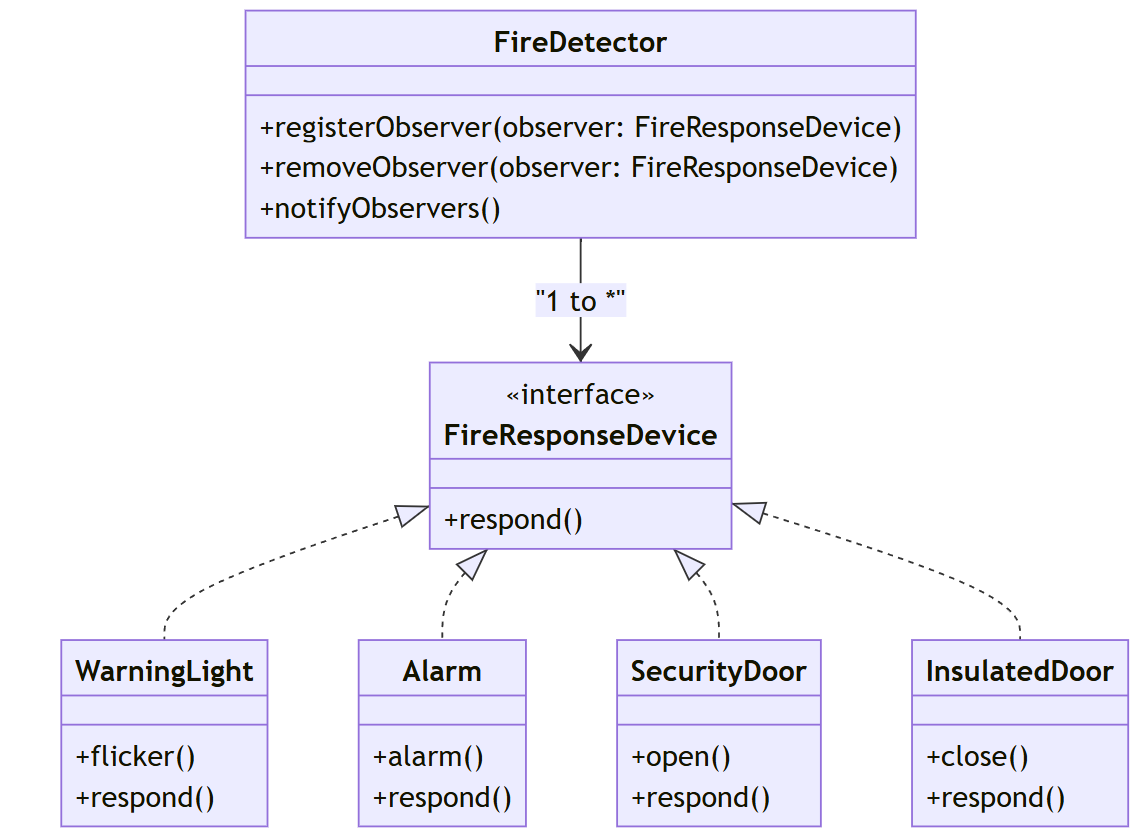
**题目：**

开发一个消防应急响应系统，火灾探测器(FireDetector)发现火灾异常后将自动传递信号给各种响应设备，例如警示灯(WarningLight)将闪烁(flicker())、报警器(Alarm)将发出警报(alarm())、安全逃生门(SecurityDoor)将自动开启(open())、隔离门(InsulatedDoor)将自动关闭(close())等，每一种响应设备的行为由专门的程序来控制。请写出你所选择的设计模式，画出类图，并给出核心代码。

**模式：**

观察者模式

**类图：**

****

|  |
| --- |
| **代码** |
| import java.util.ArrayList;  import java.util.List;  *// 观察者接口*  interface FireResponseDevice {      void respond();  }  *// 火灾探测器类*  class FireDetector {      private List<FireResponseDevice> devices = new ArrayList<>();      public void registerObserver(FireResponseDevice device) {          devices.add(device);      }      public void removeObserver(FireResponseDevice device) {          devices.remove(device);      }      public void notifyObservers() {          for (FireResponseDevice device : devices) {              device.respond();          }      }      public void detectFire() {          System.out.println("Fire detected! Notifying response devices...");          notifyObservers();      }  }  *// 警示灯类*  class WarningLight implements FireResponseDevice {      @Override      public void respond() {          flicker();      }      public void flicker() {          System.out.println("Warning light is flickering!");      }  }  *// 报警器类*  class Alarm implements FireResponseDevice {      @Override      public void respond() {          alarm();      }      public void alarm() {          System.out.println("Alarm is sounding!");      }  }  *// 安全逃生门类*  class SecurityDoor implements FireResponseDevice {      @Override      public void respond() {          open();      }      public void open() {          System.out.println("Security door is opening!");      }  }  *// 隔离门类*  class InsulatedDoor implements FireResponseDevice {      @Override      public void respond() {          close();      }      public void close() {          System.out.println("Insulated door is closing!");      }  }  *// 测试类*  public class FireEmergencySystem {      public static void main(String[] args) {  *// 创建火灾探测器*          FireDetector detector = new FireDetector();  *// 创建响应设备*          FireResponseDevice warningLight = new WarningLight();          FireResponseDevice alarm = new Alarm();          FireResponseDevice securityDoor = new SecurityDoor();          FireResponseDevice insulatedDoor = new InsulatedDoor();  *// 注册响应设备*          detector.registerObserver(warningLight);          detector.registerObserver(alarm);          detector.registerObserver(securityDoor);          detector.registerObserver(insulatedDoor);  *// 模拟火灾探测*          detector.detectFire();      }  } |