JAVA
FONDAMENTAUX

Présenté par :

Xavier TABUTEAU

Les interfaces

Une interface peut représenter un point d'échange entre un fournisseur et un ou plusieurs clients (contrat).
L'interface comporte uniquement les déclarations des méthodes présentes (il n'y a pas de code interne aux méthodes) et ne peut pas comporter des variables. Par contre, on pourra définir des constantes ou des énumérations pour créer des constantes dans l'interface.

La classe qui implémente l'interface, définit les méthodes de celle-ci.
Par convention, on nomme l'interface avec un « I » en préfixe.

L'utilisation d'une interface permet de créer une dépendance faible entre l'interface et la classe qui l'implémente. Une classe qui appelle une autre classe, crée une dépendance forte à cette classe.

Syntaxe d'une interface :

```
interface INomInterface {
        type nomMethode1(paramètres);
        type nomMethode2();
        …
}
```

Les interfaces

Exemple avec métaphore :

Plusieurs fabricants décident de proposer une façade de machine à laver standard.

L’objectif est de retrouver les mêmes fonctionnalités au même endroit sur n’importe qu’elle machine à laver.

Les fabricants définissent l’interface de la façade, qu’il peuvent donner à un sous-traitant pour la fabrication (plan, modèle).

Le sous-traitant va savoir quoi faire pour construire les façades, sans connaître les fonctionnalités propre à chaque fabriquant (démarrage, arrêt, programmes, ouverture, …)

Chaque fabricant pourra utiliser la façade pour construire ses modèle de machine à laver.

→ La façade est l’interface (contrat).
→ la machine à laver propre à un fabricant est une implémentation de cette interface.

Les interfaces

Une interface qui n’a qu’une seule méthode abstraite est appelée une interface fonctionnelle. Java fournit une annotation @FunctionalInterface, qui est utilisée pour déclarer une interface comme interface fonctionnelle.

En Java, la classe qui implémentera cette interface devra utiliser la syntaxe suivante :

```
class MaClasse implements IMonInterface {
        …
}
```

Une classe qui implémente une interface doit implémenter toutes les méthodes de celle-ci. Sinon, il y a une erreur de compilation ou il faut déclarer la classe abstraite.

Une même classe peut hériter d’une autre classe et implémenter une ou plusieurs interfaces.
Mais une classe ne peut pas hériter de plusieurs classes. L’héritage multiple n’est pas autorisé en Java.

Remarque : Une interface peut héritée d’une autre interface en utilisant la même syntaxe que celle utilisée pour les classes.

Les interfaces

Depuis la version 8 de Java, il est possible d'ajouter des méthodes statiques dans une interface et des méthodes « default ».

Les méthodes « default » et « static » ont pour particularité d'être définie dans l'interface et pourront être utilisées directement (ou redéfinie) par la classe qui implémente l'interface.

Lors d'une création d'une interface dérivée de l'interface contenant une méthode « default » on peut :

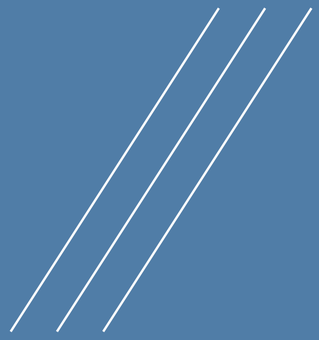
- Ne rien mentionner : l'interface hérite de la méthode par défaut.
- Redéfinir la méthode
- Redéclarer la méthode : la méthode devient abstraite.

Les méthodes « static » sont définies dans l'interface comme les méthodes « default » mais elles ne peuvent pas être redéclarées lors de la création d'une interface dérivée. Par conséquent, elles ne peuvent pas être abstraites. En effet, il n'est pas possible en Java de créer des méthodes statiques abstraites.

Remarque : Les classes abstraites et les interfaces se ressemblent. La principale différence est que les interfaces ne peuvent pas avoir d'état (variables d'instances) ni de constructeur. Elles ne peuvent avoir que de constantes.

InterfacesMultiples.java
InterfacesDerivees.java

JAVA FONDAMENTAUX



Présenté par

Xavier TABUTEAU