НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
Факультет систем управления и робототехники

Программирование
Лабораторная работа № 3

Выполнил
студент

Ракин Илья Николаевич
Группа № R3138

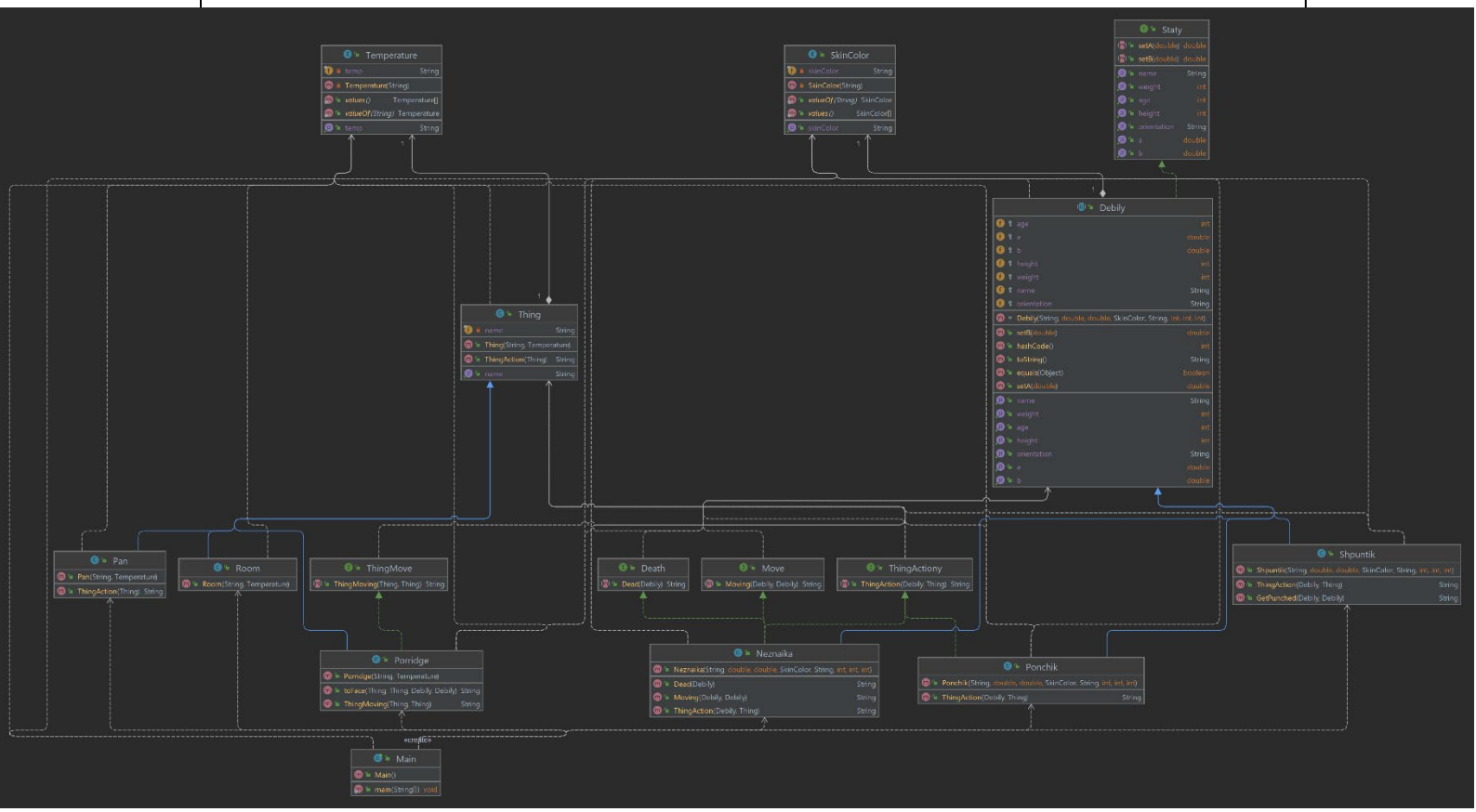
Преподаватель: Харитонова Анастасия Евгеньевна

г. Санкт-Петербург
2021

1. Доработать объектную модель приложения.
2. Перерисовать диаграмму классов в соответствии с внесёнными в модель изменениями.
3. Согласовать с преподавателем изменения, внесённые в модель.
4. Модифицировать программу в соответствии с внесёнными в модель изменениями.

Задание лабораторной работы

Диаграмма классов реализованной объектной модели



Исходный код программы:

Main-Class

```
public class Main {
    public static void main(String[] args) {
        Pan pan = new Pan("кастрюля с кашей", Temperature.HOT);
        Room room = new Room("комната", Temperature.COLD);
        Porridge porridge = new Porridge("Манная каша",Temperature.HOT);
        Neznaika neznaika = new
Neznaika("Незнайка",15,30,SkinColor.ALBO,"Натурал",48,94,190);
        Shpuntik shpuntik = new Shpuntik("Шпунтик", 138947,1840,SkinColor.BLACK,
"Гомосексуал ", 13, 45, 152);
        Ponchik ponchik = new Ponchik("Пончик", 9250, 9250, SkinColor.CHINESE,
"натурал", 75, 66, 180);
        System.out.println(neznaika.Moving(neznaika, shpuntik));
        if (Math.random() < 0.8) {
            System.out.println(shpuntik.ThingAction(shpuntik,room));
            System.out.println(pan.ThingAction(pan));
            System.out.println(porridge.ThingMoving(pan, porridge));
            System.out.println(neznaika.ThingAction(neznaika, pan));
            System.out.println(shpuntik.GetPunched(neznaika, shpuntik));
            System.out.println(porridge.toFace(pan,porridge,neznaika,ponchik));
            System.out.println(ponchik.ThingAction(ponchik,porridge));
        }
        else {
            System.out.println(neznaika.Dead(neznaika));
        }
        System.out.println("THE END");
```

Abstract class

```
import java.util.Objects;

public abstract class Debily implements Staty {
    protected double a;
    protected double b;
    protected String name;
    protected SkinColor skinColor;
    protected String orientation;
    protected int weight;
    protected int height;
    protected int age;

    Debily(String name, double a, double b, SkinColor skinColor, String orientation, int age, int weight, int height) {
        this.name = name;
        this.a = a;
        this.b = b;
        this.skinColor = skinColor;
        this.orientation = orientation;
        this.age = age;
        this.weight = weight;
        this.height = height;
    }
    @Override
    public double getA() { return this.a; }

    @Override
    public double getB() { return this.b; }

    @Override
    public double setA(double a){
        this.a = a;
```

```
        return a;
    }

    @Override
    public double setB(double b){
        this.b = b;
        return b;
    }
    @Override
    public String getName() { return this.name; }
    @Override
    public String getOrientation() { return this.orientation; }
    @Override
    public int getAge() { return this.age; }

    @Override
    public int getWeight() { return this.weight; }

    @Override
    public int getHeight() { return this.height; }
    @Override
    public String toString(){
        return "Debil{" + "name= '" + name + '\n' + ", skin color= '" + skinColor + '\n' + ", orientation= '"
                + orientation + '\n' + ", age= " + age + '\n' + ", weight=" + weight + '\n' + ", height=" + height + '}';
    }
    @Override
    public int hashCode() { return Objects.hash(name, a, b); }
    @Override
    public boolean equals(Object m) {
        if (this == m) return true;
        if (m == null || getClass() != m.getClass()) return false;
```

```
        Debily Debily = (Debily) m;
        return Objects.equals(name, Debily.name)
                && Objects.equals(a, Debily.a)
                && Objects.equals(b, Debily.b);
    }
```

Interfaces

```
public interface Death {
    public String Dead(Debily m);
}
```

```
public interface Move {
    String Moving(Debily k, Debily x);
}
```

```
public interface Staty{
    double getA();
    double getB();
    double setA(double a);
    double setB(double b);
    String getName();
    String getOrientation();
    int getAge();
    int getWeight();
    int getHeight();
}
```

```
public interface ThingActiony {
    String ThingAction(Debily k, Thing x);
}
```

```
public interface ThingMove {
    String ThingMoving(Thing x, Thing s);
}
```

Classes

```
public class Neznaika extends Debily implements ThingActiony, Move, Death {
    public Neznaika(String name, double a, double b, SkinColor skinColor, String orientation, int age, int weight, int height) {
        super(name, a, b, skinColor, orientation, age, weight, height);
    }

    @Override
    public String Moving(Debily k, Debily x) {
        return k.getOrientation() + " " + k.skinColor.getSkinColor() + " " + k.getName() +
                " взмахнул кулаком и нанес удар " + x.getName() + "у";
    }

    @Override
    public String ThingAction(Debily k, Thing x) {
        return k.getName() + " " + "от своего же удара отлетел и ударился головой о " + x.getName();
    }

    @Override
    public String Dead(Debily m) {
        return "У " + m.getName() + " отрывается тромб и он умирает из-за слишком резкого замаха руки";
    }
}
```

```java
public class Pan extends Thing{
    public Pan(String name, Temperature temp) { super(name, temp); }
    @Override
    public String ThingAction(Thing p) { return "На плите стояла " + p.getName(); }
}
```

```java
public class Ponchik extends Debily implements ThingActiony {
    public Ponchik(String name, double a, double b, SkinColor skinColor, String orientation, int age, int weight, int height) {
        super(name, a, b, skinColor, orientation, age, weight, height);
    }

    @Override
    public String ThingAction(Debily k, Thing s) {
        return s.temp.getTemp() + " " +  s.getName() + " начинает прожигать лицо бедного " + k.getName() + "а";
    }
}
```

```java
public class Porridge extends Thing implements ThingMove {
    public Porridge(String name, Temperature temp) { super(name,temp); }
    @Override
    public String ThingMoving(Thing x, Thing s){
        return "В " + x.getName() + " находится " + s.temp.getTemp() + " " + s.getName();
    }
    public String toFace(Thing k, Thing s, Debily u, Debily a){
        return "От столкновения с " + u.getName() + ", " + k.getName() + " падает и " + s.getName() +
                " выплескивается на лицо ничего не подозревающего " + a.skinColor.getSkinColor() + " "
                + a.getOrientation() + " " +  a.getAge() + "-ти лет " +  a.getName() + "а";
    }
}
```

```java
public class Room extends Thing{
    public Room(String name, Temperature temp){
        super(name, temp);
    }
}
```

```java
public class Shpuntik extends Debily {
    public Shpuntik(String name, double a, double b, SkinColor skinColor, String orientation, int age, int weight, int height) {
        super(name, a,b, skinColor, orientation, age, weight, height);
    }
    public String ThingAction(Debily k, Thing x){
        return "Получив по щам, " + k.skinColor.getSkinColor() + k.getOrientation() + k.getName() +
                " завертелся волчком и полетел через всю " + x.getName();
    }
    public String GetPunched(Debily x, Debily n){
        return "Оклемавшись от удара нанесенного " + x.getName() + ", " + n.getName() +
                " прокричал: Что ты делаешь? Мне всего " + n.getAge() + " лет";
    }
}
```

```java
public class Thing {
    private final String name;
    protected Temperature temp;

    public Thing(String name, Temperature temp){
        this.name = name;
        this.temp = temp;
    }
    public String getName() { return name; }
//    public Temperature getTemp(){
//        return temp;
//    }
    public String ThingAction(Thing o) { return ""; }
}
```

Enum’s

```java
public enum SkinColor {
    BLACK("чернокожий "),
    WHITE("белокожий "),
    ALBO("альбинос"),
    CHINESE("желтый");

    private final String skinColor;
    SkinColor(String skinColor) { this.skinColor=skinColor; }
    public String getSkinColor() { return skinColor; }
}
```

```java
public enum Temperature {
    HOT("горячая"), COLD("холодная");

    private final String temp;
    Temperature(String temp) { this.temp=temp; }
    public String getTemp() { return temp; }
}
```

Результат работы программы

80%

```
C:\Users\rakin\.jdks\corretto-1.8.0_312\bin\java.exe ...
Натурал альбинос Незнайка взмахнул кулаком и нанес удар Шпунтику
Получив по щам, чернокожий Гомосексуал Шпунтик завертелся волчком и полетел через всю комната
На плите стояла кастрюля с кашей
В кастрюля с кашей находится горячая Манная каша
Незнайка от своего же удара отлетел и ударился головой о кастрюля с кашей
Оклемавшись от удара нанесенного Незнайка, Шпунтик прокричал: Что ты делаешь? Мне всего 13 лет
От столкновения с Незнайка, кастрюля с кашей падает и Манная каша выплескивается на лицо ничего не подозревающего желтый натурал 75-ти лет Пончика
горячая Манная каша начинает прожигать лицо бедного Пончика
THE END

Process finished with exit code 0
```

20%

```
C:\Users\rakin\.jdks\corretto-1.8.0_312\bin\java.exe ...
Натурал альбинос Незнайка взмахнул кулаком и нанес удар Шпунтику
У Незнайка отрывается тромб и он умирает из-за слишком резкого замаха руки
THE END
```

Выводы в ходе выполнения лабораторной работы:

Во время выполнения данной лабораторной работы я научился применять принципы SOLID на практике, подробнее разобрал интерфейсы, абстрактные классы и перечисления.

Спасибо за внимание!