|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА  Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**ОТЧЁТ ПО ПРАКТИЧЕСКИМ РАБОТАМ №1-24**

**по дисциплине «Программирование на языке Джава»**

|  |  |  |  |
| --- | --- | --- | --- |
| Выполнил:  Студент группы ИКБО-14-22 | « » 2023 г. | (подпись) | Алексанян С.А. |
|  |  |  |  |
| Принял:  Старший преподаватель кафедры ИиППО ИИТ | « » 202 г. | (подпись) | Матчин В.Т. |

Москва 2023 г.

Оглавление

[Практическая работа №7. Реализация интерфейсов. 3](#_Toc147483247)

[Задание №1: Создайте в draw.io UML диаграмму и напишите по ней реализацию. Диаграмма должна включать в себя следующие элементы: интерфейс Movable, содержащий в себе методы для движения прямоугольника (вверх, вниз, влево, вправо) и класс MovableRectangle (движущийся прямоугольник), реализующий интерфейс Movable. 3](#_Toc147483248)

[Задание №2: Напишите по диаграмме класс MovableRectangle (движущийся прямоугольник), реализующий интерфейс Movable, класс прямоугольник, который можно представить как две движущиеся точки MovablePoint (верхняя левая и нижняя правая точки – topLeft и bottomRight), также реализующие интерфейс Movable; 3](#_Toc147483249)

[Задание №4: Разработайте интерфейс MathCalculable, который содержит 67 объявления математических функций: возведения в степень и модуль комплексного числа, также содержит число PI. Напишите класс MathFunc, который реализует, реализует этот интерфейс. Например, вычисления длины окружности, для чего используйте число PI из интерфейса. Протестируйте класс 6](#_Toc147483250)

[Задание №5: Разработайте интерфейс для работы со строками, который содержит а) функции подсчета символов в строке б) функция возвращает строку, которая образовывает строку, состоящую из символов исходной строки s, которые размещены на нечетных позициях: 1, 3, 5, ...в) функцию инвертирования строки 7](#_Toc147483251)

[Задание №6: Реализуйте интерфейс в классе ProcessStrings и протестируйте работу класса 7](#_Toc147483252)

[Задание №7: Создать статический метод printMagazines(Printable[] printable) в классе Magazine, который выводит на консоль названия только журналов. 8](#_Toc147483253)

[Вывод 10](#_Toc147483254)

# **Практическая работа №7. Реализация интерфейсов.**

**Дата**: 05.10.2023

**Цель**: цель данной практической работы – научится разрабатывать практике пользовательские интерфейсы, и применять их в программах на языке Джава.

## **Задание №1**: Создайте в draw.io UML диаграмму и напишите по ней реализацию. Диаграмма должна включать в себя следующие элементы: интерфейс Movable, содержащий в себе методы для движения прямоугольника (вверх, вниз, влево, вправо) и класс MovableRectangle (движущийся прямоугольник), реализующий интерфейс Movable.

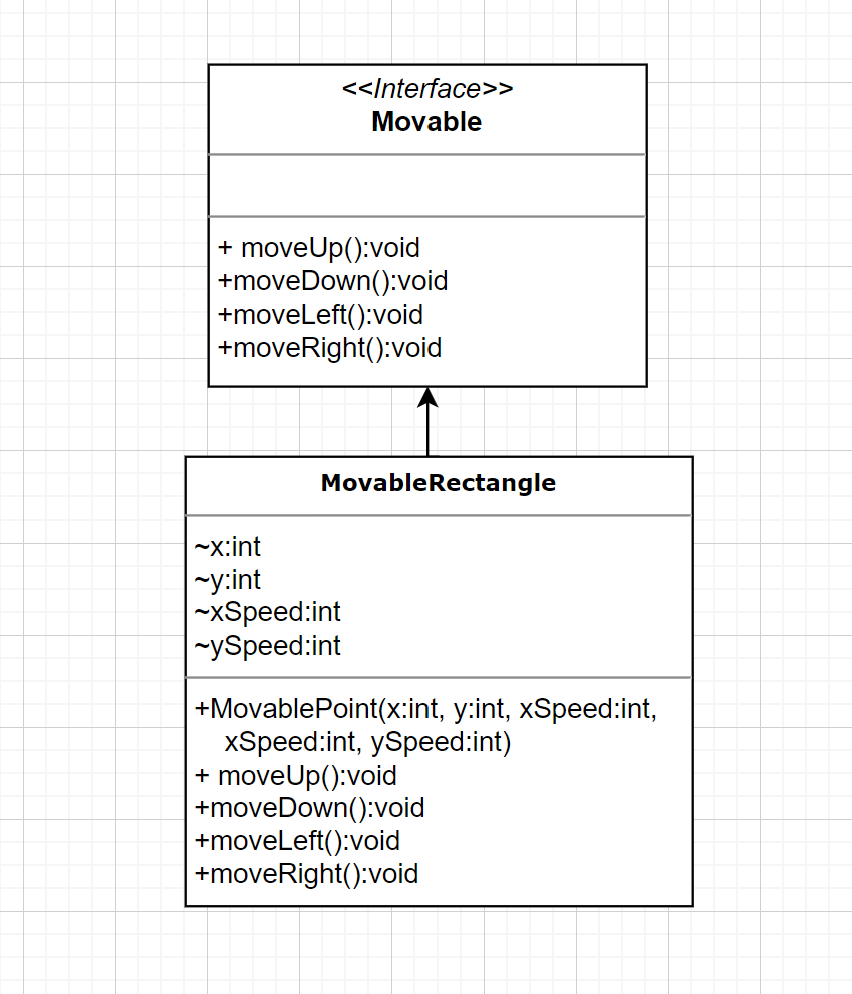


Рисунок 1 – Созданная диаграмма в draw.io.

## **Задание №2**: Напишите по диаграмме класс MovableRectangle (движущийся прямоугольник), реализующий интерфейс Movable, класс прямоугольник, который можно представить как две движущиеся точки MovablePoint (верхняя левая и нижняя правая точки – topLeft и bottomRight), также реализующие интерфейс Movable;

**Задание №3**: Напишите по диаграмме класс MovableRectangle (движущийся прямоугольник), реализующий интерфейс Movable, класс прямоугольник, который можно представить как две движущиеся точки MovablePoint (верхняя левая и нижняя правая точки – topLeft и bottomRight), также реализующие интерфейс Movable;

**Задание №4**: Добавьте в класс параметризированные конструкторы, входящие в состав классов; метод в классах для перевода числовых значений в Строку. Убедитесь, что две точки имеют одну и ту же скорость при помощи специального логического метода SpeedTest(), проверяющего это.

**public** **interface** Movable {

void moveUp();

void moveDown();

void moveLeft();

void moveRight();

}

Листинг 1 – Интерфейс Movable.

**public** **class** MovablePoint {

**public** int x,y, xSpeed, ySpeed;

MovablePoint(int x, int y, int xSpeed, int ySpeed){

**this**.x = x;

**this**.y = y;

**this**.xSpeed = xSpeed;

**this**.ySpeed = ySpeed;

}

**public** void setX(int x) {

**this**.x = x;

}

**public** void setY(int y) {

**this**.y = y;

}

**public** int getX() {

**return** x;

}

**public** int getY() {

**return** y;

}

}

Листинг 2 – Реализация класса MovablePoint.

**public** **class** MovableRectangle **implements** Movable {

**public** MovablePoint topLeft;

**public** MovablePoint bottomRight;

**public** MovableRectangle(MovablePoint topLeft, MovablePoint bottomRight) {

**if** (SpeedTest(topLeft, bottomRight))

{

**this**.topLeft = topLeft;

**this**.bottomRight = bottomRight;

}

**else**{

**System**.out.println("Скорости точек должны быть равны!");

**System**. exit(0);

}

}

**public** boolean SpeedTest(MovablePoint point1,MovablePoint point2){

**if** (!((point1.xSpeed == point2.xSpeed) && (point1.ySpeed == point2.ySpeed)))

{

**return** **false**;

}

**return** **true**;

}

**@Override**

**public** **String** toString() {

**return** "MovableRectangle{" +

"x1=" + topLeft.x +

", y1=" + topLeft.y +

", x2=" + bottomRight.x +

", y2=" + bottomRight.y +

'}';

}

**public** void moveUp(){

topLeft.setY(topLeft.y+ topLeft.ySpeed);

bottomRight.setY(bottomRight.y+topLeft.ySpeed);

}

**public** void moveDown(){

topLeft.setY(topLeft.y - topLeft.ySpeed);

bottomRight.setY(bottomRight.y - topLeft.ySpeed);

}

**public** void moveLeft(){

topLeft.setX(topLeft.x-topLeft.xSpeed);

bottomRight.setX(bottomRight.x-topLeft.xSpeed);

}

**public** void moveRight(){

topLeft.setX(topLeft.x+topLeft.xSpeed);

bottomRight.setX(bottomRight.x+topLeft.xSpeed);

}

}

Листинг 3 – Реализация класса MovableRectangle.

**import** java.util.Scanner;

**public** **class** Test {

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

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

**System**.out.print("Введите координаты x и y верхней левой точки прямоугольника, а потом скорость точки по x и y:\n>>");

MovablePoint topLeft = **new** MovablePoint(scanner.nextInt(), scanner.nextInt(), scanner.nextInt(), scanner.nextInt());

**System**.out.print("Введите координаты x и y нижней правой точки прямоугольника, а потом скорость точки по x и y:\n>>");

MovablePoint bottomRight = **new** MovablePoint(scanner.nextInt(), scanner.nextInt(), scanner.nextInt(), scanner.nextInt());

Movable rectangle = **new** MovableRectangle(topLeft, bottomRight);

**System**.out.println("START");

**System**.out.println(rectangle.toString());

**System**.out.println("Скорость по x: " + bottomRight.xSpeed + "\nСкорость по y: " + bottomRight.ySpeed);

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

**System**.out.println("Сдвиг вниз:");

rectangle.moveDown();

**System**.out.println(rectangle.toString());

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

**System**.out.println("Сдвиг вверх:");

rectangle.moveUp();

**System**.out.println(rectangle.toString());

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

**System**.out.println("Сдвиг вправо:");

rectangle.moveRight();

**System**.out.println(rectangle.toString());

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

**System**.out.println("Сдвиг влево:");

rectangle.moveLeft();

**System**.out.println(rectangle.toString());

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

}

}

Листинг 4 – Реализация теста.

## **Задание №4**: Разработайте интерфейс MathCalculable, который содержит 67 объявления математических функций: возведения в степень и модуль комплексного числа, также содержит число PI. Напишите класс MathFunc, который реализует, реализует этот интерфейс. Например, вычисления длины окружности, для чего используйте число PI из интерфейса. Протестируйте класс

**public** **interface** MathCalculable {

**public** double num\_pi =**Math**.PI;

int power(int base, int power);

double module(double mRe, double mIm);

double circle\_length(int r);

}

Листинг 5 – Интерфейс MathCalculable.

**public** **class** MathFunc **implements** MathCalculable

{

**public** double mRe, mIm;

**public** double module(double mRe, double mIm){

**return** **Math**.sqrt(mRe\*mRe + mIm\*mIm);

}

**public** int power(int base, int power) {

int result = base;

**for** (int i=1; i<power; i++)

{

result\*=base;

}

**return** result;

}

**public** double circle\_length(int r){

**return** 2\*MathCalculable.num\_pi\*r;

}

}

Листинг 6 – Реализация класса MathFunc.

**public** **class** Test {

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

MathCalculable tool = **new** MathFunc();

**System**.out.println("Возведение в степень: ");

**System**.out.println(tool.power(2,3));

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

**System**.out.println("Модуль комплексного числа: ");

**System**.out.println(tool.module(3,6));

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

**System**.out.println("Длина окружности: ");

**System**.out.println(tool.circle\_length(5));

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

}

}

Листинг 7 – Реализация класса MathFunc.

## **Задание №5:** Разработайте интерфейс для работы со строками, который содержит а) функции подсчета символов в строке б) функция возвращает строку, которая образовывает строку, состоящую из символов исходной строки s, которые размещены на нечетных позициях: 1, 3, 5, ...в) функцию инвертирования строки

## **Задание №6:** Реализуйте интерфейс в классе ProcessStrings и протестируйте работу класса

**public** **interface** Processable {

int count\_symb();

**public** **String** new\_string();

**String** inverse\_string();

}

Листинг 8 – Интерфейс Processable.

**public** **class** ProcessStrings **implements** Processable{

**String** string;

ProcessStrings(**String** string)

{

**this**.string = string;

}

**public** int count\_symb(){

int c = 0;

**for** (int i=0; i<string.length(); i++){

**if** ((string.charAt(i)) != ' ') {

c++;

}

}

**return** c;

}

**public** **String** new\_string(){

**String** newString = "";

**for** (int i = 0; i<string.length();i++){

**if** (i%2 == 1){

newString += string.charAt(i) ;

}

}

**return** newString;

}

**public** **String** inverse\_string(){

**String** newString = "";

**for** (int i = 0; i < string.length();i++){

newString += string.charAt(string.length()- 1 - i); //-1 т.к. например "Hi" имеет длину 2

//length = 2, при первой итерации i=0, length = 2 => charAt(2), что выходит за диапазон => -1

}

**return** newString;

}

}

Листинг 9 – Реализация методов в классе Processable.

**public** **class** Test {

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

Processable string = **new** ProcessStrings("Hello World!");

**System**.out.println("Количество символов");

**System**.out.println(string.count\_symb());

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

**System**.out.println("Символы на нечётных местах");

**System**.out.println(string.new\_string());

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

**System**.out.println("Перевёрнутая строка");

**System**.out.println(string.inverse\_string());

}

}

Листинг 10 – Реализация теста.

## **Задание №7:** Создать статический метод printMagazines(Printable[] printable) в классе Magazine, который выводит на консоль названия только журналов.

**public** **interface** Printable {

Printable print();

}

Листинг 11 – Интерфейс Printable.

**public** **class** Magazine **implements** Printable{

**String** name;

**public** Magazine(**String** name) {

**this**.name = name;

}

**public** Printable print(){

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

**return** **null**;

}

**public** **static** void printMagazines(Printable[] printables){

**for** (Printable p: printables){

**if** (p **instanceof** Magazine)

p.print();}}}

Листинг 12 – Реализация класса Magazine.

**Задание №7:** Создать статический метод printBooks(Printable[] printable) в классе Book, который выводит на консоль названия только книг. Используем оператор instanceof.

**public** **class** Book **implements** Printable{

**String** name;

**public** Book(**String** name) {

**this**.name = name;

}

**public** **static** void printBooks(Printable[] printables) {

**for** (Printable p : printables) {

**if** (p **instanceof** Book)

p.print();

}

}

**public** Printable print () {

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

**return** **null**;

}

}

Листинг 13 – Реализация класса Book.

## **Вывод**

В результате практической работы получены навыки работы с интерфейсами и статическими методами. Изучено практическое применение интерфейсов.