

Le diagramme de classes

Table des matières

I. Contexte	3
II. Le diagramme de classes	3
III. Exercice : Appliquez la notion	5
IV. Les associations	5
V. Exercice : Appliquez la notion	7
VI. Les associations complexes	7
VII. Exercice : Appliquez la notion	11
VIII. Types d'associations	11
IX. Exercice : Appliquez la notion	13
X. Essentiel	14
XI. Auto-évaluation	14
A. Exercice final	14
B. Exercice : Défi.....	16
Solutions des exercices	16

I. Contexte

Durée : 1 h

Environnement de travail : Local

Pré-requis : Avoir les bases de l'UML

Contexte

Parmi les diagrammes qui composent les spécifications techniques, le diagramme de classes est celui qui est majoritairement utilisé. À lui seul, il permet à la fois de structurer le code de l'application et d'établir le modèle relationnel du projet, ce qui en fait un élément indispensable. La qualité du diagramme de classes peut faire la différence entre un code facilement maintenable et un projet souffrant de dette technique.

II. Le diagramme de classes

Objectifs

- Savoir ce qu'est un diagramme de classes
- Représenter une classe dans un diagramme de classes

Mise en situation

Le **diagramme de classes** permet de représenter les différentes classes de l'application, avec leurs attributs et leurs méthodes, et de préciser les relations entre elles. Il sera ainsi possible de décrire une classe « Article » pour un blog et une classe « Auteur », puis de préciser qu'un article est rédigé par un auteur.

Syntaxe d'une classe

Dans un diagramme de classes, une classe est représentée par un rectangle séparé en 3 parties :

- Le nom de la classe est inscrit dans la partie haute du rectangle. Il doit respecter le formalisme habituel des noms de classe, c'est-à-dire être écrit en PascalCase.
- La partie du milieu est dédiée aux propriétés de la classe. À chaque propriété doit correspondre un type grâce à la syntaxe `nom : Type`. Les propriétés doivent également être préfixées par un symbole précisant leur visibilité : `+` pour une visibilité publique, `-` pour privée et `#` pour protégée.
- Enfin, la partie du bas permet d'ajouter les différentes méthodes. Chaque méthode doit être accompagnée de sa signature : paramètres et type de retour.

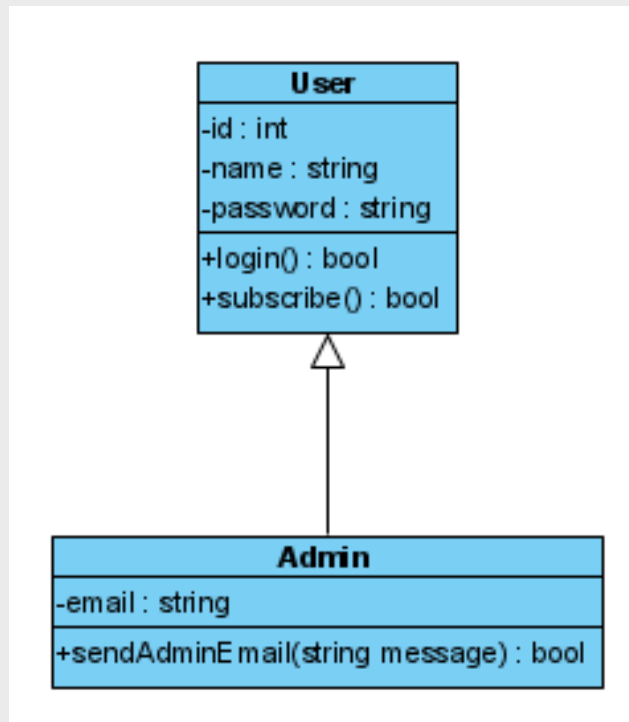
Il est possible de spécifier une relation d'héritage entre deux classes par une flèche triangulaire.

Attention

Le type des propriétés doit être présent, même si l'application finale sera écrite dans un langage non typé. En effet, le diagramme de classes sera également utilisé afin de créer la structure de la base de données, dans laquelle les types sont indispensables.

Exemple

Voici l'exemple d'une classe `User`, permettant de représenter les utilisateurs d'une application, et de sa classe fille `Admin`. Les administrateurs sont des utilisateurs, mais à qui l'application pourra envoyer des notifications par e-mail en cas de problème.



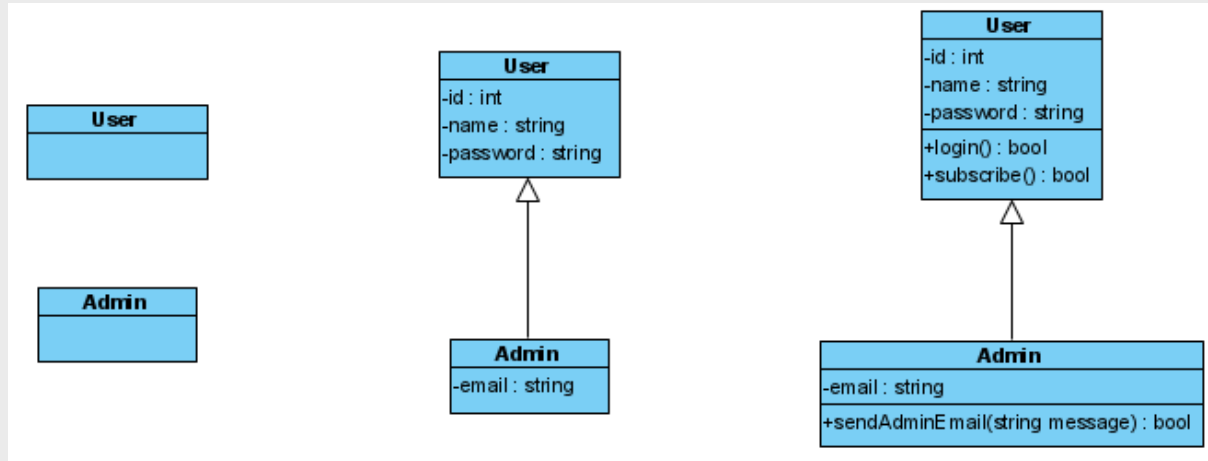
Un diagramme évolutif

Lors de la conception d'un projet, les phases de spécifications fonctionnelles et de spécifications techniques forment un cycle. Ainsi, il n'y a pas une unique phase fonctionnelle, puis une unique phase technique : c'est plutôt une boucle où le fonctionnel influence le technique, qui va lui-même modifier certains éléments fonctionnels, etc.

Le diagramme de classes va donc se construire sur plusieurs itérations. Il est ainsi possible d'avoir des classes ayant plusieurs niveaux de complexité : d'abord seulement un nom, puis en y rajoutant ensuite des attributs, puis des méthodes.

Exemple

La classe `User`, présentée ci-dessus, n'a pas eu cette forme finale directement. Lors de la première itération, les spécifications n'indiquaient que la présence d'utilisateurs et d'administrateurs. Ce n'est que plus tard que leurs propriétés ont été définies, ce qui a permis de déduire une relation entre ces deux classes. Enfin, il a été possible de répartir les différentes opérations à réaliser sur les utilisateurs :

**Syntaxe****À retenir**

- Le diagramme de classes permet de représenter les différentes classes de notre système. Chaque classe du diagramme est représentée par un rectangle séparé en trois parties : le nom, les propriétés et les méthodes.

III. Exercice : Appliquez la notion

Question

[solution n°1 p.17]

Une bibliothèque souhaiterait avoir un outil informatique pour recenser ses livres et pouvoir les rechercher selon divers critères. Les informations importantes d'un livre sont le titre, l'auteur, le nombre de pages, le synopsis et le nom du personnage principal.

En plus de la recherche, la bibliothèque souhaiterait également que l'outil lui permette de comparer un livre à un autre pour savoir s'ils sont du même auteur ou non. À des fins de statistiques, elle souhaiterait aussi pouvoir compter le nombre de mots d'un synopsis.

Réalisez le diagramme de la classe `Book` permettant de répondre à ce besoin.

Indice :

La comparaison d'un livre avec un autre va se faire avec une méthode de la classe `Book` qui prend en paramètre un autre `Book`.

IV. Les associations

Objectifs

- Lier les classes de notre diagramme entre elles
- Savoir déterminer les cardinalités d'une association

Mise en situation

La plupart des classes d'une application sont liées entre elles : une classe représentant une entreprise va être liée à la classe représentant les employés, elle-même liée à la classe des activités d'entreprise. Dans cette partie, nous allons voir comment faire apparaître ces associations dans le diagramme de classes et quelles sont les différentes associations qui existent.

Les associations

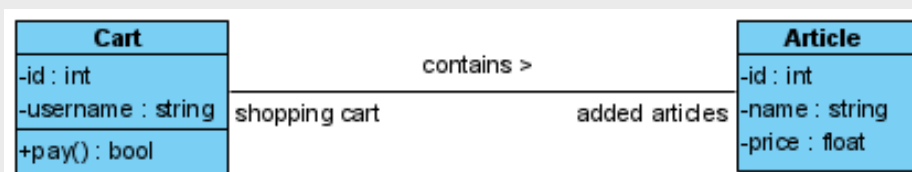
Une association est une relation entre deux objets. Elle permet de symboliser le lien entre deux objets : dans le code final, l'une des classes liées sera stockée dans les propriétés de l'autre. Ainsi, une classe `Panier` et une classe `Article` seront liées, puisqu'on peut facilement imaginer qu'un objet *Panier* possédera un tableau d'objets *Article*.

Les associations sont représentées par un **trait plein** et peuvent posséder un nom qui permet de définir la sémantique de l'association. Une **flèche** peut parfois indiquer le sens de lecture de l'association.

Il est également possible de préciser le rôle de chaque classe dans l'association en ajoutant un texte à côté de la classe. Il est facultatif, mais peut permettre de décrire le rôle de la classe dans l'association.

Exemple

La relation entre une classe représentant des articles et celle représentant des paniers est représentée par le diagramme suivant :



Ce diagramme se lit « Un panier (*shopping cart*) contient des articles (*added articles*) ». Les rôles et le nom de l'association sont ici assez explicites, il aurait donc été possible de ne pas les mettre.

Les cardinalités

Les **cardinalités** permettent de définir la quantité d'éléments de l'association. Elles permettent d'indiquer, pour une instance de la classe A, combien d'instances de la classe B peuvent y être liées. Les cardinalités ne représentent pas des quantités précises, mais des ordres de grandeur. Par exemple, qu'un panier possède plusieurs articles ou qu'un employé travaille pour une seule entreprise.

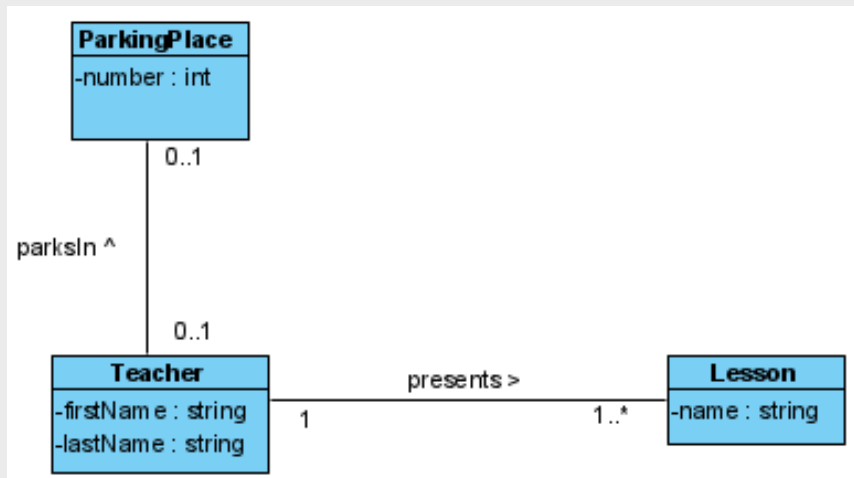
Il existe en tout quatre types de cardinalités :

- **1** désigne une cardinalité unitaire simple. Elle se lit « un A est lié à *un seul* B ». Par exemple, un cours est animé par un seul professeur : s'il n'y a pas de professeur, il n'y a pas de cours.
- **0..1** désigne une cardinalité unitaire pouvant être nulle. Elle se lit « un A est lié à *au plus un* B ». Par exemple, un professeur possède au plus une place de parking. La différence avec la cardinalité de 1 est qu'ici, certains professeurs peuvent ne pas avoir de place de parking.
- **1..*** désigne une cardinalité multiple simple. Elle se lit « un A est lié à *plusieurs* B ». Par exemple, un cours est suivi par plusieurs élèves : s'il n'y a pas d'élèves, il n'y a pas de cours.
- **0..*** désigne une cardinalité multiple pouvant être nulle. Elle se lit « un A *peut* être lié à *plusieurs* B ». Par exemple, un professeur peut posséder plusieurs voitures. Certains professeurs n'en ont pas, certains en ont une et certains en ont plusieurs.

Tout comme les noms, les cardinalités se placent des deux côtés de l'association et peuvent être différentes de chaque côté : un cours est animé par un seul professeur, mais un professeur anime plusieurs cours.

Exemple

Un professeur peut présenter plusieurs leçons, mais chaque leçon n'est présentée que par un professeur. De plus, certains professeurs ont une place de parking attitrée, mais certaines places de parking peuvent être vides. Cette situation peut être représentée par le diagramme suivant :



Le sens de lecture est donc « Teacher parks in 0..1 ParkingPlace » et « Teacher presents 1..* Lesson ».

Complément De l'UML au code

Dans le code final, chaque association deviendra une propriété dans la classe concernée. Le type de la propriété va dépendre de la cardinalité de l'association :

- **1 : propriété non nullable.** Si un cours est animé par un seul professeur, alors la classe Lesson aura une propriété de type Teacher qui ne pourra pas être nulle.
- **0..1 : propriété nullable.**
- **1..* : tableau non vide.** Si un cours doit être suivi par au moins un élève, alors la classe Lesson aura une propriété de type Student[], donc un tableau de Student. Ce tableau devra être initialisé avec au moins un élément à l'intérieur.
- **0..* : tableau pouvant être vide.**

Syntaxe À retenir

- Les associations permettent de lier les classes. Chaque association possède un nom et une cardinalité pour chaque classe. Il existe 4 cardinalités : 1, 0..1, 1..* et 0..*.

V. Exercice : Appliquez la notion**Question**

[solution n°2 p.17]

Une école de musique aimerait créer une application pour gérer les groupes de musique que ses élèves forment. Un groupe possède un nom et un style musical, tandis que les membres ont un nom, un prénom et un instrument. Sachant qu'un membre ne peut appartenir qu'à un groupe, réalisez le diagramme de classes correspondant à cette application.

VI. Les associations complexes

Objectifs

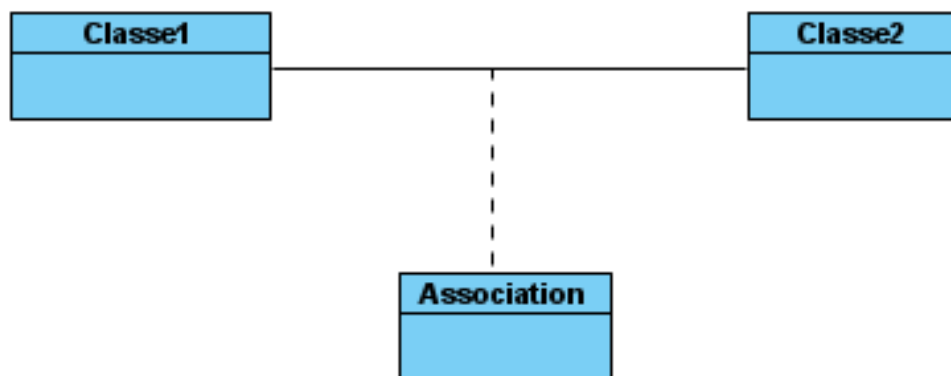
- Ajouter des informations à une association
- Lier plus de deux classes entre elles

Mise en situation

Maintenant que nous savons comment lier deux classes entre elles, nous allons découvrir comment ajouter des informations à une association. La relation entre un professeur et un cours, par exemple, aurait tout intérêt à être enrichie du numéro de la salle dans laquelle il va se dérouler ou de la date du cours. Nous allons voir comment procéder et en quoi ce cas diffère d'une association entre plus de deux classes.

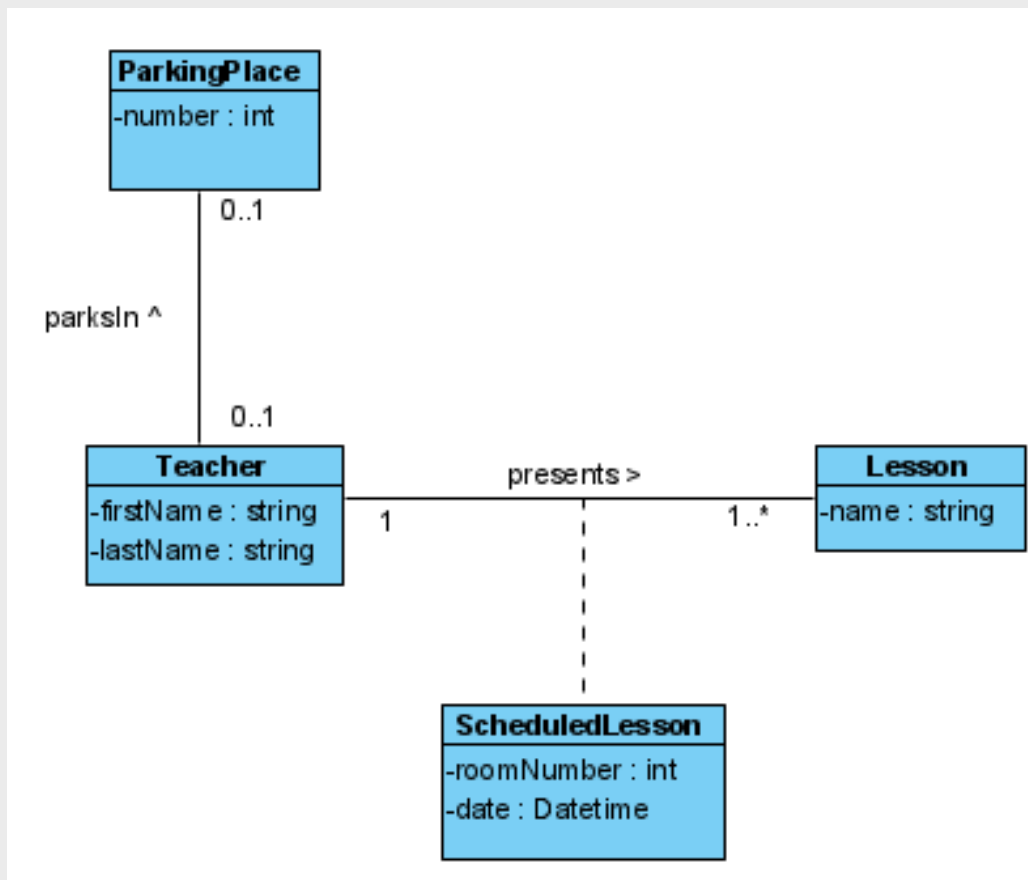
Les classes d'association

Pour ajouter des données à une association, il va falloir créer une nouvelle classe qui va contenir ces données, appelée « classe d'association ». Cette classe va être liée à l'association qu'elle enrichit par un trait pointillé. Contrairement aux autres classes, une classe d'association ne va pas avoir de cardinalités. Sa quantité est implicite : il y aura autant d'instances de cette classe que d'associations entre les deux autres classes.

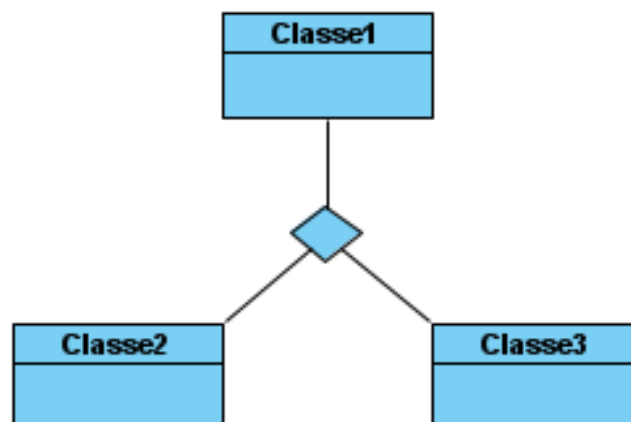


Exemple

Si, pour chaque cours donné par chaque professeur, on veut ajouter la date du cours et le numéro de la salle, alors il va falloir créer une classe d'association qui va posséder ces attributs :

**Les associations n-aires**

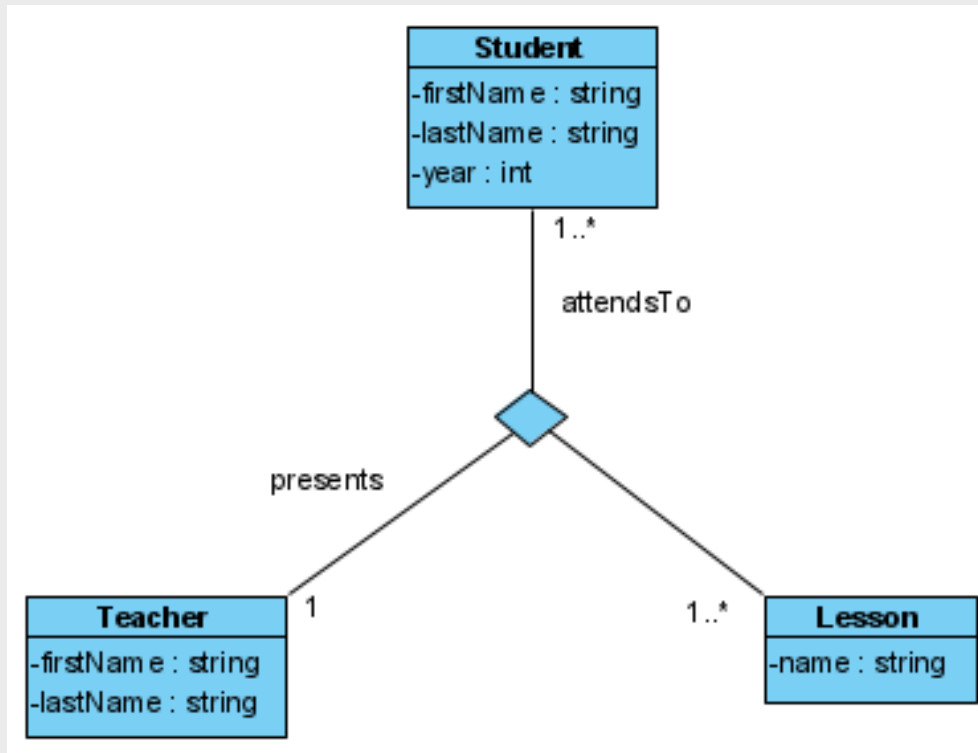
Une association **n-aire** est une association entre plus de deux classes. Pour cette association multiple, chaque lien rejoint les autres dans un **losange**.



Contrairement à une classe d'association, toutes les cardinalités doivent être présentes. Les noms et les rôles restent facultatifs.

Exemple

Plusieurs élèves participent également à la leçon donnée par le professeur. Pour les ajouter à la relation, il faut utiliser une association n-aire :



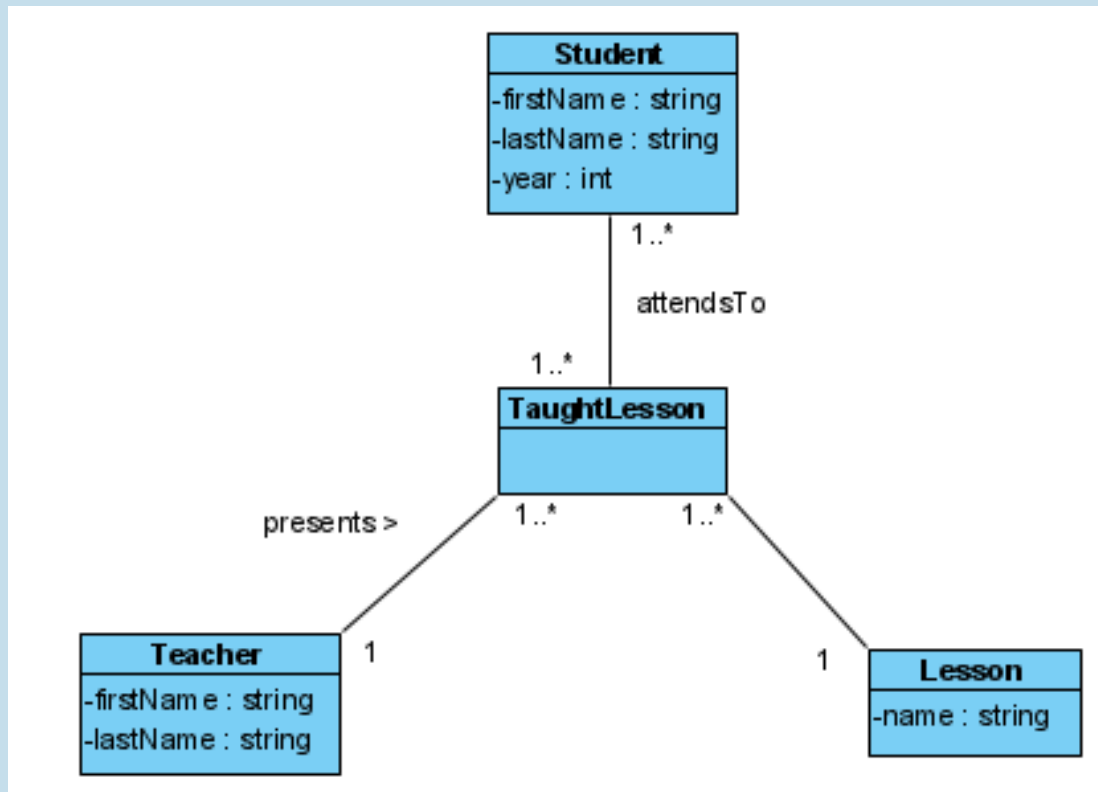
On peut lire ce diagramme de la manière suivante : « Un professeur peut donner plusieurs cours à un élève (d'où la cardinalité `1..*` pour **Lesson**), un professeur enseigne un cours à plusieurs élèves (la cardinalité `1..*` de **Student**) et, pour un élève et un cours donné, il ne peut y avoir qu'un seul professeur en présentation (la cardinalité `1` du **Teacher**) ».

Remarque

Dans la pratique, il est souvent possible de simplifier le diagramme en transformant les classes d'association en classes normales. Cela permet d'apporter plus de sémantique aux associations en leur attribuant une vraie classe et des cardinalités qui lui sont propres.

Pour les classes d'associations, il suffit de créer une vraie classe, et, pour les association n-aires, il est possible de créer une nouvelle classe permettant de faire le lien avec les autres.

Ainsi, l'association n-aire de l'exemple précédent peut s'écrire :



Syntaxe À retenir

- Les classes d'association permettent d'ajouter des informations directement sur une association. La classe est alors liée à l'association grâce à un trait pointillé.
- Pour relier plusieurs classes entre elles, il faut utiliser une association n-aire, représentée par un losange.
- Il est souvent possible de transformer des classes d'association ou des associations n-aires en associations binaires pour simplifier le diagramme.

VII. Exercice : Appliquez la notion

Question

[solution n°3 p.17]

Une association de volley-ball souhaiterait créer une application pour gérer les entraînements des différentes équipes : chaque entraînement se déroule dans un gymnase d'une ville de la région, et l'association doit réserver un créneau horaire pour que les équipes puissent avoir le gymnase pour elles, sans être dérangées par les autres équipes.

Une équipe possède un nom et un nombre de joueurs. Un gymnase possède un nom, une adresse et un nom de ville. Pour chaque entraînement, l'association attribue une équipe à un gymnase pour une date et une heure données.

En choisissant la bonne association complexe, réalisez le diagramme de classes représentant cette situation.

VIII. Types d'associations

Objectifs

- Découvrir les différents types d'associations
- Comprendre la différence entre eux

Mise en situation

Les associations simples ne permettent pas toujours d'exprimer toutes les subtilités de la relation entre les classes. C'est pourquoi, en plus de l'association de base, représentée par un trait plein, il existe deux autres types d'associations : les agrégations et les compositions, qui vont permettre d'ajouter de la sémantique aux relations.

Méthode La composition

La **composition** est une association particulière qui permet d'indiquer une dépendance forte entre deux objets et de lier leurs durées de vie : si une classe A est composée d'une classe B, alors la classe B n'existera plus si la classe A est détruite. La classe A agit ainsi comme un conteneur sans lequel la classe B ne peut survivre, et l'instance de la classe B appartient à l'instance de la classe A : l'instance de la classe B ne peut pas être utilisée ailleurs.

Une composition est symbolisée par un **diamant noir** au bout de l'association, placé du côté du conteneur.

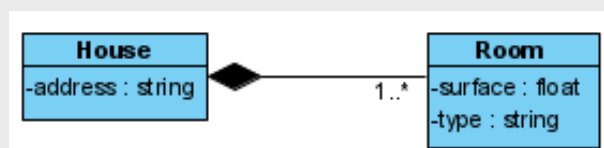


À noter qu'il n'est pas utile de préciser la cardinalité du côté du conteneur : un composant appartient forcément à un seul conteneur.

Exemple

Prenons l'exemple d'une application de gestion de biens immobiliers : chaque maison possède une adresse et un certain nombre de pièces, avec leur type (cuisine, salle de bains, etc.) et leur surface.

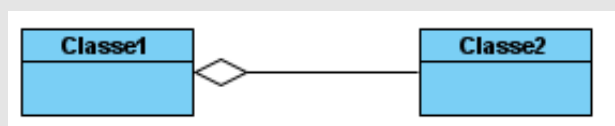
Une pièce ne peut pas exister sans maison : si la maison est supprimée de l'application, alors il faudra obligatoirement supprimer les pièces. De plus, il n'est pas possible de déplacer une pièce d'une maison à une autre : une pièce n'appartient qu'à une seule maison. Il y a donc une relation de composition entre ces deux entités.



Méthode L'agrégation

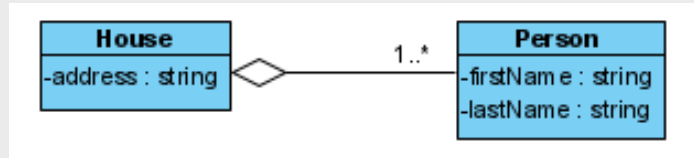
Au contraire de la composition, l'**agrégation** permet de définir une dépendance faible entre deux classes : si une classe A agrège une classe B, alors la classe A a besoin de la classe B pour fonctionner, mais chaque instance possède sa propre durée de vie. Le composant agrégé peut donc être utilisé par d'autres classes.

L'agrégation est représentée par un **diamant blanc** au bout de l'association.



Exemple

Dans le logiciel de gestion immobilière, une maison a besoin de locataires : si elle est inhabitée, il sera impossible de faire le calcul des loyers, l'état des lieux, etc. Cependant, n'importe qui peut emménager, et les personnes actuellement en place dans une maison pourront déménager dans une autre maison au besoin. La relation entre les maisons et les personnes est donc une agrégation.

**Remarque** Agrégation et association

La différence entre une agrégation et une association simple se situe dans la notion de besoin.

Dans l'exemple ci-dessus, une maison pourrait également contenir des accessoires, comme une climatisation, des panneaux solaires, etc. Cependant, une maison sans accessoires peut quand même être louée et son loyer peut être calculé : le lien entre une maison et un accessoire est donc une association simple.

Complément De l'UML au code

Dans le code final, la différence entre une agrégation et une composition se fait au moment d'initialiser la valeur des propriétés. *

Dans une composition, la propriété sera chargée dans le constructeur de son conteneur, tandis qu'avec une agrégation, elle sera passée par injection de dépendance : l'objet sera chargé à l'extérieur de la classe et seule une référence sera passée en paramètre du constructeur.

Dans le code, une agrégation sera donc similaire à une association simple, mais l'absence d'une agrégation pourra provoquer une exception, étant donné qu'elle est indispensable au bon fonctionnement de la classe.

Syntaxe À retenir

- Une composition permet de définir une dépendance forte entre deux classes : si le conteneur est détruit, ses composants sont également détruits.
- Au contraire, une agrégation définit une dépendance faible : la classe parente a quand même besoin de la classe agrégée, mais cette dernière peut être utilisée dans d'autres classes et a une durée de vie qui lui est propre.

IX. Exercice : Appliquez la notion**Question**

[solution n°4 p.18]

La Fédération Française de Motocyclisme souhaiterait avoir un logiciel pour gérer les championnats de course de moto. Un championnat est composé de plusieurs circuits. Chaque circuit se situe dans une ville et, même si c'est rare, une ville peut posséder plusieurs circuits.

Voici les informations à gérer pour chaque entité :

- Un championnat a un nom et un tarif d'inscription
- Un circuit possède une distance
- Une ville a un nom

Réalisez le diagramme de classes de cette application, en donnant le plus de sens possible à vos associations.

X. Essentiel

XI. Auto-évaluation

A. Exercice final

Exercice 1

[solution n°5 p.18]

Exercice

Quel symbole permet d'attribuer une visibilité privée à un attribut ou une méthode ?

- ☐ +
- ☐ -
- ☐ #
- ☐ !

Exercice

Les classes d'un diagramme de classes doivent obligatoirement avoir des attributs et des méthodes, sinon les spécifications ne sont pas valides et ne peuvent pas être montrées au client.

- ☐ Vrai
- ☐ Faux

Exercice

À quoi sert une association ?

- ☐ À lier deux classes entre elles
- ☐ À spécifier les attributs d'une classe
- ☐ À lier une classe avec ses instances

Exercice

Comment se lit la cardinalité 1 . . * entre deux classes A et B ?

- ☐ « Un A est lié à un seul B »
- ☐ « Un A est lié à au plus un B »
- ☐ « Un A est lié à plusieurs B »
- ☐ « Un A peut être lié à plusieurs B »

Exercice

Soit deux classes `Keyboard`, représentant un clavier, et `Key`, représentant une touche de clavier, quelle est la cardinalité de l'association, dans le sens `Keyboard > Key` ? (On ne prend pas en compte le fait qu'on peut démonter les composants.)

- ☐ 0
- ☐ 1
- ☐ 0..*
- ☐ 1..*

Exercice

Quelle est la cardinalité de l'association précédente, mais cette fois dans le sens `Key > Keyboard` ?

- ☐ 0
- ☐ 1
- ☐ 0..*
- ☐ 1..*

Exercice

Quel type d'association pourrait lier les classes `Keyboard` et `Key` ?

- ☐ Association simple
- ☐ Composition
- ☐ Agrégation

Exercice

Comment est représentée une classe d'association ?

- ☐ Un trait plein
- ☐ Un trait plein avec un diamant blanc au bout
- ☐ Un trait plein avec un diamant noir au bout
- ☐ Un trait pointillé
- ☐ Une flèche

Exercice

Comment transformer une association n-aire en associations binaires ?

- ☐ En créant une nouvelle classe qui va permettre de relier les autres (à la place du diamant)
- ☐ En créant des associations entre chaque classe de l'association n-aire
- ☐ En remplaçant l'association n-aire par une classe d'association
- ☐ Ce n'est pas possible

Exercice

Quelle est la couleur du diamant permettant de définir une relation de composition ?

- ☐ Blanc
- ☐ Noir
- ☐ Vert
- ☐ Il n'y a pas de diamant pour les compositions

B. Exercice : Défi

Une entreprise spécialisée dans l'organisation d'événements souhaiterait ajouter une partie « blog » à son site Internet afin de pouvoir afficher ses actualités. Vous allez devoir créer le diagramme de classes nécessaire à l'élaboration d'un tel projet.

Question

[solution n°6 p.20]

Afin d'améliorer son site Internet, un organisateur d'événements souhaiterait pouvoir créer des articles et les afficher dans une partie « blog », accessible aux visiteurs.

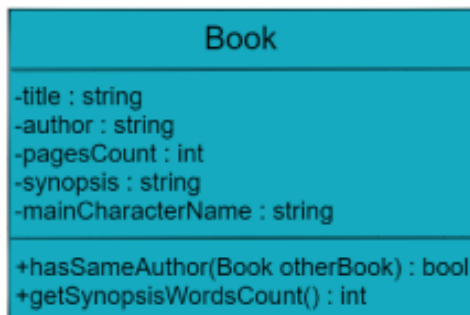
- Les utilisateurs connectés doivent pouvoir ajouter des commentaires aux articles, et certains utilisateurs particuliers, employés de l'entreprise, doivent également pouvoir rédiger les articles et y associer des mots-clés.
- Chaque utilisateur possède un nom d'utilisateur et un mot de passe, mais les auteurs ont également une signature, qui sera affichée en bas des articles qu'ils ont rédigés.
- Les articles sont composés d'un titre, d'un contenu et d'une date de publication. N'importe quel auteur peut modifier un article, et ainsi en devenir l'auteur officiel : en cas de suppression du compte d'un auteur, alors les articles seront simplement transférés à un autre auteur. Il est également possible d'associer un ou plusieurs mots-clés, composé(s) d'un simple nom, à un article, mais ce n'est pas obligatoire.
- Enfin, les utilisateurs peuvent se rendre sur la page d'un article pour poster des commentaires sous forme de messages textuels. Cependant, en cas de suppression du compte utilisateur ou de l'article, les commentaires seront détruits.

À partir des informations fournies, réalisez le diagramme de classes de ce blog.

Solutions des exercices

p. 5 Solution n°1

Le livre va posséder 5 attributs, ainsi que deux méthodes : une méthode `hasSameAuthor` qui prend un autre livre en paramètre et retourne un booléen, et une méthode `getSynopsisWordsCount` qui retourne un entier.



Le nom des attributs et des méthodes peuvent être différents, sans que ce soit considéré comme une erreur.

p. 7 Solution n°2

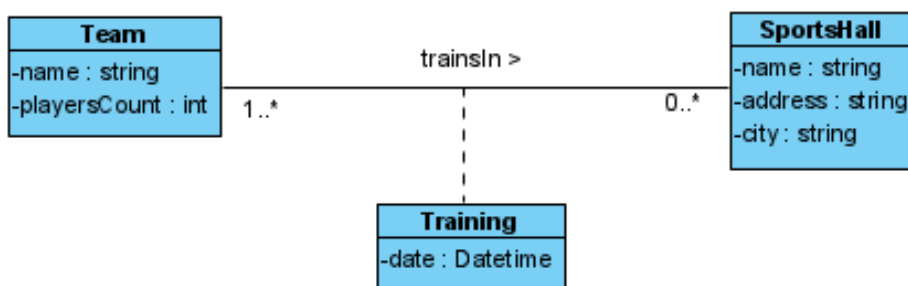
Nous savons qu'un membre n'appartient qu'à un seul groupe, donc la cardinalité est de 1 (ou 0..1, si on estime que l'application permet d'avoir des personnes sans groupe). Un groupe de musique est composé de plusieurs personnes, mais il faut au moins un membre, donc la cardinalité de l'autre côté de l'association sera 1..*. Le diagramme est donc le suivant :



p. 11 Solution n°3

Étant donné que la date doit être liée à chaque entraînement, alors il est préférable d'utiliser une classe d'association : chaque association entre une équipe et un gymnase aura ainsi l'information sur le créneau horaire.

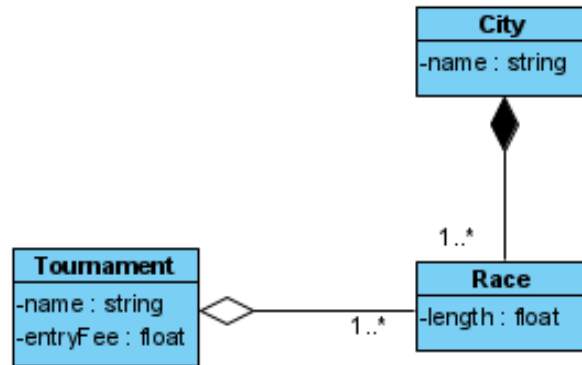
Pour les cardinalités, chaque équipe va s'entraîner dans plusieurs gymnases (mais au moins un), et chaque gymnase va accueillir plusieurs équipes (on peut imaginer que certains gymnases n'accueilleront aucune équipe, mais une cardinalité 1..* fonctionne également).



p. 13 Solution n°4

La relation entre un tournoi et un circuit est une relation d'agrégation : un circuit peut être utilisé pour plusieurs championnats différents.

En revanche, la relation liant un circuit à sa ville est une relation de composition : il n'est pas possible de déplacer un circuit d'une ville à l'autre, et, si une ville est détruite, alors son circuit le sera également.




Exercice p. 14 Solution n°5

Exercice

Quel symbole permet d'attribuer une visibilité privée à un attribut ou une méthode ?


- ☐ +
- ☒ -
- ☐ #
- ☐ !

 Le symbole - permet d'attribuer une visibilité privée. Le + est une visibilité publique, et # une visibilité protégée, et le ! n'existe pas.

Exercice


Les classes d'un diagramme de classes doivent obligatoirement avoir des attributs et des méthodes, sinon les spécifications ne sont pas valides et ne peuvent pas être montrées au client.

- ☐ Vrai
- ☒ Faux

 La rédaction des spécifications, qu'elles soient techniques ou fonctionnelles, se fait par itérations. Il est souvent nécessaire de réaliser un premier diagramme de classes simpliste, sans attributs ni méthodes, et de le faire valider avec le client avant de continuer.


Exercice

À quoi sert une association ?

- ☒ À lier deux classes entre elles
- ☐ À spécifier les attributs d'une classe
- ☐ À lier une classe avec ses instances
-  Une association permet de lier deux classes entre elles.


Exercice

Comment se lit la cardinalité $1..*$ entre deux classes A et B ?

- ☐ « Un A est lié à un seul B »
 - ☐ « Un A est lié à au plus un B »
 - ☒ « Un A est lié à plusieurs B »
 - ☐ « Un A peut être lié à plusieurs B »
-  La cardinalité $1..*$ permet de préciser que, pour un A, il peut y avoir au minimum un B, et le nombre maximum n'est pas défini.


Exercice

Soit deux classes `Keyboard`, représentant un clavier, et `Key`, représentant une touche de clavier, quelle est la cardinalité de l'association, dans le sens `Keyboard > Key` ? (On ne prend pas en compte le fait qu'on peut démonter les composants.)

- ☐ 0
 - ☐ 1
 - ☐ 0..*
 - ☒ 1..*
-  Le clavier possède $1..*$ touches.


Exercice

Quelle est la cardinalité de l'association précédente, mais cette fois dans le sens `Key > Keyboard` ?

- ☐ 0
 - ☒ 1
 - ☐ 0..*
 - ☐ 1..*
-  Une touche appartient à 1 clavier.


Exercice

Quel type d'association pourrait lier les classes `Keyboard` et `Key` ?

- ☐ Association simple
- ☒ Composition
- ☐ Agrégation
-  Étant donné qu'on ne prend pas en compte le fait que l'on peut démonter les composants, alors la relation entre un clavier et une touche est une relation de composition : si le clavier est détruit, ses touches le seront aussi.


Exercice

Comment est représentée une classe d'association ?

- ☐ Un trait plein
- ☐ Un trait plein avec un diamant blanc au bout
- ☐ Un trait plein avec un diamant noir au bout
- ☒ Un trait pointillé
- ☐ Une flèche
-  Une classe d'association est représentée par un trait pointillé.


Exercice

Comment transformer une association n-aire en associations binaires ?

- ☒ En créant une nouvelle classe qui va permettre de relier les autres (à la place du diamant)
- ☐ En créant des associations entre chaque classe de l'association n-aire
- ☐ En remplaçant l'association n-aire par une classe d'association
- ☐ Ce n'est pas possible
-  Il est recommandé de n'avoir que des associations binaires dans son diagramme de classes. Ainsi, il est possible de transformer une association n-aire en plusieurs associations binaires en créant une nouvelle classe servant à faire le lien.

Exercice

Quelle est la couleur du diamant permettant de définir une relation de composition ?

- ☐ Blanc
- ☒ Noir
- ☐ Vert
- ☐ Il n'y a pas de diamant pour les compositions
-  La composition est précisée par l'ajout d'un diamant noir, et l'agrégation par un diamant blanc.

Les auteurs sont dérivés des utilisateurs : il y a donc une relation d'héritage entre la classe des utilisateurs et celle des auteurs.

Étant donné que l'affichage des articles dépend de leur auteur (puisque leur signature est présente à la fin), mais que les auteurs peuvent être inter-changés, il y a donc une relation d'agrégation entre un article et un auteur.

En revanche, les commentaires étant détruits en cas de suppression de leur utilisateur ou de leur article, alors la classe gérant les commentaires a une relation de composition avec eux.

Enfin, un article n'a pas besoin d'une étiquette : ce n'est qu'une information additionnelle. Leur relation est donc une association simple.

