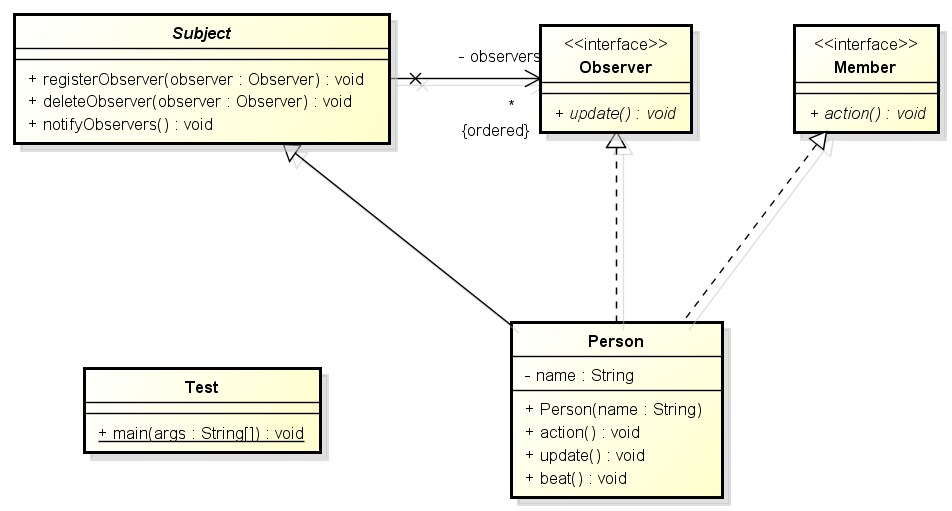
# 面向对象系统分析与设计课后作业02

## In a multiplayer game, many players can join the same team. As one member of the team was attacked by the enemy, all the other members will received the notice and take the action. Your job is to program according to the following class diagram.（一款多人联机对战游戏,在该游戏中,多个玩家可以加入同一战队组成联盟,当战队中某一个成员受到敌人的攻击时将给所有其他盟友发送通知,盟友收到通知后将做出响应。下面是类图，请根据类图编写代码。）



package MG2;

/\*\*

 \* @author QiuQian

 \*/

public interface Member {

    public abstract void action();

}

package MG2;

/\*\*

 \* @author QiuQian

 \*/

public interface Observer {

    public abstract void update();

}

package MG2;

/\*\*

 \* @author QiuQian

 \*/

public class Person extends Subject implements Observer, Member {

    private String name;

    public Person(String name) {

        this.name = name;

    }

    public void beat() {

        System.out.println(name + "受到了攻击\n");

    }

    @Override

    public void action() {

        System.out.println(name + "做出了反应\n");

    }

    @Override

    public void update() {

        System.out.println(name + "收到了通知");

        action();

    }

    @Override

    public void notifyObservers() {

        for ( Observer observer : observerList ) {

            observer.update();

        }

    }

}

package MG2;

import java.util.ArrayList;

import java.util.List;

/\*\*

 \* @author QiuQian

 \*/

public abstract class Subject {

    protected List<Observer> observerList = new ArrayList<>();

    public void registerObserver(Observer observer) {

        observerList.add(observer);

    }

    public void deleteObserver(Observer observer) {

        observerList.remove(observer);

    }

    public void notifyObservers() {

    }

}

package MG2;

/\*\*

 \* @author QiuQian

 \*/

public class Test {

    public static void main(String[] args) {

        Person person = new Person("A");

        person.registerObserver(new Person("B1"));

        person.registerObserver(new Person("B2"));

        person.registerObserver(new Person("B3"));

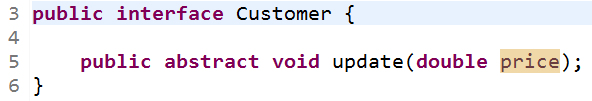
        person.beat();

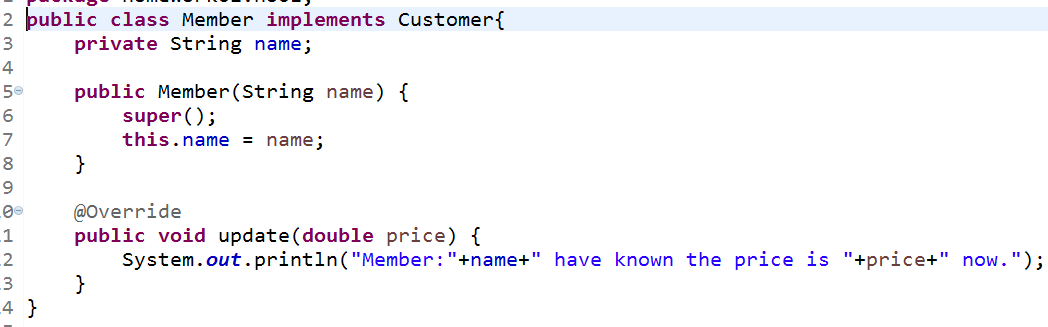
        person.notifyObservers();

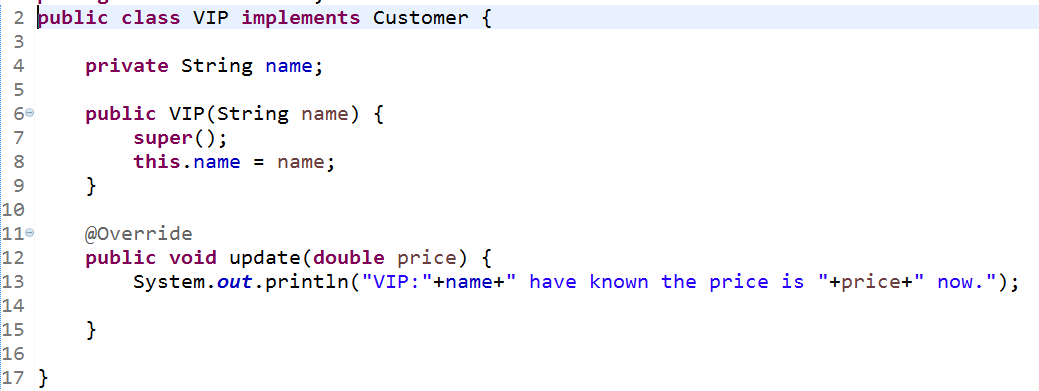
    }

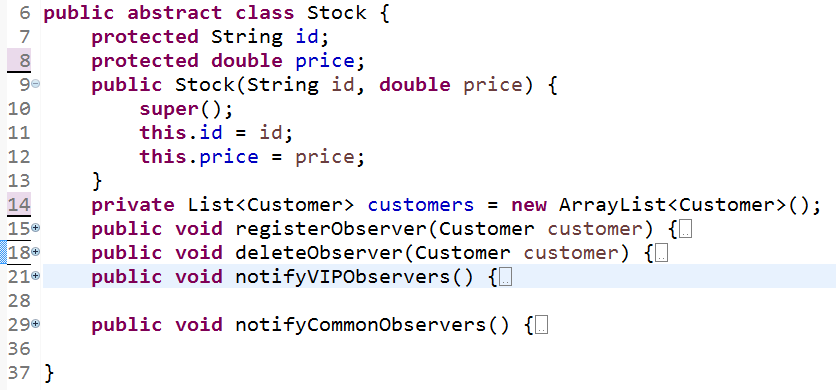
}

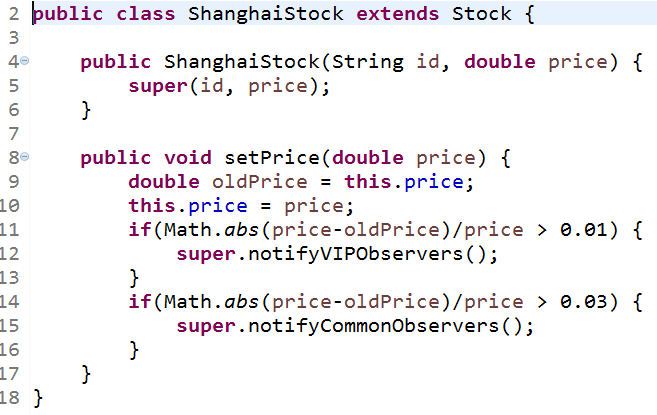
## Designing an online stock software which has the following functions: 1. Supports two securities trading markets in Shanghai and Shenzhen;2. Support common and VIP stock customers;3. When the price of shares traded on the Shanghai stock exchange fluctuate 3%, the ordinary customer will be notified, and the VIP customers will be notified when the price changes 1%.When the price of the shares traded on the Shenzhen stock exchange is 5%, the ordinary customer will be notified, and the VIP customers will be notified when the price changes 1%. Your job is to draw the class diagram according to the following code in Java.（设计一款在线股票软件,该软件具有如下功能:1.支持上海及深圳两个证券交易市场;2.支持普通和VIP两类股票客户;3.上证交易所上交易的股票价格变动3%时,通知普通客户,价格变动1%时,通知VIP客户; 深证交易所上交易的股票价格变动5%时,通知普通客户,价格变动2%时,通知VIP客户。实现这个系统的代码如下，请根据代码画出类图。）

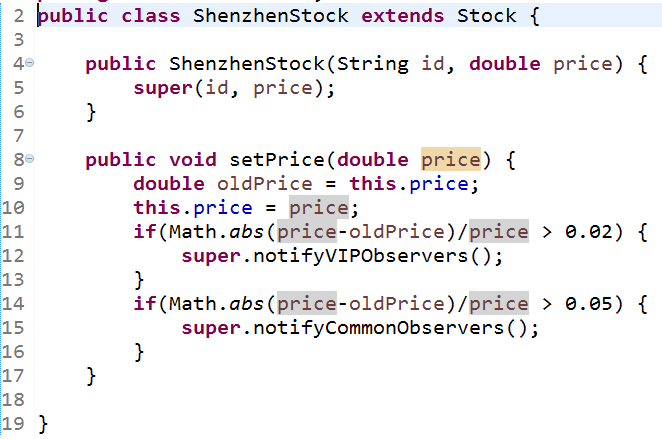


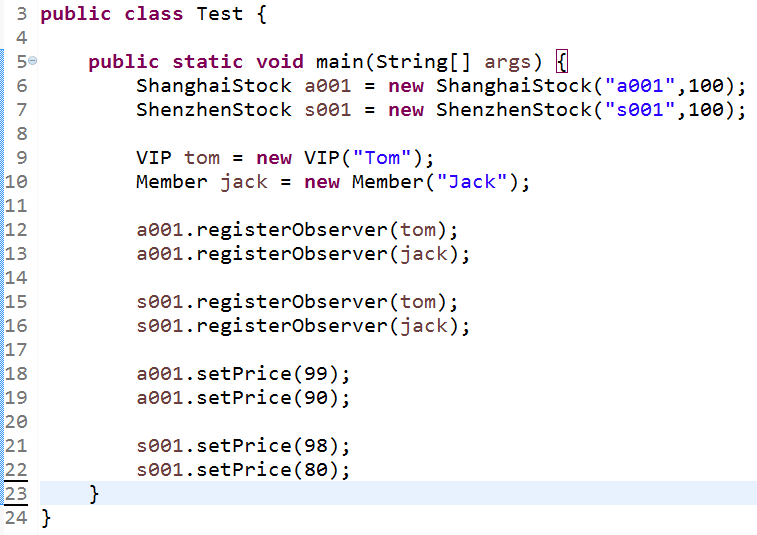


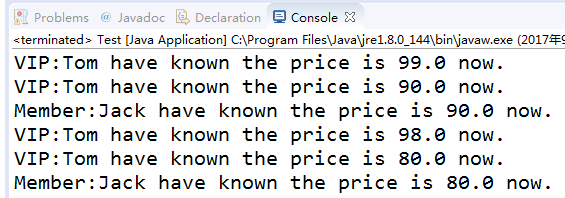


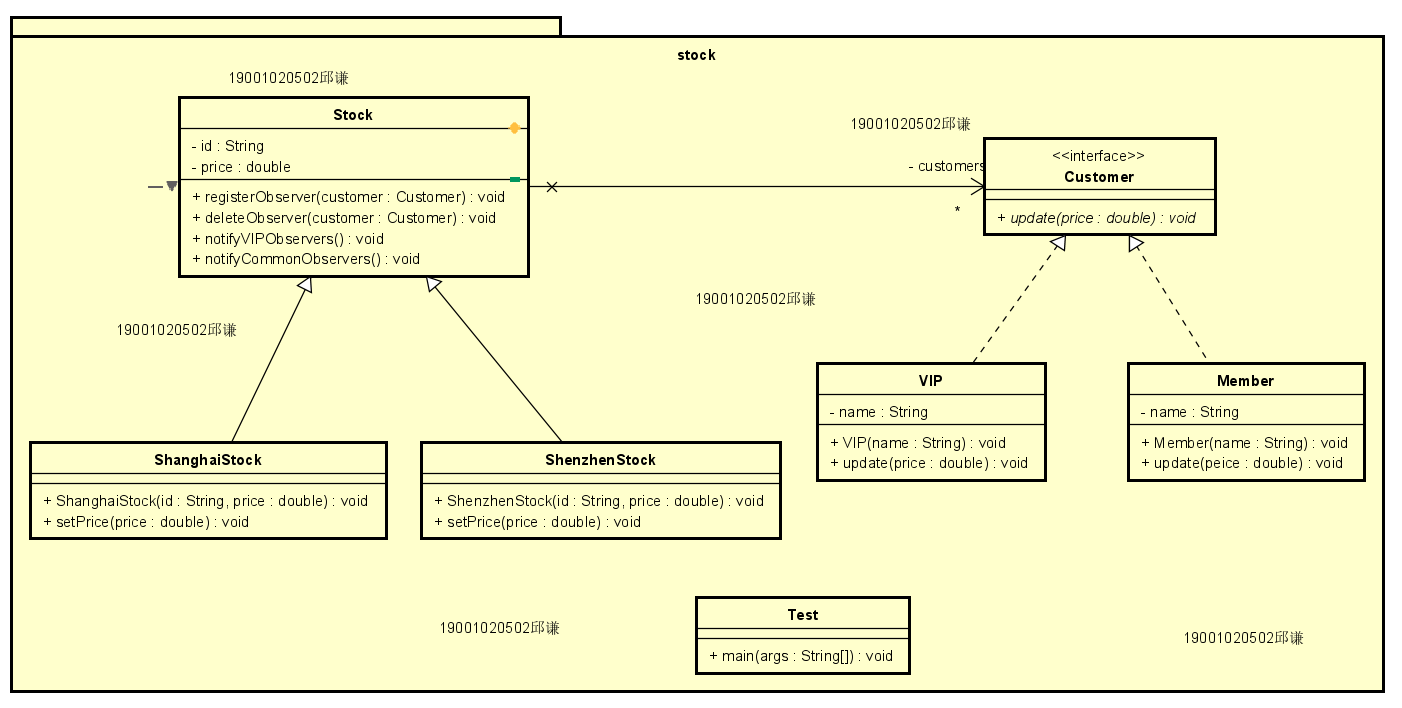












## 3. National treasure game system.

**系统名称：国家宝藏游戏系统（National treasure game system）**

人物角色：凯奇（Cage）

两个守护神(Eudemon)：老虎（Tiger）、狮子（Lion）

两种宝藏（Treasure）：黄金（Gold）、钻石（Diamond）

游戏场景（Scene）：场景中包括1个角色，5只老虎，7只狮子，500箱黄金，100箱钻石

功能需求：

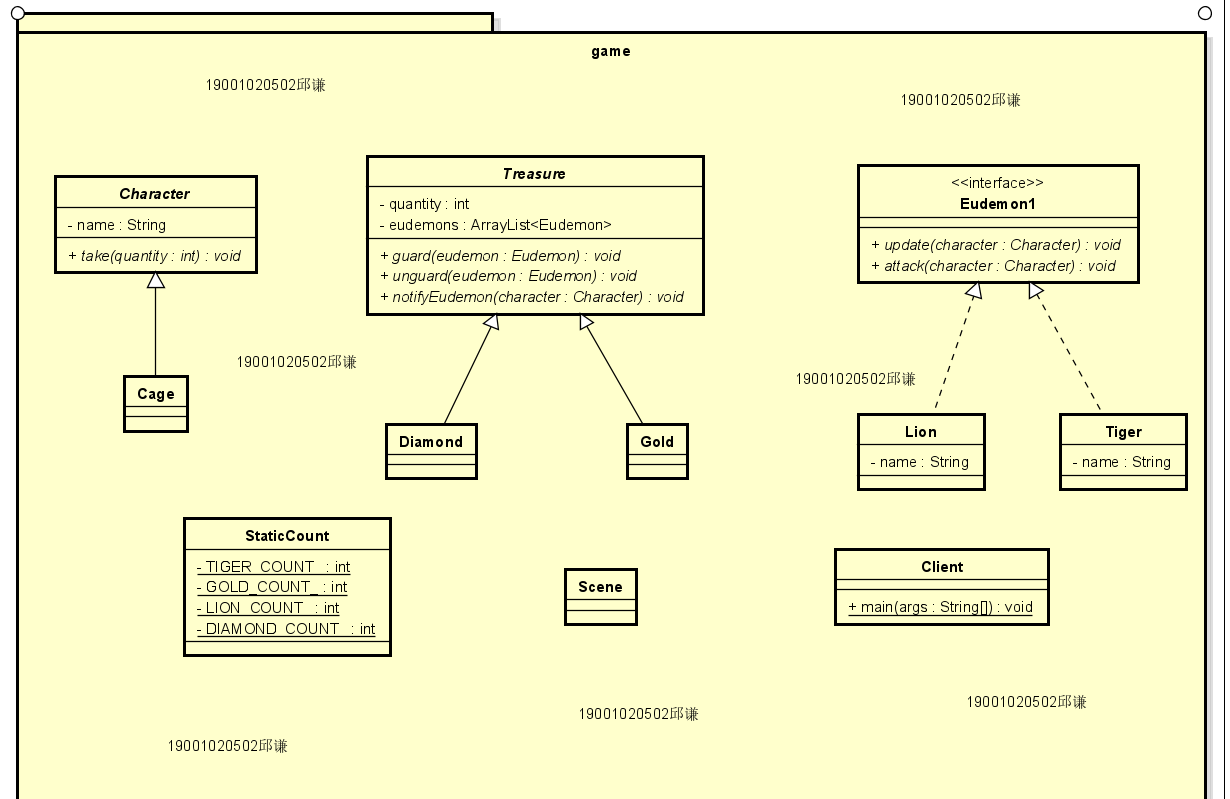
1、凯奇可以拿走一定数量的宝藏take（Treature treature,double quantity）

2、宝藏可以安排数位守护神看守宝藏guard（Eudemon eudemon）

3、每当凯奇拿走宝藏时，都会惊动所有看守该宝藏的守护神notifyAll（）

4、守护神在得到通知update（）后，都会清点count（）数量，并发出攻击attacks（）

5、场景中的情节是凯奇拿走了132#整箱、357#半箱黄金，以及3#整箱钻石



package game;

/\*\*

 \* @author QiuQian

 \*/

public class Cage extends Character {

    public Cage(String name) {

        super(name);

    }

    @Override

    public void take(int quantity, Treasure treasure) {

        System.out.println(name + "拿走了 " + quantity + " 个treasure");

        treasure.setQuantity(treasure.getQuantity() - quantity);

        treasure.notifyEudemon(this);

    }

}

package game;

/\*\*

 \* @author QiuQian

 \*/

public abstract class Character {

    protected String name;

    public Character(String name) {

        this.name = name;

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    public abstract void take(int quantity, Treasure treasure);

}

package game;

/\*\*

 \* @author QiuQian

 \*/

public class Client {

    public static void main(String[] args) {

        Scene scene = new Scene();

        Gold gold = scene.golds.get(13);

        Diamond diamond = scene.diamonds.get(3);

        scene.cage.take(50, gold);

        scene.cage.take(100, diamond);

    }

}

package game;

/\*\*

 \* @author QiuQian

 \*/

public class Diamond extends Treasure {

    public Diamond(int quantity) {

        super(quantity);

    }

    @Override

    public void guard(Eudemon eudemon) {

        getEudemons().add(eudemon);

    }

    @Override

    public void unguard(Eudemon eudemon) {

        getEudemons().remove(eudemon);

    }

    @Override

    public void notifyEudemon(Character character) {

        for ( Eudemon eudemon : getEudemons() ) {

            eudemon.update(character);

        }

    }

}

package game;

/\*\*

 \* @author QiuQian

 \*/

public interface Eudemon {

    public abstract void update(Character character);

    public abstract void attack(Character character);

}

package game;

/\*\*

 \* @author QiuQian

 \*/

public class Gold extends Treasure {

    public Gold(int quantity) {

        super(quantity);

    }

    @Override

    public void guard(Eudemon eudemon) {

        getEudemons().add(eudemon);

    }

    @Override

    public void unguard(Eudemon eudemon) {

        getEudemons().remove(eudemon);

    }

    @Override

    public void notifyEudemon(Character character) {

        for ( Eudemon eudemon : getEudemons() ) {

            eudemon.update(character);

        }

    }

}

package game;

/\*\*

 \* @author QiuQian

 \*/

public class Lion implements Eudemon {

    private String name;

    public Lion(String name) {

        this.name = name;

    }

    /\*\*

     \* @see Eudemon#update(Character)

     \*

     \*/

    @Override

    public void update(Character character) {

        attack(character);

    }

    /\*\*

     \* @see Eudemon#attack(Character)

     \*

     \*/

    @Override

    public void attack(Character character) {

        System.out.println("守护神攻击了 " + character.getName());

    }

}

package game;

import java.util.ArrayList;

import java.util.List;

/\*\*

 \* @author QiuQian

 \*/

public class Scene {

    Cage cage = new Cage("凯奇");

    List<Tiger> tigers = new ArrayList<>();

    List<Lion> lions = new ArrayList<>();

    List<Gold> golds = new ArrayList<>();

    List<Diamond> diamonds = new ArrayList<>();

    private int TIGER\_COUNT = 5;

    private int GOLD\_COUNT = 500;

    private int LION\_COUNT = 7;

    private int DIAMOND\_COUNT = 100;

    public Scene() {

        for ( int i = 0; i < TIGER\_COUNT; i++ ) {

            tigers.add(new Tiger("Tiger" + i));

        }

        for ( int i = 0; i < LION\_COUNT; i++ ) {

            lions.add(new Lion("Lion" + i));

        }

        for ( int i = 0; i < GOLD\_COUNT; i++ ) {

            Gold gold = new Gold(500);

            for ( Tiger tiger : tigers ) {

                gold.guard(tiger);

            }

            for ( Lion lion : lions ) {

                gold.guard(lion);

            }

            golds.add(gold);

        }

        for ( int i = 0; i < DIAMOND\_COUNT; i++ ) {

            Diamond diamond = new Diamond(100);

            for ( Tiger tiger : tigers ) {

                diamond.guard(tiger);

            }

            for ( Lion lion : lions ) {

                diamond.guard(lion);

            }

            diamonds.add(diamond);

        }

    }

}

package game;

/\*\*

 \* @author QiuQian

 \*/

public class Tiger implements Eudemon {

    private String name;

    public Tiger(String name) {

        this.name = name;

    }

    /\*\*

     \* @see Eudemon#update(Character)

     \*

     \*/

    @Override

    public void update(Character character) {

        attack(character);

    }

    /\*\*

     \* @see Eudemon#attack(Character)

     \*

     \*/

    @Override

    public void attack(Character character) {

        System.out.println("守护神攻击了 " + name);

    }

}

package game;

import java.util.ArrayList;

/\*\*

 \* @author QiuQian

 \*/

public abstract class Treasure {

    protected int quantity;

    public Treasure(int quantity) {

        this.quantity = quantity;

    }

    public int getQuantity() {

        return quantity;

    }

    public void setQuantity(int quantity) {

        this.quantity = quantity;

    }

    protected ArrayList<Eudemon> eudemons = new ArrayList<>();

    public ArrayList<Eudemon> getEudemons() {

        return eudemons;

    }

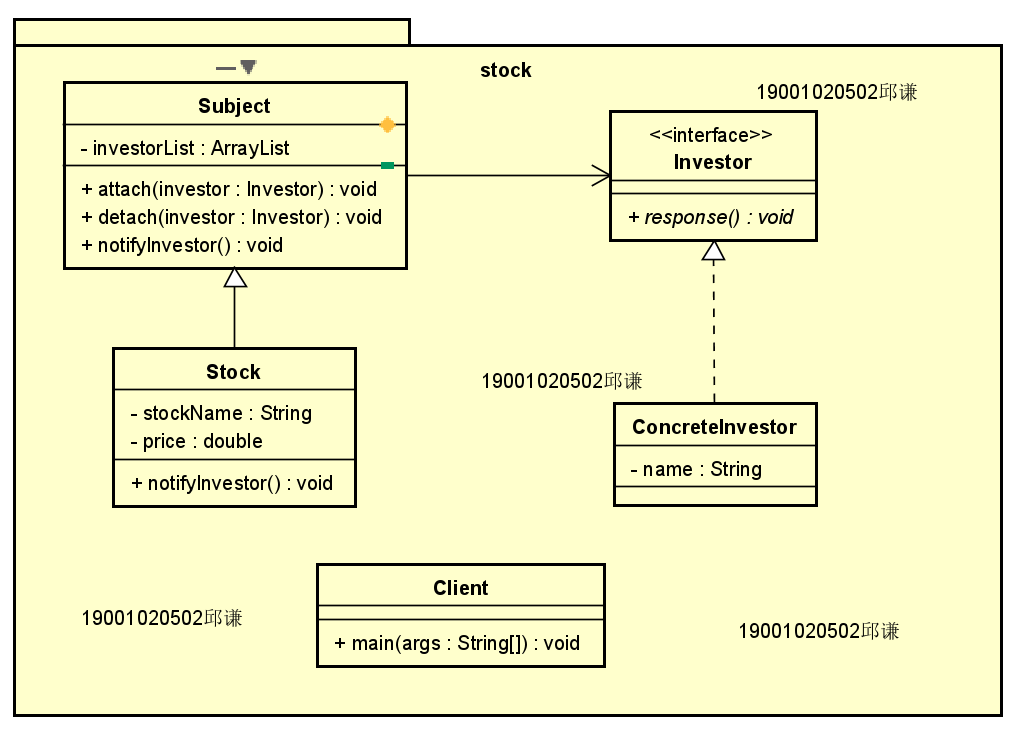
    public abstract void guard(Eudemon eudemon);

    public abstract void unguard(Eudemon eudemon);

    public abstract void notifyEudemon(Character character);

}

## 股票监测



package stock1;

/\*\*

 \* @author QiuQian

 \*/

public class Client {

    public static void main(String[] args) {

        Stock stock = new Stock("Neusoft", 100);

        stock.attach(new ConcreteInvestor("A"));

        stock.attach(new ConcreteInvestor("B"));

        stock.notifyInvestor(98);

//        stock.attach(new ConcreteInvestor("C"));

//        stock.attach(new ConcreteInvestor("D"));

//        stock.notifyInvestor(92);

    }

}

package stock1;

/\*\*

 \* @author QiuQian

 \*/

public class ConcreteInvestor implements Investor {

    private String name;

    public ConcreteInvestor(String name) {

        this.name = name;

    }

    @Override

    public void response() {

        System.out.println(name + "请注意，您购买的股票价格浮动超过了5%");

    }

}

package stock1;

/\*\*

 \* @author QiuQian

 \*/

public interface Investor {

    public abstract void response();

}

package stock1;

/\*\*

 \* @author QiuQian

 \*/

public class Stock extends Subject {

    private String stockName;

    private double price;

    public Stock(String stockName, double price) {

        this.stockName = stockName;

        this.price = price;

    }

    @Override

    public void notifyInvestor(double newPrice) {

        if ( Math.abs(newPrice - price) > price \* 0.05 ) {

            for ( Investor investor : investorList ) {

                investor.response();

            }

        }

    }

}

package stock1;

import java.util.ArrayList;

import java.util.List;

/\*\*

 \* @author QiuQian

 \*/

public abstract class Subject {

    protected List<Investor> investorList = new ArrayList<>();

    public void attach(Investor investor) {

        investorList.add(investor);

    }

    public void detach(Investor investor) {

        investorList.remove(investor);

    }

    public void notifyInvestor(double newPrice) {

    }

}