








Labo 6 - PHP - XSLTProcessor pour gérer la transformation xml/xsl côté serveur.

Remarque préalable : les bases minimales de php sont données sur Connected.

Dans le dossier « Eco » vous trouverez une série de fichiers-exemples qui vont permettre de découvrir progressivement l'utilisation de xsl avec php. A utiliser bien entendu sur Wampserver (ou tout autre serveur Web intégrant PHP)

Disque local (C:) > wamp64_3.2.0 > www > Eco				
<input type="checkbox"/> Nom	Modifié le	Type	Taille	
 eco.xml	18-11-20 07:43	Fichier XML	78 Ko	
 eco_liste_classe.xsl	17-11-20 12:30	Fichier XSL	1 Ko	
 eco1.xsl	18-11-20 07:43	Fichier XSL	1 Ko	
 eco2.xsl	17-11-20 10:41	Fichier XSL	1 Ko	
 index0.php	18-11-20 07:47	Fichier PHP	1 Ko	
 index1.php	17-11-20 10:39	Fichier PHP	2 Ko	
 index2.php	17-11-20 12:27	Fichier PHP	2 Ko	

1) Vous commencez par ouvrir eco.xml dans un navigateur. Comme le fichier est lié à eco1.xsl, une transformation aura lieu et vous aurez un résultat dans le navigateur.

1BS1
LEROY Michaël 1BS1
ASSEZ Geoffrey 1BS1
CIALMA Florina Sonira 1BS1
LANDERCY Amandine 1BS1
LAZZARO Laura 1BS1
LIBERT Stéphanie 1BS1
BOUGEÂTRE Marie 1BS1
DORCHY Nathalie 1BS1
LEQUANTRE Amélie 1BS1

Le fichier xsl permet d'extraire les étudiants pour une classe donnée ; remarquez dans le fichier eco1.xsl, le paramètre cl (pour classe) qui a comme valeur par défaut 1BS1. Si vous supprimez la valeur par défaut (eco2.xsl), vous n'aurez plus aucun étudiant puisque le test à la ligne 8 ne donnera aucun résultat : ***student[classe=\$cl]***.

2) Commençons maintenant à voir comment utiliser php pour effectuer la transformation xsl :

Premier pas avec le fichier index0.html :

On va initialiser la valeur de la classe dans une variable php : ***\$classe="1BI2"***;

Et par la suite, on fera passer la valeur de la classe vers xsl : ***\$proc->setParameter("", 'cl', \$classe)***;

Si on ouvre index0.php, on aura les étudiants de la classe 1BI2.

3) Faire interagir l'utilisateur via un formulaire HTML : on va remplacer l'initialisation de la variable classe par la saisie de cette valeur dans une zone de texte (index1.php) ou, mieux, via une liste déroulante contenant les classes existant dans le fichier eco.xml (index2.php).

Dans le fichier index1.php, il faut attirer l'attention sur la ligne *if (isset(\$_POST["classe"]))* : il s'agit d'un test sur la non-nullité de la variable \$_POST[« classe »]. La variable « superglobale » \$_POST n'est nulle que lorsque le formulaire n'a pas encore été posté (bouton Submit). Cela va éviter d'exécuter le code php lors du premier chargement de la page.

Il faut remarquer aussi que « classe » est la valeur de l'**attribut name** de l'<input> du formulaire et qu'une **méthode POST** a été spécifiée pour le formulaire.

Dans le fichier index2.php, on va utiliser 2 transformations xsl. Une première pour générer la liste des classe (eco_liste_classe.xsl) et ensuite la deuxième eco1.xsl déjà utilisée auparavant pour filtrer les étudiants sur base de la classe.

Ce que vous devez faire :

1) Optimiser le fichier index2.php : pour l'instant, la liste des classes contient des doublons et n'est pas triée. On a vu au labo4 qu'il était possible de supprimer les doublons et ainsi de proposer une liste bien plus présentable.

2) En partant du labo 5, on va transférer les compétences qui viennent d'être vues pour proposer plus d'interactivité :

2a) Dans une interface PHP/HTML, créer une liste des pays et une liste des villes disponibles afin de permettre à l'utilisateur de filtrer les clients.

2b) Dans une interface PHP/HTML, permettre d'encoder la quantité en stock qui sera transmise au fichier xsl. Permettre aussi de choisir la « clé » de tri (pas uniquement sur le niveau de réapprovisionnement – cf. remarque Connected sur le paramétrage de xsl :sort).