Association/Agrégation/Composition

1. Classes d'usage courant

1. System(java.lang)

Cette classe possède de nombreuses fonctionnalités pour utiliser des services du système d'exploitation.

La classe System définit trois variables statiques qui permettent d'utiliser les flux d'entrée/sortie standards du système d'exploitation.

Variable	Type	Rôle
in	InputStream	Entrée standard du système. Par défaut, c'est le clavier
out	PrintStream	Sortie standard du système. Par défaut, c'est le moniteur
err	PrintStream	Sortie standard des erreurs du système. Par défaut, c'est le moniteur

La classe System possède trois méthodes qui permettent de rediriger ces flux : setIn(InputStream), setOut(PrintStream) et setErr(PrintStream).
le mode de fonctionnement bien connu dans le langage C a été repris pour être ajouté dans l'API Java avec la méthode printf(). (Java5)

2. String (java.lang)

Une chaîne de caractères est contenue dans un objet de la classe java.lang.String.
On peut initialiser une variable String sans appeler explicitement un constructeur : le compilateur se charge de créer un objet.La classe String possède de nombreuses méthodes dont voici les principales :

Méthodes la classe String	Rôle
charAt(int)	renvoie le nième caractère de la chaîne
compareTo(String)	compare la chaîne avec l'argument
concat(String)	ajoute l'argument à la chaîne et renvoie la nouvelle chaîne
endsWith(String)	vérifie si la chaîne se termine par l'argument
equalsIgnoreCase(String)	compare la chaîne sans tenir compte de la casse
indexOf(String)	renvoie la première position de l'argument dans la chaîne ou null
lastIndexOf(String)	renvoie la dernière position de l'argument dans la chaîne ou null
length()	renvoie la longueur de la chaîne
replace(char,char)	renvoie la chaîne en remplaçant les occurrences du caractère fourni
startsWith(String int)	vérifie si la chaîne commence par la sous chaîne
substring(int,int)	renvoie une partie de la chaine
toLowerCase()	renvoie la chaîne en minuscule
toUpperCase()	renvoie la chaîne en majuscule
trim()	enlève les caractères non significatifs de la chaîne

3. Vector (java.util)

Un objet de la classe Vector peut être considéré comme un tableau évolué qui peut contenir un nombre indéterminé d'objets. Les méthodes principales sont les suivantes :

Méthode	Rôle
void addElement(Object)	ajouter un objet dans le vecteur
Object elementAt(int)	retourne l'objet à l'index indiqué
Enumeration elements()	retourne une énumeration contenant tous les éléments du vecteur
Object firstElement()	retourne le premier élément du vecteur (celui dont l'index est égal à zéro)
int indexOf(Object)	renvoie le rang de l'élément ou -1
void insertElementAt(Object, int)	insérer un objet à l'index indiqué
boolean isEmpty()	retourne un booléen si le vecteur est vide
Objet lastElement()	retourne le dernier élément du vecteur
void removeAllElements()	vider le vecteur
void removeElementAt(int)	supprime l'objet à l'index indiqué
void setElementAt(object, int)	remplacer l'élément à l'index par l'objet
int size()	nombre d'objets du vecteur

2. Association

Une association est une relation entre deux classes (association binaire) ou plus (association n-aire), qui décrit les connexions structurelles entre leurs instances. Une association indique donc qu'il peut y avoir des liens entre des instances des classes associées.

3. Aggrégation

Une association simple entre deux classes représente une relation structurelle entre pairs, c'est-à-dire entre deux classes de même niveau conceptuel : aucune des deux n'est plus importante que l'autre. Lorsque l'on souhaite modéliser une relation tout/partie où une classe constitue un élément plus grand (tout) composé d'éléments plus petits (partie), il faut utiliser une agrégation.

Une agrégation est une association qui représente une relation d'inclusion structurelle ou comportementale d'un élément dans un ensemble. Graphiquement, on ajoute un losange vide du côté de l'agrégat (cf. figure 3.19). La signification de cette forme simple d'agrégation est uniquement conceptuelle. Elle ne contraint pas la navigabilité ou les multiplicités de l'association. Elle n'entraîne pas non plus de contrainte sur la durée de vie des parties par rapport au tout.

4. Composition

La composition, également appelée agrégation composite, décrit une contenance structurelle entre instances. Ainsi, la destruction de l'objet composite implique la destruction de ses composants. Une instance de la partie appartient toujours à au plus une instance de l'élément composite : la multiplicité du côté composite ne doit pas être supérieure à 1 (i.e. 1 ou 0..1). Graphiquement, on ajoute un losange plein du côté de l'agrégat.

5. Implémentation

- Unidirectionnelle/bidirectionnelle
- Cardinalité 0,1,…,n

Références

[1] http://tahe.developpez.com/java/cours/temp/?page=current_classes#LIII-C

[2] https://www.jmdoudoux.fr/java/dej/chap-packages-base.htm#packages-base-4

[3] https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-aggregation-vs-composition/

[4] https://www.infoworld.com/article/3029325/application-development/exploring-association-aggregation-and-composition-in-oop.html

[5] http://laurent-audibert.developpez.com/Cours-UML/?page=diagramme-classes

[6] https://beginnersbook.com/2013/05/association/