TD 1 Design Patterns
adaptateur, observateur

Ce TD a pour objectif de vous exercer à passer de Java à UML et vice-versa. Il introduit les premiers modèles de conception de façon informelle.

Pour les diagrammes UML, vous pouvez utiliser un outil ou les dessiner à la main. Pour le code Java, vous pouvez utiliser l'environnement de développement de votre choix mais les corrections seront fournies sous forme de projet Eclipse.

Exercice 1 (de Java à UML)

Observer le code Java fourni.

1) *Fournir le diagramme de classe UML associé.*

2) *Fournir le diagramme de séquence UML correspondant à la méthode MainClass.main.*

Exercice 2 (Améliorer la conception)

Pour cet exercice, nous repartons du diagramme de classes construit à la question 2 de l'exercice 1.

1) *Identifier le modèle de conception sous-jacent.*

2) *Fournir le diagramme de classe en appliquant le modèle de conception identifié (ne pas oublier l'interface et la classe d'évènement).*

3) *Introduire dans le diagramme une classe TemperatureAlarm qui affiche un message d'alerte si la température est inférieure à 4 et supérieure à 18.*

4) *Coder le diagramme de classe. Ajouter une alarme sur la sonde de température de la cave. Faire fonctionner le programme principal et faisant chuter la température de la cave à 3°.*

Exercice 3 (Intégrer du code existant)

Vous venez de récupérer un composant existant : le thermomètre. Son fonctionnement est différent de celui de la sonde puisque le positionnement de la température se fait en augmentant ou diminuant la température courante. Voici le code du thermomètre :

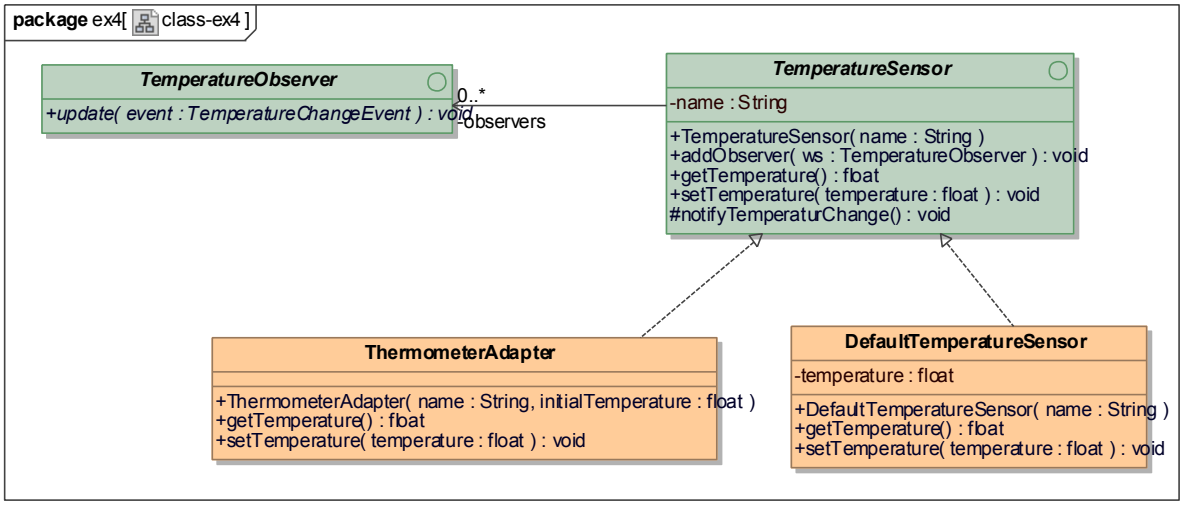
```
public class Thermometer
{
	private float temperature;

	public Thermometer(float initialTemperature)
	{
		this.temperature = initialTemperature;
	}

	public void increaseTemperature(float value)
	{
		temperature += value;
		temperature += 0.1; // moins précis le thermomètre !
	}

	public void decreaseTemperature(float value)
	{
		temperature -= value;
		temperature -= 0.1; // moins précis le thermomètre !
	}

	public float getTemperature()
	{
		return temperature;
	}
}
```

1) *Proposer des modifications pour que le code du thermomètre s'adapte comme un capteur de température dans notre système ?*

2) *Identifier les limites d'une telle adaptation et utiliser un modèle de conception pour éviter ce problème.*

3) *Fournir le diagramme de classe et le code Java correspondant.*

Exercice 4 (Problème de « mapping »)



1) *Quelles modifications doit subir le programme Java réalisé à l'exercice précédent pour répondre au diagramme de classes ci-dessus ?*