**[观察者模式及Java实现例子](http://www.cnblogs.com/mengdd/archive/2013/02/07/2908929.html)**

**观察者模式**

**观察者模式 Observer**

　　观察者模式定义了一种一对多的依赖关系，让多个观察者对象同时监听某一个主题对象。

　　这个主题对象在状态上发生变化时，会通知所有观察者对象，让它们能够自动更新自己。

**观察者模式的组成**

**抽象主题角色**：把所有对观察者对象的引用保存在一个集合中，每个抽象主题角色都可以有任意数量的观察者。抽象主题提供一个接口，可以增加和删除观察者角色。一般用一个抽象类和接口来实现。

**抽象观察者角色**：为所有具体的观察者定义一个接口，在得到主题的通知时更新自己。

**具体主题角色**：在具体主题内部状态改变时，给所有登记过的观察者发出通知。具体主题角色通常用一个子类实现。

**具体观察者角色**：该角色实现抽象观察者角色所要求的更新接口，以便使本身的状态与主题的状态相协调。通常用一个子类实现。如果需要，具体观察者角色可以保存一个指向具体主题角色的引用。

**程序实例**

　　通过程序实例来说明观察者模式：

　　首先定义抽象的观察者：

[复制代码](javascript:void(0);)

//抽象观察者角色

public interface Watcher

{

public void update(String str);

}

[复制代码](javascript:void(0);)

　　然后定义抽象的主题角色，即抽象的被观察者，在其中声明方法（添加、移除观察者，通知观察者）：

[复制代码](javascript:void(0);)

//抽象主题角色，watched：被观察

public interface Watched

{

public void addWatcher(Watcher watcher);

public void removeWatcher(Watcher watcher);

public void notifyWatchers(String str);

}

[复制代码](javascript:void(0);)

　　然后定义具体的观察者：

[复制代码](javascript:void(0);)

public class ConcreteWatcher implements Watcher

{

@Override

public void update(String str)

{

System.out.println(str);

}

}

[复制代码](javascript:void(0);)

　　之后是具体的主题角色：

[复制代码](javascript:void(0);)

import java.util.ArrayList;

import java.util.List;

public class ConcreteWatched implements Watched

{

// 存放观察者

private List<Watcher> list = new ArrayList<Watcher>();

@Override

public void addWatcher(Watcher watcher)

{

list.add(watcher);

}

@Override

public void removeWatcher(Watcher watcher)

{

list.remove(watcher);

}

@Override

public void notifyWatchers(String str)

{

// 自动调用实际上是主题进行调用的

for (Watcher watcher : list)

{

watcher.update(str);

}

}

}

[复制代码](javascript:void(0);)

　　编写测试类：

[复制代码](javascript:void(0);)

public class Test

{

public static void main(String[] args)

{

Watched girl = new ConcreteWatched();

Watcher watcher1 = new ConcreteWatcher();

Watcher watcher2 = new ConcreteWatcher();

Watcher watcher3 = new ConcreteWatcher();

girl.addWatcher(watcher1);

girl.addWatcher(watcher2);

girl.addWatcher(watcher3);

girl.notifyWatchers("开心");

}

}

[复制代码](javascript:void(0);)