

Факультет программной инженерии и компьютерной техники

Дисциплина: Программирование

Лабораторная работа №3

Вариант № 311584

Выполнил: Васильев А.Ю.

№ группы: P3115

Преподаватель: Письмак А.Е.

2020 год, 1 семестр

1.Текст задания.

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

Потом они пропели Малышу "Многие лета", а Малыш выстрелил, и треск получился оглушительный. Все утро Малыш то и дело стрелял из пистолета, ждал гостей и все время размышлял о словах папы, что подарки могут появиться и днем. На какой-то счастливый миг он вдруг поверил, что свершится чудо -- ему подарят собаку. Но тут же понял, что это невозможно, и даже рассердился на себя за то, что так глупо размечтался. Ведь он твердо решил не думать сегодня о собаке и всему радоваться. И Малыш действительно всему радовался. Сразу же после обеда мама стала накрывать на стол у него в комнате. Она поставила в вазу большой букет цветов и принесла самые красивые розовые чашки. Три штуки.

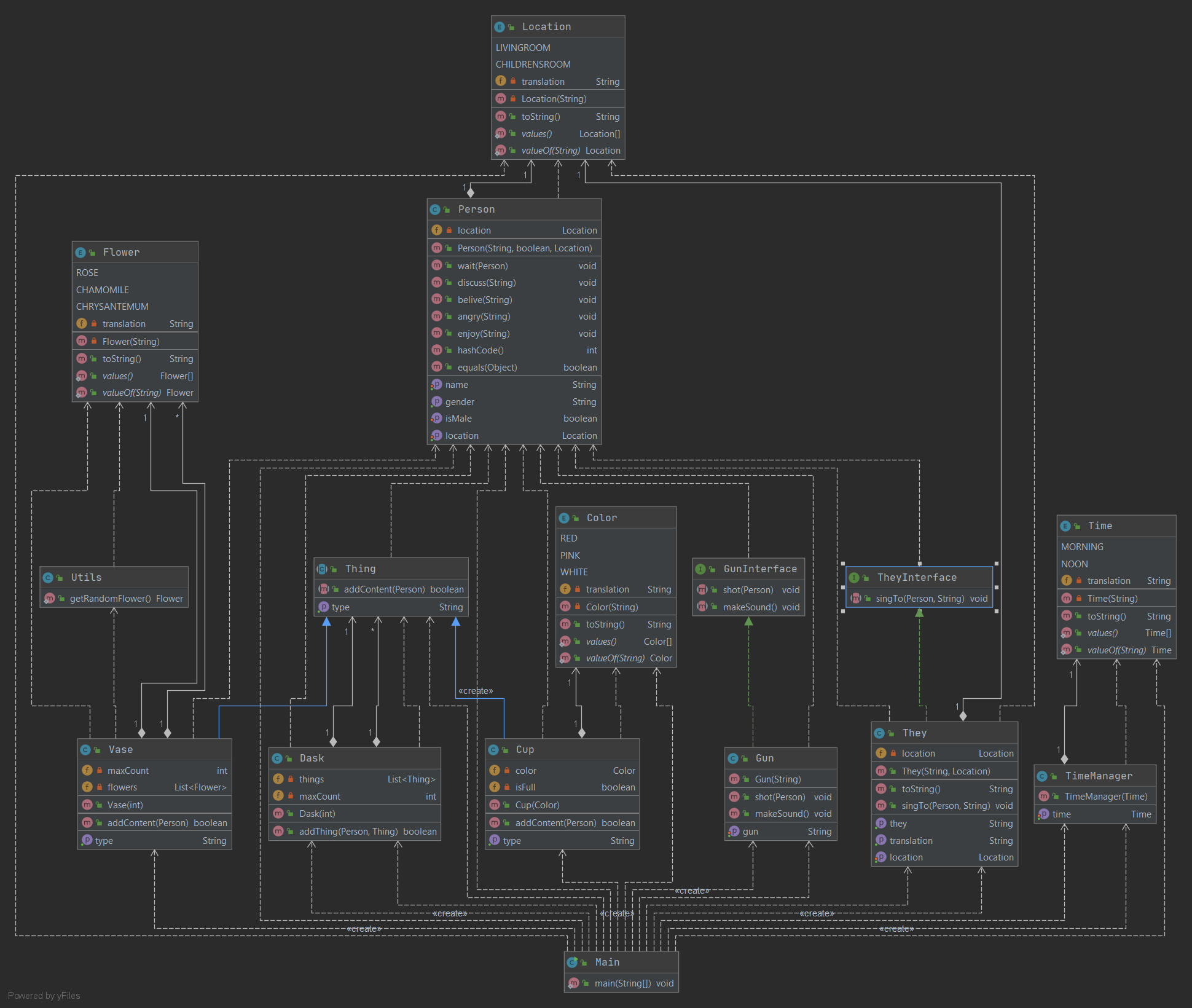
**Программа должна удовлетворять следующим требованиям:**

1. Доработанная модель должна соответствовать [принципам SOLID](https://en.wikipedia.org/wiki/SOLID_(object-oriented_design)).
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

2. Диаграмма классов объектной модель.



3.Исходный код.

Главный класс.

package prog;  
public class Main {  
 public static void main(String[] args) {  
 TimeManager manager = new TimeManager(Time.*MORNING*);  
 Person child = new Person("Малыш", true, Location.*LIVINGROOM*);  
 Person mom = new Person("Мама", false, Location.*LIVINGROOM*);  
 Person guest1 = new Person("гость Лёня", true, Location.*LIVINGROOM*);  
 Person guest2 = new Person("гость Маша", false, Location.*LIVINGROOM*);  
 Person guest3 = new Person("гость Саша", true, Location.*LIVINGROOM*);  
 They they = new They("Они",Location.*LIVINGROOM*);  
 child.getLocation();  
 System.*out*.println(manager.getTime());  
 they.getLocation();  
 they.singTo(child, "\"Многие лета\"");  
 Gun gun = new Gun("пистолет");  
 gun.shot(child);  
 child.wait(guest1);  
 child.wait(guest2);  
 child.wait(guest3);  
 child.discuss("словах папы, что подарки могут появиться и днем");  
 child.belive("свершится чудо - ему подарят собаку");  
 child.angry("размечтался");  
 child.enjoy("не думал о собаке");  
 child.setLocation(Location.*CHILDRENSROOM*);  
 manager.setTime(Time.*NOON*);  
 child.getLocation();  
 System.*out*.println(manager.getTime());  
 Dask dask = new Dask(4);  
 Thing vase =new Vase(11);  
 for (int i=0;i<12;i++) {  
 vase.addContent(mom);  
 }  
 dask.addThing(mom,vase);  
 for (int i=0;i<3;i++) {  
 Thing temp = new Cup(Color.*PINK*);  
 temp.addContent(mom);  
 dask.addThing(mom,temp);  
 }  
 }  
}

Класс Utils.

package prog;

public class Utils {  
  
 public static Flower getRandomFlower() {  
 int random = 1 + (int) (Math.*random*() \* 4);  
 if(random == 1){  
 return Flower.*ROSE*;  
 }  
 else if(random == 2){  
 return Flower.*CHAMOMILE*;  
 }  
 else{  
 return Flower.*CHRYSANTEMUM*;  
 }  
 }  
  
}

Класс TimeManager.

package prog;  
  
public class TimeManager{  
 private Time time;  
 public TimeManager(Time time){  
 this.time = time;  
 }  
  
 public void setTime(Time time) {  
 this.time = time;  
 }  
  
 public Time getTime() {  
 return time;  
 }  
  
}

Enum Time.

package prog;  
public enum Time{  
 *MORNING*("утро"), *NOON*("полдень");  
 private String translation;  
  
 Time(String translation){  
 this.translation = translation;  
 }  
  
 @Override  
 public String toString() {  
 return translation;  
 }  
}

Enum Location.

package prog;  
public enum Location {  
 *LIVINGROOM*("гостинная"), *CHILDRENSROOM*("детская");  
 private String translation;  
  
 Location(String translation){  
 this.translation = translation;  
 }  
  
 @Override  
 public String toString() {  
 return translation;  
 }  
  
  
}

Класс Person.

package prog;  
import java.util.Objects;  
  
public class Person {  
 private String name;  
 private boolean isMale;  
 private Location location;  
  
 public Person(String name, boolean isMale, Location location){  
 this.name = name;  
 this.isMale = isMale;  
 this.location = location;  
 }  
  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
  
 public void wait(Person person){  
 System.*out*.println(getName() + " ждал " + person.getName());  
 }  
  
 public void discuss(String about){  
 System.*out*.println(getName() + " размышлял о " + about);  
 }  
  
 public void belive(String about){  
 System.*out*.println(getName() + " поверил в то, что " + about);  
 }  
  
 public void setLocation(Location location) {  
 this.location = location;  
 }  
  
 public Location getLocation() {  
 System.*out*.println(getName() + " находится в " + location);  
 return location;  
 }  
  
 public void angry(String about){  
 System.*out*.println(getName()+" рассердился на себя за то, что "+about);  
 }  
 public void enjoy(String about){  
 System.*out*.println(getName()+" радовался, потому что "+about);  
 }  
  
 public void setIsMale(boolean isMale) {  
 this.isMale = isMale;  
 }  
 public String getGender() {  
 if(isMale == true){  
 return "мужчина";  
 }else{  
 return "женщина";  
 }  
 }  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(getName(), getGender(), getLocation());  
 }  
 @Override  
 public boolean equals(Object object){  
 if (object == null || object.getClass() != getClass()) {  
 return false;  
 }  
 else{  
 Person temp = (Person) object;  
 return Objects.*equals*(name, temp.name);  
 }  
 }  
}

Класс They.

package prog;  
  
public class They implements TheyInterface{  
 private String they;  
 private Location location;  
 public They(String they, Location location){  
 this.they=they;  
 this.location = location;  
 }  
 public String getTranslation(){  
 return they;  
 }  
 @Override  
 public String toString(){  
 return "Перевод на русский: " + they;  
 }  
 @Override  
 public void singTo(Person person, String what) {  
 System.*out*.println(getTranslation() + " спели " + person.getName() + " " + what);  
 }  
 public void setLocation(Location location) {  
 this.location = location;  
 }  
  
 public Location getLocation() {  
 System.*out*.println(getThey() + " находится в " + location);  
 return location;  
 }  
  
 public String getThey() {  
 return they;  
 }  
}

Интерфейс TheyInterface.

package prog;  
  
public interface TheyInterface {  
 public void singTo(Person person, String what);  
}

Класс Gun.

package prog;  
  
public class Gun implements GunInterface {  
 private String gun;  
 public Gun(String gun){  
 this.gun = gun;  
 }  
  
 public void setGun(String gun){  
 this.gun = gun;  
 }  
  
 public String getGun() {  
 return gun;  
 }  
 @Override  
 public void shot(Person person){  
 System.*out*.print(person.getName() + " выстрелил из " + getGun() + ", ");  
 makeSound();  
 }  
 @Override  
 public void makeSound() {  
 System.*out*.println("c оглушительным треском");  
 }  
  
}

Интерфейс GunInterface.

package prog;  
  
public interface GunInterface {  
 public void shot(Person person);  
 public void makeSound();  
}

Класс Dask.

package prog;  
import java.util.ArrayList;  
import java.util.List;  
public class Dask {  
 private List<Thing> things = new ArrayList<Thing>();  
 private int maxCount;  
 public Dask(int maxCount){  
 this.maxCount = maxCount;  
 }  
 public boolean addThing(Person person,Thing thing){  
 if(things.size()!=maxCount) {  
 System.*out*.println(person.getName() + " добавила на стол " + thing.getType());  
 things.add(thing);  
 return true;  
 }else{  
 return false;  
 }  
 }  
}

Абстрактный класс Thing.

package prog;  
  
public abstract class Thing{  
 public abstract boolean addContent(Person person);  
 public abstract String getType();  
}

Класс Vase.

package prog;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class Vase extends Thing {  
 private int maxCount;  
 private List<Flower> flowers = new ArrayList<Flower>();  
 public Vase(int maxFlowers){  
 this.maxCount = maxFlowers;  
 }  
 @Override  
 public boolean addContent(Person person){  
 if(flowers.size()!=maxCount) {  
 Flower temp = Utils.*getRandomFlower*();  
 flowers.add(temp);  
 System.*out*.println(person.getName()+" добваила " + temp);  
 return true;  
 }else{  
 System.*out*.println("Ваза полная");  
 return false;  
 }  
 }  
  
 @Override  
 public String getType() {  
 return "ваза";  
 }  
  
  
}

Enum Flower.

package prog;  
  
public enum Flower{  
 *ROSE*("роза"),*CHAMOMILE*("ромашка"),*CHRYSANTEMUM*("хризантема");  
 private String translation;  
  
 Flower(String translation){  
 this.translation = translation;  
 }  
  
 @Override  
 public String toString() {  
 return translation;  
 }  
}

Класс Cup.

package prog;  
  
public class Cup extends Thing {  
 private Color color;  
 private boolean isFull;  
 public Cup(Color color){  
 this.color = color;  
 }  
 @Override  
 public boolean addContent(Person person){  
 if(!isFull) {  
 System.*out*.println(person.getName() + " налила чай в кружку.");  
 isFull = true;  
 return true;  
 }else{  
 System.*out*.println(person.getName() + " не может налить чай в кружку, потому что она полная.");  
 return false;  
 }  
 }  
 @Override  
 public String getType() {  
 return color + " кружка";  
 }  
}

Enum Color.

package prog;  
  
public enum Color{  
 *RED*("красную"),*PINK*("розовую"),*WHITE*("белую");  
 private String translation;  
  
 Color(String translation){  
 this.translation = translation;  
 }  
  
 @Override  
 public String toString() {  
 return translation;  
 }  
}

4.Программа выводит.

Малыш находится в гостинная

утро

Они находится в гостинная

Они спели Малыш "Многие лета"

Малыш выстрелил из пистолет, c оглушительным треском

Малыш ждал гость Лёня

Малыш ждал гость Маша

Малыш ждал гость Саша

Малыш размышлял о словах папы, что подарки могут появиться и днем

Малыш поверил в то, что свершится чудо - ему подарят собаку

Малыш рассердился на себя за то, что размечтался

Малыш радовался, потому что не думал о собаке

Малыш находится в детская

полдень

Мама добваила хризантема

Мама добваила роза

Мама добваила ромашка

Мама добваила хризантема

Мама добваила роза

Мама добваила хризантема

Мама добваила ромашка

Мама добваила роза

Мама добваила роза

Мама добваила хризантема

Мама добваила хризантема

Ваза полная

Мама добавила на стол ваза

Мама налила чай в кружку.

Мама добавила на стол розовую кружка

Мама налила чай в кружку.

Мама добавила на стол розовую кружка

Мама налила чай в кружку.

Мама добавила на стол розовую кружка

5.В ходе данной лабораторной работы, была реализованна объектная модель, описанная в тексте варианта, я разобрался научился применять Enum, абстрактные классы, интерфейсы, а также познакомился с некоторыми методами класса Object. В процессе написания я применял принципы SOLID.Благодаря этому, теперь, я лучше знаю как проектировать java-проекты.