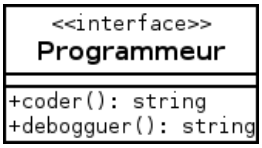
Programmation orientée objet en PHP

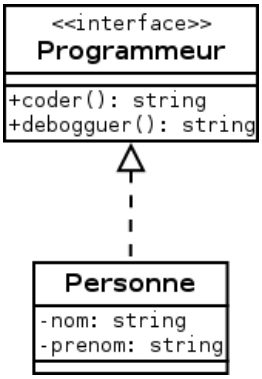
Introduction

Les interfaces sont ce que l'on appelle des contrats d'implémentation. Elles servent à indiquer les "compétences" que doit posséder une classe.
Nous trouverons dans une interface une ou plusieurs méthodes abstraites, c'est-à-dire des méthodes sans implémentation. Ces méthodes peuvent être de classe ou d'instance.
Une interface ne déclarera aucun attribut d'instance ou de classe, ni de méthode "non abstraite". Par contre elle pourra déclarer des constantes. Toute méthode déclarée dans une interface est publique et abstraite.
Les classes peuvent implémenter une ou plusieurs interfaces. Pour être "concrètes", ces classes doivent fournir une implémentation aux méthodes définies dans les interfaces. Une classe qui implémente une ou plusieurs interfaces mais qui ne fournit pas d'implémentation aux méthodes abstraites sera forcément une classe abstraite.
Dans un diagramme de classe une interface porte le stéréotype <<interface>> :




```php
<?php

// Programmeur.php
interface Programmeur {

    public function coder();

    public function debogguer();
}
```

Créons une classe Personne qui implémente l'interface Programmeur. Dans un diagramme la flèche en pointillés signifie "implémente".



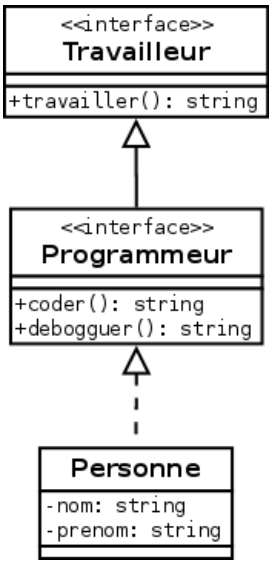
En PHP, une classe implémente une ou plusieurs interfaces en utilisant le mot clé "implements". Les noms des interfaces sont séparés par des virgules s'il y en a plusieurs.

```php
<?php

// Personne.php
class Personne implements Programmeur {

```

```
    protected $nom;
    protected $prenom;

    function __construct($nom, $prenom) {
        $this->nom = $nom;
        $this->prenom = $prenom;
    }

    public function getNom() {
        return $this->nom;
    }

    public function getPrenom() {
        return $this->prenom;
    }

    public function setNom($nom) {
        $this->nom = $nom;
    }

    public function setPrenom($prenom) {
        $this->prenom = $prenom;
    }

    public function coder() {
        echo 'je code...';
    }

    public function debogguer() {
        echo 'je déboggue...';
    }

}
```

```
<?php

// lanceur.php
include './Programmeur.php';
include './Personne.php';

$pers = new Personne('Sparrow', 'Jack');
$pers->coder();
$pers->debogguer();
```

Les interfaces peuvent aussi hériter d'autres interfaces avec le mot clé extends :




```
<?php
```

```php

// Travailleur.php

interface Travailleur {

    public function travailler();
}
```

```php
<?php

// Programmeur.php
interface Programmeur extends Travailleur {

    public function coder();

    public function debogguer();
}
```

```php
<?php

// Personne.php
class Personne implements Programmeur {

    protected $nom;
    protected $prenom;

    function __construct($nom, $prenom) {
        $this->nom = $nom;
        $this->prenom = $prenom;
    }

    public function getNom() {
        return $this->nom;
    }

    public function getPrenom() {
        return $this->prenom;
    }

    public function setNom($nom) {
        $this->nom = $nom;
    }

    public function setPrenom($prenom) {
        $this->prenom = $prenom;
    }

    public function coder() {
        echo 'je code...';
    }

    public function debogguer() {
        echo 'je déboggue...';
    }

    public function travailler() {
        echo 'je travaille...';
    }

}
```

```php
<?php

```

```php
// lanceur.php
include './Travailleur.php';
include './Programmeur.php';
include './Personne.php';

$pers = new Personne('Sparrow', 'Jack');
$pers->coder();
$pers->debogguer();
$pers->travailler();
```

Fin

NAVIGATION

Accueil
- Ma page
  - Pages du site
  - Mon profil
  - Cours actuel
    - POO PHP
      - Participants
      - Généralités
      - La programmation orientée objet : premiers pas
      - L'héritage
      - Les interfaces
      - **Introduction**
      - T.P. employés v2
      - Le typage
      - Les namespaces
      - Les exceptions
      - Les bases de données avec PDO
      - Les tests avec PHPUnit
      - Petite application version 2
      - Petitie application version 3
  - Mes cours

ADMINISTRATION

Administration du cours

Réglages de mon profil

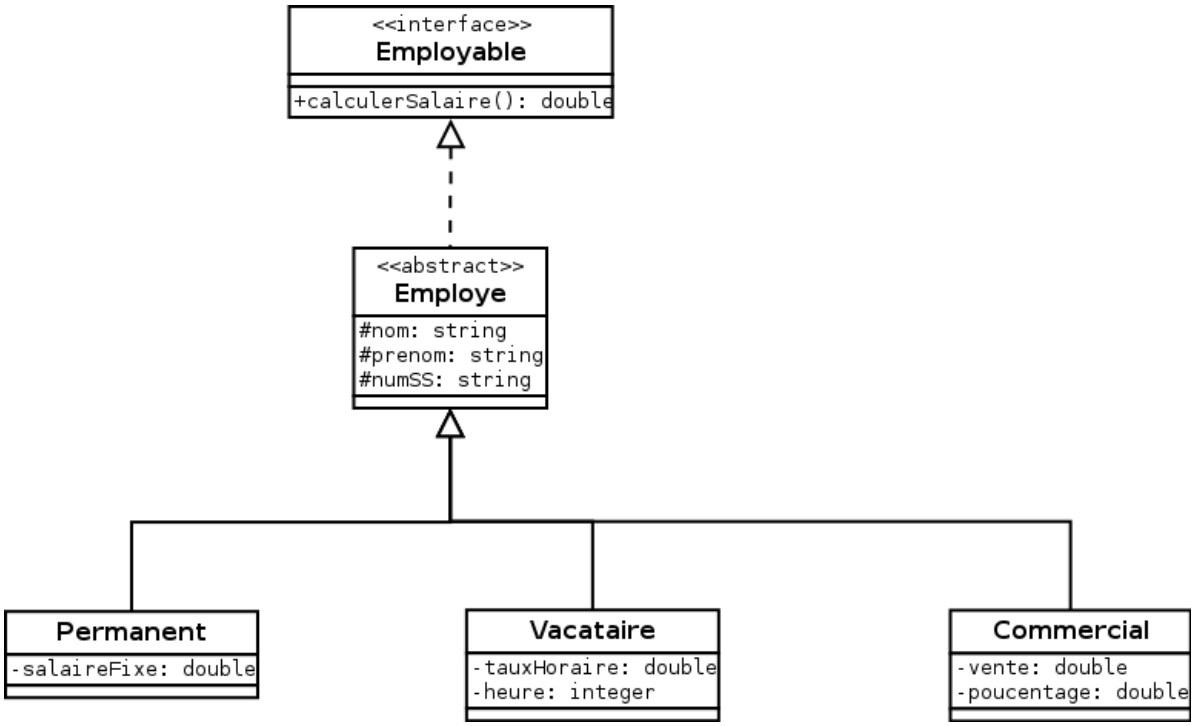
Connecté sous le nom « Arnaud Lemais » (Déconnexion)
POO PHP

fyligrane Arnaud Lemais

Programmation orientée objet en PHP

T.P. employés v2

Ajoutez au devoir employés une interface Employable qui déclare une méthode calculerSalaire(). Supprimez la déclaration de la méthode calculerSalaire() de la classe Employe. Faire en sorte que la classe Employe implémente l'interface Employable.



Les fichiers seront à remettre dans une archive zip dont le nom sera de la forme "employes_2_*nom_prenom*.zip".

État du travail remis

Numéro de tentative	Ceci est la tentative 1.
Statut des travaux remis	Aucune tentative
Statut de l'évaluation	Pas évalué
Dernière modification	vendredi 27 mars 2015, 15:32

Ajouter un travail

Modifier votre travail remis

NAVIGATION

Accueil
- Ma page

Pages du site
Mon profil
Cours actuel
POO PHP
Participants
Généralités
La programmation orientée objet : premiers pas
L'héritage
Les interfaces
Introduction
T.P. employés v2
Le typage
Les namespaces
Les exceptions
Les bases de données avec PDO
Les tests avec PHPUnit
Petite application version 2
Petitie application version 3
Mes cours

ADMINISTRATION

Administration du cours

Réglages de mon profil

Connecté sous le nom « Arnaud Lemais » (Déconnexion)
POO PHP