TPN°2 : Le pattern observer

Le patron de conception **observateur/observable** est utilisé en programmation pour envoyer un signal à des modules qui jouent le rôle **d'observateur.** En cas de notification, les observateurs effectuent l'action adéquate en fonction des informations qui parviennent depuis les modules qu'ils observent (les "**observables**").

La notion **observateur/observable** permet de coupler des modules de façon à réduire les dépendances aux seuls phénomènes observés.

1. Exemple d'illustration

Prenons comme exemple une classe ***Horloge*** (données ***observables***), visualisée à travers des panneaux (***observateur***) d'une interface graphique. On souhaite que la mise à jour d'une horloge modifie le panneau qui l'affiche. Afin d'éviter l'utilisation de thread ou encore d'inclure la notion de panneau dans l'Horloge (couplage fort) il suffit d'utiliser le patron de conception **observateur/observable.**

Le couplage entre objets est l'un des problèmes principaux concernant la réutilisabilité d'objets...Dans notre cas, si vous voulez utiliser votre objet **Horloge** ailleurs, vous serez confrontés à pas mal de problèmes puisque cet objet ne s'affichera que dans l'interface conçue pour ce dernier (JPanel)

2. L'apport du pattern observer

- Il permet de faire communiquer des objets entre eux ;
- C'est un bon moyen d'éviter le couplage entre les objets ;
- Les objets ne sont plus liés par leur type respectif, mais par leurs interfaces.

Le travail à faire

Partie 1 : Créer une interface graphique permettant d'afficher l'heure.

La classe horloge doit hériter de la classe ***Thread***.

Pour récupérer l'heure courante utilisez l'Objet ***Calendar***.

- Pour récupérer l'heure, utilisez la méthode : ***get(Calendar.HOUR_OF_DAY)***

- Pour récupérer les minutes, utilisez la méthode : ***get(Calendar.MINUTE)***
- **…**

Partie 2 :

Développez la même application en utilisant le **pattern Observer**.

Les étapes à suivre pour implémenter l'horloge avec le **Pattern Observer** :

- L'utilisation du **Pattern Observer** nécessite l'ajout des packages ; **com.sdz.model ; package com.sdz.vue**
- Créer les classes ***Horloges, Fenêtre.***
- Créer les interfaces ***Observateur, Observable.***
- La classe ***Fenêtre*** implémente l'interface ***Observateur***.
- La classe ***Horloge*** implémente l'interface ***Observable***.

Voici les deux classes **Observateur** et **Observable.**