

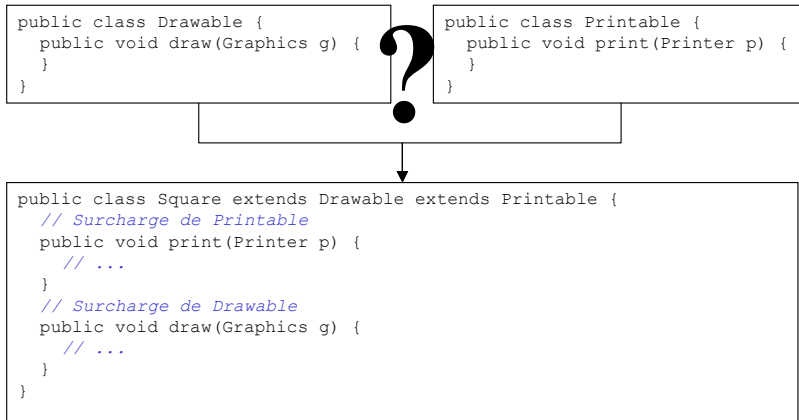
LIFI-Java 2004

Séance du Mercredi 6 sept.

Cours 5

Héritage Multiple (1/2)

- Peut-on dériver de deux classes?



Héritage Multiple (2/2)

- Peut-on dériver de deux classes? Non!
- Pourquoi?
 - comment décider quels code appliquer par défaut si deux méthodes ont le même nom?
 - des solutions dans d'autres langages (C++,...)
 - pas en Java!
- Pourtant on en a besoin?!?
 - il y a des objets printable et drawable!
 - comment organiser la hiérarchie d'objets?

Retour sur l'héritage (1/2)

- Pourquoi a-t-on introduit l'héritage?
 - pour “factoriser” du code
 - ex: le code pour avoir des numéros d'immatriculation sur tous les `Vehicle` que ce soit des `Car` ou des `Truck`
 - pour pouvoir surcharger une méthode par défaut
 - évolutivité sans changer le code source
 - pour pouvoir manipuler des objets “hétérogènes”

Retour sur l'héritage (2/2)

- Qu'est ce que permet en fait l'héritage?
 - définir une *interface*
 - indique qu'un objet sait faire certaines choses
 - ex: un objet est affichable et imprimable
 - faire varier les *implémentations*
 - évolutivité
 - polymorphisme
- Pas besoin d'une implémentation par défaut!

Interface vs. implémentation

- Pour travailler avec l'instance d'une classe:
 - ex: pouvoir écrire `o.draw()`
 - on doit:
 - vérifier qu'une méthode donnée existe
 - savoir ce que la méthode *est censée* faire
 - mais on a pas besoin de:
 - savoir *comment* elle le fait
 - savoir quelle fonction le fait
- Seule compte l'interface pas l'implémentation

Les interfaces en Java (1/2)

- Les mots clés `interface` et `implements`

```
public interface Drawable {  
    public void draw(Graphics g);  
}
```

```
public interface Printable {  
    public void print(Printer p);  
}
```

```
public class Square implements Drawable implements Printable {  
    // Implémentation de Printable  
    public void print(Printer p) {  
        // ...  
    }  
    // Implémentation de Drawable  
    public void draw(Graphics g) {  
        // ...  
    }  
}
```

Les interfaces en Java (2/2)

- On ne peut dériver que d'une classe
- On peut implémenter plusieurs interfaces
 - ça a du sens en terme d'objets
 - plus de problèmes de définition multiple
- C'est un choix "idéologique"
 - on pourrait s'en sortir avec les classes abstraites
 - ça met en évidence les concepts objets différents

Conclusion: les types de classes

- Classe *concrète*
 - chaque méthode a une implémentation
- Classe *abstraite*
 - certaines méthodes ont une implémentation
 - les autres devront être implémentées
- Classe *d'interface*
 - aucune méthode n'a d'implémentation
 - peut être vue comme une *purement* abstraite