UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Licence 2 Informatique
INFO0303 – 2023/2024
***Resp. :* C. Rabat**

Travaux dirigés n° 2

Programmation orientée objets en PHP

Exercice 1 (Oh ma zone !)
Nous souhaitons développer un site marchand proposant des articles à vendre. Chaque article est caractérisé par un intitulé, une description et un prix. Nous supposons que les articles sont stockés dans un fichier texte. Chaque ligne contient les informations d'un article séparées par un `";"` (nous considérons que `";"` ne peut pas être présent dans la description).

1°) Écrivez une classe `Item` représentant un article contenant les attributs nécessaires, un constructeur, des getters et la méthode `__toString` (qui crée une chaîne contenant simplement l'intitulé et le prix).

2°) Proposez une méthode `fromString` qui permet de créer un article depuis une chaîne de caractères contenant les données séparées par un `";"` (ce qui correspond à une ligne de notre fichier). Elle retourne `null` si la chaîne ne contient pas exactement 3 éléments (on ne vérifie pas la validité des données).

Nous souhaitons maintenant afficher la liste des articles. Un script `list.php` charge la liste des articles depuis le fichier puis les affiche les uns après les autres à l'aide du script `view.php` qui affiche une instance de `Item` contenue dans une variable globale `$item`.

3°) Donnez le contenu du script `view.php`.

4°) Donnez le contenu du script `list.php`.

Exercice 2 (Ajout et modification d'articles)
Nous souhaitons ajouter des méthodes dans la classe `Item` pour faciliter la création et la modification d'un article. Pour cela, nous proposons les méthodes `displayFrom` et `fromForm` pour respectivement afficher un formulaire contenant les informations de l'article courant et récupérer un objet `Item` depuis le formulaire.

1°) Donnez la signature de la méthode `displayForm` et expliquez ce qu'elle doit contenir.

2°) Donnez le code de la méthode `fromForm`.

3°) Expliquez comment utiliser ces méthodes pour éditer un article.

Exercice 3 (Bibliothèque de gestion de formulaires)
La gestion des formulaires dans une application Web peut être rapidement fastidieuse. Nous souhaitons développer une bibliothèque pour nous permettre de créer des formulaires rapidement, automatiser la vérification des données saisies et éviter le rejeu des formulaires.
n.b. : de telles bibliothèques existent déjà en PHP et sont parfois intégrées dans les *frameworks*.

Nous considérons la classe principale `Form` qui représente un formulaire et qui contient un ensemble de champs. Chaque champ du formulaire correspond à un objet de la classe `FormElement`. La classe `FormElement` est la classe mère de chaque type de champ (champ texte ou mot de passe, bouton, zone de texte, liste de saisie, etc.). Une fois le formulaire créé, une méthode `display` permet de l'afficher en HTML.

1. Quelles données sont nécessaires dans les différentes classes ? Quelles méthodes ?
2. En sachant que le nom et l'identifiant peuvent être communs aux classes `Form` et `FormElement` mais qu'il n'y a aucune relation d'héritage entre elles, proposez une solution fournie par PHP.
3. Quelle solution proposez-vous pour spécifier le type d'un champ texte (entre `text`, `password`, `number`, etc.) ?
4. Proposez un diagramme de classes de la bibliothèque.
5. Expliquez comment il est possible de vérifier les données saisies d'un formulaire. Donnez un exemple précis avec le code associé.
6. Si les classes de la bibliothèque se trouvent dans l'espace de noms *myform*, rappelez comment utiliser les classes dans un script externe.
7. Expliquez ce qu'est le rejeu d'un formulaire.

Pour éviter le rejeu d'un formulaire, le plus simple est d'ajouter une clé unique dans le formulaire. Pour cela, nous pouvons utiliser la fonction PHP `uniqid` qui retourne une chaîne aléatoire.

8. Expliquez comment mettre en place la vérification du rejeu d'un formulaire dans notre bibliothèque.

Annexes

Utilisation des *cards* de *bootstrap* :

```
<div class="card">
  <div class="card-header">Titre</div>
  <div class="card-body">
    Contenu qui peut tout contenir... ou presque !
  </div>
  <div class="card-footer"></div>
</div>
```

Il est possible d'ajouter des classes spécifiques dans les attributs `class` des éléments comme la couleur de fond (`bg-danger`, `bg-primary`, `bg-light`...), la couleur du texte (`text-white`) ou l'alignement du texte (`text-align`).