Министерство образования и науки РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

**факультет программной инженерии и компьютерной техники**

**ЛАБОРАТОРНАЯ РАБОТА № 3**

по дисциплине

‘ПРОГРАММИРОВАНИЕ’

Вариант №0

*Выполнил:*

Студент группы P3113

Свиридов Дмитрий Витальевич

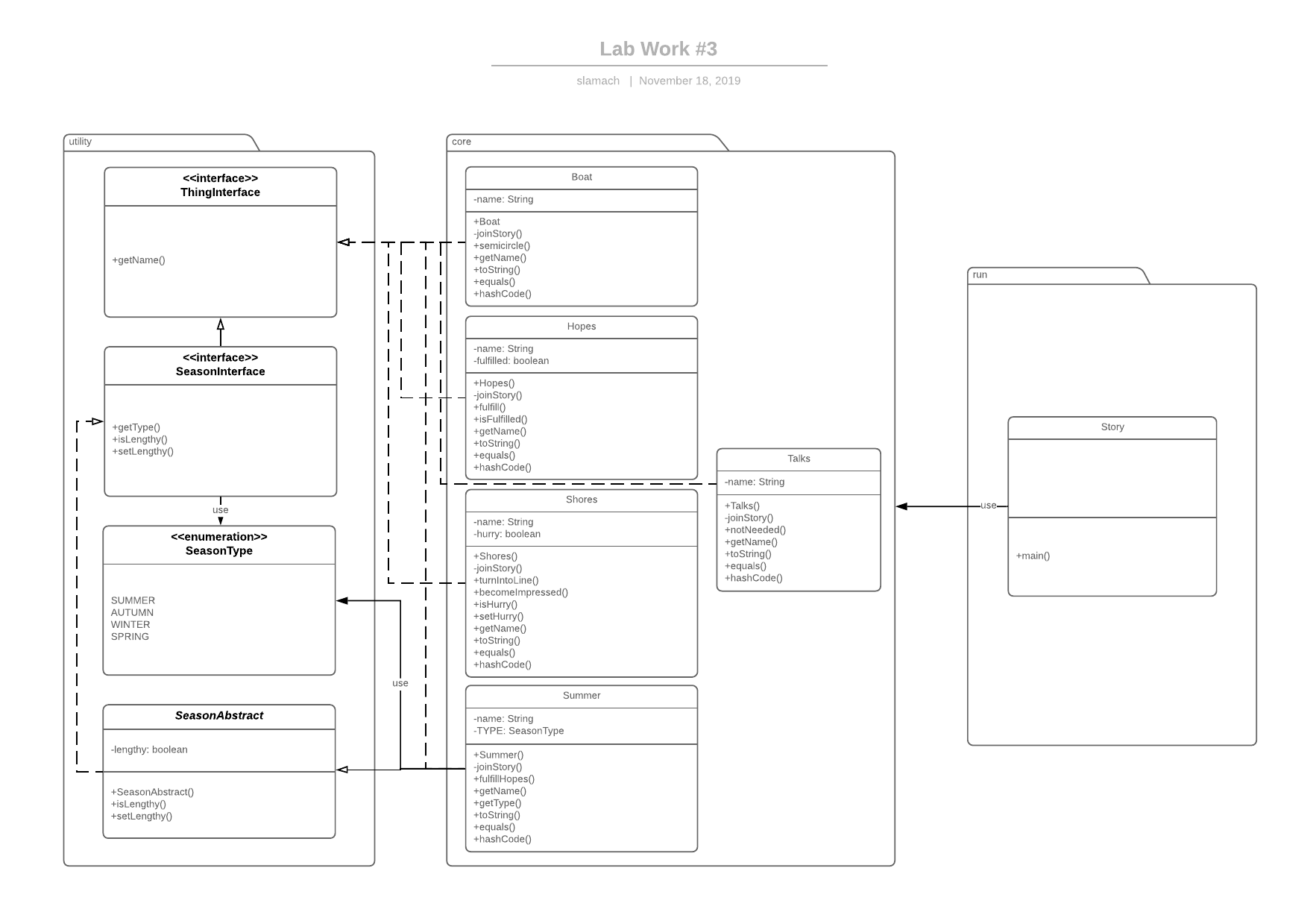
*Преподаватель:*

Письмак Алексей Евгеньевич

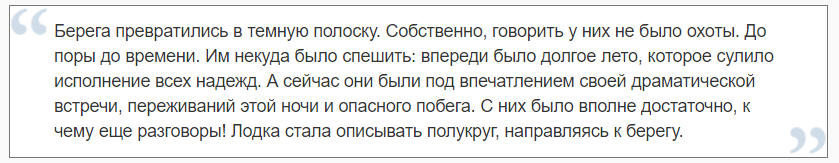


Санкт-Петербург, 2019

Диаграмма классов:



Описание предметной области:



Исходный код:

**Story.java**

package run;

import core.\*;

public class Story {

    public static void main(String[] args) {

        Shores aShores = new Shores();

        Summer aSummer = new Summer();

        Hopes aHopes = new Hopes();

        aSummer.fulfillHopes(aHopes);

        aShores.becomeImpressed();

        Talks aTalks = new Talks();

        aTalks.notNeeded();

        Boat aBoat = new Boat();

        aBoat.semicircle();

    }

}

**SeasonAbstract.java**

package utility;

public abstract class SeasonAbstract implements SeasonInterface {

    private boolean lengthy;

    public SeasonAbstract(boolean lengthy) {

        this.lengthy = lengthy;

    }

    @Override

    public boolean isLengthy() {

        return lengthy;

    }

    @Override

    public void setLengthy(boolean lengthy) {

        this.lengthy = lengthy;

    }

}

**SeasonInterface.java**

package utility;

public interface SeasonInterface extends ThingInterface {

    SeasonType getType();

    boolean isLengthy();

    void setLengthy(boolean lengthy);

}

**SeasonType.java**

package utility;

public enum SeasonType {

    SUMMER,

    AUTUMN,

    WINTER,

    SPRING

}

**ThingInterface.java**

package utility;

public interface ThingInterface {

    String getName();

}

**Boat.java**

package core;

import utility.ThingInterface;

public class Boat implements ThingInterface {

    private String name;

    public Boat() {

        name = "Лодка";

        joinStory();

    }

    public Boat(String name) {

        this.name = name;

        joinStory();

    }

    private void joinStory() {

        System.out.println("Лодка '" + name + "' присоединилась к истории.");

    }

    public void semicircle() {

        System.out.println("Лодка '" + name + "' стала описывать полукруг, направляясь к берегу.");

    }

    @Override

    public String getName() {

        return name;

    }

    @Override

    public String toString() {

        return "Лодка '" + name + "'";

    }

    @Override

    public boolean equals(Object obj) {

        if (this == obj) return true;

        if (obj instanceof Boat) {

            return name.equals(((Boat) obj).getName());

        }

        return false;

    }

    @Override

    public int hashCode() {

        return name.hashCode();

    }

}

**Hopes.java**

package core;

import utility.ThingInterface;

public class Hopes implements ThingInterface {

    private String name;

    private boolean fulfilled;

    public Hopes() {

        name = "Надежды";

        fulfilled = false;

        joinStory();

    }

    public Hopes(String name) {

        this.name = name;

        fulfilled = false;

        joinStory();

    }

    private void joinStory() {

        System.out.println("Надежды '" + name + "' присоединились к истории.");

    }

    public void fulfill() {

        if (!fulfilled) {

            fulfilled = true;

            System.out.println("Надежды '" + name + "' исполнены!");

        } else {

            System.out.println("Надежды '" + name + "' уже и так исполнены!");

        }

    }

    public boolean isFulfilled() {

        return fulfilled;

    }

    @Override

    public String getName() {

        return name;

    }

    @Override

    public String toString() {

        String ending;

        if (fulfilled) {

            ending = "исполнены";

        } else {

            ending = "не исполнены";

        }

        return "Надежды '" + name + "', " + ending;

    }

@Override

    public boolean equals(Object obj) {

        if (this == obj) return true;

        if (obj instanceof Hopes) {

            return name.equals(((Hopes) obj).getName()) && fulfilled == ((Hopes) obj).isFulfilled();

        }

        return false;

    }

    @Override

    public int hashCode() {

        if (fulfilled) return name.hashCode() + name.length();

        return name.hashCode();

    }

}

**Shores.java**

package core;

import utility.ThingInterface;

public class Shores implements ThingInterface {

    private String name;

    private boolean hurry;

    public Shores() {

        name = "Берега";

        hurry = false;

        joinStory();

    }

    public Shores(String name) {

        this.name = name;

        hurry = false;

        joinStory();

    }

    public Shores(String name, boolean hurry) {

        this.name = name;

        this.hurry = hurry;

        joinStory();

    }

    private void joinStory() {

        System.out.print("Берега '" + name + "' присоединились к истории ");

        if (hurry) {

            System.out.println("(они куда-то спешат).");

        } else {

            System.out.println("(им некуда спешить).");

        }

    }

    public void turnIntoLine() {

        System.out.println("Берега '" + name + "' превратились в темную полоску.");

    }

    public void becomeImpressed() {

        System.out.println("Берега '" + name + "' под впечатлением...");

    }

    public boolean isHurry() {

        return hurry;

    }

    public void setHurry(boolean hurry) {

        if (hurry != this.hurry) {

            this.hurry = hurry;

            if (hurry) {

                System.out.println("Берега '" + name + "' теперь куда-то спешат.");

            } else {

                System.out.println("Берегам '" + name + "' теперь некуда спешить.");

            }

        } else {

            if (hurry) {

                System.out.println("Берега '" + name + "' уже и так куда-то спешат.");

            } else {

                System.out.println("Берегам '" + name + "' уже и так некуда спешить.");

            }

        }

    }

    @Override

    public String getName() {

        return name;

    }

    @Override

    public String toString() {

        return "Берега '" + name + "'";

    }

    @Override

    public boolean equals(Object obj) {

        if (this == obj) return true;

        if (obj instanceof Shores) {

            return name.equals(((Shores) obj).getName()) && hurry == ((Shores) obj).isHurry();

        }

        return false;

    }

    @Override

    public int hashCode() {

        if (hurry) return name.hashCode() + name.length();

        return name.hashCode();

    }

}

**Summer.java**

package core;

import utility.SeasonAbstract;

import utility.SeasonType;

public class Summer extends SeasonAbstract {

    private String name;

    private final SeasonType TYPE = SeasonType.SUMMER;

    public Summer() {

        super(true);

        name = "Лето";

        joinStory();

    }

    public Summer(String name) {

        super(true);

        this.name = name;

        joinStory();

    }

    public Summer(String name, boolean lengthy) {

        super(lengthy);

        this.name = name;

        joinStory();

    }

    private void joinStory() {

        if (isLengthy()) {

            System.out.println("Долгое лето '" + name + "' присоединилось к истории.");

        } else {

            System.out.println("Лето '" + name + "' присоединилось к истории.");

        }

    }

    public void fulfillHopes(Hopes obj) {

        System.out.println("Лето '" + name + "' сулило исполнение всех надежд '" + obj.getName() + "'...");

        obj.fulfill();

    }

    @Override

    public String getName() {

        return name;

    }

    @Override

    public SeasonType getType() {

        return TYPE;

    }

    @Override

    public String toString() {

        if (isLengthy()) return "Долгое лето '" + name + "'";

        return "Лето '" + name + "'";

    }

@Override

    public boolean equals(Object obj) {

        if (this == obj) return true;

        if (obj instanceof Summer) {

            return name.equals(((Summer) obj).getName()) && isLengthy() == ((Summer) obj).isLengthy();

        }

        return false;

    }

    @Override

    public int hashCode() {

        if (isLengthy()) return name.hashCode() + name.length();

        return name.hashCode();

    }

}

**Talks.java**

package core;

import utility.ThingInterface;

public class Talks implements ThingInterface {

    private String name;

    public Talks() {

        name = "Разговоры";

        joinStory();

    }

    public Talks(String name) {

        this.name = name;

        joinStory();

    }

    private void joinStory() {

        System.out.println("Разговоры '" + name + "' присоединились к истории.");

    }

    public void notNeeded() {

        System.out.println("Разговоры '" + name + "' теперь ни к чему.");

    }

    @Override

    public String getName() {

        return name;

    }

    @Override

    public String toString() {

        return "Разговоры '" + name + "'";

    }

    @Override

    public boolean equals(Object obj) {

        if (this == obj) return true;

        if (obj instanceof Talks) {

            return name.equals(((Talks) obj).getName());

        }

        return false;

    }

    @Override

    public int hashCode() {

        return name.hashCode();

    }

}

Результат работы:

Берега 'Берега' присоединились к истории (им некуда спешить).

Долгое лето 'Лето' присоединилось к истории.

Надежды 'Надежды' присоединились к истории.

Лето 'Лето' сулило исполнение всех надежд 'Надежды'...

Надежды 'Надежды' исполнены!

Берега 'Берега' под впечатлением...

Разговоры 'Разговоры' присоединились к истории.

Разговоры 'Разговоры' теперь ни к чему.

Лодка 'Лодка' присоединилась к истории.

Лодка 'Лодка' стала описывать полукруг, направляясь к берегу.

Вывод:

Во время выполнения данной лабораторной работы я научился применять принципы SOLID на практике, подробнее разобрал интерфейсы, абстрактные классы и перечисления. Научился пользоваться системой сборки Gradle. Не выспался.