

Exercices de Patterns (6)

Une autre fonction de notre machine à café est la sélection de la boisson. La technique consiste à choisir une boisson de base (Moka, Dessert ou Décaféiné) et d'y ajouter éventuellement du lait, du sucre et/ou de la crème fraîche. Un programme utilisant cette sélection pourra être :

```
public class Main {
    public static void main(String[] args) {
        Boisson b = new Lait(new Sucre(new Décaféiné()));
        System.out.println(b + " au prix de " + b.prix() + " euros.");
        b = new Sucre(new Crème(new Dessert()));
        System.out.println(b + " au prix de " + b.prix() + " euros.");
    }
}
```

Ce programme donnera en sortie :

Café décaféiné sucré au lait au prix de 0.55 euros.
Café dessert crème sucré au prix de 0.57 euros.

Les classes sont implémentées comme suit :

Boisson.java :

```
public interface Boisson {
    public abstract String getNom();
    public abstract double prix();
}
```

Café.java :

```
public abstract class Café implements Boisson {
    private String nom;
    public Café(String nom) {
        this.nom = nom;
    }
    public String getNom() {
        return nom;
    }
    public abstract double prix();
    public String toString() {
        return "Café " + getNom();
    }
}
```

Dessert.java :

```
public class Dessert extends Café {
    public Dessert() {
        super("dessert");
    }
    public double prix() {
        return 0.37;
    }
}
```

Moka.java :

```
public class Moka extends Café {
    public Moka() {
        super("moka");
    }
    public double prix() {
        return 0.35;
    }
}
```

Décaféiné.java :

```
public class Décaféiné extends Café {
    public Décaféiné() {
        super("décaféiné");
    }
    public double prix() {
        return 0.40;
    }
}
```

Exercices de Patterns (6)

```
}  
}
```

Additif.java :

```
public abstract class Additif implements Boisson {  
    private Boisson boisson;  
    public Additif(Boisson boisson) {  
        if (boisson == null) throw new IllegalArgumentException();  
        this.boisson = boisson;  
    }  
    public String getNom() {  
        return boisson.getNom();  
    }  
    public double prix() {  
        return boisson.prix();  
    }  
    public String toString() {  
        return boisson.toString();  
    }  
}
```

Crème.java :

```
public class Crème extends Additif {  
    public Crème(Boisson boisson) {  
        super(boisson);  
    }  
    public double prix() {  
        return 0.15 + super.prix();  
    }  
    public String toString() {  
        return super.toString() + " crème";  
    }  
}
```

Lait.java :

```
public class Lait extends Additif {  
    public Lait(Boisson boisson) {  
        super(boisson);  
    }  
    public double prix() {  
        return 0.10 + super.prix();  
    }  
    public String toString() {  
        return super.toString() + " au lait";  
    }  
}
```

Sucre.java :

```
public class Sucre extends Additif {  
    public Sucre(Boisson boisson) {  
        super(boisson);  
    }  
    public String toString() {  
        return super.toString() + " sucré";  
    }  
    public double prix() {  
        return 0.05 + super.prix();  
    }  
}
```

Exercices de Patterns (6)

Nom du Pattern :

Participants théoriques

Noms utilisés dans le code

Nom théorique des méthodes

Nom utilisé dans le code