Université de Lille 1 SCIENCES ET TECHNOLOGIES

Semaine 4 - Technologies du Web

FIL FORMATIONS EN INFORMATIQUE DE LILLE 1

février 2016

PHP - Formulaires

Exercice 1 : Recopiez le fichier `testSVG.html` dans votre répertoire. Il s'agit d'un fichier HTML 5 qui comporte un dessin en SVG. SVG est un langage XML consacré au dessin vectoriel (les figures sont décrites par leur caractéristiques géométriques). Ouvrez le fichier dans un éditeur et examinez attentivement le contenu de l'élément d'identifiant `id=dessin` et les commentaires qui expliquent la signification des balises `circle`, `rectangle` et `polygon`.

Question 1.1 : Créez une bibliothèque `fonctionsSVG.php` contenant

- une fonction `cercle($cx,$cy,$r)` qui renvoie le texte de la balise SVG `<circle>` correspondante.
  Par exemple `cercle(0,100,30)` doit renvoyer la chaîne `<circle cx="0" cy="100" r="30" />`
- `carreInscrit($cx,$cy,$r,$angle=0)` qui renvoie la balise SVG dessinant un carré de côté $c = r\sqrt{2}$ et de coin inférieur $(cx - \frac{c}{2}, cy - \frac{c}{2})$ tourné de angle `autour` du point `cx,cy`
  Par exemple `carreInscrit(0,100,30,10)` doit renvoyer la chaîne
  `<rect x="-21.21" y="78.79" width="42.42" height="42.42" transform="rotate(10,0,100)" />`
- `triangleInscrit($cx,$cy,$r,$angle=0)` qui renvoie la balise SVG dessinant un triangle équilatéral dont les sommets sont $(cx, cy + r)$, $(cx - \frac{r\sqrt{3}}{2}, cy - \frac{r}{2})$, $(cx + \frac{r\sqrt{3}}{2}, cy - \frac{r}{2})$, tourné de angle `autour` du point `cx,cy`.

puis créez un fichier `testSVG.php`, dérivé du fichier HTML d'origine en remplaçant le contenu de l'élément d'identifiant `dessin` par des appels de fonctions.

Question 1.2 : Créez un fichier `dessiner.php` qui prend en compte des paramètres passés en mode GET :

- `figure` : peut prendre pour valeur : `cercle, rectangle, triangle`
- `cx` : un entier
- `cy` : un entier
- `r` : un entier positif

puis qui dessine la figure demandée.
Testez ce script en codant les arguments dans l'URL.

Question 1.3 : Créez une page `formulaireDessiner.html` qui contient un formulaire permettant d'appeler le script `dessiner.php`.

Question 1.4 : Vous allez adapter le formulaire de façon à pouvoir choisir de dessiner plusieurs types de figures en même temps, Vous allez modifier le champ `select` :

- ajouter l'attribut `multiple="multiple"`
- modifier le nom : `name="figure[]"`

Maintenant le formulaire renvoie non pas une valeur simple mais un tableau avec toutes les valeurs selectionnées.
Il reste à adapter `dessiner.php` de façon à prendre en compte le fait que `$_GET['figure']` est maintenant un tableau.

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.

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'adhésion, 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`).

Question 2.5 : Amélioration du formulaire
En l'état, il possible d'envoyer un formulaire incomplet. Complétez le code HTML pour faire en

sorte que l'envoi devienne impossible en l'absence d'information dans les champs `nom, prenom, voie, cp, commune`

Question 2.6 : De même, contraignez le champ code postal pour n'y valider qu'une chaîne de 4 caractères Vous ferez en sorte qu'une chaîne invalide s'affiche en rouge.