Héritage

Définition

L'héritage permet la définition de classes comme des **extensions d'autres classes**. Lorsqu'une classe hérite d'une autre classe, cela signifie qu'elle incorpore le fonctionnement interne de la classe dont elle hérite.

On dit que :

- la classe « **enfant** » dérive de la classe « parent » ;
- la classe « **parent** » est la classe de base ou superclasse de la classe « enfant » ;

La classe dérivée **hérite** des caractéristiques (attributs et méthodes) de sa classe de base, et par ce fait permet d'économiser beaucoup d'efforts de développement. En effet l'héritage apporte la possibilité de :

- factoriser le code en le spécialisant (dans la classe mère) ;
- réutiliser le code en le personnalisant (dans les classes enfants) ;

Exemples dans la documentation Java

- https://docs.oracle.com/javase/8/docs/api/javax/swing/JTable.html
- https://docs.oracle.com/javase/8/docs/api/javax/swing/JList.html
- https://docs.oracle.com/javase/8/docs/api/javax/swing/JPasswordField.html

---

Aspects syntaxiques Java

- **extends** : indique le nom de la classe mère dans l'entête d'une classe
- **super** : désigne l'instance courante de la classe mère
- **super(…)** : désigne l'appel au constructeur de la classe mère. Le constructeur d'une classe fille devrait toujours faire appel au constructeur de la classe mère en toute première instruction
- **protected** : désigne une visibilité publique dans la hiérarchie d'héritage et privée pour le reste

---

Polymorphisme

- Une variable d'un super-type peut contenir des objets de ses sous-types.
- Les sous-types peuvent être utilisés partout où un super-type est attendu

Redéfinition (override)

Lorsqu'une méthode existe dans une super-classe et que la sous-classe nécessite que cette méthode se comporte différemment, il est possible de redéfinir la méthode dans la sous-classe (même nom, même signature).

Dans ce cas, c'est au moment de l'exécution, selon le type réel d'un objet que l'environnement décidera d'exécuter la méthode du sous-type ou celle du supertype (liaison retardée).

Transtypage (Cast)

À partir du moment où des variables peuvent être polymorphes, il est nécessaire de disposer d'un outil qui permet de forcer la reconnaissance d'un objet comme un sous-type précis alors qu'il est perçu comme un super-type par le formalisme du code.

Interrogation de type

Opérer un transtypage, c'est bien. Mais parfois on ignore jusqu'au moment de l'exécution le type effectif de l'objet concerné.

Java offre deux moyens pour obtenir une indication sur le type effectif de l'objet :

- L'opérateur **instanceOf** permet de comparer un objet avec son type supposé
- La méthode **getClass()** de la classe Object qui renvoie le type effectif d'un objet sous la forme d'un objet de type Class

From:
https://wiki.siochaptalqper.fr/ - **Wiki SIO Chaptal**

Permanent link:
https://wiki.siochaptalqper.fr/doku.php?id=bloc2:prog:poo:heritage&rev=1670947260

Last update: **2022/12/13 17:01**