PHP : La Réflexion

L3 Informatique - UE Développement Web

David Lesaint

david.lesaint@univ-angers.fr

Janvier 2019

Réflexion (Reflection)

Mécanisme pour examiner ou modifier programmatiquement (reverse engineering)

- Les classes et objets (propriétés et méthodes).
- Les interfaces.
- Les fonctions.
- Les extensions.
- Les commentaires.

PHP est un langage réflexif

Depuis PHP 5, on dispose d'une API (classes et interfaces) dédiée à la réflexion.

Réflexion : usages

- Le typage dynamique (duck typing).
- La programmation orientée aspect.
- La méta-programmation.
- Les frameworks Web : initialisation de modèles, création d'objets vues, injection de dépendances . . .
- Les "mocking frameworks" (pour objets simulés), e.g., PHPUnit.
- Les frameworks d'analyse de code.

Réflexion procédurale

Examen du contenu d'un objet avec les fonctions suivantes

`string get_class(o)`	Renvoie le nom de la classe d'un objet.
`string get_parent_class(o)`	Renvoie le nom de la classe parente de l'objet.
`array get_class_vars(o)`	Renvoie les valeurs par défaut des attributs d'une classe.
`array get_object_vars(o)`	Renvoie les attributs non statiques de l'objet qui sont accessibles depuis le contexte courant.
`array get_class_methods(o)`	Renvoie les noms des méthodes d'une classe.

Autres fonctions

`class_exists()`, `get_called_class()`, `get_declared_interfaces()`, `method_exists()` ...

get_object_vars

Ne donne pas accès aux membres privés/protégés si elle est invoquée en dehors de la classe.

get-object-vars.php

```php
class Person {
      public $pub;
      protected $pro;
   private $pri;
   public function __construct($pub,$pro,$pri) {
      $this->pub=$pub;
      $this->pro=$pro;
      $this->pri=$pri;
   }
   public function __get($name) {
      $a=get_object_vars($this);  print_r($a);
      if (isset($a[$name])) return $a[$name];
      return null;
   }
   public function __toString() {
      return $this->pub." ".$this->pro." ".$this->pri;
   }
}
$p=new Person("Snowden","Edward","Moscow");
echo $p."\n"; // Snowden Edward Moscow via __toString
echo $p->pub."\n"; // Snowden par accès direct
echo $p->pro."\n";
//Array([pub] => Snowden [pro] => Edward [pri] => Moscow) par __get
//Edward par __get
print_r(get_object_vars($p));//Array([pub] => Snowden [pro])
```

Réflexion avec l'API `ReflectionClass`

Donne accès aux propriétés et méthodes d'une classe cible

Permet d'invoquer accesseurs (getters) et mutateurs (setters) si l'on s'impose une convention de nommage, e.g.,

- CamelCase : `setProp()` pour la propriété `$prop`.
- snake-case : `set_Prop()` pour la propriété `$prop`.

reflection-class.php

```php
require('employee.php');
$reflector=new ReflectionClass('Employee');
$e=new Employee("turing",100,"IQ_society");
$properties=$reflector->getProperties();
foreach ($properties as $property) {
   $pname = $property->getName(); // objet ReflectionProperty
   if ($property->isPublic())
      echo $pname, " = ", $property->getValue($e),"\n";
   else
      if ($reflector->hasMethod("set".ucfirst($pname))) {
         $method=$reflector->getMethod("set".ucfirst($pname)); // objet ReflectionMethod
         $method->invoke($e,1000000);
      }
}
//propriétés affichées et méthode setSalaire invoquée !
var_dump($e);
```

Particularité de la réflexion

Le transtypage d'un objet en tableau permet d'obtenir toutes ses propriétés.

reflection-array.php

```php
class A {
        public $pubA;
        protected $proA;
        private $priA;
}
class B extends A {
        public $pubB;
        protected $proB;
        private $priB;
}
$b = new B();
$arr = (array) $b; //transtypage explicite
print_r($arr);
//Array([pubB] => [*proB] => [BpriB] => [pubA] => [*proA] => [ApriA] =>)
```