Design Mode阅读笔记

by chenxiangjie3@jd.com (mailto:chenxiangjie3@jd.com) 8/5/2020

Chapter2 观察者模式

I.Backgroud & Demands

1.对象间存在一对多关系，且一个对象被修改时，需要通知依赖它的对象。
2.考虑易用、低耦合、可拓展性。

II.Motivation

1.Strive for loosely coupled designs between objects that interact.
为交互对象之间的松耦合设计而努力。

III.Our Mode

Definition

观察者模式定义了对象间的一对多依赖，当一个对象改变时，他的所有依赖者都会收到通知并自动更新。

Interface

需要设计两类接口分别由发布消息方与获取消息方实现(继承)。
java.util.Observer与java.util.Observable是java自带的观察者模式接口。

其中，Observable用于消息提供方继承(它是个实现好的类)。

java.util.Observable

变量列表:

变量名	类型	备注

变量名	类型	备注
changed	boolean	是否更新
obs	Vector<Observer>	订阅者数组

方法列表：

方法名	类型	参数	备注
Observable	N/A	N/A	constructor,初始化Vector
addObserver	void	Observer o	增加订阅者
deleteObserver	void	Observer o	删除订阅者
deleteObservers	void	N/A	删除所有订阅者
setChanged	void	N/A	置真
clearChanged	void	N/A	置否
hasChanged	boolean	N/A	返回changed
notifyObserver	void	N/A	调用重载有参方法,参数为null
notifyObserver	void	Object arg	调用订阅者的update方法，只通知该参数
countObservers	int	N/A	订阅者计数

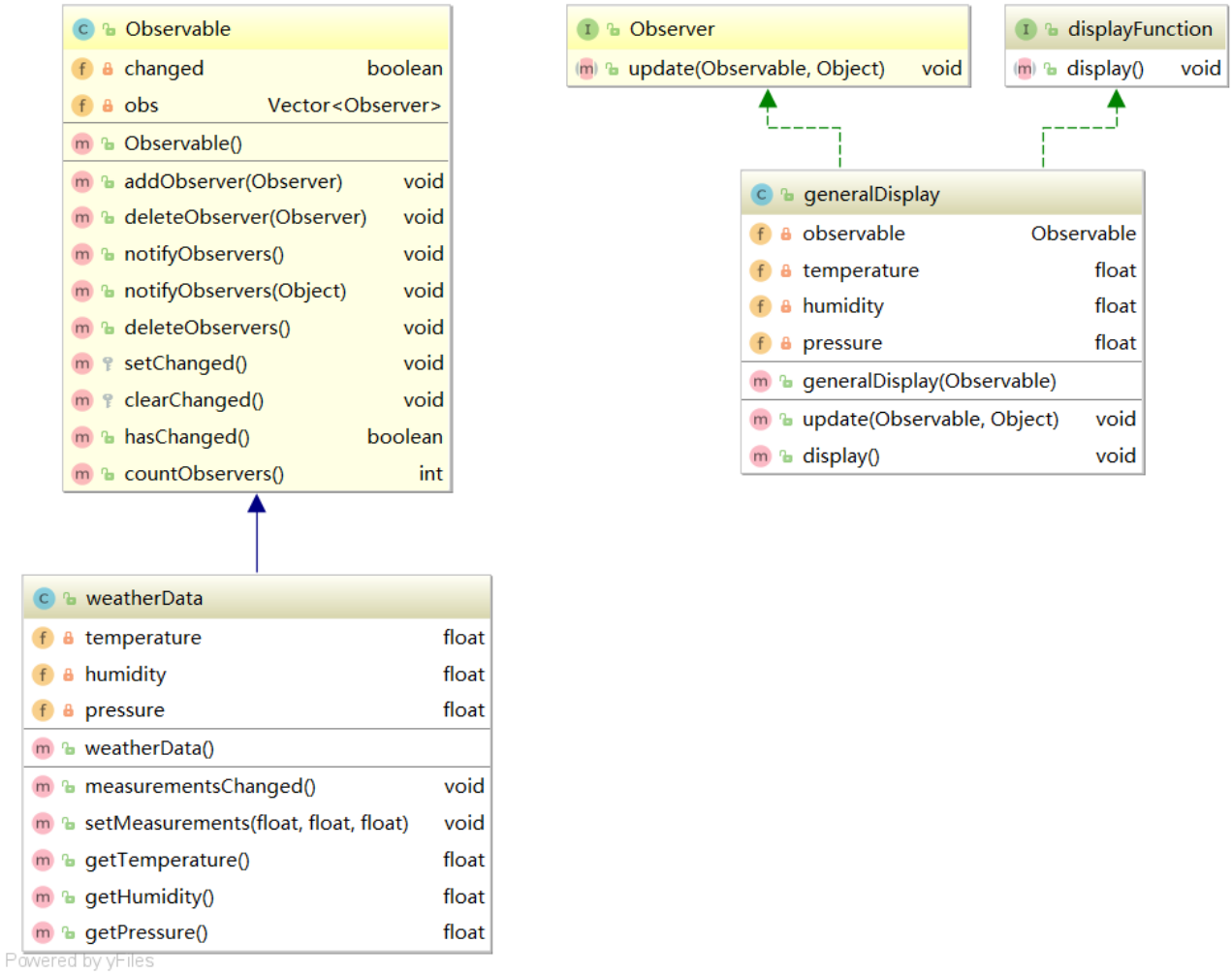
java.util.Observer

方法名	类型	参数	备注
update	void	Observable o, Object arg	每次被订阅者更新时调用

Implementation

IV.Example

UML Graph



Test Class

```
        Observable weather = new weatherData();
        Observer obs = new generalDisplay(weather);

        ((weatherData) weather).setMeasurements(29,20,101.5f);
        TimeUnit.SECONDS.sleep(10);
        ((weatherData) weather).setMeasurements(30,22,101.2f);
        TimeUnit.SECONDS.sleep(10);
        ((weatherData) weather).setMeasurements(27,25,100.3f);
```

Result

```
"C:\Program Files\Java\jdk1.8.0_241\bin\java.exe" ...

Data updated from weather monitor center.
Current time: Wed Aug 05 13:53:00 CST 2020
Current conditions: 29.0 °C degrees and 20.0 % humidity and 101.5 kpa.

Data updated from weather monitor center.
Current time: Wed Aug 05 13:53:10 CST 2020
Current conditions: 30.0 °C degrees and 22.0 % humidity and 101.2 kpa.

Data updated from weather monitor center.
Current time: Wed Aug 05 13:53:20 CST 2020
Current conditions: 27.0 °C degrees and 25.0 % humidity and 100.3 kpa.

Process finished with exit code 0
```

V.Evaluation

prons

1.主题与观察者之间是抽象耦合的。
2.每次主题更新，可以根据自己的需求弹性地通知观察者。

cons

1.若一个被观察者对象有很多直接或间接观察者，通知耗时会随之增加。
2.若观察者与被观察对象之间存在循环依赖，观察目标会触发它们之间进行循环调用，导致系统崩溃。
3.观察者模式没有相应的机制让观察者知道所观察的目标对象是怎么发生变化的，仅仅知道发生了变化。

VI.Scenarios

1.一个抽象模型有两个方面，一个方面依赖于另一个方面。将它们封装在独立的对象中使它们可以独自地改变和复用。
2.一个对象的改变将导致其他一个或多个对象将发生改变，可以降低对象之间的耦合度。
3.一个对象必须通知其他对象，而并不知道这些对象是谁。
4.需要在系统中建立一套触发机制，使得依赖之间的行为影响具有传递性。

Attention: MVC模型中大量运用Observer Mode.