Découverte de la Programmation Orientée Objet en php

Nos animaux familiers sont des objets..... ou comment une animalerie vous permettra de comprendre les concepts de base de la POO et les mécanismes associés ;-)

Objectifs :

- ✓ Découvrir la Programmation Orientée Objet,
- ✓ Se familiariser avec le vocabulaire consacré,
- ✓ Instancier des objets, utiliser des propriétés, utiliser des méthodes.

Préambule

La programmation est passée d'une ère artisanale à une ère industrielle :
Des logiciels de plus en plus complexes doivent être réalisés dans des délais de plus en plus courts, tout en maintenant le meilleur niveau de qualité possible. Cette migration s'est accompagnée de nouveaux langages et de nouvelles techniques de programmation qui tournent autour du concept d'objet. Ces nouvelles technologies permettent de développer des applications informatiques par assemblage d'éléments composites.
Pourquoi apprendre à programmer en objet ?
Tout simplement parce que tous les langages modernes utilisent cette technologie de Java à VB.Net en passant par C#. Tous non seulement fournissent des structures et des composants objets, mais aussi tous permettent à des degrés divers de concevoir ses propres objets.
Tous les outils du programmeur utilisent la technologie Objet : une liste déroulante est un objet, une fenêtre est un objet, un TextBox est un objet etc ...
Plus question donc d'apprendre un langage de programmation sans comprendre cette technologie. Ce support va vous permettre découvrir la programmation objet de manière pratique.

Au palais de la découverte

Vous avez commencé à vous habituer à utiliser des objets, notamment graphiques. Ces objets fonctionnent comme des " boîtes noires " dont vous découvrez petit à petit les fonctionnalités. Voici une petite " boîte noire " que vous allez découvrir et qui va vous permettre de commencer à comprendre les mécanismes objet.
Cette découverte se fait de manière ludique, donc au travers de classes qui n'ont pas d'utilité technique, mais qui permettent d'illustrer l'utilisation de briques pour créer une application.
L'idée de la programmation objet, c'est en effet de :

- ✓ Créer des composants pour soi-même pour d'autres développeurs,
- ✓ Utiliser des composants conçus par soi ou par d'autres développeurs, pour construire rapidement une application.

Lorsqu'un développeur conçoit un modèle d'objet, il fabrique une sorte de moule à gâteau, qui va pouvoir être utilisé et réutilisé à souhait, pour cuire plein de gâteaux.
Le moule est appelé **classe** ; le gâteau c'est **l'objet**, créé, on dit **instancié**, grâce au moule.
Si on raisonne de manière générale, un objet possède des caractéristiques (une couleur, une forme, une taille, etc.) et a un comportement qui lui est propre (un vélo roule, une lampe éclaire, s'allume, s'éteint, etc.)
En programmation objet :

- ✓ Les caractéristiques s'appellent des **propriétés**
- ✓ Le comportement est décrit par un certain nombre d'actions appelées **méthodes**

Découverte de la Programmation Orientée Objet en php

Comment créer une ménagerie avec l'aide du prof ?

La boîte noire fournie par votre professeur va vous permettre de créer une ménagerie.

- ✓ Une ménagerie est un regroupement d'animaux,
- ✓ Un animal est caractérisé par son nom, son espèce, sa couleur,
- ✓ Le comportement d'un animal peut se définir entre autre par la façon dont il " parle ", " mange ", "dort ".

Parmi les animaux, il existe les chiens. Un chien est un animal particulier, qui plus précisément " aboie " (aboyer, c'est parler " Waf ! Waf " par exemple), mais qui aboie plus ou moins fort en fonction de sa taille.

- ✓ Un chien est donc un animal, pour lequel on s'intéresse particulièrement à sa taille, mais aussi à sa race.

Voilà vous avez presque tout ce qu'il faut savoir sur cette fameuse boîte noire.
Vous allez maintenant créer votre propre ménagerie, comportant un certain nombre d'animaux, que vous allez faire parler, etc.
Bon courage !

1) **Copier le dossier AppAnimalerie dans le dossier où vous pouvez interpréter du php.**
2) **Vous y découvrirez les scripts " animalclass.php ", " chienclass.php " et "animalerieclass.php " qui vous permettront de développer votre application.**
3) **Observer les classes définies dans ces fichiers :**
   Quelles sont propriétés de la classe " Animal " ?
   Quelles sont les méthodes de la classe " Animalerie " ?

Comme on déclare une variable de type entier, on peut également déclarer un objet appartenant à une des classes fournies dans les scripts présents.
Une fois ces constatations faites, nous pouvons écrire le code de notre application dont le rôle est, rappelons-le, la création d'une ménagerie.

Utilisation des classes existantes dans l'application

Notre application " AppAnimalerie " va nous permettre de créer une ménagerie comportant quelques animaux, et de surveiller leur comportement ...
La ménagerie comportera dans un premier temps :
Un " Poisson " " rouge " nommé " Nemo " qui bulle : " Gloup !! Gloup !! Gloup!! Gloup !! "
Un " Chat " " noir " nommé " Félix " qui miaule : " Miaou !! Miaou !! Miaou !! Miaou !! "
Une « Perruche » nommée « Pistache » qui chante : « Cui !! Cui !! Cui !! Cui !! »

Commençons à coder cette application :

```
<?php
include "animalclass.php";
$unanimal = new Animal("Babar","elephant");
?>
```

Notion de constructeur

Animal est le **constructeur** de la classe Animal : il s'agit de la méthode qui permet de créer l'objet, **d'instancier** ici un Animal, de lui " donner vie " en quelque sorte, selon le modèle défini dans la classe "Animal ".

Quelques règles :

- ✓ En php le **constructeur** porte toujours le nom de la classe.
- ✓ Un **constructeur** peut admettre des paramètres, qui servent généralement à initialiser ses propriétés.
- ✓ Le mot-clé pour instancier un objet est **new**.

Créons la ménagerie souhaitée :

```
<?php
include "animalclass.php";
$nemo = new Animal("Nemo","Poisson");
$felix = new Animal("Felix","Chat");
$pistache= new Animal("Pistache","Perruche");
//la vie de Nemo se résume ainsi
$nemo->sePresenter();
$nemo->parler("Gloup");
$nemo->manger();
$nemo->dormir();
echo "<BR/>";
//la vie de Felix se résume ainsi
$felix->sePresenter();
$felix->parler("Miaou");
$felix->manger();
$felix->dormir();
echo "<BR/>";
//la vie de Pistache se résume ainsi
$pistache->sePresenter();
$pistache->parler("Hi");
$pistache->manger();
$pistache->dormir();
?>
```

On obtient le résultat suivant à l'exécution :

Je me presente : Poisson Nemo Gloup !! Gloup !! Gloup !! Gloup !!
J ai si faim... Donne-moi un biscuit !
Zzz Zzz Zzz.... (Nemo s'est endormi.e)

Je me presente : Chat Felix
Miaou !! Miaou !! Miaou !! Miaou !! J ai si faim... Donne-moi un biscuit !
Zzz Zzz Zzz.... (Felix s'est endormi.e)

Je me presente : Perruche Pistache Hi !! Hi !! Hi !! Hi !!
J ai si faim... Donne-moi un biscuit !
Zzz Zzz Zzz.... (Pistache s'est endormi.e)

Utilisation de la classe Menagerie

Nous allons maintenant utiliser la classe Ménagerie, qui va nous permettre de rassembler les animaux dans un " zoo ". On en déduit que l'on peut donc avoir un objet (une Ménagerie) qui peut contenir ou être composé d'autres objets (ici des animaux), même si pour l'instant on ne comprend pas exactement ni pourquoi ni comment.
Au vu de cette classe, on peut manifestement :

- ✓ Connaître le nombre d'animaux de la ménagerie
- ✓ Enregistrer l'arrivée d'un animal (dans la ménagerie)
- ✓ Enregistrer le départ d'un animal (de la ménagerie)
- ✓ Présenter l'ensemble des animaux de la ménagerie.

Complétons l'application pour utiliser les différentes possibilités offertes par cette classe :

```php
<?php
include "animalerieclass.php";
include "animalclass.php";
$nemo = new Animal("Nemo","Poisson");
$felix = new Animal("Felix","Chat");
$pistache= new Animal("Pistache","Perruche");
//Création d'une ménagerie
$zoo= new Menagerie;
//enregistrement des animaux qui arrivent dans la ménagerie
$zoo->arriver($nemo);
$zoo->arriver($felix);
$zoo->arriver($pistache);
//Présentation de la ménagerie ainsi constituée
echo "<BR/>"."Le zoo comporte ".$zoo->count()." animaux"."<BR/>";
$zoo->presenter();
//Enregistrement du départ de Félix
echo "<BR/>"."Felix s'en va"."<BR/>";
$zoo->partir($felix);
//Re-Présentation de la ménagerie
echo "<BR/>"."Le zoo comporte maintenant ".$zoo->count()." animaux"."<BR/>";
$zoo->presenter();
//Changement de nom du poisson
$nemo->nom="Maurice";
//Re-Présentation de la ménagerie
echo "<BR/>"."Le zoo comporte maintenant ".$zoo->count()." animaux"."<BR/>";
$zoo -> presenter();
?>
```

On obtient le résultat suivant :

```
Le zoo comporte 3 animaux
***** PRESENTATION DE LA MENAGERIE *****
Je me presente : Poisson Nemo
Je me presente : Chat Felix
Je me presente : Perruche Pistache

Felix s'en va

Le zoo comporte maintenant 2 animaux
***** PRESENTATION DE LA MENAGERIE *****
Je me presente : Poisson Nemo
Je me presente : Perruche Pistache

Le zoo comporte maintenant 2 animaux
***** PRESENTATION DE LA MENAGERIE *****
Je me presente : Poisson Maurice
... Il parait que je pousse le bouchon un peu trop loin ...!!! ;-)
Je me presente : Perruche Pistache
```

4) Est-ce que le chat Félix existe encore, bien qu'il ne soit plus dans notre ménagerie ?
   Comment faites-vous pour vérifier ?
5) Peut-on modifier l'espèce d'un animal, une fois qu'il a été créé ?
   Comment faites-vous pour vérifier ?
   Et quelle est votre conclusion ?
6) Peut-on modifier le nombre d'animaux de la ménagerie directement ?
   Comment faites-vous pour vérifier ?
   Quel est le résultat de votre vérification ?

Découverte de la Programmation Orientée Objet en php

Les chiens prennent pension dans un chenil ...

Dans la suite du TP, nous allons nous occuper d'un type particulier d'animaux : les chiens.
Un chien est bien un animal, mais dans notre contexte, il possède en plus deux propriétés :

- ✓ Une race (exemple : " Fox Terrier ")
- ✓ Une taille en cm, 10 cm minimum

La particularité d'un chien c'est d'aboyer, autrement dit de Parler(" Ouah ! ").
Mais un molosse n'aboie pas comme un petit roquet ; on a donc décidé que l'aboiement dépendait de la taille du chien :

- ✓ Un chien " normal " (entre 20 et 60 cm) fait " Wouaf !! Wouaf !! Wouaf !! "
- ✓ Un gros chien (plus de 60 cm) fait plutôt " Grrr !! Grrr !! Grrr !! "
- ✓ Un petit chien (moins de 20 cm) fait lui " Kaii !! Kaii !! Kaii !! "

Vous pourrez vérifier, en écrivant par exemple le programme suivant, dans une nouvelle application, que vous ajouterez à votre solution, et que vous appellerez " AppliChenil " :

```
< ?php
include "animalerieclass.php";
include "animalclass.php";
include "chienclass.php";
//les chiens prennent vie
$medor = new Chien("Médor","noir","batard",19);
$saucisse = new Chien("Saucisse","roux","teckel",9);
$pongo = new Chien("Pongo","pie","dalmatien",70);
$perdita = new Chien("Perdita","pie","dalmatien",50);
//Les chiens aboient quand
echo "<BR/>"." La caravane passe .."."<BR/>";
$medor->aboyer();  $saucisse->aboyer();
echo "<BR/>"."Quelle est la taille de Saucisse ? ".$saucisse->taille()." cm"."<BR/>";
$pongo->aboyer();
$perdita->aboyer();
//Les chiens sont en pension au chenil
$LeChenil = new Menagerie;
$LeChenil->arriver($medor);
$LeChenil->arriver($saucisse);
$LeChenil->arriver($pongo);
$LeChenil->arriver($perdita);
//Le chenil présente ses pensionnaires
$LeChenil->presenter();
?>
```

NOTE IMPORTANTE

On verra que les propriétés réelles des objets ne sont généralement pas accessibles pour différentes raisons.
L'une de ces raisons, c'est que l'on veut parfois contrôler la valeur de ces propriétés. C'est le cas ici de la taille d'un chien. On n'enregistrera pas une taille<10cm.
Une autre de ces raisons, c'est que certaines propriétés ne sont accessibles que partiellement, par exemple seulement en lecture.
Les propriétés sont en fait rendues visibles (ou non) par des " accesseurs ", qui limitent éventuellement cette visibilité.

7) Quelle taille fait saucisse ?
   A-t-elle la taille fixée initialement ?
   Conclusion ?
8) Peut-on modifier la taille d'un chien après coup ?
   Avec n'importe quelle valeur ?
   Comment testez-vous ?
   Quel est le résultat ?
9) Peut-on modifier la race d'un chien après son instanciation ?
   Comment testez-vous ?
10) Peut-on afficher le nom d'un chien ?
11) Comment se fait-il qu'un chien possède un nom alors que cette propriété n'existe pas dans la classe Chien ?
12) Peut-on faire dormir un chien ? le faire manger ?
    le faire parler (au lieu d'aboyer) ?
13) Un chien possède-t-il une espèce ?
    quelle est-elle ?
14) Créer une S.P.A. (sorte de ménagerie ;-) dans laquelle il y a 2 chats, et 2 chiens.
    Que se passe-t-il lorsque la S.P.A. présente ses animaux ?
15) Créer une classe oiseau dérivée de la classe animal, les oiseaux possèdent en plus un pays d'origine et ils peuvent voler.
16) Instancier une volière (de la classe Ménagerie), puis des oiseaux qui arrivent et quittent la volière.
17) Présenter la volière après chaque mouvement.