DepannetonPC

Rechercher une définition :

programmation orientee objet Chercher

Définition : Programmation orientée objet

Publié par coyotte49 le Lundi 20 août 2007

La programmation orientée objet (POO) est un type de programmation qui a pour avantage de posséder une meilleure organisation, surtout dans les gros programmes. Ces derniers seront agencés de façon plus logique et seront donc plus facilement modifiables. La POO est cependant plus difficile à maîtriser.

Concrètement, elle consiste à faire "coïncider" la réalité et les lignes de codes.
Si vous voulez coder un jeu où il y a des véhicules, vous allez créer une catégorie, une classe (ne vous souciez pas de la signification de ce mot pour le moment) nommée *véhicule*. A l'intérieur, vous pourrez créer des sous-classe *Voiture*, *Bateau*, *A pied*, *Avion* etc. Dans *Voiture*, vous pourrez mettre *Jeep*, *Ferrari* etc.
Cette métaphore est donc **logique**, mais elle est loin d'être représentative de la totalité de la POO ! C'est juste un exemple d'utilisation, elle n'englobe pas du tout toutes les possibilités de la POO. Elle supporte en outre :

- L'encapsulation : Consiste à empêcher l'accès à certaine partie du programme pour sa propre sécurité (sécurité dans le sens stabilité du programme, le protéger lui même de fausses manoeuvres, pas une protection contre le piratage).
- Polymorphisme : Vient du Grec et signifie "Plusieurs formes". Concept difficile à comprendre sans connaissance en POO. Nous ne le développerons donc pas ici.
- L'héritage, que nous avons expliqué ci-dessus, en guise d'exemple.

On représentera, en POO, un objet réel comme une voiture par un objet aussi, comprenant les même propriétés et facultés de l'objet réel, mais celui-ci sera purement virtuel.

Si je décide de m'occuper d'une Ferrari, je créerai un objet 'voiture' qui aura comme attribut "nombre_de_roues = 4; marque = audi; moteur = v8" etc... Et je lui donnerai comme "faculté": 'rouler, reculer, tourner a gauche, a droite etc...

information :

Nous parlions il y a quelques instants de classe. Une classe est un type de chose. L'objet est l'une de ces choses. Concrètement, si nous avons la classe chien, un objet sera un chien en particulier, Bouffy et Rex par exemple. Pour reprendre ce qui a été dis ci-dessus, la classe est Voiture et un objet serait Ferrari. Un objet est une instance d'une classe.

Le C++, le Java, C#, Delphi, Python, Visual Basic sont quelques des langages supportant la programmation orientée objet.

Voici un exemple de code source, que notre compatriote Loïc a tenu que je publie. Merci à lui.
(je déconseille à ceux qui ne connaissent pas de lire ce qui suit ; les effets secondaires pourraient être nausée, vomissement et suicide. DTPC ne sera tenu pour responsable dans aucun de ces cas ;)).

En PHP (5) :

code :

```
<?php

class Voiture
{
private $nb_roues;
private $marque;
private $moteur; //etc...

function rouler()
{
/* ... */
}

function reculer()
{
/* ... */
}

function tourner($direction)
{
/* ... */
} //etc...

function setValue($variable, $value) //Fonction pour pouvoir changer la valeur d'une variable de la class
{
$this -> $variable = $value;
}

function getValue($variable)
{
return $this -> $variable;
}
/*On doit faire appel à une fonction interne de la class pour changer ou lire la valeur d'une variable de cette classe car elle est définie en 'private' c-à-d que seule la class a le droit de lecture et d'écriture sur ces variables */

}

//je veux que ma voiture aille 4 roues, soit une audi et aille un V8:

$voiture = new Voiture; // On instancie la class à l'aide du mot clé "new"

$voiture -> setValue('$nb_roues', '4');
$voiture -> setValue('$marque', 'audi');
```

```
$voiture -> setValue('$moteur', 'v8');

//Maintenant la voiture a donc les attributs d'une audi ayant un moteur v8, si je voulais la faire tourner (à droite), il faudrait faire $voiture -> tourner('droite');
//Afficher la marque de la voiture:

echo $voiture -> getValue('$marque');

//Et si je fais "echo $voiture"? Php chercher la fonction __toString() dans la class et fera ce que lui dit cette fonction (:p)
?>
```