Initiation à la Programmation orientée objet en C#

NORA IZRI – DÉPARTEMENT IBO

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

Pourquoi ?

Soit les deux exemples suivants :

1. Calculer le périmètre d'un rectangle avec la formule : (largeur + hauteur)/2
   - Définir deux variables de type « double » largeur et hauteur
   - Demander à l'utilisateur la saisie des valeurs
   - Faire le calcul selon la formule et retourner le résultat
2. Comparer deux voitures selon le modèle, la marque et la puissance
   - Définir deux variables de type « Voiture »
   - Comparer selon un ou plusieurs critères
   - Retourner « true » si les deux voitures sont équivalentes

   ➔ Qu'est ce que le type « Voiture »?

   ➔ Comment l'utiliser? Existe-t-il? Doit-on le créer?

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

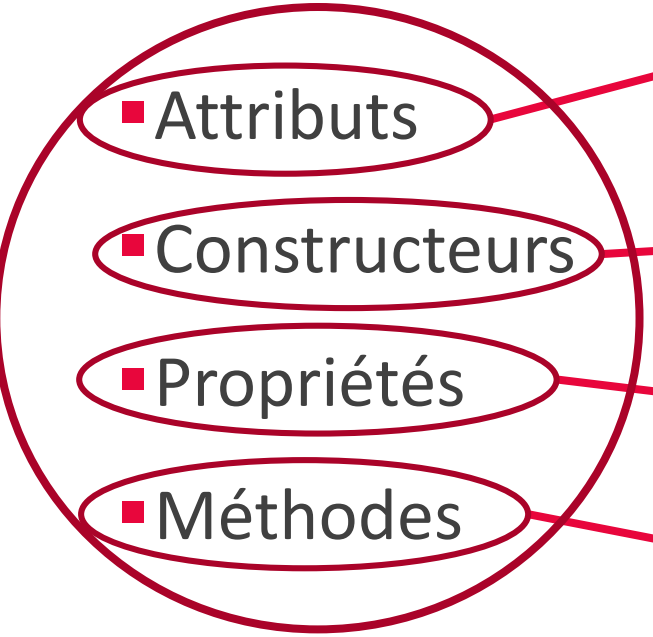
POO ?

Programmation orientée objets

- **Principe d'encapsulation des données** :
  - Consiste à protéger l'information (caractéristiques ou attributs) contenue dans un objet et de ne proposer que des méthodes de manipulation de cet objet.
  - Les attributs sont généralement inaccessibles depuis l'extérieur de l'objet : seules les méthodes sont accessibles
- **Classe** : description d'un ensemble d'objets ayant une structure de données commune et disposant des mêmes méthodes
- **Objet** : **instance** d'un type **classe**
- Exemples :
  - Personne est une classe. « Durand Daniel » est une instance de la classe Personne.
  - String est une classe. « Bonjour » est une instance de cette classe donc un objet.

Classe ?

- Une classe comporte :



Corps d'une classe

Décrire l'état de l'objet.
Exemple:
Une personne : un nom, un prénom, une date de naissance, une adresse.

Initialisation des attributs .
Exemple:
Une personne : Durand (nom), Daniel (prénom).

Accès aux attributs (setter/getter).

Actions qu'un objet peut effectuer.

Comment définir une classe?

- Chaque classe est définie dans un **fichier « .cs » indépendant**
- Mot-clé **class**
- Syntaxe :

```
class NomDeLaClasse {
      Corps de la classe
}
```

- Exemple :

```
public class Voiture
    {
        //attributs
        private int kilometres;
        private double essence;
        private bool roule;
    }
```

public : accessible depuis n'importe quelle partie du programme

private : accessible uniquement au sein de la classe. Les attributs private ne sont pas accessibles en dehors de la classe Voiture

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

Instance d'une classe (1/2)

- Mot-clé **new** ➔ allouer l'espace mémoire nécessaire pour l'objet
- Soit en deux étapes :
  - Déclaration de la variable de type classe

```
Voiture maVoiture;
```

  - Création de l'objet via l'un des constructeurs de la classe

```
maVoiture = new Voiture();
```

Instance d'une classe (2/2)

- Instanciation en une instruction (version combinée) :

```
Voiture maVoiture = new Voiture();
```

- Le « **new** » doit être exécuté pour que l'espace mémoire destinés aux attributs de l'objet soit alloué.

Exemple > La classe Voiture

```
public class Voiture
    {
        //attributs
    private int kilometres;
    private double essence;
    private bool roule;
    private int capacite;
    public const int CAPACITE_RESERVOIR = 50;
         //constructeurs
        public Voiture()
        {
            this.kilometres = 0;
            this.essence = 0;
            this.roule = false;
            this.capacite = 0;
        }
```

```
public Voiture(int k, int c)
        {
            this.kilometres = k;
            this.essence = CAPACITE_RESERVOIR;
            this.roule = false;
            this.capacite = c;
        }

//méthodes
public string Affichage()
        {
            return "Kilomètres : " +
                    this.kilometres + " ,
Capacité : " + this.capacite + " \n Il reste :
" + this.essence + " dans le réservoir";
        }
```

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

Constructeur?

- Constructeur
  - ➔ méthode portant toujours **le nom de la classe**.
  - ➔ ne retourne aucun type, même pas void
  - ➔ initialisation des attributs d'une instance de classe.
- Deux types :
  - constructeur par défaut : est un constructeur qui ne reçoit pas de paramètres d'entrée
  - constructeur paramétré : tout comme pour une méthode, il peut attendre des paramètres d'entrée

Initialisation d'un objet

- Après avoir créé une instance de la classe, il faut initialiser les membres (les attributs).

```
class Program
    {
        static void Test1(){ …  }

        static void Test2(){ …}

        static void Main(string[] args)
         {
             Test1();
             Test2();
             Console.ReadKey();
         }
}
```

Initialisation d'un objet

```
static void Test1()
        {
            Voiture maVoiture = new Voiture(1000, 5);

            maVoiture.kilometres = 200; // FAUX : Car kilometres n'est pas
                                        // visible à l'extérieur
            Console.WriteLine(maVoiture.capacite);

        }
```

```
static void Test2()
        {
            Voiture maVoit;
            maVoit = new Voiture();
            maVoit.Kilometres = 1000;
        }
```

Il faut rajouter les propriétés!!!!

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

Propriétés ?

- Les propriétés permettent d'accéder de manière indirecte aux attributs privés de la classe.
- Une propriété porte le nom de l'attribut à rendre accessible (en commençant par une majuscule). Elle peut contenir l'un des deux blocs de code ou les deux :
  - le bloc de code **get** → **lecture de la propriété** → lecture
  - le bloc de code **set** → **affectation d'une valeur à la propriété** → écriture
- Il est nécessaire de définir les propriétés pour respecter le principe d'encapsulation → garantir l'intégrité des données contenues dans un objet.

Exemple > Définition des propriétés

```
//propriétés
        public int Kilometres
        {
            get { return this.kilometres;}
        }
        public bool Roule
        {
            get { return this.roule; }
            set { this.roule = value; }
        }
```

```
 public int Capacite
 {
        get { return this.capacite; }
        set { this.capacite = value; //en cas de
modification du nombre de places
             }
 }
  public double Essence
  {
        get { return this.essence; }
        set { this.essence = value; }
  }
```

Exemple > La classe Voiture

Après avoir défini toutes les propriétés. Est-ce que le code ci-dessous est juste?

```
class Program
    {
        static void Test()
        {
            Voiture maVoiture = new Voiture(1000, 5);

            maVoiture.kilometres= 200; // FAUX : Car kilometres est accessible en
lecture seulement

           Console.WriteLine(maVoiture.Capacite);

        }
```

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

Méthodes d'une classe?

- Sous-programmes prêt à agir sur les objets d'une classe.
- Définir les méthodes liées à la classe au sein de la classe
- Les méthodes s'appliquent à des instances de la classe ➔ elles ne sont pas statiques.

Exemple > La classe Voiture

On complète cette classe Voiture, par les deux méthodes suivantes :

```
        //méthodes
        public void Rouler()
        {
            this.roule = true;
            while(this.roule && this.essence > 0)
            {
                this.kilometres++;
                this.essence -= CONSOMMATION;
            }
        }

        public void Stopper()
        {
            this.roule = false;
        }
    }
```

Exemple > Appel aux méthodes

```
class Program
    {
        static void Test()
        {
            Voiture maVoiture = new Voiture(1000, 5);

            maVoiture.Rouler();
            maVoiture.Stopper();

            Console.WriteLine(maVoiture.Capacite);
            Console.WriteLine(maVoiture. Affichage());

        }
```

Visibilité d'une classe

Mot clé	signification
public	Accès non limité
private	Accès restreint au type conteneur (modificateur d'accès au membre). Les membres privés sont accessibles uniquement dans le corps de la classe où ils sont déclarés.
protected	Accès restreint à la classe conteneur ou aux types dérivés de la classe conteneur
abstract	Une classe abstraite sert de classe de base dans une relation d'héritage (à voir dans le cours de Programmation Orientée Objet Avancée)

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

Membres partagés/statiques ?

- Mot-clé : **static**
- Gérer dans une classe, des informations qui ne sont pas spécifiques à une instance de classe mais à la classe elle-même.
- La modification de la valeur d'un membre partagé modifie la valeur pour toutes les instances de la classe.
- Assimilables à des variables globales. Ils sont utilisables uniquement en y faisant **référence par le nom de la classe**.
- L'utilisation d'un membre statique **par une instance** de classe **est interdite**.
- La portée d'un attribut statique est **la classe et non l'objet**.

➔ tous les objets partagent la même valeur.

Exercice1

1. Créer la classe Client. Un client est caractérisé par son nom, son prénom, son numéro de sécurité sociale, son numéro de portable, son adresse postale et son adresse Email.
2. Créer un premier client «Dupont Jean » dans le fichier « Program.cs ».
3. Afficher les informations d'un client ➔ Rajouter la méthode Affichage dans la classe Client
4. Modifier le numéro de téléphone d'un client ➔ Rajouter des propriétés pour les attributs dans la classe Client. ATTENTION: pas de set et get systématiques pour tous les attributs
5. Dans « Program.cs »; créer un second client.
6. Comment mettre en place un compteur au sein de la classe Client pour gérer le nombre d'objets construits