Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа №4

Вариант 32856.6

Группа: P3132

Выполнил: Овчаренко Александр Андреевич

г. Санкт-Петербург

2021 г.

Оглавление

Задание

Доработать программу из лабораторной работы #3, обновив реализацию объектной модели в соответствии с новой версией описания предметной области.

Программа должна удовлетворять следующим требованиям:

- В программе должны быть реализованы 2 собственных класса исключений (checked и unchecked), а также обработка исключений этих классов.
- В программу необходимо добавить использование локальных, анонимных и вложенных классов (static и non-static).

Порядок выполнения работы:

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

Вариант 32856.6

Выполнение задания

Korotishi

```
package lab_4.kids;

public class Korotishi {
    private final String name;
    private final Gender gender;
    public static Korotishi malishi = new Korotishi("Malishi", Gender.Malishi);
    public static Korotishi malishki = new Korotishi("Malishki",
Gender.Malishki);

    private Korotishi(String name, Gender gender) {
        this.name = name;
        this.gender = gender;
    }

    public String getGender() {
        return gender.getGender();
    }

    @Override
    public int hashCode() {
        return name.hashCode() + gender.hashCode();
    }

    @Override
    public String toString() {
        return name;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true; // сравнение ссылок, та же ли ссылка
        if (obj == null || getClass() != obj.getClass()) return false;
        Korotishi that = (Korotishi) obj; // сужеющее приведение, чтобы взять
поля
        return name.equals(that.name) && gender == that.gender;
    }

    public static Korotishi createMalish(String name) {
        if (name.equals("Malishi")) throw new IllegalArgumentException(name + " -
wrong name");
        else {return new Korotishi(name, Gender.Malishi);}
    }

    public static Korotishi createMalishka(String name) {
        if (name.equals("Malishki")) throw new IllegalArgumentException(name + "
- wrong name");
```

```
        else {return new Korotishi(name, Gender.Malishki);}
    }

    public static class Reaction{
        public static void goodReaction(String thing){
            System.out.println("They like " + thing);
        }

        public static void badReaction(String thing){
            System.out.println("They didn't like " + thing);
        }
    }
}
```

Gender

```
package lab_4.kids;

public enum Gender {
    Malishi("Malish"),
    Malishki("Malishki");

    private String gender;

    Gender(String gender) {
        this.gender = gender;
    }

    public String getGender() {
        return gender;
    }
}
```

ThisObject

```
package lab_4.objects;

public abstract class ThisObject {
    private final String object;

    public ThisObject (String object) {
        this.object = object;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        ThisObject that = (ThisObject) obj;
        return object == that.object;
```

```
    }

    @Override
    public String toString() {
        return object;
    }
}
```

Story

```
package lab_4.objects;

public class Story extends ThisObject{
    private String topic;

    public Story(String topic){
        super("story");
        this.topic = topic;
    }

    public String getTopic(){
        return topic;
    }
}
```

WindowCloseException

```
package lab_4.exception;

public class WindowCloseException extends Exception {
    public WindowCloseException(String message){
        super(message);
    }
}
```

JumpInWindow

```
package lab_4.actions;
import lab_4.exception.*;
import lab_4.kids.Korotishi;

public class JumpInWindow {
    public static void jumpInWindow(Korotishi who) throws WindowCloseException{
        try{
        if(Math.random() < 0.5) throw new WindowCloseException("the window is
closed. ");
        else{System.out.println(who + " jumps out in window.");}
        }
        catch(WindowCloseException e){
```

```
            System.out.print("Exception: " + e.getMessage());
            System.out.println("Befor jump open window.");
        }
    }
}
```

CarryAction

```
package lab_4.actions;
import lab_4.kids.*;

public class CarryAction {
    private final String name = "carry";
    private final Korotishi[] who;
    private final Korotishi whom;
    private final String place;

    public CarryAction(Korotishi[] who, Korotishi whom, String place) {
        this.who = who;
        this.whom = whom;
        this.place = place;
    }

    public Korotishi[] getWho(){
        return who;
    }

    public Korotishi getWhom(){
        return whom;
    }

    public void print() {
        System.out.print(who[0]);
        for(int i = 1; i<who.length; i++) System.out.print(", " + who[i]);
        System.out.println(" " + name + " " + whom + " " + place + ".");
    }
}
```

CheckInterface

```
package lab_4.actions;

import lab_4.objects.ThisObject;

public interface CheckInterface {
    public boolean check(ThisObject object);
}
```

TellStoryAction

```
package lab_4.actions;

import lab_4.objects.*;

public class TellStoryAction {

    public static void tellStories(Story[] stories){
        CheckInterface check = new CheckInterface() {
            public boolean check(ThisObject object){
                Story story = (Story) object;
                if (story.getTopic() == "journey") return true;
                else{return false;}
              }
        };
        for(Story st: stories){

            System.out.println(who + " tells story about " + st.getTopic() + " to
" + whom + ".");
            if (check.check(st)) {Korotishi.Reaction.goodReaction("story about "
+ st.getTopic());}
            else{Korotishi.Reaction.badReaction("story about " + st.getTopic());}
        }
    }
}
```

PutAction

```
package lab_4.actions;

import lab_4.kids.Korotishi;

public class PutAction {
    private Korotishi whom;
    private Korotishi[] who;
    private String place;

    public PutAction(Korotishi[] who, Korotishi whom, String place){
        this.who = who;
        this.whom = whom;
        this.place = place;
    }

    public void print(){
        System.out.print(who[0]);
        for(int i = 1; i<who.length; i++) System.out.print(", " + who[i]);
        System.out.println(" put " + whom + " " + place + ".");
    }
}
```

ReadBookAction

```
package lab_4.actions;

import java.util.ArrayList;
import java.util.List;

import lab_4.kids.Korotishi;

public class ReadBookAction {
    private final Korotishi who;

    public ReadBookAction(Korotishi who){
        this.who = who;
    }

    public class Book{
        private final String nameOfBook;
        private final String themeOfBook;
        private static List<String> arrayNameOfBooks = new ArrayList<String>();
        private static int countOfAllBook;

        public Book(String name, String theme){
            nameOfBook = name;
            themeOfBook = theme;
            countOfAllBook += 1;
            arrayNameOfBooks.add(name);
        }

        public String getNameOfBook(){
            return nameOfBook;
        }

        public String getThemeOfBook(){
            return themeOfBook;
        }

        public void printBook(){
            System.out.println("Name of Book: " + nameOfBook + "\n" + "Theme of
Book: " + themeOfBook);
        }

        public static int getcountOfAllBook(){
            return countOfAllBook;
        }

        public static void printNameOfAllBooks(){
            for(String book: arrayNameOfBooks){System.out.println("- " + book);}
```

```
        }
    }

    public void printReadAction(){
        System.out.println(who + " read:");
        Book.printNameOfAllBooks();
    }
}
```

SeeAction

```
package lab_4.actions;

import java.util.List;

import lab_4.kids.*;

public class SeeAction {
    public static void printSeeAction(List<Object> objects, Korotishi who){
        class ObjectOfInspection{
            Object object;

            ObjectOfInspection(Object object){
                this.object = object;
            }

            public Object getObjectOfInspection(){
                return object;
            }
        }

        for(int i=0; i<objects.size(); i++){
            ObjectOfInspection object = new ObjectOfInspection(objects.get(i));
            System.out.println(who + " see " + object.getObjectOfInspection() +
".");
        }
    }

    public static void openEyes(Korotishi who){
        System.out.println(who + " opens eyes.");
    }
}
```

NullIdeaException

```
package lab_4.exception;

public class NullIdeaException extends RuntimeException {
    public NullIdeaException(String message){
        super(message);
```

```
    }
}
```

ThinkAction

```
package lab_4.actions;

import lab_4.exception.NullIdeaException;
import lab_4.kids.Korotishi;

public class ThinkAction {
    private final String name = "think";
    private final Korotishi who;
    private final int num_day;
    private final int num_night;
    private final String result;

    public ThinkAction(Korotishi who, int day, int night, String result) {
        this.who = who;
        num_day = day;
        num_night = night;
        this.result = result;
    }

    public Korotishi getWho(){
        return who;
    }

    public void printDays(){
        System.out.println("num_days = " + num_day + ", num_night = " +
num_night);
    }

    public String getResult(){
        return result;
    }

    public void print() {
        if (result == null) throw new NullIdeaException(" doesn't has any
ideas.");
        else{
            System.out.println(who + " " + name + ". He has been thinking for " +
num_day +
            " days and " + num_night + " night. Result - " + result + ".");
            }
    }
}
```

Main

```
package lab_4;

import java.util.ArrayList;
import java.util.List;

import lab_4.actions.*;
import lab_4.exception.WindowCloseException;
import lab_4.kids.Korotishi;
import lab_4.objects.Story;


public class Main{
    public static void main(String[] args) throws WindowCloseException{
        //creat heros
        Korotishi znayka = Korotishi.createMalish("Znayka");
        Korotishi neznayka = Korotishi.createMalish("Neznayka");
        Korotishi avoska = Korotishi.createMalish("Avoska");
        Korotishi vintik = Korotishi.createMalish("Vintik");
        Korotishi pilylkin = Korotishi.createMalish("Pilylkin");
        Korotishi[] korotArray = new Korotishi[]{znayka, avoska, vintik,
pilylkin};
        //creat Actions
        CarryAction carryNeznayka = new CarryAction(korotArray, neznayka,
"home");
        PutAction putNeznayka = new PutAction(korotArray, neznayka, "on the
bed");
        List<Object> objectsOfSeeAction = new ArrayList<Object>();
        objectsOfSeeAction.add("on the left");
        objectsOfSeeAction.add("on the right");
        ReadBookAction znaykaRead = new ReadBookAction(znayka);
        ReadBookAction.Book bookFirst = znaykaRead.new Book("first book",
"journey");
        ReadBookAction.Book bookSecond= znaykaRead.new Book("second book",
"education");
        ReadBookAction.Book bookThird= znaykaRead.new Book("third book",
"roman");
        ThinkAction znaykaThinkFirst = new ThinkAction(znayka, 3, 3, "make a big
ball");
        ThinkAction znaykaThinkSecond = new ThinkAction(znayka, 3, 3, null);
        Story storyFirst = new Story(bookFirst.getThemeOfBook());
        Story storySecond = new Story(bookSecond.getThemeOfBook());
        Story storyThird = new Story(bookThird.getThemeOfBook());
        Story[] arrayOfStory = new Story[]{storyFirst, storySecond, storyThird};
        //print story
        carryNeznayka.print();
```

```
        putNeznayka.print();
        SeeAction.openEyes(neznayka);
        SeeAction.printSeeAction(objectsOfSeeAction, neznayka);
        JumpInWindow.jumpInWindow(neznayka);
        znaykaRead.printReadAction();
        TellStoryAction.tellStories(arrayOfStory, znayka, Korotishi.malishi);
        znaykaThinkFirst.print();
    }
}
```

Вывод программы

Znayka, Avoska, Vintik, Pilylkin carry Neznayka home.

Znayka, Avoska, Vintik, Pilylkin put Neznayka on the bed.

Neznayka opens eyes.

Neznayka see on the left.

Neznayka see on the right.

Exception: the window is closed. Before jump open window.

Znayka read:

- first book

- second book

- third book

Znayka tells story about journey to Malishi.

They like story about journey.

Znayka tells story about education to Malishi.

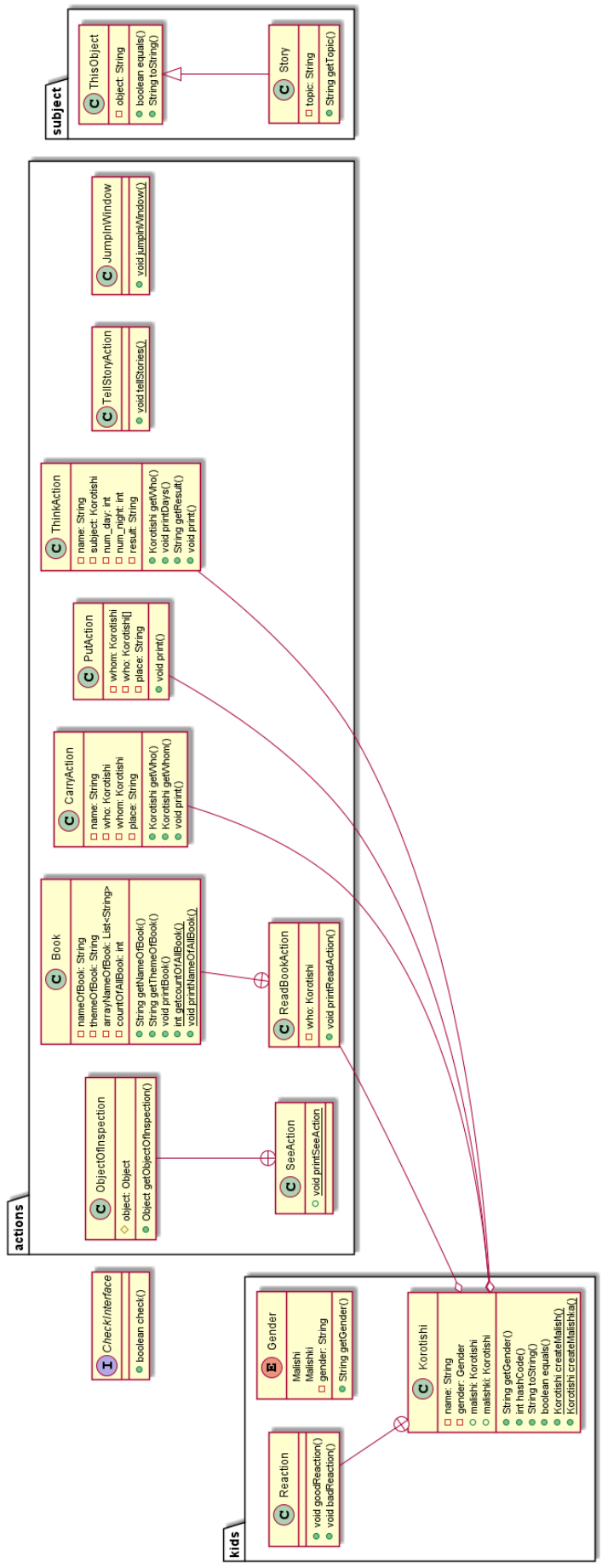
They don't like story about education.

Znayka tells story about roman to Malishi.

They don't like story about roman.

Znayka think. He has been thinking for 3 days and 3 night. Result - make a big ball.

UML


subject
ThisObject
object: String
boolean equals()
String toString()
Story
topic: String
String getTopic()
actions
JumpInWindow
void jumpInWindow()
TellStoryAction
void tellStories()
ThinkAction
name: String
subject: Korotishi
num_day: int
num_night: int
result: String
Korotishi getWho()
void printDays()
String getResult()
void print()
PutAction
whom: Korotishi
who: Korotishi[]
place: String
void print()
CarryAction
name: String
who: Korotishi
whom: Korotishi
place: String
Korotishi getWho()
Korotishi getWhom()
void print()
Book
nameOfBook: String
themeOfBook: String
arrayNameOfBook: List<String>
countOfAllBook: int
String getNameOfBook()
String getThemeOfBook()
void printBook()
int getcountOfAllBook()
void printNameOfAllBook()
ReadBookAction
who: Korotishi
void printReadAction()
ObjectOfInspection
object: Object
Object getObjectOfInspection()
SeeAction
void printSeeAction
CheckInterface
boolean check()
kids
Gender
Malishi
Malishki
gender: String
String getGender()
Korotishi
name: String
gender: Gender
malishi: Korotishi
malishki: Korotishi
String getGender()
int hashCode()
String toString()
boolean equals()
Korotishi createMalishi()
Korotishi createMalishka()
Reaction
void goodReaction()
void badReaction()

Итоги

В результате выполнения лабораторной работы были изучены вложенный, внутренний, локальный и анонимный классы. Была доработанная программа, в которую были включены эти классы.