Université de Lille 1 SCIENCES ET TECHNOLOGIES

Semaine 4 - Technologies du Web

FORMATIONS EN INFORMATIQUE DE LILLE 1

février 2015

PHP - Formulaires

Exercice 1 : Une classe en PHP

On dispose, pour une commune, des informations suivantes :

- nom (plus exactement le nom de base, nous y reviendrons)
- code de département (ex : 59)
- code de commune (à 3 chiffres, par exemple 350 pour Lille)
- code d'arrondissement (1 chiffre)
- tncc (1 chiffre, voir plus loin)
- population (entier)
- latitude (1 nombre flottant)
- longitude (1 nombre flottant)

Question 1.1 : Vous créerez une classe `Commune` (dans un fichier `Commune.class.php`) permettant, entre autres, de représenter ces informations.

Le constructeur de la classe aura pour argument un tableau de 8 chaînes (les 7 informations ci-dessus, dans cet ordre).

Voici une liste minimale de méthodes à implémener :

```
public function nom();          //simple accesseur
public function codeDept();     //simple accesseur
public function codeCommune(); //simple accesseur
public function codeArrondissement();  //simple accesseur
public function population();  //simple accesseur
public function codeINSEE();   //concatenation des codes département et commune
public function coordonnees(); // tableau de 2 flottants : latitude et longitude
public function nomComplet();  // nom avec article éventuel
public function nomCharniere();// nom avec charnière
```

À propos des noms de commune :

Le nom d'une commune est composé d'un nom de base, et éventuellement d'un article. Par exemple pour «La Madeleine», le nom de base est «Madeleine». Le TNCC indique l'article à utiliser. Par exemple TNCC=3 indique que le nom doit être précédé de l'article «La».

Dans certaines expressions (par exemple «La commune d...») la dénomination est précédée d'une charnière qui dépend du son de la première syllabe et de la présence ou non d'un article. Par exemple : Commune de Lille, Commune d'Halluin, Commune de La Madeleine. Le code TNCC indique également la charnière à utiliser. Voici le tableau complet :

tncc	article	charnière
0		de
1		d'
2	Le	du
3	La	de La
4	Les	des
5	L'	de L'
6	Aux	des
7	Las	de Las
8	Los	de Los

(NB : un tableau PHP contenant ces informations vous est fourni dans le fichier `tableauTNCC.php`)
Vous pourrez tester votre classe avec le script `testClasseCommune.php` qui vous est fourni.

Question 1.2 : La ressource `http://www.fil.univ-lille1.fr/technoweb/exos/communes5962.csv` est un fichier des communes du Nord-Pas-de-Calais, au format CSV. Chaque ligne du fichier contient les informations d'une commune.
Pour la lecture de ce fichier vous pourrez utiliser la fonction `fgetcsv($fichier)`. Comme la fonction `fgets($fichier)`, elle lit une ligne de fichier, mais elle renvoie un tableau PHP où le contenu de la ligne a été découpé selon le séparateur (par défaut, la virgule).
Construire une fonction `loadCommunes($file)` qui renvoie un tableau PHP associatif dont chaque clé est un code INSEE des communes et la valeur correspondantes est l'objet instance de Commune.
Faire un script PHP `tableCommunes.php` qui présente une page HTML contenant une table des communes. Les communes seront présentées à raison d'une par ligne, avec deux colonnes : son n°INSEE, son nom complet.

Question 1.3 : (À faire **en dehors** de la séance) Présenter les communes par ordre alphabétique. Générez en PHP un index selon la lettre initiale, c'est à dire une liste des initiales des communes. Chaque initiale est un lien cliquable qui désigne la première commune commençant par cette lettre.

Question 1.4 : Construire un script `presentationCommune.php` générant une page HTML dédiée à la présentation d'une seule commune. Cette page affichera toutes les informations disponibles pour la commune.
Le script recevra dans une variable HTTP envoyée en mode GET le code INSEE de la commune à afficher. (cette variable s'appellera "insee").
Testez votre page en complétant (dans la barre URL du navigateur) l'URL de la page par l'envoi de cette variable.
Vous ferez en sorte que si la variable "insee" est absente ou si son contenu ne correspond pas au code d'une commune connue, la page affiche un message d'erreur.

Question 1.5 : Reprendre le script `tableCommunes.php` pour ajouter une colonne contenant un lien cliquable renvoyant à la page `presentationCommune.php`, avec le code INSEE de la commune.

Exercice 2 : Recopiez le fichier `starWars.html` dans votre répertoire. Visualisez-le avec un navigateur et ouvrez-le avec un éditeur.
Il s'agit d'un formulaire HTML qui permet de commander des articles à une association, le *club des fans de Star Wars*. Les articles sont des figurines de collection des personnages de Star Wars.

Question 2.1 : Mettez en commentaire les éléments `input` permettant de choisir une figurine. Remplacez-les par un élément `select` à sélection multiple (même nom de variable, mêmes textes affichés et mêmes valeurs renvoyées)

Question 2.2 : Transférez le fichier `starWars.html` obtenu à la question précédente dans votre espace webtp.
Écrivez un programme PHP qui, à partir de ce formulaire, génère une page HTML présentant la facture. Voici comment se calcule le montant de la facture ;

- Chaque figurine coûte 15 euros HT,
- Une réduction de 20% est consentie pour toute commande de 5 figurines ou plus.
- L'ensemble des 12 figurines coûte 135 euros HT (ne pas appliquer la réduction de 20% sur ce prix).

Les adhérents au club bénéficient d'une remise de 10% qui s'applique aux prix ci-dessus (donc cumulable). L'adhésion coûte 5 euros HT.
Pour terminer, la TVA est de 20%, et les frais de port de 7,5 euros HT.

La facture fera apparaître la liste des figurines commandées, les prix HT et TTC ainsi que les coordonnées de l'acheteur (nom, adresse, etc...)
Dans cette question il n'est, **pour l'instant**, pas demandé de vérifier la validité des paramètres.

La vérification de validité des paramètres reçus est indispensable afin d'éviter un comportement anormal du script qui pourrait produire des résultats erronés ou incohérents. Et surtout, c'est un élément crucial de la sécurité des sites web.
La vérification faite au niveau du formulaire (premières questions de cette fiche) est un confort apporté à l'utilisateur et un plus dans l'ergonomie du site, mais **n'apporte aucune sécurité** : il est en effet toujours possible d'envoyer une requête au serveur sans passer par le formulaire.
S'agissant de notre exemple, si on envoie une requête avec une URL comme celle-ci :

```
factureStarWars.php?fig[]=KingKong&fig[]=SpiderMan& ....
```

La facture affichera des figurines totalement inconnues du club des fans de Star Wars. Certes, pas de gros problème de sécurité sur cet exemple (peut-être une problème de ridicule !), mais un fonctionnement erroné du site.
C'est pourquoi, il faut toujours tester chacun des paramètres pour vérifier s'il correspond à ce que l'on attend.

Cas des variables « énumérées »

Quand une variable doit appartenir à un ensemble fini de valeurs, une solution consiste à construire un tableau ayant ces valeurs pour clés (on utilise le tableau comme une table de hachage). Supposons, par exemple, que l'on attende un argument *fruit* dont la valeur serait nécessairement *pomme, poire* ou *orange*. on peut utiliser une portion de code comme celle-ci :

```
$fruitsAutorises = array('pomme'=>TRUE,'poire'=>TRUE,'orange'=>TRUE);
if (! isset($_REQUEST['fruit']))
  throw new Exception("argument fruit non fourni");
else if (! isset($fruitsAutorises[$_REQUEST['fruit']]))
  throw new Exception("argument fruit {$_REQUEST['fruit']} invalide");
```

Quelques remarques :

- On aurait obtenu la même table en l'initialisant par

  ```
  $fruitsAutorises = array_fill_keys(array('pomme','poire','orange'),TRUE);
  ```

  l'utilisation de cette fonction permet un code plus compact quand les ensembles de valeurs sont grands.
- Notez que l'on n'utilise pas ici la valeur booléenne du tableau `$fruitsAutorises`, mais simplement le fait que la clé est définie. On pourrait utiliser ce tableau pour associer aux valeurs attendues non pas TRUE, mais une information utile à la suite du script. Par exemple, si l'on avait besoin d'une traduction en anglais :

  ```
  $fruitsAutorises = array('pomme'=>'Apple','poire'=>'Pear','orange'=>'Orange');
  ```

Utilisation des filtres

PHP propose une fonction d'import filtré des variables externes. Voici comment récupérer un paramètre entier nommé `nombreEntier` dans le tableau `$_GET` :

```
$n = filter_input(INPUT_GET, 'nombreEntier', FILTER_VALIDATE_INT);
 if ($n === NULL)
  throw new Exception("argument nombreEntier non fourni");
else if ($n === FALSE)
  throw new Exception("argument nombreEntier {$_REQUEST['nombreEntier']} invalide");
```

Nous venons d'utiliser le filtre prédéfini pour les entiers, nommé `FILTER_VALIDATE_INT`. Il existe d'autres filtres que vous pouvez découvrir dans la documentation PHP
Si 'nombreEntier' n'est pas fourni, la fonction `input_filter` renvoie NULL. S'il est fourni mais ne convient pas au filtre, le résultat vaut FALSE.
Certains filtres demandent un paramètre supplémentaire. C'est le cas de `FILTER_VALIDATE_REGEXP`. Voici comment on pourrait filtrer un argument attendu sous la forme d'une lettre suivie d'une suite de chiffres décimaux :

```
$n = filter_input(INPUT_GET, 'nombreEntier', FILTER_VALIDATE_REGEXP,
                  array('options'=> array('regexp'=>'/^[a-zA-Z][0-9]+$/'))
                 );
```

Note ; il s'agit ici de la notation des expressions régulières Perl qui présente quelques particularités. Pour une recherche par motif exact, commencer par `/^` et terminer par `$/`

Question 2.3 :

Ajouter au fichier PHP un contrôle des paramètres entrés. Vous vérifierez en particulier l'adhesion, la civilité, les noms des figurines, la validité du code postal. Vous vérifierez que les autres champs obligatoires sont non vides.
En cas d'erreur vous afficherez une page HTML d'erreur.

Question 2.4 :
Transformez le fichier contenant le formulaire `starWars.html` en `starWars.php`. Vous insèrerez une portion de code PHP permettant d'intégrer un message d'erreur. Le comportement sera le suivant :

- Si la variable `$error` est définie, alors elle est supposée contenir un message d'erreur qu'il faut afficher avant le formulaire, dans un paragraphe d'identifiant `errorMessage`.
- Sinon, ce paragraphe n'est pas créé.

Puis vous ferez en sorte que lorsqu'une erreur est détectée, le script `factureStarWars.php` renvoie non plus la page d'erreur, mais le formulaire précédé du message d'erreur (indication : utiliser `require`).