

# DIAGRAMME DE CLASSES & DIAGRAMME D'OBJETS

Mme Sfaxi Lilia

Mme Zoubeir Najet

MCOO-Chapitre 3

# Diagramme de Classes

- Le diagramme le plus important de la modélisation orientée objet : le seul obligatoire
- Décrit la structure interne du système
- Fournit une représentation abstraite des objets du système qui interagissent ensemble pour réaliser le cas d'utilisation
- Vue statique : pas de facteur temporel
- Modélisation des classes du système et leurs relations
  - ✓ Indépendant du langage de programmation

# Classes

- Description formelle d'un ensemble d'objets ayant une sémantique et des caractéristiques communes
- Un objet est une instance de classe
- Représentée par un rectangle divisé en 3 compartiments obligatoires et 2 optionnels
  - ✓ Nom de la classe
  - ✓ Attributs
  - ✓ Méthodes
  - ✓ Responsabilités (op) : ensemble des tâches devant être assurées par la classe mais pour lesquelles on ne dispose pas d'assez d'informations
  - ✓ Exceptions (op) : Situations exceptionnelles devant être gérées par les classes

# Caractéristiques d'une Classe

## ■ Visibilité

- ✓ Public ou + : tout élément qui peut voir la classe courante peut également voir l'élément indiqué
- ✓ Protected ou # : seul un élément situé dans la classe courante ou un de ses descendants peut voir l'élément indiqué.
- ✓ Private ou – : seul un élément situé dans la classe courante peut voir l'élément.
- ✓ Package ou ~ ou rien : seul un élément déclaré dans le même paquetage peut voir l'élément.

## ■ Nom de la classe

- ✓ [ <Nom\_du\_paquetage\_1>:: ... ::<Nom\_du\_paquetage\_N> ]  
<Nom\_de\_la\_classe> [ { [abstract], [<auteur>], [<date>], ... } ]
- ✓ Exemple : Pack1::Class1 {abstract, Dupond}

# Stéréotypes d'une classe

- Le stéréotype est le rôle principal d'une classe
- On définit principalement 4 stéréotypes:

✓ Entité



✓ Contrôleur



✓ Interface



✓ Acteur



# Attributs

- Données encapsulées dans les objets de cette classe
- Définis par un nom, un type de données, et une visibilité
- A la forme: <visibilité> [ / ] <nom\_attribut> : <type>  
[ '['<multiplicité>' ]' [ {<contrainte>} ] ] [ = <valeur\_par\_déf.> ]
- Exemple : + couleur : int [3] {list}
- Attribut de classe
  - ✓ Attribut propre à la classe, pas à l'instance(*static* en java)
  - ✓ Garde une valeur unique et partagée par toutes les instances de la classe
  - ✓ Graphiquement : souligné

# Méthodes

- Décrit une fonctionnalité de la classe
- Doit contenir un nom, un type de retour et des paramètres
- A la forme :
  - ✓ <visibilité> <nom\_méthode> ([<paramètre\_1>, ..., <paramètre\_N>]) : [<type\_renvoyé>] [{<propriétés>}]
- Un paramètre a la forme :
  - ✓ [<direction>] <nom\_paramètre>:<type> ['['<multiplicité>'']' [= <valeur\_par\_défaut>]]
- Exemple
  - ✓ + déplacer (in distance : int = 2) : void {abstract}

# Méthodes abstraites

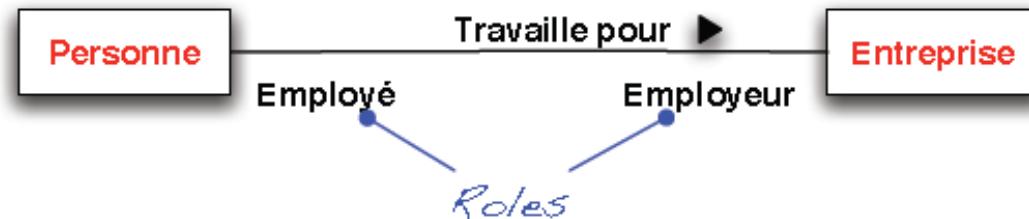
- Ne possède pas d'implémentation
- Doit être surchargée
- Si on appelle une méthode abstraite qui n'est pas surchargée, on déclenche une erreur
- Utilisée si on désire bâtir un squelette d'objet qui a plusieurs descendants devant tous avoir un comportement analogue
- Exemple
  - ✓ Dans la classe mère : `void print( String message );` //sans implémentation
  - ✓ Pour les descendants : `void print( String message ) { ... // implémentation }`

# Interfaces

- Type particulier de classes
- Classe où toutes les méthodes sont abstraites
  - ✓ Stéréotype « interface »
- Permet de regrouper un ensemble de propriétés et d'opérations assurant un service cohérent
- Doit être réalisée (implémentée) par au moins une classe et peut l'être par plusieurs
  - ✓ Stéréotype « realize»
- Une classe peut dépendre d'une interface
  - ✓ Stéréotype « use »

# Relation d'Association

- Relation entre deux classes ou plus décrivant les connexions structurelles entre leurs instances
- Relie des classes au même niveau hiérarchique
- 4 décorations permettent de spécifier le lien entre objets :
  - ✓ Nom : nature des relations entre les objets
  - ✓ Direction : direction d'application du nom
  - ✓ Rôle : rôle spécifique de chacune des classes dans l'association
  - ✓ Cardinalité : nombre d'éléments affectés
- Exemple



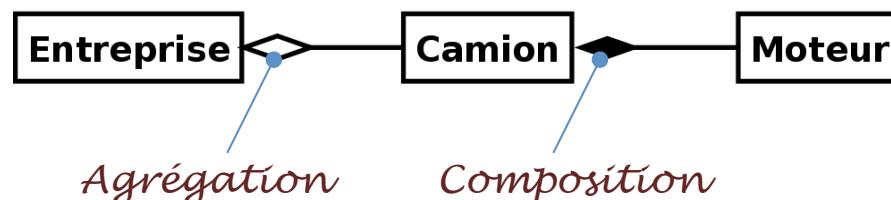
# Relations d'Aggrégation et de Composition

## ■ Agrégation

- ✓ Définit une relation hiérarchique entre les entités
- ✓ Définit la relation : « se compose de » et modélise la notion de « possession » ou de « tout et partie »

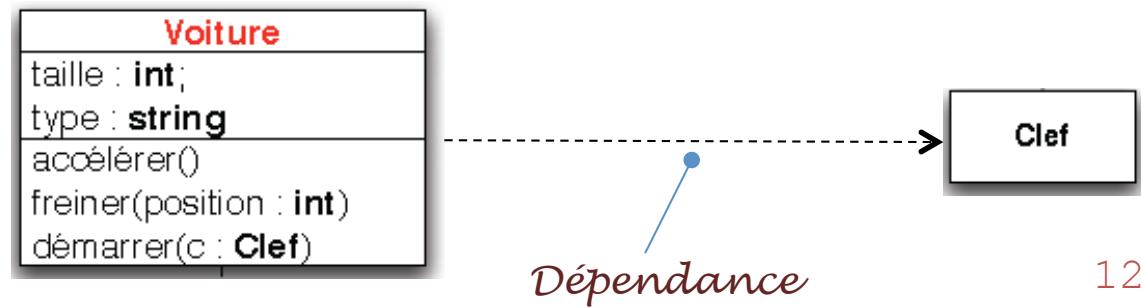
## ■ Composition

- ✓ Définit une contenance structurelle entre les instances
- ✓ La destruction de l'objet **composite** implique la destruction de ses **composants**
- ✓ Une instance du composant appartient au plus à une instance du composite



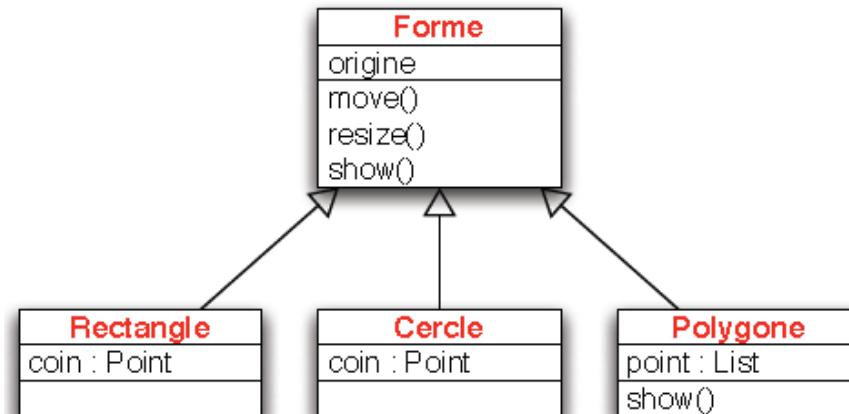
# Relation de Dépendance

- La dépendance établit une relation d'utilisation entre 2 entités d'un même diagramme.
- La plupart du temps il s'agit d'une dépendance d'utilisation :
  - ✓ Argument d'une méthode par exemple
  - ✓ On parle alors de “relation d'utilisation”
- Cela permet d'identifier implications possibles des modifications à apporter dans une entité
  - ✓ Changement du comportement d'une des classes par exemple



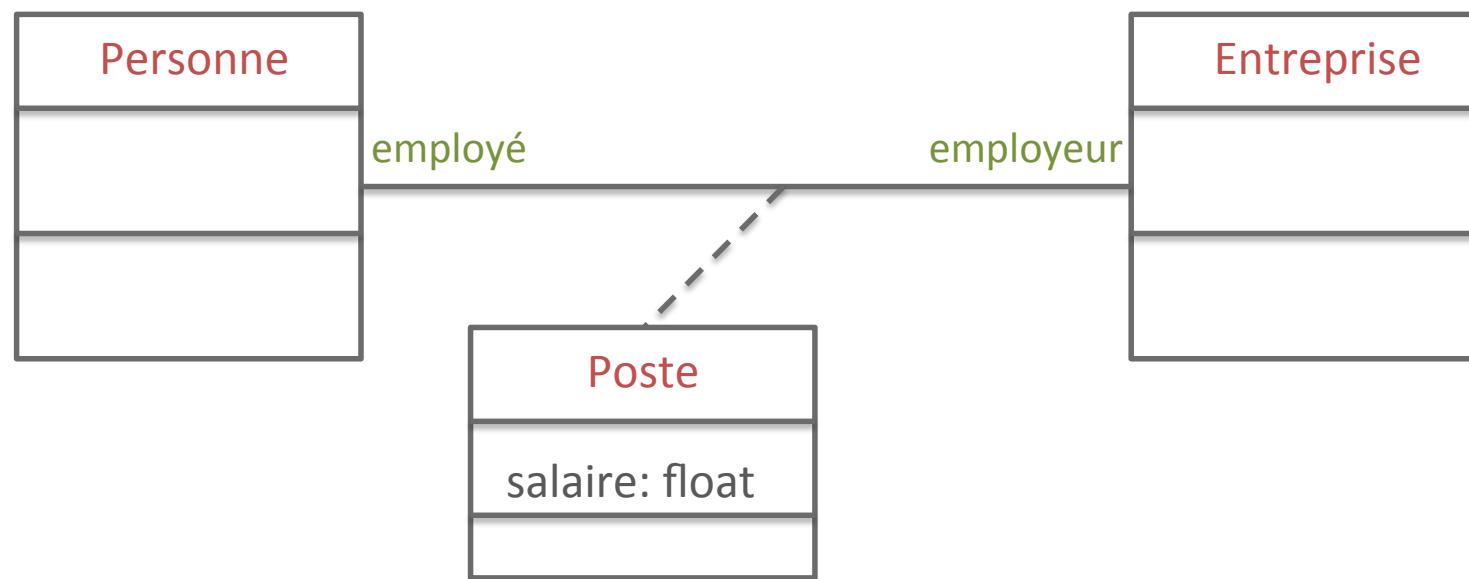
# Relation de Généralisation

- Modélise la relation d'héritage
  - ✓ La généralisation correspond à la notion “est une sorte de”
  - ✓ Modélisation des relations parents / enfants
- Les entités issues d'une généralisation sont utilisables partout où leur classe mère peut l'être (mais pas l'inverse)
- Généralisation (classe, classe) ou (classe, interface)
- Cette relation est modélisée par une flèche pointant sur la classe mère



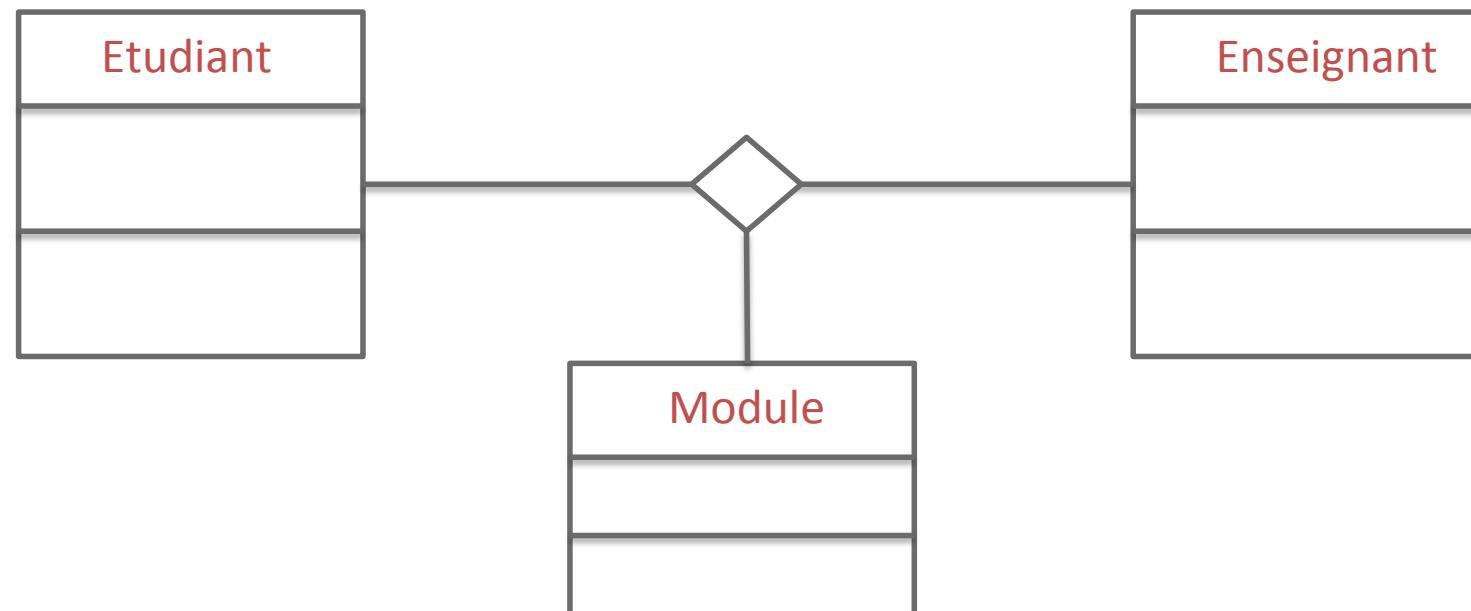
# Classe-Association

- Une association peut avoir des propriétés, qui ne sont disponibles dans aucune des classes qu'elle lie
  - ✓ On définit alors une classe-association



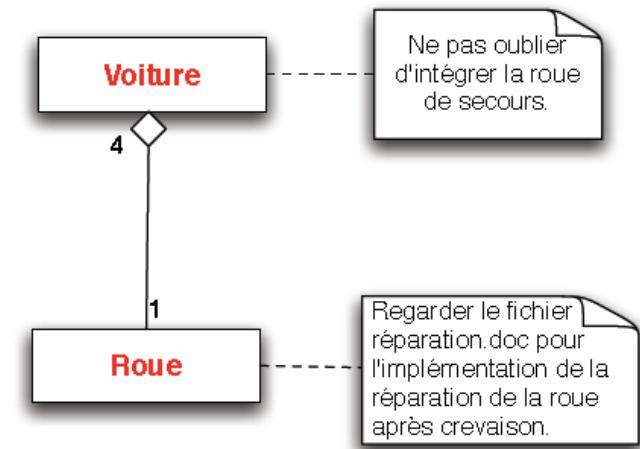
# Association n-aire

- Association qui lie plus que 2 classes
- Peu utilisée
- ✓ Gestion des multiplicités délicate

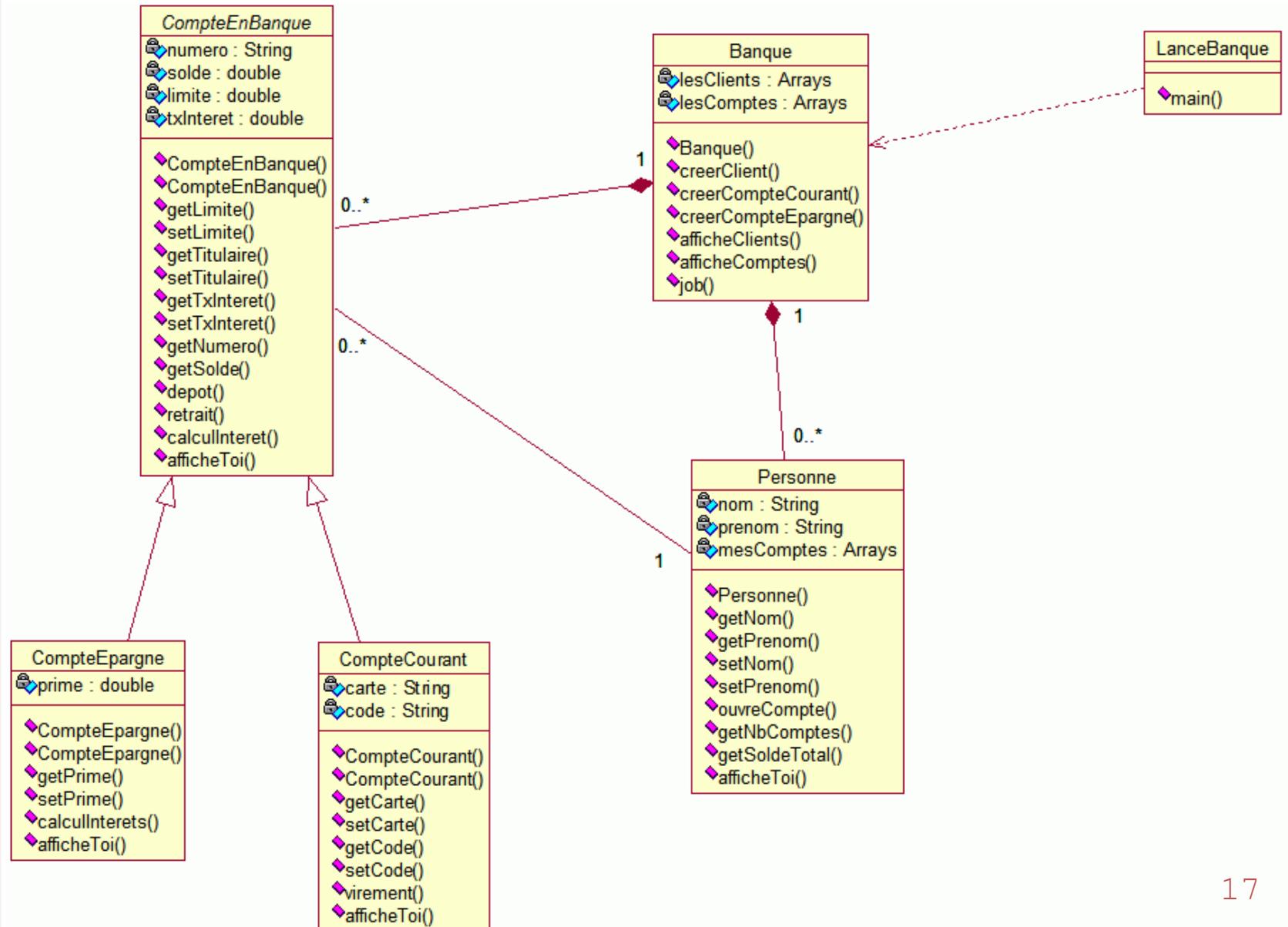


# Les Notes (Commentaires)

- Lors des phases de modélisation et de spécification, il est nécessaire de documenter ses modèles :
  - ✓ Contraintes matérielles
  - ✓ Contraintes de performance
  - ✓ Choix techniques réalisés
  - ✓ Références à d'autres docs
  - ✓ Explications techniques, etc.
- En UML, on utilise les *Notes*



# Exemple : Banque



# Activité 1

## Hôtel

- Un hôtel est composé d'au moins deux chambres. Chaque chambre dispose d'une salle d'eau qui peut être une douche ou une salle de bain. L'hôtel héberge des personnes. Il peut employer du personnel et est dirigé par un des employés. L'hôtel a les caractéristiques suivantes : une adresse, le nombre de pièces, la catégorie. Une chambre est caractérisée par le nombre et le type de lits, le prix et le numéro. On peut calculer le chiffre d'affaires, le loyer en fonction des occupants.
- Donnez le diagramme de classes.

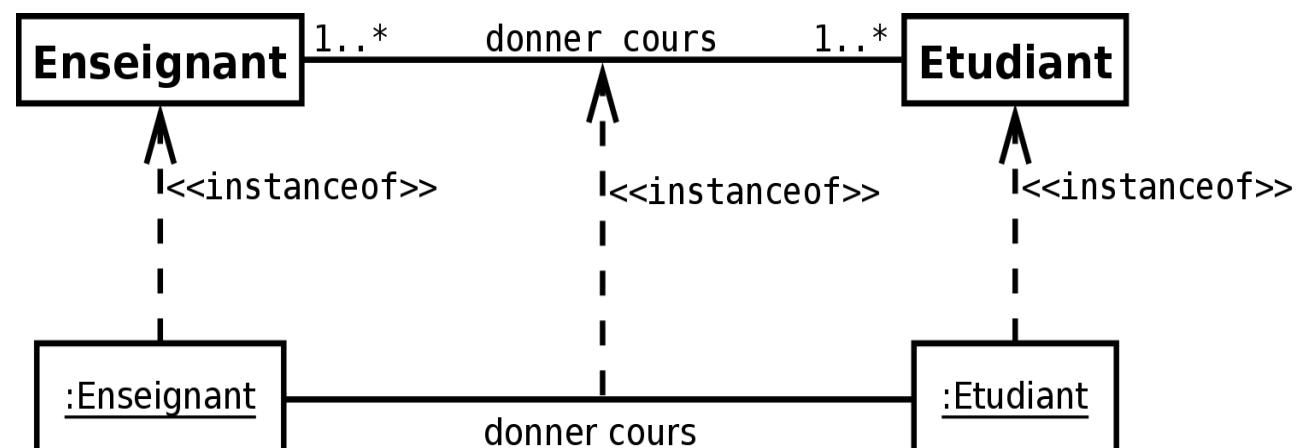
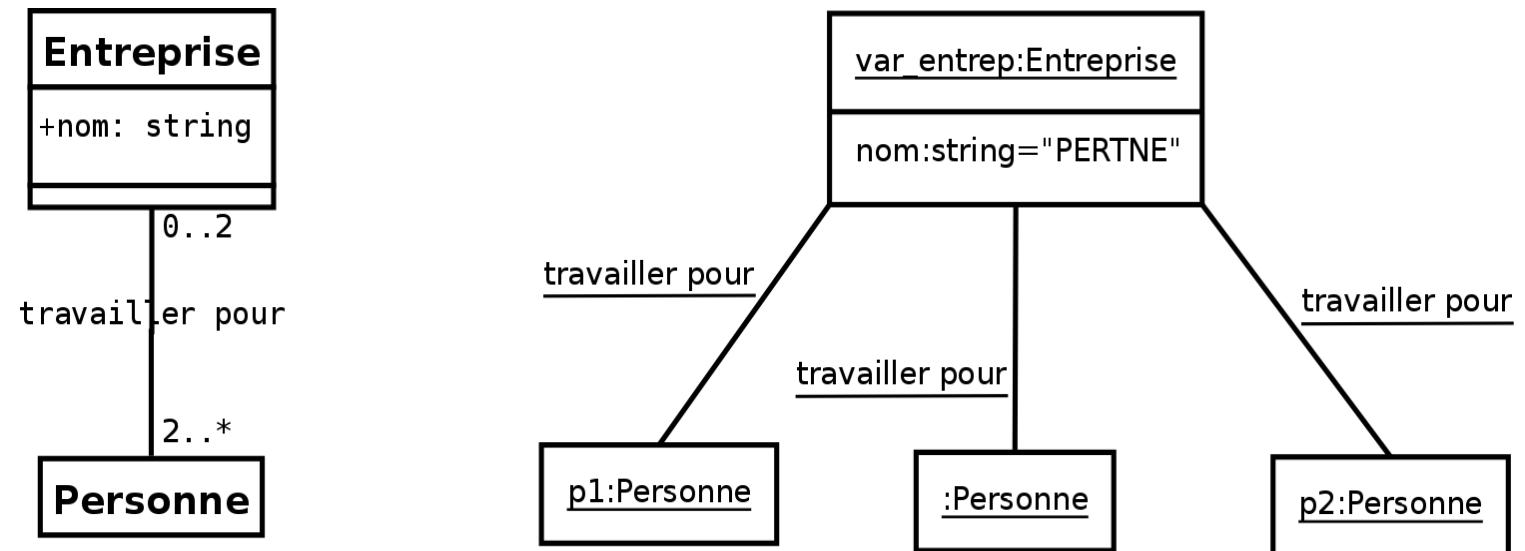
# Diagramme d' Objets

- Représente des objets (i.e. instances de classes) et leurs liens (i.e. instances de relations) pour donner une vue figée de l'état d'un système à un instant donné
- Peut être utilisé pour
  - ✓ illustrer le modèle de classes en montrant un exemple qui explique le modèle
  - ✓ préciser certains aspects du système en mettant en évidence des détails imperceptibles dans le diagramme de classes
  - ✓ exprimer une exception en modélisant des cas particuliers ou des connaissances non généralisables qui ne sont pas modélisés dans un diagramme de classe
- **Le diagramme de classes modélise les règles et le diagramme d'objets modélise des faits**

# Graphiquement

- Un objet est représenté comme une classe, mais le compartiment des méthodes n'est pas indiqué
- Le nom de l'objet est composé du nom de l'instance, suivi de celui de la classe, et est souligné
- Les attributs reçoivent des valeurs
  - ✓ Si certaines valeurs ne sont pas renseignées, l'objet est **partiellement défini**
- La relation de généralisation n'est **jamais** représentée
- Les multiplicités ne sont pas représentées
- On peut représenter la dépendance d'instanciation
  - ✓ Stéréotype « instance of »

# Exemples



## Activité 2: Bibliothèque

- Une bibliothèque compte les exemplaires des titres suivants parmi les livres dont elle dispose : « Histoire de la 2ème guerre mondiale », « Les Aventures de Robin Hood », et deux exemplaires de « Harry Potter ».
- Felix et Alain sont des utilisateurs abonnés. Alain a emprunté « Les Aventures de Robin Hood » tandis que Felix a emprunté deux livres: « Histoire de la 2ème guerre mondiale » et un exemplaire de « Harry Potter ».
- Représenter le diagramme de classes et le diagramme d'objets modélisant ce cas de figure.