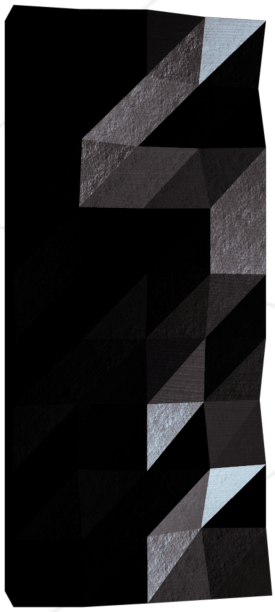

Formation PHP - Symfony

D02 - Les bases PHP OOP

Résumé: Dans ce jour de la piscine PHP Symfony,vous allez (ré)apprendre les bases de la programmation orientée objet et des fonctionnalités avancées de PHP.

Table des matières

Chapitre I

Préambule

Vous allez également apprendre à écrire des programmes modulaires et scalables en PHP.

Chapitre II

Consignes

Sauf contradiction explicite, les consignes suivantes seront valables pour tous les jours de cette Piscine.

- Seul ce sujet sert de référence : ne vous fiez pas aux bruits de couloir.
- Le sujet peut changer jusqu'à une heure avant le rendu.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne porterons attention ni ne prendrons en compte un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Attention aux droits de vos fichiers et de vos répertoires.
- Vous devez suivre la procédure de rendu pour tous vos exercices. L'url de votre dépot `GIT` pour cette journée est disponible sur votre intranet.
- Vos exercices seront évalués par vos camarades de Piscine.
- En plus de vos camarades, vous pouvez être évalués par un programme appelé la Moulinette. La Moulinette est très stricte dans sa notation car elle est totalement automatisée. Il est donc impossible de discuter de sa note avec elle. Soyez d'une rigueur irréprochable pour éviter les mauvaises surprises.
- Les exercices shell doivent s'éxcuter avec `/bin/sh`.
- Vous ne devez laisser aucun autre fichier que ceux explicitement specifiés par les énoncés des exercices dans votre dépot de rendu.
- Vous avez une question ? Demandez à votre voisin de droite. Sinon, essayez avec votre voisin de gauche.
- Toutes les réponses à vos questions techniques se trouvent dans les `man` ou sur Internet.
- Pensez à discuter sur le forum Piscine de votre Intra et sur Slack !
- Lisez attentivement les exemples car ils peuvent vous permettre d'identifier un travail à réaliser qui n'est pas précisé dans le sujet à première vue.
- Réfléchissez. Par pitié, par Thor, par Odin !

Chapitre III

Règles spécifiques de la journée

- Vous devez suivre scrupuleusement les demandes de chaque exercice et respecter les conventions de nommage des fichiers.

Chapitre IV

Exercice 00

	Exercice : 00
Exercice 00 : Les paramètres HTML	
Dossier de rendu : $ex00/$	
Fichiers à rendre : `TemplateEngine.php,index.php`	
Fonctions Autorisées :	

A l'aide du fichier template **book_description.html**, vous devez créer un fichier **TemplateEngine.php** avec une classe **TemplateEngine** contenant une méthode **createFile($fileName, $templateName, $parameters)**. Cette méthode devrait générer un nouveau fichier HTML avec le nom **$fileName** en remplaçant les paramètres entre **{}** dans le fichier template (ici **book_description.html**) avec les valeurs des paramètres en provenance du tableau **$parameters**.

Vous allez également créer un fichier **index.php** qui devra instancier la classe et appeler la méthode décrite avec un ensemble d'arguments que vous allez spécifier.

Chapitre V

Exercice 01

	Exercice : 01
Exercice 01 : Affichage de l'object	
Dossier de rendu : $ex01/$	
Fichiers à rendre : `TemplateEngine.php,index.php,Text.php`	
Fonctions Autorisées :	

Modifier la méthode **createFile($fileName, Text $text)** de la classe **TemplateEngine** afin de prendre un objet **Text** comme dernier paramètre.

Le classe **Text** devrait avoir un constructeur qui prend en paramètre un tableau de chaines de caractères, une méthode pour lui ajouter de nouvelles chaînes de caractères et une méthode permettant d'afficher toutes les chaînes de caractères en HTML, chacune incorporée dans une balise **<p>**.

La méthode **createFile** devrait créer un fichier HTML statique similaire à celui généré dans l'**Exercice 00** et dont le corps devrait inclure le contenu rendu de la classe **Text**.

Chapitre VI

Exercice 02

	Exercice : 02
Exercice 02 : Affichage de l'object 2	
Dossier de rendu : $ex02/$	
Fichiers à rendre : `TemplateEngine.php,index.php,HotBeverage.php,Coffee.php,Tea.php`	
Fonctions Autorisées :	

Créer une classe **HotBeverage** avec les attributs 'name', 'price' et 'resistence' et créer des "getter" (méthodes permettant de retourner les valeurs) pour tous les attributs.

Créer deux classe qui étendent la classe **HotBeverage**, **Coffee** et **Tea**, les deux attribuant une valeur aux attributs 'name', 'price' et 'resistence' et ayant en plus les attributs privés 'description' et 'comment' qui toutes les deux devrait avoir une valeur et un getter.

Modifier la méthode **createFile(HotBeverage $text)** de la classe **TemplateEngine** afin qu'elle prenne en paramètre un objet **HotBeverage** et, en utilisant le fichier template **template.html**, créer un fichier HTML statique ayant le nom de la classe de l'objet **HotBeverage** en remplaçant les paramètres dans le template avec la valeur des attributs de l'objet. La classe PHP **ReflectionClass** devrait être utilisée afin de retrouver les attributs de la classe et ensuite récupérer leurs valeurs en utilisant des appels aux getters respectifs.

Le fichier **index.php** devrait créer un objet **Coffee** et un objet **Tea** et appeler la fonction **createFile** pour les deux objets.

Chapitre VII

Exercice 03

	Exercice : 03
Exercice 03 : Générer HTML	
Dossier de rendu : $ex03/$	
Fichiers à rendre : `TemplateEngine.php,index.php,Elem.php`	
Fonctions Autorisées :	

Créer une classe **Elem** représentant une balise HTML. La classe devrait avoir un constructeur avec les paramètres suivants :

- element - le nom de la balise HTML
- content - le contenu de la balise (optionnel)

La classe devrait supporter les balises HTML suivantes : meta, img, hr, br, html, head, body, title, h1, h2, h3, h4, h5, h6, p, span, div.

La classe devrait également contenir la méthode **pushElement** permettant d'ajouter un nouvel élément au contenu de la balise. Un méthode **getHTML** devrait retourner le code HTML de la balise. Si un élément/balise contient d'autres éléments, le code HTML des sous-éléments devrait aussi être inclus.

Exemple :

```
$elem = new Elem('html');
$body = new Elem('body');
$body->pushElement(new Elem('p', 'Lorem ipsum'));
$elem->pushElement($body);

echo $elem->getHTML();
```

Devrait générer le code HTML suivant :

```
<html>
    <body>
        <p>Lorem ipsum</p>
    </body>
</html>
```

Modifier la classe **TemplateEngine** en y ajoutant un constructeur qui prend en paramètre un objet de la classe **Elem**. La méthode **createFile($fileName)** devra retourner le code HTML généré par l'objet dans un fichier.

Le fichier **index.php** devrait créer plusieurs objets **Elem** et sauvegarder le contenu généré dans un fichier.

Chapitre VIII

Exercice 04

	Exercice : 04
Exercice 04 : Générer HTML 2	
Dossier de rendu : $ex04/$	
Fichiers à rendre : `TemplateEngine.php,index.php,Elem.php,MyException.php`	
Fonctions Autorisées :	

Modifier la classe **Elem** de l'exercice précédent afin de lever l'exception **MyException** si une balise non autorisée est fournie en paramètre dans le constructeur.

Ajouter la prise ne charge des balises suivantes : table, tr, th, td, ul, ol, li.

Modifiez également le constructeur afin de prendre un nouveau paramètre optionnel, **attributes**, qui contiendra un tableau avec les attributs à ajouter dans le rendu HTML de la balise.

Exemple :

```
$elem = new Elem('html');
$body = new Elem('body');
$body->pushElement(new Elem('p', 'Lorem ipsum', ['class' => 'text-muted']));
$elem->pushElement($body);

echo $elem->getHTML();

$elem = new Elem('undefined'); // Leve une exception de type MyException$
```

Devrait générer le code HTML suivant :

```
<html>
    <body>
        <p class="text-muted">Lorem ipsum</p>
    </body>
</html>
```

Chapitre IX

Exercice 05

	Exercice : 05
Exercice 05 : Valider HTML	
Dossier de rendu : $ex05/$	
Fichiers à rendre : `TemplateEngine.php,index.php,Elem.php,MyException.php`	
Fonctions Autorisées :	

Ajouter une nouvelle méthode à la classe **Elem**, appelée **validPage()**, qui retournera soit *true* soit *false* selon que le code HTML d'un élémént donné est une page HTML valide.

Cette méthode devrait traverser l'ensemble des éléments et vérifier que les conditions suivantes sont vraies :

- l'élément/balise parent est **html** et contient exactement un seul élément **head** suivi par un élément **body**
- l'élément **head** devrait contenir un seul élément **title** et un seul élément **meta charset**
- la balise **p** ne devrait contenir aucune autre balise, seulement du texte
- la balise **table** ne devrait contenir que des balises **tr**, ces dernières contenant que des balises **th** ou **td**
- les éléments **ul** et **ol** ne devrait contenir que éléments **li**

Ajouter des tests afin de valider si toutes les conditions ci-dessus sont respectées.