Chapitre	3. Solutions technologiques
Objectif général de formation	• Identifier une solution technique, • Développer une culture des solutions technologiques.
Paragraphe	3.1 Structures matérielles et/ou logicielles
Sous paragraphe	3.1.4 Traitement de l'information
Connaissances	Programmation objet : structures élémentaires de classe, concept d'instanciation
Niveau d'enseignement	Première Terminale
Niveau taxonomique	**2.** Le contenu est relatif à **l'acquisition de moyens d'expression et de communication** : définir, utiliser les termes composant la discipline. Il s'agit de maîtriser un savoir « appris ».
Commentaire	*Les opérandes simples (somme, différence, multiplication, retard, comparaison) sont extraites de bibliothèques graphiques fournies.* *On se limite aux principes de la programmation objet.* *Pour les systèmes événementiels on utilise les composants programmables intégrés.*
Liens	

Programmation orientée objet (P.O.O.)

La programmation orientée objet est un paradigme de programmation basé sur des éléments logiciels appelés objets.

La P.O.O. est une évolution de la programmation structurée. En programmation structurée, les variables sont basées sur des types de données (caractère, entier, flottant, …), en P.O.O., les variables sont des objets basés sur des classes.

Concept d'objet

Un objet est une **structure logicielle** qui regroupe :

- Des **champs**, ou **attributs**, qui détiennent les données de l'objet ;
- Des **fonctions membres**, ou **méthodes**, destinée à manipuler les champs de l'objet et à interagir avec les autres objets.

Classe et instanciation

Un objet est une **instance** d'une classe. La classe décrit la structure interne des données et elle définit les méthodes qui s'appliqueront aux objets de même famille (même classe).

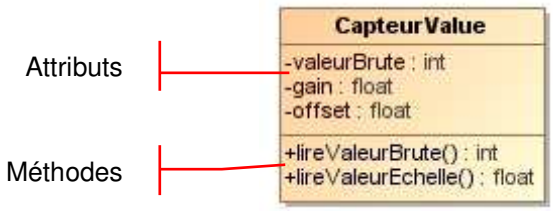

Figure 1 : représentation d'une classe en UML

```
public class CapteurValue
{
    private int valeur;
    private float gain;
    private float offset;

    public int lireValeur() {
        return valeur;
    }

    public float lireValeurEchelle() {
        return valeur * gain - offset ;
    }
}
```

Figure 2 : Définition d'un classe en langage C#

```
CapteurValue temperature = new CapteurValue();
CapteurValue pression = new CapteurValue();
```

Figure 3 : instanciation de 2 objets en C#

Encapsulation

L'accès aux données des objets peut être réglementé

- Données publiques : accès direct (non protégé)
- Données privées (encapsulées) : accès uniquement par les fonctions membres

Conséquences de l'encapsulation :

- Un objet n'est vu que par ses spécifications
- Une modification interne est sans effet pour le fonctionnement général du programme
- Meilleure réutilisation de l'objet

La plupart des environnements objets modernes définissent la notion de **propriétés** d'objets. Les propriétés sont des attributs privés (encapsulés) qui sont associés à 2 méthodes destinées :

- L'une à lire la valeur de la propriété (l' « *accesseur* », noté « ***get*** » en Anglais)
- L'autre à écrire la valeur de la propriété (le « mutateur », noté « **set** » en Anglais)

Héritage

Non traité.
Hors programme

Polymorphisme

Non traité.
Hors programme

Références

[1] Introduction à la P.O.O. : http://hdd34.developpez.com/cours/artpoo/
[2] Introduction aux objets : http://www.developpez.biz/download/introobj.pdf