观察者模式

问题：现在有一个气象站的需求，已知一个WeatherData类，其中有getTemperature(),getHumidity(),getPressure()方法和一个measurementChaned()，这个方法在气象数据发生变化时会自动调用

现在需要根据这个类来设计三个布告板，目前状况，气象统计，天气预报

方法：

最直观的解决方法如下

public class WeatherData {

public void measurementsChanged(){

float temp = getTemperature();

float humidity = getHumidity();

float pressure = getPressure();

currentCoditionsDisplay.update(temp,humidity,pressure);

statisticsDisplay.update(temp,humidity,pressure);

forecaseDisplay.update(temp,humidity,pressure);

}

}

先了解观察者模式：定义了对象之间的一对多的依赖，当一个对象状态改变时，他所依赖者都会受到通知并自动更新

为此想象两类对象，一类是主题相当于报社，一类是观察者相当于订阅报纸的人，当报社有新报纸时会送到订阅报纸的人手上，将注意力转移到气象站问题上，气象站相当于主题，各个面板相当于订阅者

首先气象站得有订阅者列表，当有新数据读取时，按照列表传送给订阅者

//主题

public class WeatherData {

float temp = getTemperature();

float humidity = getHumidity();

float pressure = getPressure();

public void measurementsChanged(){

for(订阅者列表){

订阅者的接口,参数为气象数据类似于：

订阅者.update(temp,humidity,pressure)

}

}

}

//观察者1

public class currentCoditionsDisplay{

float temp;

float humidity;

flaot pressure;

public void update(temp,humidity,pressure){

temp = temp;

humidity = humidity ;

pressure = pressure ;

//处理算法并显示

printf(“new data…\n”);

}

}

//观察者2

public class statisticsDisplay{

float temp;

float humidity;

flaot pressure;

public void update(temp,humidity,pressure){

temp = temp;

humidity = humidity ;

pressure = pressure ;

//处理算法并显示

printf(“new data…\n”);

}

}

根据上面大致的三个类，进一步补充内容

1:观察者1，2共性很多，将其独立出来为接口

2:主题的订阅者列表如何形成？

interface observer{

public abstract update();

}

//主题

public class WeatherData {

float temp = getTemperature();

float humidity = getHumidity();

float pressure = getPressure();

public void measurementsChanged(){

for(订阅者列表){

订阅者的接口,参数为气象数据类似于：

订阅者.update(temp,humidity,pressure)

}

}

//订阅者列表的获取，现在问题是订阅者都是用observer引用的，所有的订阅者要以observer类为父类或者为接口

public void registerobservers(observer ob){

arraylist.add(ob);

}

public void unregisterobservers(observer ob){

arraylist.del(ob);

}

}

//观察者1

public class currentCoditionsDisplay implements observer{

float temp;

float humidity;

flaot pressure;

public void update(temp,humidity,pressure){

temp = temp;

humidity = humidity ;

pressure = pressure ;

//处理算法并显示

printf(“new data…\n”);

}

}

//观察者2

public class statisticsDisplay implements observer{

float temp;

float humidity;

flaot pressure;

public void update(temp,humidity,pressure){

temp = temp;

humidity = humidity ;

pressure = pressure ;

//处理算法并显示

printf(“new data…\n”);

}

}

进一步优化

将主题和观察者分别进一步优化，设置成类似模块的方式

主题的通用方法独立出来作为接口

//主题--------------------------------------------------------------------------

interface subject{

public abstract void registerobservers(observer ob);

public abstract void unregisterobservers(observer ob);

}

public class WeatherData implements subject {

float temp = getTemperature();

float humidity = getHumidity();

float pressure = getPressure();

public void measurementsChanged(){

for(订阅者列表){

订阅者的接口,参数为气象数据类似于：

订阅者.update(temp,humidity,pressure)

}

}

//订阅者列表的获取，现在问题是订阅者都是用observer引用的，所有的订阅者要以observer类为父类或者为接口

public void registerobservers(observer ob){

arraylist.add(ob);

}

public void unregisterobservers(observer ob){

arraylist.del(ob);

}

}

//观察者--------------------------------------------------------------------------

interface observer{

public abstract update();

}

//观察者1

public class currentCoditionsDisplay implements observer{

float temp;

float humidity;

flaot pressure;

public void update(temp,humidity,pressure){

temp = temp;

humidity = humidity ;

pressure = pressure ;

//处理算法并显示

printf(“new data…\n”);

}

}

//观察者2

public class statisticsDisplay implements observer{

float temp;

float humidity;

flaot pressure;

public void update(temp,humidity,pressure){

temp = temp;

humidity = humidity ;

pressure = pressure ;

//处理算法并显示

printf(“new data…\n”);

}

}

问题似乎得到解决，但是还有个问题，就是不同的观察者需要得到的数据不同，有的只需要一个数据，有的需要多个数据，就跟订阅不同的报纸一样,这个问题暂不讨论，读者可以自行解决，提示在注册的时候指定需要订阅的类型即可

完整的例子见cpp文件