Le polymorphisme

Le polymorphisme permet à différents objets d'être traités de manière interchangeable s'ils répondent à une même interface ou s'ils partagent une classe parente. Cela permet d'écrire du code plus générique et réutilisable.

Exemple de polymorphisme par l'héritage :

le polymorphisme peut être réalisé grâce à l'héritage en PHP. Montrez un exemple où une classe parente définit une méthode, et les classes enfants la redéfinissent pour implémenter leur propre comportement. Malgré cela, vous pouvez utiliser un objet de n'importe quelle classe enfant où un objet de la classe parente est attendu.

Exemple de polymorphisme par l'interface :

le polymorphisme peut également être réalisé à travers les interfaces en PHP. plusieurs classes peuvent implémenter la même interface mais fournir une implémentation différente pour les méthodes définies par cette interface. Cela permet à ces classes d'être traitées de manière uniforme lorsqu'elles sont utilisées via leur interface commune.

Supposons que nous avons une classe parente Animal et deux classes enfants Chien et Chat. Chacune de ces classes a une méthode crier(), mais chaque sous-classe implémente cette méthode de manière différente.

```
<?php
// Classe parente Animal
class Animal {
    // Méthode crier() par défaut pour tous les animaux
    public function crier() {
        return "Cri indéfini";
    }
}
// Classe enfant Chien
class Chien extends Animal {
    // Implémentation spécifique de la méthode crier() pour les chiens
    public function crier() {
        return "Le chien aboie : Wouf wouf!";
    }
}
// Classe enfant Chat
class Chat extends Animal {
    // Implémentation spécifique de la méthode crier() pour les chats
    public function crier() {
        return "Le chat miaule : Miaou!";
    }
}
// Fonction utilisant le polymorphisme pour afficher le cri d'un animal
function afficherCriAnimal(Animal $animal) {
    echo $animal->crier() . "\n";
}
// Création d'objets de différentes classes et appel de la fonction
$chien = new Chien();
$chat = new Chat();
afficherCriAnimal($chien); // Affiche : Le chien aboie : Wouf wouf!
afficherCriAnimal($chat); // Affiche : Le chat miaule : Miaou!
?>
```

Dans cet exemple, les classes Chien et Chat héritent de la classe parente Animal et remplacent la méthode crier() avec leur propre implémentation spécifique. Lorsque nous passons des objets de ces classes à la fonction afficherCriAnimal(), le polymorphisme permet d'appeler la méthode crier() appropriée en fonction du type réel de l'objet passé, ce qui nous permet d'afficher le cri spécifique de chaque animal.