République islamique de la Mauritanie
Ministère de l'enseignement supérieur et de la recherche scientifique

Université de Nouakchott
Faculté des sciences et technique
Département Mathématiques et Informatique
Master M1
Option : Système d'information et Réseaux et Systèmes Communicants

Rapport TP3

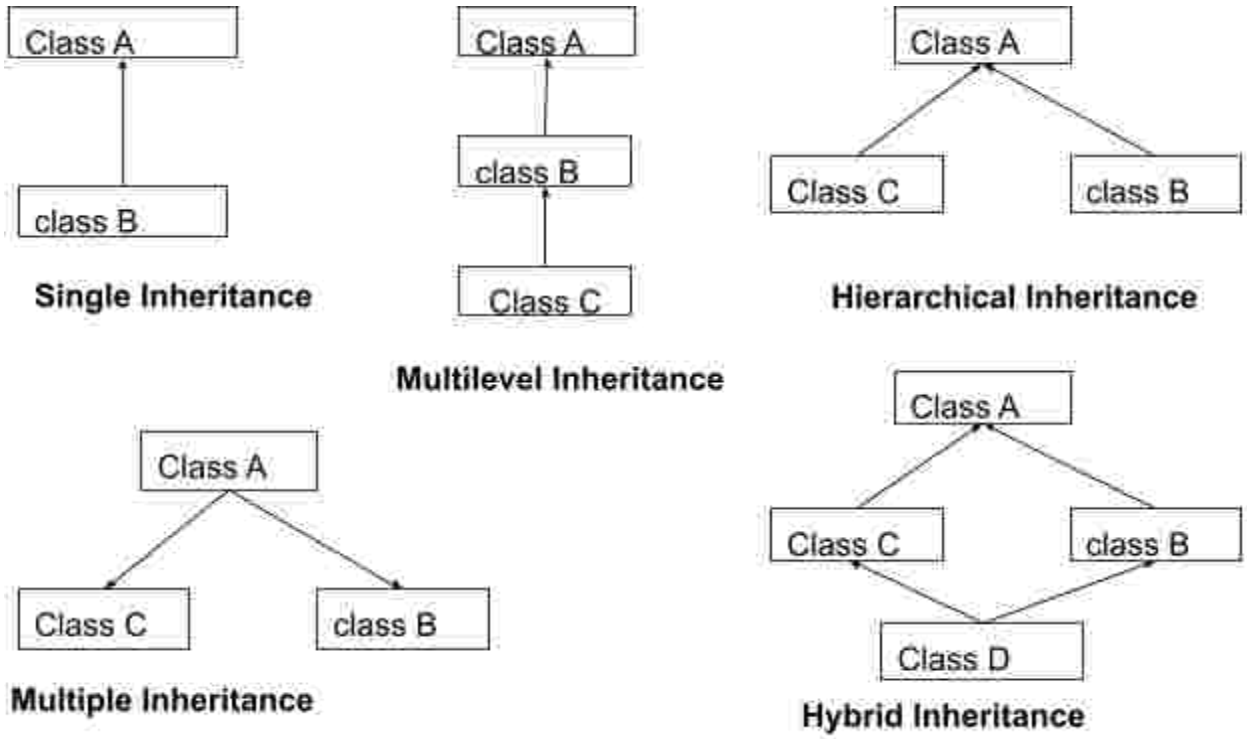
2023-2024

L'Héritage en JAVA

Présenter par :
-SALEM Ahmedou C17051
-Cheikh Abdellahi C16649

Héritage :

L'héritage vise à promouvoir la réutilisation du code en regroupant des fonctionnalités communes dans une classe parente, simplifiant ainsi la maintenance du code. Différents types d'héritage ont été identifiés :



Redéfinition :

L'objectif de la redéfinition est de fournir une nouvelle implémentation pour une méthode déjà définie dans la classe parente. Cela permet aux classes dérivées de fournir des versions spécifiques des méthodes héritées de leur classe parente.

Polymorphisme :

Le polymorphisme permet aux objets de différentes classes d'être traités de manière uniforme. Il peut être statique (lié à la compilation) ou dynamique (lié à l'exécution). L'héritage et les interfaces contribuent à la réalisation du polymorphisme, offrant ainsi une flexibilité dans l'utilisation des objets .

Interfaces :

Les interfaces permettent la réalisation du polymorphisme en définissant des contrats que les classes doivent respecter. Elles favorisent la séparation des responsabilités, la flexibilité dans l'héritage, la facilitation de la maintenance du code, l'encapsulation des signatures de méthodes, et la communication entre classes .

Classes abstraites :

Les classes abstraites sont des modèles incomplets pour d'autres classes. Elles regroupent des fonctionnalités communes, favorisent la réutilisation du code, et permettent une organisation modulaire. Les classes abstraites encouragent le polymorphisme, mais ne peuvent pas être instanciées directement.

Classes et méthodes génériques :

Les génériques offrent une flexibilité en permettant la création de composants logiciels qui travaillent avec différents types de données sans duplication de code. Ils assurent une sécurité de type à la compilation, améliorent la lisibilité du code, et sont rétrocompatibles avec le code existant. En résumé, les génériques fournissent une approche flexible et sûre pour créer des composants logiciels génériques et réutilisables .

Conclusion :

Dans l'ensemble, les concepts d'héritage, de redéfinition, de polymorphisme, d'interfaces, de classes abstraites et de génériques contribuent à une conception orientée objet robuste et flexible. Ils facilitent la réutilisation du code, améliorent la maintenabilité et permettent une gestion plus efficace des relations entre les classes. Ces concepts offrent des solutions puissantes pour créer des programmes modulaires et adaptables.