Feuille 2 - Technologies du Web

janvier 2020

PHP

Exercice 1 :
Préparation de l'espace de travail.

- Sur webtp créez un dossier `feuille2` consacré au travail de cette semaine
- Dans ce dossier `feuille2` , transférez tous les fichiers et dossiers de l'archive et leur contenus. Voici le rôle des dossiers :
  - `couvertures` contient des images
  - `data` contient les données qui serviront à engendrer la page HTML
  - `lib` est destiné à un ou plusieurs fichiers PHP qui n'ont pas vocation à être exécutés directement mais à être inclus dans d'autres.
  - `views` contient des « pages template » : des pages décrites essentiellement en HTML mais incluant quelques chaînes PHP.

  N'oubliez pas de mettre les droits convenables sur l'ensemble des dossiers et fichiers.
- **Les fonctions que vous écrirez dans le cadre de cet exercice figureront dans le fichier** `lib/fonctionsLivre.php`

Le but final de l'exercice sera de générer en PHP un document HTML à partir des données figurant dans le fichier. Nous procèderons par étapes, en structurant le code PHP.

Le fichier `exempleLivre.txt` contient la description d'un ouvrage :

```
couverture : scorpion.jpg
titre : La marque du diable
serie : Le Scorpion
auteurs : Marini - Desberg
année : 2000
catégorie : bandes-dessinées
```

Chaque ligne définit une « propriété » du livre. Elle **se compose de 2 parties**, de part et d'autre du premier caractère `:`

- d'abord, le **nom de la propriété**
- ensuite, la **valeur de la propriété**

Les espaces éventuels situés avant ou après le nom de la propriété ne sont pas significatifs et devront être éliminés. Idem pour la valeur de la propriété.
Par exemple la première propriété s'appelle `couverture` ; sa valeur est `scorpion.jpg`.
La description d'un livre est une suite de propriétés. Elle est suivie soit d'une ligne vide, soit de la fin de fichier.

Q 1 . *Représentation d'un livre dans une structure PHP*
Un livre sera représenté par une table associative, dont les clés sont les noms des propriétés :

clé	valeur
`"couverture"`	`"scorpion.jpg"`
`"titre"`	`"La marque du diable"`
`"serie"`	`"Le Scorpion"`
`"auteurs"`	`"Marini - Desberg"`
`"année"`	`"2000"`
`"catégorie"`	`"bandes-dessinées"`

Dans le fichier `lib/fonctionsLivre.php`, écrire une fonction `readBook($file)` qui reçoit en argument un descripteur de fichier (supposé déja ouvert). Le fichier est censé commencer par la description d'au moins un livre, suivie d'une ligne vide **ou** de la fin de fichier. Le résultat de la fonction est une table PHP représentant ce livre.

Si une des lignes de la description est incorrecte (ne contient pas le caractère `:`) une exception doit être déclenchée

Pour cette question, vous consulterez la documentation des fonctions PHP `implode` et `trim`

Test unitaire de la fonction que vous venez d'écrire

Un fichier `debug.php` vous est fourni, il vous permet de tester la fonction que vous venez d'écrire.

Important : le seul rôle de ce script est de faire des tests unitaires de votre fonction. C'est un script qui n'est destiné qu'à vous, pour les besoins du développement, et en aucun cas un script de production. Par exemple, l'usage de la fonction `print_r()` qui permet d'afficher le contenu d'une valeur PHP élaborée est strictement réservé à un usage de débogage.

Ouvrez `debug.php` dans un **éditeur** de texte et lisez les commentaires, qui vous expliquent son fonctionnement.

Ensuite, ouvrez `debug.php` dans un **navigateur**. Si le résultat n'est pas celui attendu, continuez à mettre au point votre fonction `readBook()`

Q 2 . *Construction du code HTML à partir des données.*

Le modèle de document à produire vous est donné dans le fichier `exempleLivre.html`. Examinez-en le contenu. Vous remarquerez que chaque propriété se traduit par un élément HTML (pas toujours le même : `div`, `time`, `h2` ...)

- cet élément possède un attribut `class`. La classe de l'élément est le nom de la propriété.
- le contenu de l'élément est en général la valeur de la propriété. Dans certains cas cette valeur a été transformée (pour les auteurs ou la couverture).

Vous allez maintenant construire les fonctions nécessaires à la production du code HTML en procédant de façon modulaire.

→ Notez bien que chacune des fonctions que vous allez écrire maintenant **ne doit rien envoyer sur la sortie standard** (**une fonction ne comporte aucun echo, aucun print, aucun printf ...**). Seul le **résultat** de chaque fonction est important.

1. Construire une fonction `elementBuilder($elementType, $content, $elementClass="")`. Les 3 arguments sont des chaînes. Le résultat est une chaîne contenant le code HTML d'un élément de type `$elementType`, et de contenu `$content`. Si `$elementClass` n'est pas la chaîne vide, il désigne la valeur de l'attribut `class` de l'élément.

   Exemples :
   - `elementBuilder('p','bla bla')` renvoie `'<p>bla bla</p>';`
   - `elementBuilder('h2','La marque du diable','titre')` renvoie `'<h2 class="titre">La marque du diable</h2>'`

   NB : longueur approximative : 2-3 lignes environ

   Procédez au test unitaire de la fonction, en adaptant le script de débogage.

2. Construire une fonction `authorsToHTML($authors)` qui renvoie une chaîne contenant le code HTML représentant les auteurs. Dans `authors`, les noms des auteurs sont séparés par `' - '`. Dans le résultat, chaque nom d'auteur doit être inséré dans un élément `span`. Les spans sont séparés par un espace.

   par exemple `authorsToHTML('Marini - Desberg')` a pour résultat la chaîne

   `'<span>Marini</span><span>Desberg</span>'`

   NB : vous utiliserez les fonctions `implode` et `explode` (doc à consulter)

   NB : longueur approximative : 2 lignes environ

   Procédez au test unitaire de la fonction, en adaptant le script de débogage.

3. Construire une fonction `coverToHTML($fileName)` qui renvoie une chaîne contenant le code HTML représentant l'image de couverture, c'est à dire un élément `img` dont l'attribut `src` renvoie à l'URL du fichier, dans le dossier `couvertures` et dont l'attribut `alt` est défini. Par exemple `coverToHTML('scorpion.jpg')` a pour résultat la chaîne

   `'<img src="couvertures/scorpion.jpg" alt="image de couverture" />'`

   NB : longueur approximative : 1 ligne environ

   Procédez au test unitaire de la fonction, en adaptant le script de débogage.

4. Construire une fonction `propertyToHTML($propName,$propValue)` qui renvoie une chaîne contenant le code HTML représentant la propriété, selon le principe :

propriété	type d'élément	contenu de l'élément
titre	h2	valeur (telle quelle)
couverture	div	élément `img`
auteurs	div	suite d'éléments `span`
année	time	valeur (telle quelle)
autres propriétés	div	valeur (telle quelle)

Par exemple `propertyToHTML('titre', 'La marque du diable')` a pour résultat la chaîne

`'<h2 class="titre">La marque du diable</h2>'`

`propertyToHTML('auteurs', 'Marini - Desberg')` a pour résultat la chaîne

`'<div class="auteurs"><span>Marini</span> <span>Desberg</span></div>'`

NB : longueur approximative : 10 lignes environ

Procédez au test unitaire de la fonction, en adaptant le script de débogage.

5. Enfin, construisez une fonction `bookToHTML($book)` dont l'argument est une table PHP représentant un livre (voir question précédente) et dont le résultat est une chaîne qui contient le code HTML correspondant (c'est à dire le code d'un élément `article`).

   Pour la structure HTML à produire, vous vous réfèrerez au contenu de fichier `exempleLivre.html`. Vous remarquerez ainsi que l'ensemble des propriétés est regroupé dans un élément `div` de classe `description`, à l'exception de la propriété `couverture` qui figure en premier et n'est pas groupée avec les autres.

   NB : longueur approximative : 10 lignes environ

   Procédez au test unitaire de la fonction, en adaptant le script de débogage.

Q 3 . *Le travail final : la page PHP*

Cette partie vous est donnée, mais vous devez absolument examiner le code fourni pour vous en inspirer dans l'exercice suivant. Avec l'éditeur, examinez les contenus des deux fichiers :

- `livreUnique.php` est le script principal. Il inclut la bibliothèque de fonctions et effectue les traitements. En fin de script il affiche le contenu de la page en incluant la vue `views/pageLivreUnique.php`
- `views/pageLivreUnique.php` a quasiment la forme d'un fichier HTML. Il utilise cependant une variable PHP globale `$bookHTML` qui a été définie par le script appelant (`livreUnique.php`). Cette variable doit contenir une chaîne avec du code HTML à insérer dans la page.

Vérifiez le bon fonctionnement en ouvrant `livreUnique.php` dans le navigateur

Exercice 2 :

Q 1 . Vous allez modifier le code de la fonction `readBook($file)` développée dans l'exercice précédent (conservez une copie de la version précédente que vous mettrez en commentaires).

On suppose maintenant que dans `$file`,

- on peut **éventuellement** trouver une ou plusieurs lignes vides avant la description du livre. Il faut passer ces lignes.
- le fichier peut ne comporter que des lignes vides ou même aucune ligne du tout (et donc ne comporter aucune definition de propriété). Dans ce cas la fonction `readBook($file)` doit renvoyer `FALSE`.

La nouvelle version de la fonction `readBook()` ne devrait pas impacter le fonctionnement du script de l'exercice précédent : vérifiez qu'il fonctionne toujours.

Q 2 . Créez une fonction `libraryToHTML($file)`. Son argument `$file` est supposé être un descripteur d'un fichier ouvert contenant une suite (éventuellement vide) de descriptions de livres. Un exemple de tel fichier est `livres.txt`

Le résultat de la fonction doit être une chaîne qui contient le code HTML présentant l'ensemble des livres du fichier (donc une suite d'articles).

Q 3 . Créez une page `bibliotheque.php` qui présente (en HTML) l'ensemble des livres figurant dans `livres.txt` **Vous procéderez de la même façon qu'à la question précédente** (pour `livreUnique.php`) : vous créerez une vue dans le répertoire `views` et cette vue sera incluse par une `require` à la fin du script `bibliotheque.php`.

Q 4 . (à faire en dehors des séances) Associer aux pages `livreUnique.php` et `bibliotheque.php` un style permettant d'obtenir une meilleure présentation. Voir à ce sujet l'exercice du l'UE TW1 consacré à cette question.

Exercice 3 :
Sécurisation du site
Dans l'état actuel, l'ensemble des fichiers est accessible directement depuis n'importe quel client du site web, comme vous allez le voir

- Dans le navigateur, entrez l'URL suivante :
  `http://webtp.fil.univ-lille1.fr/~votre_login/feuille2/data/livres.txt`
  Le navigateur affiche alors le fichier de données.
- Dans le navigateur, entrez l'URL suivante :
  `http://webtp.fil.univ-lille1.fr/~votre_login/feuille2/lib/fonctionsLivre.php`
  Le navigateur affiche une page blanche mais pas de message d'erreur : le script `fonctionsLivre.php` vient d'être exécuté, mais comme il ne produit pas de sorties, rien ne s'affiche.

Les deux manipulations ci-dessus mettent en évidence des problèmes.

- l'utilisateur a pu visualiser directement des données brutes qui ne lui étaient pas, en principe, destinées (le fichier `livres.txt`)
- l'utilisateur a déclenché l'exécution d'un script qui n'était pas prévu pour être exécuté directement mais pour être inclus dans un autre.

Un principe basique de sécurisation d'un site est que **les seuls fichiers accessibles directement sont les pages PHP, les fichiers nécessaires à la présentation de ces pages (CSS, JS), les media inclus dans ces pages (images).**
Ainsi l'organisateur du site ne doit PAS se demander « devrais-je interdire l'accès à tel fichier ? » mais « faut-il vraiment que je laisse libre accès à tel fichier ? »

Q 1 . *Protection des dossiers*
Créez un fichier nommé `.htaccess` contenant les lignes :

```
Order deny, allow
Deny from all
```

Il s'agit d'un fichier de directives pour le serveur web. Ici, la directive donnée est de ne pas laisser libre accès au contenu du dossier où se trouve le fichier `.htaccess`
Un fichier d'un dossier ainsi protégé n'est plus accessible directement depuis un client, mais il reste néanmoins accessible depuis les autres scripts : il peut toujours être inclus (par un `require`) ou manipulé en tant que fichier (`fopen, fgets` ...)

- Placez une copie de ce fichier dans le dossier `data` et une autre dans le dossier `lib`.
- Tentez d'accéder de nouveau aux deux fichiers qui étaient accessibles tout à l'heure

Dossiers devant être interdits en lecture directe (en y plaçant un fichier `.htaccess` comme ci-dessus) :

- `data`
- `lib`
- `views`

Notez que le dossier `couvertures` doit rester accessible : il contient des images faisant partie des pages web du site.
Notez aussi que le script `debug.php` doit être supprimé du site (ou déplacé dans un répertoire protégé) quand le site est en phase de production.