软件构造第2次实验

实验报告

**班级**：软件工程14-3

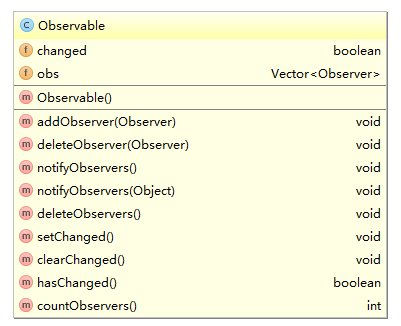
**学号**：201401061038

**姓名**：张正锟

2016年3月16日

# 设计

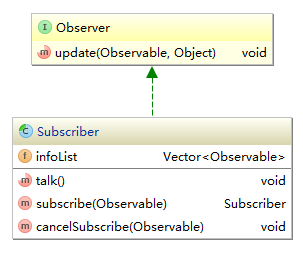
## java.util. Observable类图



JDK提供的被观察者，里面有一个Vector，存放观察他的所有Observer

含有多个方法添加和删除观察者，当发生变化时，可以调用所有Observer的update方法

## 订阅者（继承自java.util.Observer）类图

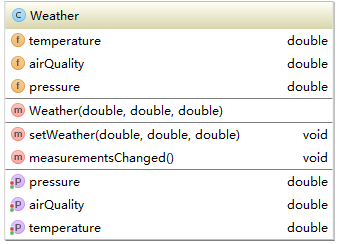


继承自JDK的Observer

为了实现多对多，在每个Observer中含有一个Vector，存放这个Observer所观察的所有被观察者

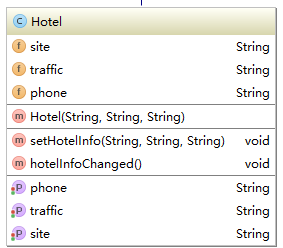
可以添加和删除被观察者

## 天气类图



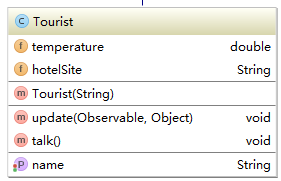
提供天气信息的Observable，可以提供温度、空气质量、气压三个信息

## 旅馆类图



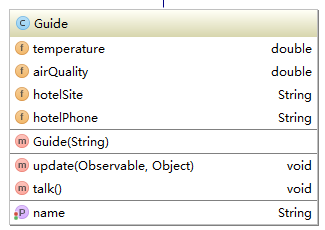
提供旅馆信息的Observable，可以提供旅馆地址、附近交通状况和旅馆电话信息

## 游客类图



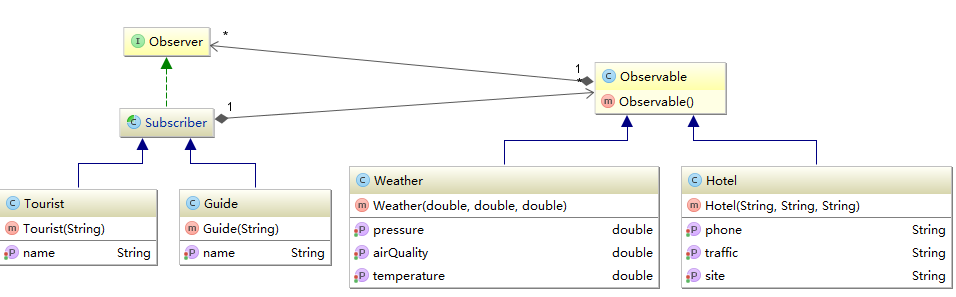
可以订阅多重被观察者的游客Observer，当温度和旅馆地点发生变化时可以自动更新并且调用talk()方法输出信息

## 导游类图



可以订阅多重被观察者的导游Observer，当温度、空气质量、旅馆地点、旅馆电话发生变化时可以自动更新并且调用talk()方法输出信息

## 整体继承图



每个Subscriber可以订阅多个Observable，一个Observable可以被多个Subscriber订阅

并且当一个Observable更新的时候，订阅他的所有Subscriber都将收到更新

# 实现

## 订阅者（Subscriber implements Observer）

**public abstract class** Subscriber **implements** Observer {  
  
 **private** Vector<Observable> **infoList** = **new** Vector<>();  
  
 **public abstract void** talk();  
  
 **public** Subscriber subscribe(Observable observable) {  
 **infoList**.add(observable);  
 observable.addObserver(**this**);  
 update(observable, **null**);  
 **return this**;  
 }  
  
 **public void** cancelSubscribe(Observable observable) {  
 observable.deleteObserver(**this**);  
 **infoList**.remove(observable);  
 }  
}

## 游客实现

**public class** Tourist **extends** Subscriber {  
  
 **private** String **name**;  
 **private double temperature**;  
 **private** String **hotelSite**;  
  
 **public** Tourist(String name) {  
 **this**.**name** = name;  
 }  
  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public void** setName(String name) {  
 **this**.**name** = name;  
 }  
  
 @Override  
 **public void** update(Observable o, Object arg) {  
 **if**(o **instanceof** Weather) {  
 Weather weather = (Weather) o;  
 **temperature** = weather.getTemperature();  
 } **else if**(o **instanceof** Hotel) {  
 Hotel hotel = (Hotel) o;  
 **hotelSite** = hotel.getSite();  
 }  
 talk();  
 }  
  
 @Override  
 **public void** talk() {  
 System.***out***.println(**"游客"** + **name** + **"：\n当前天气："** + **temperature** + **" 目标旅馆地址："** + **hotelSite**);  
 }  
}

## 导游实现

**public class** Guide **extends** Subscriber {  
  
 **private** String **name**;  
 **private double temperature**;  
 **private double airQuality**;  
 **private** String **hotelSite**;  
 **private** String **hotelPhone**;  
  
 **public** Guide(String name) {  
 **this**.**name** = name;  
 }  
  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public void** setName(String name) {  
 **this**.**name** = name;  
 }  
  
 @Override  
 **public void** update(Observable o, Object arg) {  
 **if**(o **instanceof** Weather) {  
 Weather weather = (Weather) o;  
 **temperature** = weather.getTemperature();  
 **airQuality** = weather.getAirQuality();  
 } **else if**(o **instanceof** Hotel) {  
 Hotel hotel = (Hotel) o;  
 **hotelSite** = hotel.getSite();  
 **hotelPhone** = hotel.getPhone();  
 }  
 talk();  
 }  
  
 @Override  
 **public void** talk() {  
 System.***out***.println(**"导游"** + **name** + **"：\n当前天气："** + **temperature** + **" 当前空气质量状况："** + **airQuality** +  
 **" 目标旅馆地址："** + **hotelSite** + **" 旅馆电话："** + **hotelPhone**);  
 }  
}

## 天气信息实现

**public class** Weather **extends** Observable {  
  
 **private double temperature**;  
 **private double airQuality**;  
 **private double pressure**;  
  
 **public** Weather(**double** temperature, **double** airQuality, **double** pressure) {  
 **this**.**temperature** = temperature;  
 **this**.**airQuality** = airQuality;  
 **this**.**pressure** = pressure;  
 measurementsChanged();  
 }  
  
  
 **public void** setWeather(**double** temperature, **double** airQuality, **double** pressure) {  
 **this**.**temperature** = temperature;  
 **this**.**airQuality** = airQuality;  
 **this**.**pressure** = pressure;  
 }  
  
 **public void** measurementsChanged() {  
 setChanged();  
 notifyObservers();  
 }  
  
 **public double** getTemperature() {  
 **return temperature**;  
 }  
  
 **public double** getAirQuality() {  
 **return airQuality**;  
 }  
  
 **public double** getPressure() {  
 **return pressure**;  
 }  
  
 **public void** setTemperature(**double** temperature) {  
 **this**.**temperature** = temperature;  
 measurementsChanged();  
 }  
  
 **public void** setAirQuality(**double** airQuality) {  
 **this**.**airQuality** = airQuality;  
 measurementsChanged();  
 }  
  
 **public void** setPressure(**double** pressure) {  
 **this**.**pressure** = pressure;  
 measurementsChanged();  
 }  
}

## 旅馆信息实现

**public class** Hotel **extends** Observable {  
  
 **private** String **site**;  
 **private** String **traffic**;  
 **private** String **phone**;  
  
 **public** Hotel(String site, String traffic, String phone) {  
 **this**.**site** = site;  
 **this**.**traffic** = traffic;  
 **this**.**phone** = phone;  
 hotelInfoChanged();  
 }  
  
 **public void** setHotelInfo(String site, String traffic, String phone) {  
 **this**.**site** = site;  
 **this**.**traffic** = traffic;  
 **this**.**phone** = phone;  
 hotelInfoChanged();  
 }  
  
 **public void** hotelInfoChanged() {  
 setChanged();  
 notifyObservers();  
 }  
  
 **public** String getSite() {  
 **return site**;  
 }  
  
 **public** String getTraffic() {  
 **return traffic**;  
 }  
  
 **public** String getPhone() {  
 **return phone**;  
 }  
  
 **public void** setSite(String site) {  
 **this**.**site** = site;  
 hotelInfoChanged();  
 }  
  
 **public void** setTraffic(String traffic) {  
 **this**.**traffic** = traffic;  
 hotelInfoChanged();  
 }  
  
 **public void** setPhone(String phone) {  
 **this**.**phone** = phone;  
 hotelInfoChanged();  
 }  
}

## 整体测试

**public class** MyTest {  
 @Test  
 **public void** test() {  
 Weather weather = **new** Weather(23, 100, 101);  
 Hotel hotel = **new** Hotel(**"青岛皇家旅馆"**, **"良好"**, **"17854258245"**);  
 Guide guide = **new** Guide(**"小王"**);  
 Tourist tourist = **new** Tourist(**"小张"**);  
  
 guide.subscribe(weather).subscribe(hotel);  
 tourist.subscribe(weather).subscribe(hotel);  
  
 weather.setTemperature(30);  
 }  
}