4) Les méthodes d'accès (ou accesseurs)

Un des grands principes de la conception objet se résume dans la phrase suivante :

Un objet est seul responsable des informations dont il a la charge !

Cela signifie qu'il faut se poser les questions suivantes, en tant qu'objet (moi en tant qu'Article) faut-il, oui ou non que je donne accès à mes attributs, et si oui faut-il donner l'accès en lecture et/ou en écriture ?

Par exemple je décide d'appliquer la visibilité suivante :

	lecture	écriture
ref	oui	non
libelle	oui	non
prixVente	oui	non
prixAchat	non	non
stock	oui	oui

Il faut prévoir en conséquence des méthodes d'accès (ou accesseurs) de type **get** (pour un accès en lecture), de type **set** (pour un accès en écriture).

Ici je suis en conception de classe !

```
public class Article{

    // ...

    //Accesseurs

    //En lecture
    public String getRef(){
        return this.ref;
    }

    // En écriture
    public void setStock(int s){
        this.stock = s;
    }

    // à compléter...
}
```

Remarque : un accesseur en écriture (set) est également appelé un **mutateur**

Exercice 6 :

- Compléter la classe Article pour créer tous les accesseurs en lecture et en écriture conformes à la visibilité décidée,

	lecture	écriture
ref	oui	non
libelle	oui	non
prixVente	oui	non
prixAchat	non	non
stock	oui	oui

- Dans le programme de test, instancier un objet Article avec les valeurs initiales suivantes :

  La référence : **CBMXE**
  Le libellé : **Casque BMX enfant**
  Le prix de vente : **35.00** €
  Le prix d'achat : **29.90** €
  La quantité en stock : **3**

- Afficher la description de l'article en utilisant la méthode **toString(),**

- Afficher uniquement le libellé, le prix de vente et le prix d'achat (est-ce possible ?),

- Le magasin vient de recevoir une livraison de l'article "***casque BMX enfant***", le nouveau stock est maintenant de 10 unités,

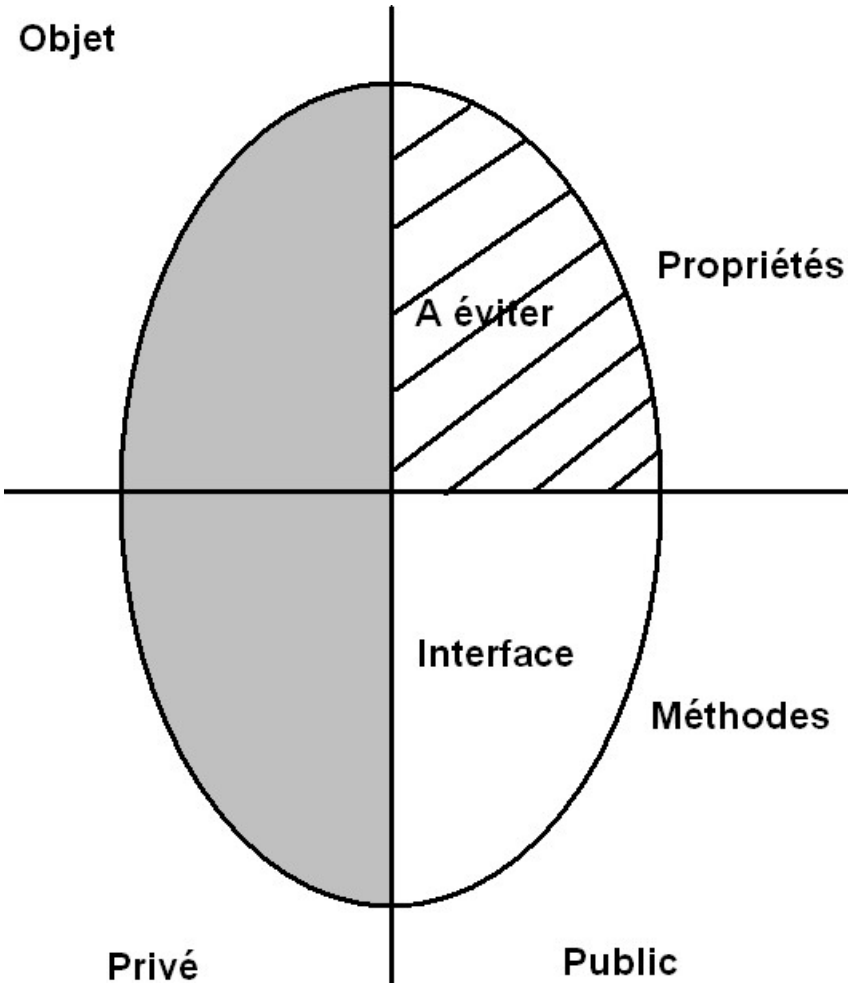
- Afficher la description de l'article en utilisant la méthode **toString()** afin de vérifier le nouveau stock.

- Il apparait plus pratique de créer une méthode stocker() qui recevra en paramètre le nombre d'articles à cumuler au stock actuel. Créer cette méthode et supprimer la méthode setStock() désormais inutile (voire dangereuse).

- A l'aide de l'outil Violet, traduire en UML la classe Article avec ses nouvelles méthodes.

Pour résumer :

Un **objet** peut embarquer à la fois des **attributs** et des **méthodes** (des fonctions et/ou des procédures), ce principe est appelé **encapsulation**.



Un objet est décrit par une structure appelée **classe**.

Dans une classe toutes les propriétés sont déclarées en **privée**. Si l'on veut donner des droits de lecture ou d'écriture sur les attributs d'un objet il faut embarquer des méthodes d'accès (accesseurs) de type **get** ou **set** en respectant le principe :

Un objet est seul responsable des informations dont il a la charge !

L'ensemble des méthodes **publiques** de la classe représente **l'interface** (et donc la seule possibilité de manipuler l'objet).