M2103 – Bases de la Programmation Orientée Objets

Java – Cours 3

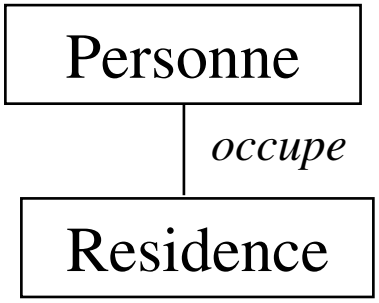
Associations (Partie I)

Plan du Cours

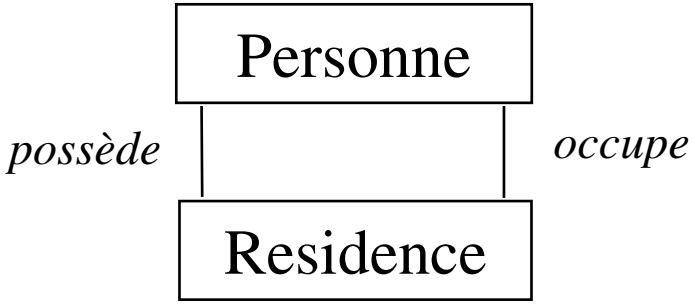
- Exemples et Définition
- Association unidirectionnelle
- Association bidirectionnelle

Associations – Exemples et Définition

- Une Personne habite dans une Residence:



- Il peut y avoir plus d'une association entre 2 classes
  - Pour une 'Personne', on peut compter différents objets pour chacune des associations…



- Définition : Abstraction de **liens** potentiels entre objets instanciés à partir des 2 classes

Classes Personne & Residence – Code Java

```
class Personne {
    // la résidence possédée par la personne
    private Residence possResidence;

    // la résidence occupée par la personne
     private Residence occupResidence;
    …
}
```

```
class Residence {
    private String adresse;
    …
    public void nettoie()
    { … }
}
```

On caractérise dans la classe *Personne* plusieurs objets *Residence*

- Un objet *Personne* peut envoyer des messages aux objets *Residence*

Association Unidirectionnelle

- Seulement l'une des deux classes envoie des messages
  - Cible – classe qui reçoit les messages
  - Expéditrice – classe qui envoie les messages
- L'expéditeur maintient les références aux objets cible
  - La référence cible est établie par :
    - Le constructeur de la classe expéditrice
    - Un mutateur de la classe expéditrice
    - En étant passée comme paramètre d'une méthode de la classe expéditrice qui appelera la cible

Référence Cible – Etablie par le Constructeur (1)

```
class Personne {
    private Residence maResidence;
    public Personne ( Residence maResidence )
    {
        this. maResidence = maResidence; // enregistre la référence,
                                         // met en oeuvre le lien
    }
}
```

- Le paramètre est de type `Residence`.

Référence Cible – Etablie par le Constructeur (2)

```
class Personne {
    private Residence maResidence;
    public Personne()                  // pas de paramètres
    {
        maResidence = new Residence();     // crée objet et lien
    }
}
```

Référence Cible – Etablie par un Mutateur

```
class Personne {
    private Residence maResidence;
    …
    public setMaResidence (Residence myResidence)
    {
        this.maResidence = maResidence;
    }
}
```

Référence Cible – Passage de Paramètre

```
class Personne {
    …
    public nettoieUneResidence (Residence residenceANettoyer)
    {
        residenceANettoyer.nettoie();      // n'enregistre pas, mais utilise
                                           // l'objet.
                                           // lien existe pendant que la
                                           // méthode est exécutée
    }   // lien n'existe plus lorsque la méthode a fini d'être exécutée
}
```

Association Bidirectionnelle

- Chacune des deux classes peut envoyer des messages à l'autre
  - L'une est la cible pendant que l'autre est l'expéditrice
  - Les rôles peuvent toutefois changer sans avoir besoin de mettre en place une nouvelle relation
- La mise en oeuvre de liens est plus compliquée :
  - Les 2 objets doivent avoir été instanciés, mais l'un est instancié avant l'autre
  - L'ordre de création dépend des autres liens requis
  - Une méthode *set* d'une classe appelle la méthode *set* de l'autre classe
  - Peut nécessiter de passer la référence 'this' comme paramètre du mutateur de l'autre objet

Association Bidirectionnelle

```
class Personne
{
    private Chat monChat;
    public void setMonChat(Chat chat)
    {
        // 1 direction établie
        this.monChat = chat;

        chat.setMonProprio(this);
    }
}
```

```
class Chat
{
    private Personne monProprio;
    public void setMonProprio(
    Personne proprio)
    {
        // Autre direction établie
        this.monProprio = proprio;
    }
    …
}
```

Application

- Un `Locataire` est caractérisé par : son identité, son adresse, le montant de son loyer et celui de sa taxe d'habitation. Mettre en œuvre la classe correspondante avec les attributs et méthodes requis.
- Un `Propriétaire` est caractérisé par : son identité, son adresse, le montant de sa taxe foncière et celui de sa taxe d'habitation. Mettre en œuvre la classe correspondante.
- Modifier l'implémentation de la classe `BienImmobilier` de manière à caractériser le propriétaire et le locataire d'un bien (attributs et méthodes).
- Mettre en œuvre une classe de test.