	Methods Concrete Methods Method	Description
Modifier and Type	brightDisplayMinus(int change)	Method simulates decrement monitor's display brightness
void	brightDisplayPlus(int change)	Method simulates increment monitor's display brightness
void	changeHz(int hz)	Method simulates changing monitor's refresh rate
void	changeMatrix(String® mat)	Method simulates changing monitor`s matrix type
void	changeMonitorFormat(String® monitorFormat)	Method simulates changing monitor`s screen format
void	changeName(String® name)	Method simulates changing monitor`s name
void	changeResolution(String® resolution)	Method simulates changing monitor`s resolution
void	dispose()	Method releases used recourses
double	getBrightDisplay()	Method returns monitor`s display brightness
double	getDiagonal()	Method returns monitor`s diagonal
double	getHz()	Method returns monitor`s refresh rate
String [®]	getMatrix()	Method returns monitor`s matrix type
String [®]	getName()	Method returns monitor`s name
String [®]	getResolution()	Method returns monitor`s resolution
void	getSettings()	Method return monitor`s settings which we set
void	off_onMonitor(Status comand)	Method simulates monitor`s off/on
void	setBrightDisplay(double brightDisplay)	Method sets the new monitor`s display brightness
void	setDiagonal(double diagonal)	Method sets the new monitor's diagonal
void	setHz(double hz)	Method sets the new monitor's refresh rate
void	setMatrix(String [®] matrix)	Method sets the new monitor`s matrix type
void	setName(String [®] name)	Method sets the new monitor`s name
void	setResolution(String [®] resolution, String [®] monitorF	Format) Method sets the new monitor's resolution and format

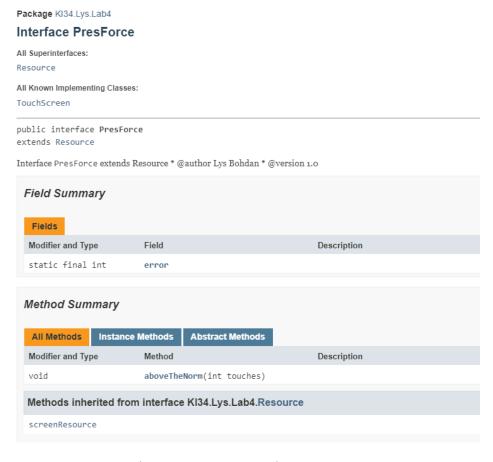
Інформація про клас Monitor



Інформація про клас MonitorApp



Інформація про клас EnumStatus



Інформація про інтерфейс PresForce

Package KI34.Lys.Lab4

Interface Resource

All Known Subinterfaces:

PresForce

All Known Implementing Classes:

TouchScreen

public interface Resource

Interface Resource * @author Lys Bohdan * @version 1.0

Method Summary

All Methods	Instance Methods	Abstract Methods
Modifier and Typ	e Method	
void	screenRe	esource(int touches

Method Details

screenResource

void screenResource(int touches)

Інформація про інтерфейс Resource

Package KI34.Lvs.Lab4 Class TouchScreen java.lang.Object[@] Kl34.Lys.Lab4.Monitor Kl34.Lys.Lab4.TouchScreen PresForce, Resource public class TouchScreen extends Monitor implements PresForce Class TouchScreen implements PresForce * @author Lys Bohdan * @version 1.0 Field Summary Fields inherited from class Kl34.Lys.Lab4.Monitor myWrite, status Fields inherited from interface KI34.Lys.Lab4.PresForce Constructor Summary Constructors Constructor Description TouchScreen(int resource, double presForce, String[@] name, double diagonal, String[@] matrix, String[®] resolution, double hz, double BrightDisplay, String[®] monitorFormat) Constructor

Інформація про клас TouchScreen

Відповіді на контрольні запитання:

1. Синтаксис реалізації спадкування?

```
class Підклас extends Суперклас {
    Додаткові поля і методи
}
```

2. Що таке суперклас та підклас?

Базовий клас найчастіше називається суперкласом, а похідний клас – підкласом.

3. Як звернутися до членів суперкласу з підкласу?

4. Коли використовується статичне зв'язування при виклику методу?

Статичне зв'язування використовується коли метод є приватним, статичним, фінальним або конструктором.

5. Як відбувається динамічне зв'язування при виклику методу?

Віртуальна машина повинна викликати версію методу, що відповідає фактичному типу об'єкту на який посилається об'єктна змінна.

Оскільки на пошук необхідного методу потрібно багато часу, то віртуальна машина заздалегідь створює для кожного класу таблицю методів, в якій перелічуються сигнатури всіх методів і фактичні методи, що підлягають виклику. При виклику методу віртуальна машина просто переглядає таблицю методів.

6. Що таке абстрактний клас та як його реалізувати?

Абстрактний клас – це клас, для якого не можна створити об'єкти, призначений бути основою для розробки ієрархії класів.

Реалізується за допомогою ключового слова abstract.

7. Для чого використовується ключове слово instanceof?

Оператор **instanceof** дозволяє визначити, чи вказаний об'єкт належить до заданого типу.

8. Як перевірити чи клас є підкласом іншого класу?

Щоб перевірити чи клас ϵ підкласом іншого класу, потрібно за допомогою instanceof порівняти, чи дійсно посилання на об'єкт супертипу посилається на об'єкт підтипу.

9. Що таке інтерфейс?

Це абстрактний тип, який використовується для визначення поведінки, яку класи повинні реалізовувати

10. Як оголосити та застосувати інтерфейс?

```
[public] interface НазваІнтерфейсу
{
Прототипи методів та оголошення констант інтерфейсу
}
```

Висновок:

На цій лабораторній роботі, я ознайомитися з спадкуванням та інтерфейсами у мові Java.