Le design pattern Obserser en java

Le Design pattern Observer/Observable permet de coupler des couches logicielles en limitant leurs interactions aux seuls phénomènes observés. C'est un Design pattern utile, par exemple, pour implémenter dans le modèle MVC, les liens entre Vues et Modèle.

- La vue sera un observateur du Model.
- Le model sera observable, et permettra d'enregistrer des écouteurs (observateurs).
- La vue pourra écouter les modifications du/des modèles, et se modifier en conséquence

Exemple MVC : Liens Vue/modèle

-- Le modèle

- Le modèle hérite de Observable
- Lors de la modification de l'un de ses membres :
  - on spécifie que l'objet a changé (**setChanged**)
  - on notifie les observers du changement (**notifyObservers**)

```
public class Model extends Observable {
    private String member;

    /**
     * @return the member
     */
    public String getMember() {
        return member;
    }

    public Model() {
        super();
    }

    /**
     * @param member
     *            the member to set
     */
    public void setMember(String member) {
        this.member = member;
        this.setChanged();
        notifyObservers(member);
    }
}
```

Une vue

- La vue implémente Observer, et doit à ce titre implémenter la méthode **update**
- **update** pourra traiter les changements en fonction des paramètres passés : **Observable o** correspond à l'objet modifié

```
public class View implements Observer {

    @Override
    public void update(Observable o, Object arg) {
        System.out.println(o);
        System.out.println(arg);
    }

}
```

--Exemple de programme

```
public class Program {

    /**
     * @param args
     */
    public static void main(String[] args) {
        View v = new View();
        Model m = new Model();
        m.addObserver(v);
        m.setMember("essai");
    }

}
```

From:
http://slamwiki2.kobject.net/ - **SlamWiki 2.1**

Permanent link:
http://slamwiki2.kobject.net/slam4/poo/designpattern/observer?rev=1365590767

Last update: **2019/08/31 14:39**