DUT Informatique S3 - Programmation Java

Jean-Claude MARTIN – MARTIN@LIMSI.fr

Cours 5 : Classes abstraites et interfaces

Contexte : quel est le problème ?

En entreprise, un projet est généralement développé en équipe et sur parfois sur plusieurs mois / années avec maintenance.

Un développeur doit donc prévoir que ses programmes seront utilisés par quelqu'un d'autre que lui-même.

C'est par exemple le cas du JDK (environ 1000 classes qui sont utilisées par XX développeurs Java).

Classes abstraites

Définition

Une classe abstraite est une classe dans laquelle au moins une méthode n'est pas implémentée (cad n'a pas d'instructions).

Utilisation

Utiliser le mot-clé abstract devant le mot-clé class.

Pour les méthodes qui ne sont pas implémentées, spécifier juste leur entête.

But d'une classe abstraite

Une classe abstraite permet de définir les caractéristiques communes à plusieurs classes.

Une classe abstraite permet à un développeur :

- de fournir à d'autres développeurs une partie de l'implémentation d'une classe

- de laisser aux autres développeurs la manière d'implémenter le reste de la classe

- d'imposer aux autres développeurs d'implémenter certaines méthodes s'ils veulent pouvoir utiliser ses classes

Exemple de classe abstraite

Contexte

M. Pasquier est chef de projet et développeur.

Le but du projet est de développer un éditeur de dessin sur SmartPhone en Java.

Il ne peut pas tout developper tout seul et doit collaborer avec M. Dupont.

Répartition des tâches

M. Pasquier va prendre en charge la partie affichage graphique et stockage en mémoire de toutes les formes graphiques dans un tableau. Il devra notament écrire un programme affichant des statistiques avec toutes les formes crées classées par surface decroissante.

M. Dupont va prendre en charge la partie implémentation de différentes formes graphiques : des cercles, des rectangles ... Cette liste pourra évoluer au fur et à mesure du projet.

Répartition des classes à programmer

M. Pasquier écrit plusieurs classes :

- une classe générale EditeurGraphique qui lance le programme entier et inclue une méthode afficherStats qui affiche une fenetre graphique avec toutes les informations sur une forme graphique (notamment surface, …) passée en parametre à cette méthode.

- une classe abstraite FormeGeometrique qui comporte

deux méthodes implémentées : déplacer () et afficher ()

deux méthodes abstraites : perimetre () et surface ()

```
abstract public class FormeGeometrique {
…
}
```

M. Dupont développe la classe Rec:

```
public class Rec extends FormeGeometrique {
…
}

public class TestDupont{
…
}
```

Dans sa classe EditeurGraphique, M. Pasquier a mis des instructions du type :

```
class Editeur {
…
 }
```

M. Pasquier ne s'est pas pré-occupé du type de FormeGraphique.

M. Pasquier sait que pour utiliser ses classes, M. Dupont devra passer en paramètre à la méthode afficherStats() une instance d'une classe qui implémente les méthodes spécifiées comme abstraites dans la classe FormeGraphique.

Que se passe-t-il si M. Dupont n'implémente pas la classe abstraite FormeGraphique ?

Est-ce qu'il peut toujours utiliser les classes développées par M. Pasquier ? ……

De même, M. Dupont développe la classe Cercle :

```
public class Cercle
             extends FormeGeometrique {
  double rayon;
  Cercle(double x, double y, double r) {
    posX=x; posY=y; rayon=r;
  }
  double surface() {
    return Math.PI*Math.pow(rayon, 2.);
  }
  double perimetre() {
    return 2*rayon*Math.PI;
  }
}
```

Attention

Le compilateur n'accepte pas de créer des objets instance d'une classe abstraite.

Pourquoi : .....

Exemple du JDK

- classe Component : superclasse des objets qui ont une représentation graphique : Button, Scrollbar, Window (Cf Javadoc)

Interface

Définition

Une interface est une spécification complètement abstraite : aucune méthode n'est implémentée.

Une interface ne comporte que des constantes et des méthodes abstraites.

ATTENTION : le terme "interface" n'a rien à voir ici avec les "interfaces graphiques"

Utilisation

Une classe qui implémente une interface doit définir le corps (les instructions) de toutes ses méthodes abstraites.

Le mot-clef *implements* permet de rendre une classe conforme à une interface particulière (ou à un groupe d'interfaces). « L'interface spécifie ce à quoi la classe ressemble, mais maintenant on va spécifier comment cela fonctionne ».

But

Le but est le même que pour les classes abstraites mais à l'extrême : aucune implémentation n'est fournie !

Exemple 1

Version interface de FormeGeometrique.

Exemple 2

- MinMax
- MinMaxPaire
- MinMaxTriplet
- Mediane
- Application

Exemple du JDK

1) Set est une interface qui modèlise les fonctionnalités des ensembles :

- isEmpty() méthode qui détermine si l'ensemble est vide
- contains() détermine si un élément est dans l'ensemble
- containsAll() détermine si une collection d'élément est incluse dans l'ensemble
- add() ajoute un élément à l'ensemble
- ...

TreeSet et HashSet sont 2 classes qui implémente l'interface Set :

- TreeSet est un arbre d'éléments ordonnés selon l'ordre ascendant
- HashSet est une table d'élément accessible par une clef de recherche.

2) Comparable

- L'interface Comparable modélise les objets qui possèdent un ordre total :
- La seule méthode est compareTo(autre) qui renvoie un entier positif si l'objet est supérieur à l'autre, 0 s'ils sont égaux, et négatif sinon.
- Cette interface permet d'utiliser des collections d'objets ordonnés comme TreeSet

Applications à l'héritage et au polymorphisme

- tableau de formes graphiques dans l'éditeur : on peut parcourir le tableau et appeler sur les cases du tableau des méthodes de la classe abstraite / interface
- Héritage “multiple” : une classe C2 peut étendre une classe C1 et implémenter une interface I

Liens

- http://www.u-picardie.fr/~ferment/java/cours/chap12_d.html
- http://www.siteduzero.com/tutoriel-3-10373-les-classes-abstraites.html