МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА № 3

по дисциплине

‘ПРОГРАММИРОВАНИЕ’

Вариант № 3118008

Выполнил:
Студент группы P3118
Павлов Александр
Сергеевич
Преподаватель:
Письмак Алексей
Евгеньевич

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург, 2021

Оглавление

Задание:

Введите вариант: 3118008

Описание предметной области, по которой должна быть построена объектная модель:

> И привидение разразилось долгим глухим смехом. Но фрекен Бок было не до смеха. Она кинулась к двери и стала расшвыривать мебель. В мгновение ока разобрав баррикаду, она с громким криком выбежала в переднюю. Привидение полетело следом, а Малыш побежал за ним. Последним мчался Бимбо и заливисто лаял. Он узнал привидение по запаху и думал, что началась веселая игра. Привидение, впрочем, тоже так думало.

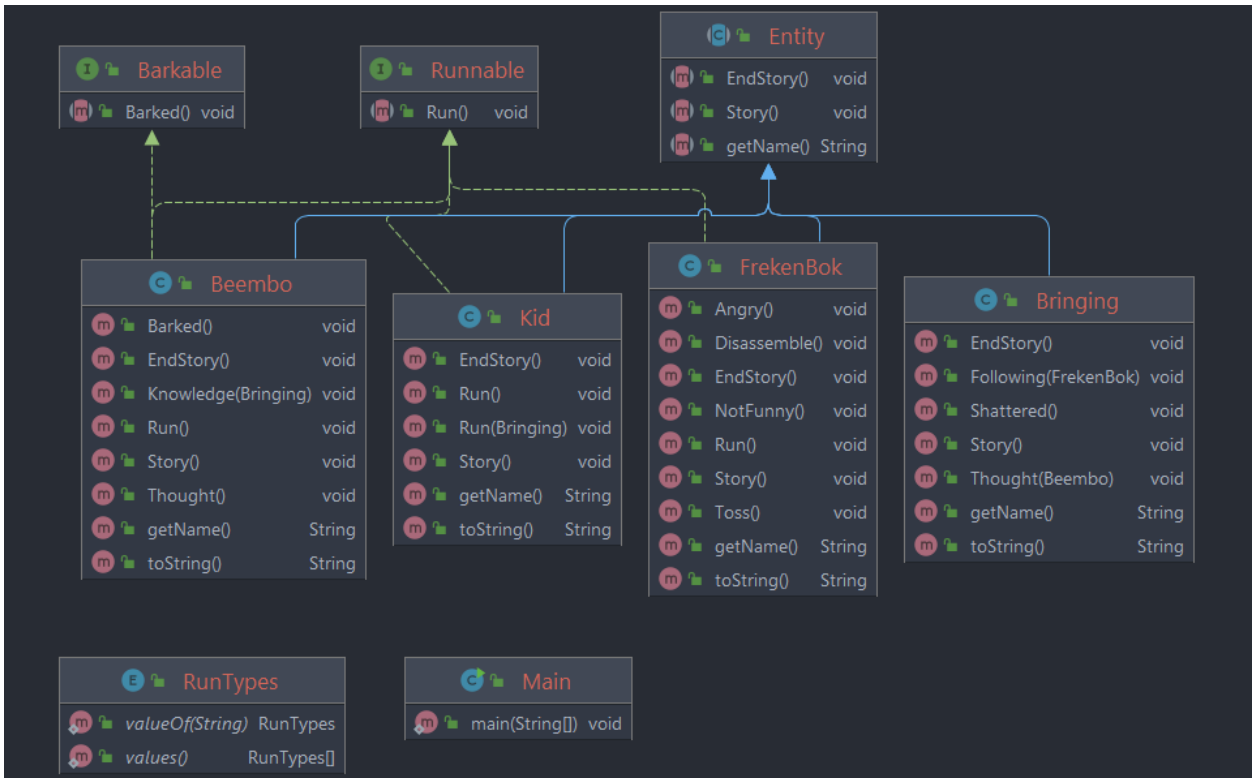
Программа должна удовлетворять следующим требованиям:

1. Доработанная модель должна соответствовать принципам SOLID.
2. Программа должна содержать как минимум два интерфейса и один абстрактный класс (номенклатура должна быть согласована с преподавателем).
3. В разработанных классах должны быть переопределены методы `equals()`, `toString()` и `hashCode()`.
4. Программа должна содержать как минимум один перечисляемый тип (enum).

Порядок выполнения работы:

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.

Диаграмма классов:



Исходный код программы:

Main.java

```
package com.company;

public class Main {

    public static void main(String[] args) {
        Bringing privedenie = new Bringing("4 лаба");
        FrekenBok freken = new FrekenBok("Злюка");
        freken.Angry();
        freken.Toss();
        freken.Disassemble();
        freken.Run();
        freken.EndStory();
        privedenie.Following(freken);
        Kid malish = new Kid("эмокид");
        malish.Run(privedenie);
        malish.EndStory();
        Beembo bimbo = new Beembo("коронавирус");
        bimbo.Run();
        bimbo.Barked();
        bimbo.Knowledge(privedenie);
        bimbo.Thought();
        bimbo.EndStory();
        privedenie.Thought(bimbo);
        privedenie.EndStory();
    }
}
```

Bringing.java

```
package com.company;

public class Bringing extends Entity{
    private final String name;
    private final RunTypes type = RunTypes.FLEW;
    public Bringing(){
        super();
        name = "Приведение";
        Story();
        Shattered();
    }
    public Bringing(String name){
        super(name);
        this.name = name;
        Story();
        Shattered();
    }
    public void Story(){
        System.out.println("Приведение '"+name+"' ворвалось в рассказ");
    }
    public void Shattered(){
        System.out.println("Приведение '"+name+"' разразилось долгим глухим
смехом");
    }
    public void Following(FrekenBok frek){
        if (type == RunTypes.FLEW) {
            System.out.println("Приведение '" + name + "' полетело следом за
фрекен Бок '"+frek.getName()+"'");
```

```
        }
    }
    public void Thought(Beembo bim){
        System.out.println("Приведение '" + name + "' думало также, как и
'"+bim.getName()+"'");
    }
    public void EndStory(){
        System.out.println("Приведение '" + name + "' вылетело из истории");
    }
    public String toString(){
        return "Bringing name - "+ name;
    }
    public String getName(){
        return name;
    }
}
```

FrekenBok.java

```
package com.company;

public class FrekenBok extends Entity implements Runnable{
    private final String name;
    private RunTypes type = RunTypes.RUSHED;
    public FrekenBok(){
        super();
        name = "фрекен Бок";
        Story();
        NotFunny();
    }
    public FrekenBok(String name){
        super(name);
        this.name = name;
        Story();
        NotFunny();
    }
    public void Story(){
        System.out.println("фрекен Бок '"+name+"' ворвалась в рассказ");
    }
    public void NotFunny(){
        System.out.println("фрекен Бок '"+name+"' было не до смеха");
    }
    public void Angry(){
        if (type == RunTypes.RUN){
            System.out.println("фрекен Бок '"+name+"' побежала к двери");
        }
        else if (type == RunTypes.RUSHED){
            System.out.println("фрекен Бок '"+name+"' кинулась к двери");
        }
    }
    public void Toss(){
        System.out.println("фрекен Бок '"+name+"' стала расшвыривать
мебель");
    }
    public void Run(){
        type = RunTypes.RUN_OUT;
        System.out.println("фрекен Бок '"+name+"' с громким криком выбежала в
переднюю");
    }
    public void Disassemble(){
        System.out.println("фрекен Бок '"+name+"' разобрала баррикаду в
мнгновение ока");
    }
    public String toString(){
```

```
        return "FrekenBok name - "+ name;
    }
    public void EndStory(){
        System.out.println("фрекен Бок '"+name+"' вырвалась из рассказа");
    }
    public String getName(){
        return name;
    }
}
```

Entity.java

```
package com.company;

public abstract class Entity{
    private final String name;
    public Entity(String name){
        this.name = name;
    }
    public Entity() {
        name = null;
    }
    public abstract String getName();
    public abstract void Story();
    public abstract void EndStory();
}
```

RunTypes.java

```
package com.company;

public enum RunTypes {
    RUSHED,
    RUN_OUT,
    FLEW,
    RUN,
    RACED
}
```

Runnable.java

```
package com.company;

public interface Runnable {
    public void Run();
}
```

Barkable.java

```
package com.company;

public interface Barkable {
    public void Barked();
}
```

Beembo.java

```
package com.company;

public class Beembo extends Entity implements Runnable,Barkable{
    private final String name;
    private final RunTypes type = RunTypes.RACED;
    public Beembo(){
        super();
        name = "Бимбо";
        Story();
    }
```

```
    public Beembo(String name){
        super(name);
        this.name = name;
    }
    public void Story() {
        System.out.println("Бимбо '"+name+"' вбежал в историю с четырех
лап");
    }
    public void Run(){
        if (type == RunTypes.RACED){
            System.out.println("Бимбо по имени '"+name+"' мчался");
        }
    }
    public void Barked(){
        System.out.println("Бимбо '"+name+"' заливисто лаял");
    }
    public void Knowledge(Bringing prizrak){
        System.out.println("Бимбо '"+name+"' узнал приведение
'"+prizrak.getName()+"' по запаху");
    }
    public void Thought(){
        System.out.println("Бимбо '"+name+"' думает, что началась весёлая
игра");
    }
    public void EndStory(){
        System.out.println("Бимбо '"+name+"' вышел из рассказа");
    }
    public String getName() {
        return name;
    }
    public String toString(){
        return "Beembo name - "+ name;
    }
}
```

Kid.java

```
package com.company;

public class Kid extends Entity implements Runnable{
    private final String name;
    private final RunTypes type = RunTypes.RUN;
    public Kid(){
        super();
        name = "ребеночек";
        Story();
    }
    public Kid(String name){
        super(name);
        this.name = name;
        Story();
    }
    public void Story(){
        System.out.println("Малыш '"+name+"' вполз в рассказ");
    }
    public void Run(Bringing prizrak){
        if (type == RunTypes.RUN){
            System.out.println("Малыш '"+name+"' побежал за приведением
'"+prizrak.getName()+"'");
        }
    }
    public void EndStory(){
        System.out.println("Малыш '"+name+"' выполз из рассказа");
    }
```

```
    public String getName(){
        return name;
    }
    public String toString(){
        return "Kid name - "+ name;
    }
    public void Run() {}
}
```

Результаты работы программы:

Приведение '4 лаба' ворвалось в рассказ

Приведение '4 лаба' разразилось долгим глухим смехом

фрекен Бок 'Злюка' ворвалась в рассказ

фрекен Бок 'Злюка' было не до смеха

фрекен Бок 'Злюка' кинулась к двери

фрекен Бок 'Злюка' стала расшвыривать мебель

фрекен Бок 'Злюка' разобрала баррикаду в мнгновение ока

фрекен Бок 'Злюка' с громким криком выбежала в переднюю

фрекен Бок 'Злюка' вырвалась из рассказа

Приведение '4 лаба' полетело следом за фрекен Бок 'Злюка'

Малыш 'эмокид' вполз в рассказ

Малыш 'эмокид' побежал за приведением '4 лаба'

Малыш 'эмокид' выполз из рассказа

Бимбо по имени 'коронавирус' мчался

Бимбо 'коронавирус' заливисто лаял

Бимбо 'коронавирус' узнал приведение '4 лаба' по запаху

Бимбо 'коронавирус' думает, что началась весёлая игра

Бимбо 'коронавирус' вышел из рассказа

Приведение '4 лаба' думало также, как и 'коронавирус'

Приведение '4 лаба' вылетело из истории

Вывод:

В ходе выполнения данной лабораторной работы я познакомился с принципами SOLID, улучшил навыки работы с ООП, познакомился с понятием интерфейса и абстрактного класса.