Observer Pattern

The **Observer Pattern** is a **behavioural pattern** that is used for cases where we need a **subscription mechanism**, i.e., there is a **publisher** or **subject** that needs to be observed, which must notify the **observers** when some event, such as a change, occurs. Thus, the objects have a **one-to-many** relationship.

The pattern is quite simple.

1. The subject stores the list of observers and provides methods for observers to add or remove themselves from the list.
2. Each observer implements a common interface which includes a method which the subject can call. This method will result in the observer retrieving the new information from the subject.
3. Upon any change, the subject goes over the list of observers and notifies each one.

import java.util.ArrayList;  
import java.util.*List*;  
  
public class Subject {  
 private *List*<Observer> observers = new ArrayList<>();  
 private int state;  
  
 public int getState() {  
 return state;  
 }  
  
 public void setState(int state) {  
 this.state = state;  
 notifyAllObservers();  
 }

public void attach(Observer observer) {  
 observers.add(observer);  
 }  
  
 public void notifyAllObservers() {  
 for (Observer observer : observers) {  
 observer.update();  
 }  
 }  
}  
  
public abstract class Observer {  
 protected Subject subject;  
 public abstract void update();  
}  
  
public class BinaryObserver extends Observer {  
 public BinaryObserver(Subject subject) {  
 this.subject = subject;  
 subject.attach(this);  
 }  
  
 @Override  
 public void update() {  
 int state = subject.getState();  
 String binaryState = Integer.*toBinaryString*(state);  
 System.*out*.println(binaryState);  
 }  
}  
  
public class OctalObserver extends Observer {  
 public OctalObserver(Subject subject) {  
 this.subject = subject;  
 subject.attach(this);  
 }  
  
 @Override  
 public void update() {  
 int state = subject.getState();  
 String octalState = Integer.*toOctalString*(state);  
 System.*out*.println(octalState);  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 Subject subject = new Subject();  
  
 new BinaryObserver(subject);  
 new OctalObserver(subject);  
  
 subject.setState(15);  
 // Two lines printed:  
 // 1111  
 // 17  
 }  
}

JAVA

Notice that in the main code, we do not even need to keep track of the objects we created (although that is a terrible idea for other reasons). The print statements execute automatically.