Algorithmique et programmation orienté objets en C#

Maher REBAI: Enseignant en informatique à
Ecole supérieure d'ingénieurs Léonard de Vinci
Paris la défense(ESILV)
ECE de Paris

Année Universitaire : 2018-2019

Chapitres

- Révision
- Conception de classe
- Collections
- Héritage/polymorphisme
- Classe abstraite et interface
- La délégation
- Suivi de projet

Concept de classe

Qu'est-ce que la programmation orientée objet ?

- Une technique visant à faire interagir des objets entre eux, pour créer un programme.
- Exemple: Traitement des comptes bancaires dans une agence
  - On crée un objet « Client » et un autre objet « Compte »
  - On associe à chaque objet « Compte » un objet « Client »
  - Les objets « Client » et « Compte » vont interagir ensemble pour permettre de créer un programme de gestion des comptes bancaires de l'agence.

Intérêts la programmation orientée objet ?

- Faciliter l'évolution du code.
- Améliorer la conception du code
- Améliorer la maintenance du code

Un objet ?

- La POO permet de manipuler des objets, mais qu'est-ce que c'est concrètement des objets ? .
- Un **objet** est un conteneur symbolique et autonome qui contient des propriétés (caractéristiques) et des actions possibles (méthodes) qu'on peut leur demander.
- **Exemple**: l'objet compteBancaire. il a comme caractéristiques: un numéro, une valeur du solde, un propriétaire, etc.

Comme mécanisme, on peut demander à ce compte d'augmenter sa valeur du solde d'un montant donné ou de la diminuer, etc.

Notion de Classe

- Pour créer un objet, on doit définir ce qu'on appelle une classe.
- Une classe est la « description » d'une collection objets homogènes.
- Une classe regroupe d'une part des variables qui seront les carcatéristiques de l'objet, et d'autre part des fonctions qui seront les méthodes (fonctionnalités) de cet objet.

Notion de Classe

- Une classe doit nécessairement comporter les éléments suivants :
  - les attributs: Variables permettant de définir l'état de l'objet.
  - Un ou plusieurs constructeurs : ce sont des fonctions indiquant de quelle façon l'objet doit être déclaré et initialisé.
  - Les méthodes : ce sont les fonctions permettant d'agir sur les objets d'une classe.

Exemple de classe: Personne

Exemple de classe Personne dont on veut stocker les nom, prénom, âge et numéro de téléphone tout étant capable de modifier et d'afficher ces informations.

```
public class Personne
{
     //Déclaration des champs
...

     //Déclaration des contructeurs
...

     //Déclaration des méthodes
...
}
```

Les attributs

- Représentent les « variables » traduisant l'état de l'objet.
- Ils peuvent être de toutes natures : des entiers, des chaînes de caractères, des listes, d'autres objets, etc...
- Par convention, le nom d'un attribut commence toujours par une minuscule.

```
public class Personne
{
        private string nom;
        private string prenom;
        private int age;
        private int telephone;
}
```

Fig 1: déclaration des attributs de la classe *Personne*

Les attributs

- Le mot clef ***public*** devant le mot clef class indique que la classe *Personne* est accessible depuis n'importe quelle partie du programme.
- Le mot clef ***private*** devant les attributs indique que les attributs ne sont pas accessibles en dehors de la classe personne.
- ***public*** ou ***private*** devant un attribut représente la portée de l'attribut. Il existe d'autres portée comme ***static*** et ***protected*** (nous allons les voir dans la suite du cours)

Constructeur

- Fonction (méthode) appellée au moment de la création d'un objet à partir d'une classe.
- Il permet d'initialiser correctement un objet, à travers la réservation de la mémoire nécessaire pour tous les attributs de l'objet.
- Le constructeur doit avoir le même nom de la classe
- Le constructeur peut ne pas prendre des paramètres comme il peut prendre des paramètres
- Plusieurs constructeurs peuvent être définis dans une classe.

Constructeur

```
public class Personne
{
    //Champs
    private string nom;
    private string prenom;
    private int age;
    private int telephone;

    //Constructeurs
    public personne(){} //Constructeur par défaut

    public personne(string nom, string prenom,
         int age, int telephone)
    {
         this.nom = nom;
         this.prenom = prenom;
         this.age = age;
         this.telephone = telephone;
    }

    public personne(personne p)
    {
         this.nom = p.nom;
         this.prenom = p.prenom;
         this.age = p.age;
         this.telephone = p.telephone;
    }
}
```

Propriétés

- Méthodes méthode qui va permettre d'accéder aux variables des objets en lecture (accesseur) et en écriture (mutateur).
- Grâce aux accesseurs, on pourra afficher les valeurs des variables de l'objets (attributs)
- Grâce aux mutateurs, on pourra modifier les valeurs des variables de l'objets (attributs)
- Une propriété permet de définir des accès en consultation (get) et/ou en modification (set).

Propriétés

En rendant les champs privés, un programmeur utilisateur de la classe personne ne pourra pas lire, ni écrire de valeur dans ces variables. Cependant, il se peut que le programmeur ait besoin de connaitre ces informations et ait aussi besoin de les mettre à jour. Pour cela, il est possible de définir des propriétés pour accéder à ces variables

```
public int Age
{
    get
    {
        return this.age;
    }

    set
    {
        this.age = value;
    }
}
```

Méthodes

- Sont des « sous-porgrammes » prêt à agir sur les objets d'une classe.
- Elles sont assimilables à des fonctions sauf qu'elles sont généralement

appelés par des objets et agissent potentiellement sur eux.

- Quand une méthode s'effectue sur une instance particulière de la classe (c'est-à-dire sur un seul objet) on l'appelle : **méthode d'instance.**
- Quand une méthode s'effectue sur chaque instance de la classe (c'est-à-dire sur chaque objet créé) on l'appelle : **méthode de classe.**

Méthodes

Note:

1. Ce principe de méthode d'instance et de méthode de classe fonctionne de la même façon pour les attributs et pour les propriétés.
2. Pour déclarer un attribut de classe (ou méthode ou propriété), il suffit d'ajouter le mot clef ***static*** qui indique que cet attribut est un attribut de classe.

Principe d'encapsulation

- Lorsqu'un objet dont les variables sont protégées de l'extérieure de la classe, elles ne sont accessibles que via les propriétés (accesseurs et mutateurs) que l'objet propose. C'est le principe d'encapsulation.
- L'encapsulation permet donc de garantir l'intégrité des données contenues dans l'objet.
- L'encapsulation permet de définir des niveaux de visibilité des éléments de la classe. Ces niveaux de visibilité définissent les droits d'accès aux données

Modélisation des objets avec UML (diagramme des classes)

Le langage de modélisation unifiée [unified modeling langage (UML)] est un ***langage de modélisation graphique*** à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet

http://fr.wikipedia.org/wiki/UML_(informatique)

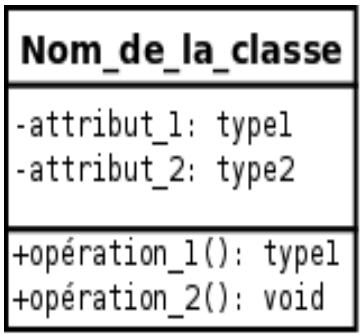
Modélisation des objets avec UML (diagramme des classes)

Les diagramme d'UML:

- Vue statique:
  - ✓ Diagramme des classe
  - ✓ Diagramme d'objet
  - ✓ Diagramme de cas d'utilisation
  - ✓ Diagramme de déploiement
- Vue dynamique:
  - ✓ Diagramme de séquence
  - ✓ Diagramme de collaboration
  - ✓ Diagramme d'activités

Modélisation des objets avec UML (diagramme des classes)

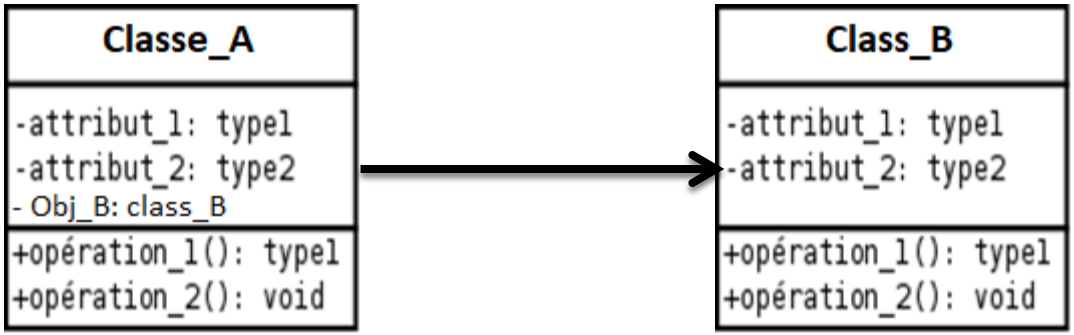
- Modélisation d'une classe:



- une classe est modélisée sous la forme représentée sur la figure ci-dessus
- La portée des attributs et des méthodes peut être:
  - ✓ **+ (Public) :** l'attribut ou la méthode peut être vu en dehors de la classe.
  - ✓ **- (Private):** l'attribut ou la méthode ne peut pas être vu en dehors de la classe.

Modélisation des objets avec UML (diagramme des classes)

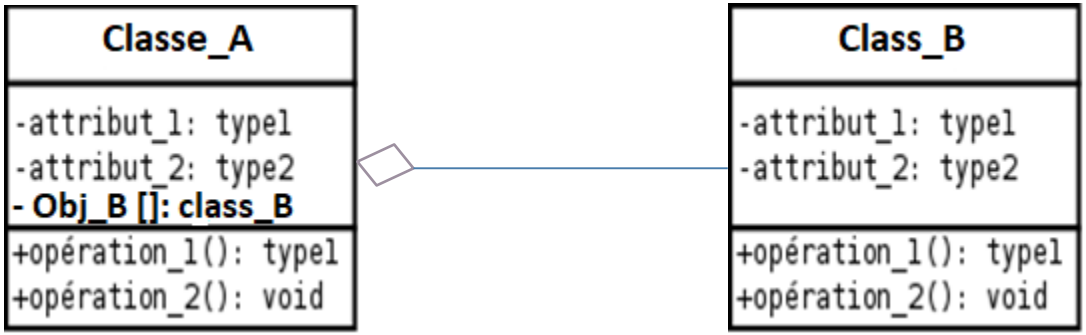
- Modélisation des liens entre les classes: Contenance



- tout objet de la classe A possède une variable de classe de type class_B.
- *une seule* instance d'ObjetB est présente dans ObjetA.

Modélisation des objets avec UML (diagramme des classes)

- Modélisation des liens entre les classes: Agrégation



- tout objet de la classe A possède plusieurs variables de classe de type class_B.
- *Plusieurs* instances d'ObjetB sont présents dans ObjetA.

Exercice

Déterminer le diagramme UML de l'exercice du TD