Atelier 1 – Bases de l'orienté Objet et notations UML

Fonctionnement des ateliers

Les ateliers permettent de préparer des exercices pour les séances de laboratoire. Certains ateliers introduisent également de nouveaux concepts simples qui seront approfondis par la suite.

Pour le bon déroulement du cours et pour une efficacité optimale du dispositif d'apprentissage, chacun doit jouer le jeu. Nous avons tenu compte de vos attentes et remarques (à savoir une meilleure clarté sur le déroulement, un plus grand suivi dans la découverte de la matière et de meilleures bases théoriques pour la réalisation des ateliers) en adaptant la pédagogie du cours.

Nous attendons, dès lors, de votre part un engagement quant au travail à fournir. Concernant les ateliers, il vous est demandé de les avoir faits avant le cours suivant. Et ce, afin de pouvoir passer du temps pendant les séances de théorie et de laboratoire sur ce qui est le plus important. Le temps est prévu dans votre horaire pour faire ces ateliers et, si vous travaillez bien pendant les heures prévues à cet effet, il n'y aura que très peu de travail à préparer chez vous.

En cas d'incompréhension de la matière, n'hésitez pas à poses des questions. C'est le meilleur moyen d'apprendre. Ne vous laissez pas déborder car la matière est évolutive et une incompréhension pourrait vous handicaper pour la suite.

Cet atelier est prévu pour deux séances d'atelier. Bon travail !

A. Extrait du premier atelier d'intro progra Q1

Vous avez déjà entendu parler d'objet et de classe au Q1. Vous ne vous rappelez plus ? Voici un extrait :

Dans le Shell, tapez l'instruction suivante.

```
type(x)
```

Vous devriez obtenir le résultat présenté à la Figure 1.

```
>>> x = 5
>>> id(x)
140723744531408
>>> type(x)
<class 'int'>
```

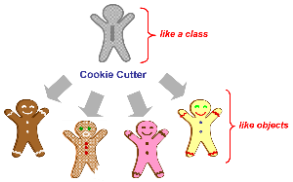
Figure 1 - Type de variable

À nouveau, procédez de la même façon pour les variables proposées ci-dessus.

Classe et objet

Devant le nom du type (ici `'int'`) est inscrit le mot `'class'`, qui signifie classe.

Comme vous l'avez vu lors de la présentation du cours, il existe plusieurs paradigmes de programmation (procédural, fonctionnel, logique…). Il existe aussi plusieurs façons d'analyser un problème et donc d'écrire le programme qui permet d'y apporter une solution. L'une d'entre elles est la programmation orientée objet. Cette façon de rédiger un programme repose sur les concepts de classe et d'objet. Vous aborderez les nombreux autres concepts liés à cette façon de faire dans la suite du cours voire même de votre formation.



Bien que, dans les scripts que vous allez écrire dans le cadre de ce cours, vous n'allez jamais écrire vos propres classes, il semble utile de comprendre ce que ces termes cachent. En effet, en Python, toutes les valeurs que vous allez manipuler sont en fait des objets.

On pourrait voir la classe comme un moule à gâteau et les objets comme les gâteaux fait à partir de ce moule.

Une **classe** correspond à un « type ». Les **objets** sont les « valeurs » produites lorsqu'on instancie une classe. Les objets générés sur base de la classe sont appelés les instances de la classe.

Vous vous rappelez maintenant? Et bien, on va se lancer dans la programmation orienté objet !

B. Quelques questions

A ce stade, le premier cours théorique vous aura normalement permis de comprendre pas mal de choses concernant les objets et les classes… Voici quelques questions pour tester votre compréhension :

Citez 2 grandes différences entre la programmation orientée objet et la programmation procédurale.

Que signifie « couplage fort » ?

1. le fait qu'un objet regroupe à la fois des données et du code
2. le fait que deux bouts de code dépendent fortement l'un de l'autre
3. le fait que deux objets puissent s'envoyer des messages mutuellement
4. le fait que deux fonctions s'appellent l'une l'autre

Lequel de ces élément suivant n’a rien à voir avec l’encapsulation ?

1. Les données et le code associé forment un tout.
2. On peut utiliser un objet sans connaître son fonctionnement interne.
3. On peut contrôler les modifications des valeurs.
4. Le code généré est forcément plus clair et plus "clean".

Parmi les propositions suivantes, laquelle n’est pas une conséquence de l’encapsulation ?

1. une sécurité accrue
2. un couplage plus faible
3. un code plus facile à modifier
4. un code plus efficace

En général, dans quelle partie de la capsule se trouvent les valeur de l’objet ?

Qu’est ce qui peut distinguer des objets différents ?

Qu’est-ce que la « granularité » ?

Que sont les sélecteurs et les modificateurs ?

Par convention, comment écrit-on les noms de classes ?

C. Modéliser des classes

Par **convention** entre nous, lors des exercices et sauf avis contraire, il vous est demandé quand vous modélisez une classe de spécifier :
- Le nom de la classe
- Les attributs de la classe
- Le type des attributs booléens
- Si un attribut est facultatif ou composite
- Si un attribut demandé est dérivable

Pour chaque phrase suivante, modélisez la classe (avec attribut) correspondant :
- Un point possède une abscisse et une ordonnée
- La Belgique a pour capitale Bruxelles. Paris est la capitale de la France. La ville-capitale du Luxembourg est Luxembourg.
- Pour chaque employé, on doit pouvoir connaître son nom, son ou ses prénoms (de 1 à 3), éventuellement son nom de jeune fille, s'il est syndiqué ou pas et, le cas échéant, son syndicat, sa date d'embauche et le nombre d'années d'ancienneté dans l'entreprise.

Quelles critiques/remarques pouvez-vous faire sur ces classes ?

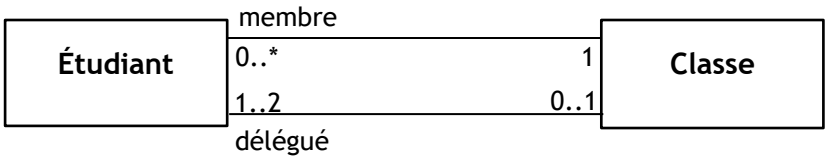
Étudiants
nom dateNaissance email âge possèdeUnKot [0..1]

article
code libellé qttStock valeurUnitaire valeurStockTotal

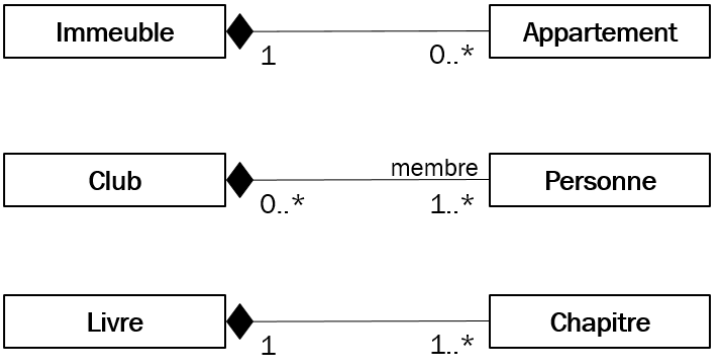
D. Associations

Qu'est ce qu'une association en UML ? Et quels sont les deux cas particuliers ?

Comment doit-on lire les associations suivantes :



Les exemples de compositions suivants sont-ils correctes ?



E. Relations entre les classes

Modélisez sous la forme de diagrammes UML.

- Un électeur vote pour au moins un candidat

- Un électeur vote pour au plus un candidat

- Une droite est déterminée par deux points. La droite porte un nom et chaque point est déterminé par ses coordonnées en X et en Y.

- Un cours, caractérisé par un titre et un nombre d'heures, aborde un ou plusieurs sujets. Chaque sujet est décrit par un nom et un niveau d'importance (entre 1 et 5). À chaque sujet est associé des questions (d'interrogation) décrites par un énoncé et un niveau de difficulté. Une question peut couvrir entre 1 et 3 sujets.
  Une interrogation, caractérisée par une date, comporte entre 5 et 10 questions principales auxquelles viennent s'ajouter au maximum 3 questions facultatives.

- Un plat est décrit par son titre, son concepteur et le fait qu'il est végétarien ou pas. Tout plat est associé à une catégorie caractérisée par un libellé (entrée froide, entrée chaude, dessert…).

Modélisez les classes suivantes ainsi que leurs relations :

- Une classe **Section** avec un nom.
- une classe **Bloc**, chaque bloc (d'étude à l'Henallux) étant décrit par un numéro (1, 2 ou 3) et une section ;
- une classe **Étudiant**, chaque étudiant étant décrit par ses nom et prénom, sa date de naissance, son bloc d'étude et le fait qu'il est finançable ou pas ;
- une classe **Professeur**, chaque professeur étant décrit par son nom, son prénom, sa spécialité et éventuellement la section à laquelle il est attaché.
- une classe **Cours**, chaque cours étant décrit par un nom, un nombre d'heures, le bloc d'étude où il est donné et le(s) professeur(s) qui s'en charge(nt) ;
- une classe **Voiture**, chaque voiture (d'un professeur) étant décrite par sa plaque et sa marque (pour le contrôle du parking).

F. Exercices supplémentaires

Faites les diagrammes UML correspondant aux énoncés suivants :

Bibliothèque

Un livre est décrit par un code, un auteur, un titre, une date de parution et, éventuellement, une maison d'édition. Il peut exister en plusieurs exemplaires, chacun possédant un numéro et une localisation dans la bibliothèque. Il y a au moins un exemplaire par livre mémorisé.

Chaque livre peut être écrit par un ou plusieurs auteurs dont on retient les nom et prénom. Tous les livres ont au moins un auteur. On souhaite également préciser le nom et l'adresse complète de la maison d'édition.

Chaque client de la bibliothèque a un nom, un prénom et une adresse et peut faire un emprunt. Pour chaque emprunt, on veut connaitre la date et la date supposée de retour des livres.

Au sein de la bibliothèque, il y a un système de parrainage. En effet, un client peut en parrainer d'autres. En fonction du nombre de filleuls, le client bénéficie de réduction.

Système de fichier

Sur un serveur, chaque fichier et chaque dossier portent un nom. En plus du nom, dans le cas des fichiers, on retient également la taille et éventuellement l'extension. Un fichier peut être caché ou non.

Un utilisateur, dont on connait le login, a des droits d'accès différents selon les fichiers auxquels il accède (lecture seule, lecture et écriture, exécution…).

On veut aussi pouvoir déterminer (a) le nombre de fichiers cachés, (b) le nombre de fichiers visibles et (c) le nombre total de fichiers pour chacun des dossiers. Comment le préciser sur le diagramme UML ?

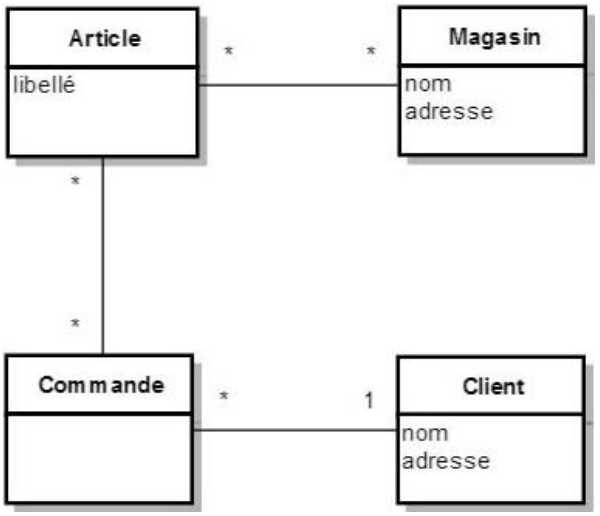
Que faudrait-il ajouter pour garder une trace (la date par exemple) de chaque accès d'un utilisateur à un fichier ?

Place dans un avion

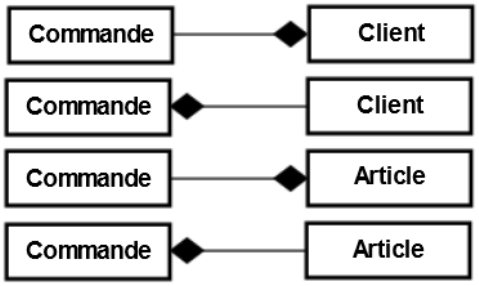
On désire modéliser les sièges dans un avion. Les sièges sont groupés en rangées identifiées par un numéro. Une rangée peut être en classe affaire ou en classe économique, et comporte un certain nombre de sièges identifié par une lettre. Un siège peut être côté hublot ou pas ; il peut offrir un plus grand espace pour les jambes.

Magasin

Considérez le diagramme UML suivant. Le contexte est celui d'un service en ligne permettant à des clients de passer commande d'un certain nombre d'articles qui seront ensuite achetés pour eux dans les magasins de la région puis livrés à leur domicile



a) Indiquez si chacune des compositions suivantes a du sens ou pas.



b) Complétez le diagramme si on désire également retenir le prix de vente de chaque article ainsi que les quantités demandées dans chaque commande.

c) Le diagramme UML ne cite aucun attribut pour la classe Commande. Cela signifie-t-il qu'à l'implémentation, cette classe n'aura que des méthodes, aucun attribut ?

d) Certains clients aimeraient pouvoir passer une commande à livrer chez une autre personne (pensez par exemple aux individus qui veulent passer une commande à livrer chez une personne âgée ou sans internet). Complétez le diagramme pour rendre cela possible.