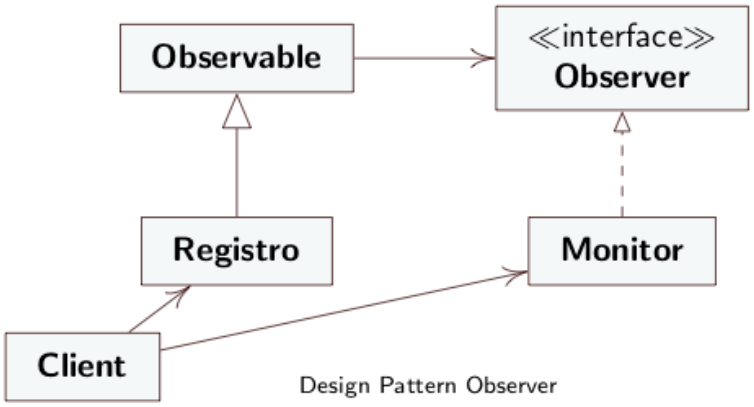
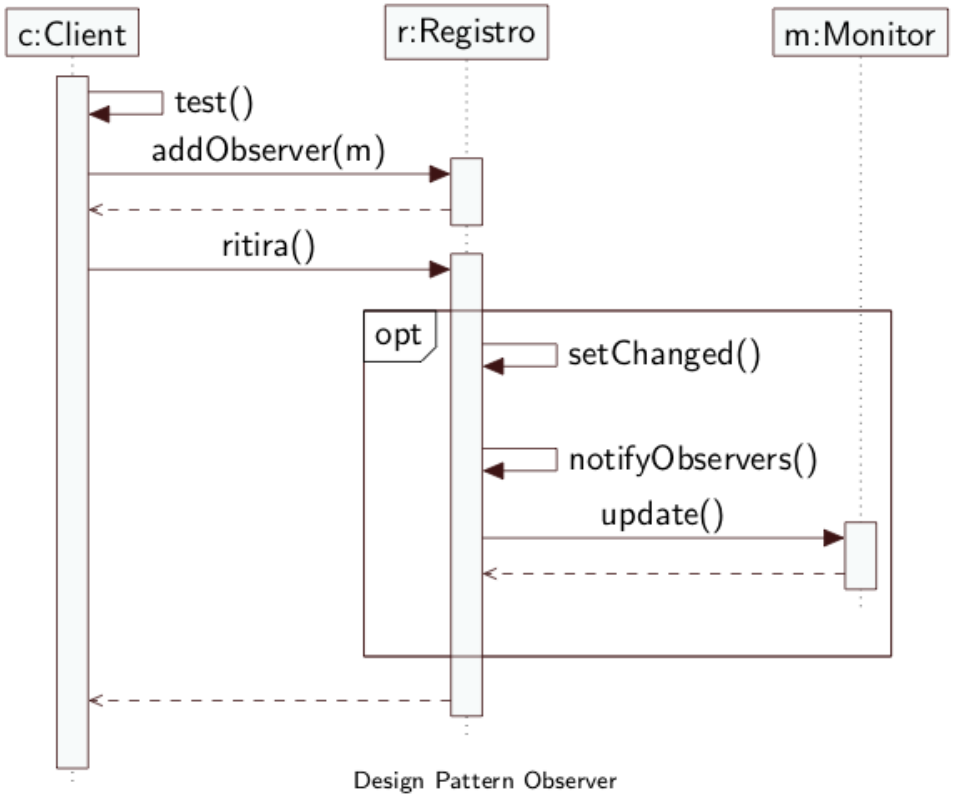


Design Pattern Observer



Design Pattern Observer

```java
// Codice Java che implementa il design pattern Observer

// Monitor e' un ConcreteObserver
import java.util.Observer;
import java.util.Observable;

public class Monitor implements Observer {
   private int numOperazioni = 0;

   @Override public void update(Observable obs, Object s) {
      System.out.println("Monitor: Il saldo e' di euro "+s);
      numOperazioni++;
   }
   public int getNumOperazioni() {
      return numOperazioni;
   }
}


// Registro e' un ConcreteSubject
import java.util.Observable;

public class Registro extends Observable {
   private double bilancio = 0;

   public boolean ritira(double somma) {
      System.out.println("Prelievo di " + somma + " euro");
      if (bilancio >= somma) {
         bilancio -= somma;
         setChanged();
         notifyObservers(bilancio);
         return true;
      }
      System.out.println("Operazione non permessa");
      return false;
   }
   public void deposita(double somma) {
      System.out.println("Deposito di " + somma + " euro");
      bilancio += somma;
      setChanged();
      notifyObservers(bilancio);
   }
   public double getBilancio() {
      return bilancio;
   }
}


// Classe con il main che usa il design pattern
public class Client {
   private Registro r = new Registro();
   private Monitor m = new Monitor();
```

```
    public static void main(String[] args) {
        Client c = new Client();
        c.test();
    }

    private void test() {
        r.addObserver(m);
        r.ritira(500);

        r.deposita(1200);
        System.out.println("Saldo attuale: " + r.getBilancio() + " euro");
        System.out.println("Il subject ha effettuato " + m.getNumOperazioni() + "
    }
}
```

Output dell'esecuzione

```
Prelievo di 500.0 euro
Operazione non permessa
Deposito di 1200.0 euro
Monitor: Il saldo e' di euro 1200.0
Saldo attuale: 1200.0 euro
Il subject ha effettuato 1 operazioni
```