Sujet 5 : DP observateur

Vous devez réaliser un programme de gestion d'abonnements à un quotidien. Un abonné choisit de s'abonner au quotidien et choisit de recevoir son journal de différentes manières (abonnement papier, abonnement électronique, abonnement n'affichant que les notifications des titres des articles du journal).

Lorsqu'un nouveau numéro paraît :

— l'abonné à la version papier reçoit une information lui indiquant d'aller chercher son exemplaire du journal dans sa boîte aux lettres (avec affichage du numéro) ;
— l'abonné à la version électronique reçoit une information avec le numéro, le titre et l'article de une ;
— l'abonné aux notifications reçoit une information avec le numéro et le titre de la une seulement.

Chaque fois qu'un exemplaire du journal paraît, l'abonné est informé et récupère les informations selon le mode choisi. Si l'abonné décide d'arrêter son abonnement, il n'est plus informé des nouvelles parutions.

Exercice 1 Une solution personnelle

1. Modéliser ce problème en utilisant le DP Observateur vu en cours.

2. Écrire le code des deux interfaces `ISujet` et `IObservateur`.

3. Coder ensuite la classe `Quotidien`. On trouvera dans le quotidien un numéro, le titre de la une et le contenu de l'article de une.

4. Coder enfin les trois classes représentant les affichages pour les abonnés.

5. Écrire une classe permettant de tester votre code.

6. Il existe une autre façon de faire dans laquelle la méthode `actualiser(ISujet sujet)` admet comme paramètre les données actualisées à traiter par les abonnés et pas le sujet lui-même. Implanter et tester cette deuxième méthode. Quels sont les avantages et les inconvénients de chacune des deux méthodes ?

Exercice 2 Une solution qui utilise l'interface *PropertyChangeListener*

Ce design pattern est très présent dans l'écriture des interfaces graphiques.Il existe dans le langage Java sous la forme de l'interface *PropertyChangeListener* qui peut être utilisée pour implanter ce que nous avons appelé dans le cours l'interface *IObservateur*.

1. Regarder la documentation de cette interface. Quelle est l'unique méthode qu'elle définit ? À quoi correspond-elle dans le cours ?

2. Regarder la documentation de la classe `PropertyChangeEvent`. Que permet-elle ?

3. Coder la classe `AbonnePapier` qui réalise cette interface.

4. Coder maintenant une classe abstraite `Observable` qui définit les trois méthodes permettant de gérer une liste d'abonnés.

5. Coder la classe `Quotidien` qui hérite d'`Observable` et la spécialise dans le cas d'un quotidien.

6. Écrire une classe permettant de tester votre code.

7. Regarder le diagramme de classe obtenu. Quel est l'inconvénient d'avoir défini `Observable` comme classe ?