Programmation orientée objet (UAA14)

Exercices : série 1

Objets d'apprentissage :

- ✔ Modéliser une logique de programmation orientée objet.
- ✔ Déclarer une classe.
- ✔ Instancier une classe.
- ✔ Utiliser les méthodes de l'objet instancié.
- ✔ Traduire un algorithme dans un langage de programmation.
- ✔ Commenter des lignes de code.
- ✔ Tester le programme conçu.
- ✔ Caractériser les attributs dans une classe (encapsulation).
- ✔ Caractériser les méthodes dans une classe (encapsulation).
- ✔ Décrire la création d'un constructeur.
- ✔ Extraire d'un cahier des charges les informations nécessaires à la programmation.
- ✔ Programmer en recourant aux instructions et types de données nécessaires au développement d'une application.
- ✔ Corriger un programme défaillant.
- ✔ Développer une classe sur la base d'un cahier des charges en respectant le paradigme de la programmation orientée objet.
- ✔ Programmer en recourant aux classes nécessaires au développement d'une application orientée objet.
- ✔ Améliorer un programme pour répondre à un besoin défini.

Exercice 1 : créer une classe

Etape 1

Crée une classe nommée `Individu`. Cette classe doit comporter les attributs suivants :

- un prénom ;
- un age ;
- un sexe (M ou F) ;
- un domicile (nom de la ville uniquement)

Etape 2

Crée un constructeur pour la classe `Individu`.

Etape 3

Teste ton code :

- déclare un objet de type `Individu` nommé `moi` ;
- initialise la variable `moi` en utilisant le constructeur avec les valeurs te décrivent ;
- affiche le prénom, le sexe et l'âge de `moi` ;
- modifie le sexe de `moi` en W et l'âge en -124 ;
- affiche de nouveau le prénom, le sexe et l'âge de `moi` ;

Exercice 2 : la sécurité avant tout !

Etape 1

En les laissant publics (visibilité par défaut), les attributs de la classe `Individu` ne sont pas protégés : le « monde extérieur » (d'autres scripts PHP) peut les modifier à sa guise, en y mettant parfois des valeurs non autorisées.

Pour résoudre ce problème, transforme les attributs de la classe `Individu` en attributs privés. Observe que le serveur lance une erreur…

Si rien ne s'affiche, c'est que PHP a désactivé l'affichage des erreurs à l'écran. Pour le réactiver, ajoute ces trois lignes au début de ton script :

```php
ini_set('display_errors', '1');
ini_set('display_startup_errors', '1');
error_reporting(E_ALL);
```

Une convention importante à la programmation O.O est que les variables d'instances d'une classe doivent toujours être privées.

Etape 2

Le fait de mettre un attribut privé empêche toute modification du monde extérieur. Ils sont maintenants bien protégés… mais un peu trop !

Pour permettre la modification des attributs privés d'une classe, il faut utiliser des *getters* / *setters* : des méthodes qui servent de « gardiens » et qui se chargent de donner les accès en lecture / écriture aux attributs, tout en les protégeant.

Définis donc les getters suivants : `getPrénom`, `getAge`, `getSexe` et `getDomicile`

Ensuite, définis les setters suivants :

- `setPrénom`
- `setSexe` : si on tente de mettre un sexe autre que M ou F, ce setter doit ignorer la modification
- `setAge` : si on tente de mettre un âge négatif ou supérieur à 120, ce setter doit ignorer la modification
- `setDomicile`

Tu peux également utiliser les méthodes magiques `__get` et `__set` si tu préfères (c'est même mieux !).

Enfin, corrige ton programme afin d'utiliser les getters / setters. Tente plusieurs modifications autorisées et non autorisées et observe le résultat.

Exercice 3 : méthode simple et efficace

Etape 1

Ajoute à la classe `Individu` une méthode nommée `sePrésenter` qui affiche le texte suivant :

« Je m'appelle `<prénom>` et je suis âgé de `<âge>` an(s). Je réside à `<domicile>`. »

Pour être méticuleux, fais en sorte que le « s » de « an(s) » ne s'affiche que si l'âge est supérieur à 1. Fais en sorte également d'afficher « âgée » ou « âgé » selon les cas.

Teste ensuite cette méthode en l'appelant !

Etape 2

Commente la méthode `sePrésenter` et écris une nouvelle version de cette méthode : cette dernière doit accepter un entier comme argument : 1 pour l'affichage en français, 2 pour l'affichage en anglais et 3 pour l'affichage en néerlandais (pas d'affichage dans les autres cas).

Teste cette nouvelle version avec les entrées 0, 1, 2 et 3.

Etape 3

Dé-commente la première version de `sePrésenter`. Il y a donc maintenant deux méthodes avec le même nom, mais pas avec la même signature. Le code est-il accepté ?

Regarde attentivement les deux versions : une grosse partie de leur code est identique. Comment pourrais-tu faire pour réutiliser du code ?

> ⓘ Le principe du « point de modification unique » est un principe de programmation qui vise à réutiliser, le plus possible, du code déjà présent dans un programme dans le but de n'avoir qu'un seul « endroit dans le code » à modifier lorsque c'est nécessaire.

Références

Les présents exercices ont été élaborés à l'aide des ressources suivantes :

- https://www.pierre-giraud.com/php-mysql-apprendre-coder-cours/introduction-programmation-orientee-objet/
- Cours de « Développement d'applications WEB », Hénallux (2019)