TP – Héritage & Interface

L'objectif de ce TP est de simuler une maison connectée contenant un ensemble de capteurs et des dispositifs connectés. Les capteurs émettent leur valeur à chaque fois que celle-ci est mise à jour. Les dispositifs connectés peuvent « s'abonner » aux capteurs qui les intéressent et ainsi réagir en fonction des valeurs envoyées.

Voici un exemple que nous mettrons en œuvre dans ce TP :
Les capteurs présents sont des capteurs de luminosité et de température. Les dispositifs connectés sont une station météo, les volets roulants.
La station météo affiche les valeurs des différents capteurs. Les volets roulants seront automatiquement ouverts si la luminosité est trop faible, et inversement seront fermés en cas de trop forte intensité.

Pour que les capteurs puissent envoyer une information à leurs abonnés, il faut que tous les abonnés aient une méthode commune à chacun d'eux par laquelle les informations pourraient transiter.
Une interface permettra de définir cette méthode. Toutes les classes qui souhaiteront s'abonner à, au moins, un capteur devront alors implémenter cette interface.

1- Créez une interface **Abonne** qui aura une méthode *actualiserEtat* qui prendra en paramètre un objet de type **Capteur**.

Les capteurs, pour pouvoir communiquer leur valeur, doivent gérer la liste de leurs abonnés. A chaque mise à jour de leur valeur, cette information sera communiquée à l'ensemble des abonnées via la méthode *actualiserEtat*.

2- Créez une classe **Capteur** qui aura comme attributs un nom pour le capteur (type String), une liste de diffusion (type ArrayList) contenant les abonnés. Cette classe doit avoir les méthodes :
   - *ajouterAbonne*(Abonne a) qui permet d'ajouter un Abonne à la liste de diffusion
   - *envoyerNotification*() qui transmet à tous les abonnés le nouvel état du capteur

La classe Capteur est une classe générique qui regroupe les informations et fonctionnalités communes à chaque type de capteur. Mais la classe capteur n'est pas un capteur en soi (elle n'a pas de valeur).

3- Créez les classes **CapteurLuminosite** et **CapteurTemperature** qui héritent de la classe Capteur et qui auront respectivement un attribut intensité (type int) et un attribut température (type float). Pour pouvoir mettre à jour et accéder à ces valeurs, les classes auront des méthodes :
   - *setData*(Object o)
   - Object *getData*()

Ces deux méthodes sont propres aux capteurs. Par conséquent, la classe Capteur qui est générique, doit être une classe abstraite.

4- Modifiez la classe Capteur pour que ce soit une classe abstraite avec les méthodes setData et getData comme méthodes abstraites.
5- Créez la classe **StationMeteo** qui affiche au fur et à mesure dans la console les informations reçues
6- Créez la classe **VoletRoulant** qui a un attribut indiquant si les volets sont ouverts ou non. En fonction des valeurs reçues par les capteurs d'intensité, si l'intensité est supérieure à 100 et que les volets sont ouverts, les volets seront alors fermés. Inversement, si les volets sont fermés et que l'intensité est inférieure à 20, les volets seront ouverts. *Remarque : L'ouverture et la fermeture des volets seront simplement représentées par un message dans la console.*
7- Pour tester l'ensemble, créez une classe Maison. Dans la méthode main, créez des objets de types CapteurLuminosité, CapteurTemperature, StationMeteo et VoletRoulant. La station météo sera abonnée à tous les capteurs. Les volets roulants seront abonnés au capteur de luminosité. Créez un petit menu pour pouvoir allouer une nouvelle valeur à un capteur de température ou à un capteur de luminosité.

Bonus : Créez une classe AlerteCanicule qui doit pouvoir recevoir les informations de température. Si la température est supérieure à 30, un message d'alerte canicule sera affiché dans la console et les volets seront fermés.