Travailler avec les variables globales à OFF

Depuis peu, peut être avez vous rencontré certains problèmes avec les versions récentes de PHP pour la manipulation de variables.
Avant, pour récupérer vos variables, vous pouviez le faire directement par leur nom, quel que soit le type de variables (passées par la méthode POST/GET, cookie, session, ...)
Cela nous permettait de faire ceci :

Récupération directe de variables

```
un appel au script :
http://www.mondomaine.fr/monscript.php
?truc=coucou
<?php
 echo 'Ma variable passée par url =>'.$truc;
?>
affichant :
Ma variable passée par url =>coucou
```

La variable $truc, passée par URL est directement accessible depuis le script PHP. Cette façon de travailler a l'air, certes, bien pratique, elle laisse néanmoins la porte ouverte à des trous de sécurité.

Ce qui nous permettait d'utiliser nos variables de cette façon−là est en fait une option se trouvant dans le fichier PHP.ini qui s'appelle : register_globals et qui est par défaut initialisée à ON dans ce fichier.
Depuis la version 4.2.0 de PHP, cette option est par défaut initialisée à OFF.

C'est pourquoi il est important de prendre dès maintenant l'habitude de travailler avec register_globals à OFF. De plus, beaucoup d'hébergeurs ayant upgradé PHP avec la nouvelle version laissent register_globals à OFF... Et puis, cette méthode permet de résoudre pas mal de trous de sécurité dans les scripts.

Une autre option importante de la configuration PHP : track_vars, positionnée à ON, celle ci nous permet de récupérer les variables passées par formulaire, url et cookies dans des tableaux prédéfinis.
Avec ces 2 options configurées telles que décrites ci−dessus, nous pouvons utiliser les tableaux associatifs suivants pour récupérer nos variables :

Tableaux associatifs	
$HTTP_GET_VARS (désormais $_GET)	Permet de récupérer les variables passées par url ou par la méthode GET d'un formulaire
$HTTP_POST_VARS (désormais $_POST)	Permet de récupérer les variables envoyées par la méthode POST d'un formulaire
$HTTP_POST_FILES (désormais $_FILES)	Permet de récupérer les variables de fichiers envoyés par un formulaire
$HTTP_COOKIE_VARS (désormais $_COOKIE)	Permet de récupérer les cookies
$HTTP_SESSION_VARS (désormais $_SESSION)	Permet de récupérer les variables de session
$HTTP_ENV_VARS (désormais $_ENV)	Permet de récupérer les variables d'environnement données par PHP
$HTTP_SERVER_VARS (désormais $_SERVER)	Permet de récupérer les variables serveur envoyées par le serveur HTTP

A partir de PHP 4.1.0, ces variables changent de nom pour prendre des noms plus courts (et plus facile à retenir :p).

Avant 4.1.0	**A partir de 4.1.0**
$HTTP_GET_VARS	$_GET
$HTTP_POST_VARS	$_POST
$HTTP_POST_FILES	$_FILES
$HTTP_COOKIE_VARS	$_COOKIE
$HTTP_SESSION_VARS	$_SESSION
$HTTP_ENV_VARS	$_ENV
$HTTP_SERVER_VARS	$_SERVER

Mise en pratique

Ok..ok, mais comment ça marche? Nous allons voir quelques petits exemples dans lesquels nous allons manipuler ces tableaux magiques...

1. Formulaires

Récupération d'informations provenant d'un formulaire utilisant la méthode POST.

Récupération d'un formulaire exemple1.php

```
<?php
$bouton = $_POST['send'];
if(!empty($bouton)) {
 $nom = trim($_POST['nom']);
 $prenom = trim($_POST['prenom']);
 if(!empty($nom) && !empty($prenom)) {
 echo 'Bonjour, '.$prenom.' '.$nom;
 }
 else {
 echo 'vous n\'avez pas rempli tous
 les champs';
 }
}
?>
```

Formulaire utilisé form.html

```
 <form action="exemple1.php" method="post">
 <table>
 <tr>
 <td>Nom :</td>
 <td>
 <input type="text" name="nom"
 value="">
 </td>
 </tr>
 <tr>
 <td>Prénom :</td>
 <td>
 <input type="text" name="prenom"
 value="">
 </td>
 </tr>
 </table>
```

```
<input type="submit" value="envoyer"
name="send">
</form>
```

Comme vous le voyez, nous récupérons la valeur des variables grâce au tableau associatif $_POST dans lequel une entrée à été créée avec le nom de chaque variable du formulaire.

Pour ceux qui se demandent à quoi sert la fonction trim(), elle retire les espaces en début et fin de chaîne, cela nous permet de ne pas traiter des informations vides (un utilisateur pourrait très bien n'entrer que des espaces dans le formulaire)

Récupération d'un fichier.

Bien, maintenant voyons comment récupérer les informations concernant l'envoi de fichiers par une formulaire

Voici comment récupérer toutes les informations du fichier envoyé avec le tableau associatif $_FILES

Entrée	**Explications**
$_FILES['variable']['name']	Le nom original du fichier qui provient de la machine de l'utilisateur
$_FILES['variable']['type']	Le type mime du fichier
$_FILES['variable']['size']	Le taille du fichier en bytes
$_FILES['variable']['tmp_name']	Le nom temporaire du fichier stocké sur le serveur
$_FILES['variable']['error']	le code erreur associé à l'upload (attention cette option à été ajoutée en PHP 4.2.0)

Cet exemple ne fait qu'afficher les informations concernant le fichier que l'on a uploadé.

Récupération de fichiers

```
<?php
$bouton = $_POST['bouton'];
if(!empty($bouton)) {
$fichier = $_FILES['fichier']['name'];
$size = $_FILES['fichier']['size'];
$tmp = $_FILES['fichier']['tmp_name'];
$type = $_FILES['fichier']['type'];
$error = $_FILES['fichier']['error'];
$max = 10000;
```

Récupération de fichiers

```
if(isset($fichier)) {
 if($size <= $max) {
 echo 'Nom d\'origine =>'.$fichier.'<br />';
 echo 'Taille =>'.$size.'<br />';
 echo 'Nom sur le serveur =>'.$tmp.'<br />';
 echo 'Type de fichier =>'.$type.'<br />';
 echo 'Code erreur =>'.$error.'<br />';
 }
 else {
 echo 'Le fichier est trop volumineux';
 }
```

```
}
else {
 echo 'aucun fichier envoyé';
}
}
?>
<form enctype="multipart/form-data"
action="exemple2.php"
method="post">
<input name="fichier" type="file">
<input type="submit" value="send" name="bouton">
</form>
```

Récupérer les informations d'un formulaire facilement.

Imaginons que vous vouliez simplement afficher les informations entrées dans un formulaire. Vous pouvez parcourir le tableau $_POST afin d'afficher tout ce qu'il contient.

La variable $key contient le nom de la variable formulaire et la variable $val contient la valeur entrée dans le formulaire.

Afficher toutes les infos d'un formulaire

```
<?php
if(isset($_POST)) {
foreach($_POST as $key=>$val) {
 echo $key.'=>'.$val.'<p>';
 }
}
else {
 echo 'le formulaire n\'a pas été envoyé';
}
?>
<form action="exemple3.php" method="post">
Nom :<input type="text" name="nom"
value="">
<br />
Prénom:<input type="text" name="prenom"
value="">
<br />
Adresse:<input type="text" name="adresse"
value="">
<br />
Ville:<input type="text" name="ville"
value="">
<br />
Pays:<input type="text" name="pays"
value="">
<br />
<input type="submit" value="envoyer">
</form>
```

2. La méthode get et le passage par URL

Rien de bien compliqué, il suffit de récupérer les variables passées en get par le tableau $_GET.Un petit exemple pour illustrer quand même.

Passage par URL

```
http://www.monsite.fr/exemple4.php?nom=cow&prenom=flying
<?php
$nom = $_GET['nom'];
$prenom = $_GET['prenom'];
echo 'Bonjour, '.$prenom.' '.$nom;
?>
```

3. Les cookies

là nom plus rien de particulier

Récupération d'un cookie

```
<?php
$cookie = $_COOKIE['moncookie'];
if(!empty($cookie)){
 echo 'Valeur du cookie=>'.$cookie;
}
?>
```

4. Les variables session

En récupérant vos variables sessions par le tableau associatif $_SESSION vous êtes sûrs que la valeur est bien une variable session. Voyons un exemple ou nous récupérons une variable session directement.

Le fichier session.php, débute une session si il n'en existe pas encore pour l'utilisateur.

Fichier ou l'on crée une entrée en session si le login et mot de passe entrés sont corrects, auquel cas, on initialise la variable $_SESSION['login'], avec la chaîne 'ok'.

Variables session

```
<?php
include 'session.php';
$bouton = $_POST['bouton'];
if(!empty($bouton)) {
$log = trim($_POST['log']);
$pass = trim($_POST['pass']);
 if ($log=='jenny' && $pass == 'foo') {
 $_SESSION['login'] = 'ok';
 echo '<a href="exemple7.php"> >> suite >></a>';
 }
 else {
 echo 'mauvais login/pass';
 }
}
?>
<form action="exemple6.php" method="post">
Login :<input type="text" name="log"
value=""><br />
Pass :<input type="text" name="pass"
value=""><br />
<input type="submit" value="envoyer"
name="bouton">
</form>
```

Voici comment récupérer les variables session par le bon tableau.

Récupération d'une session

```
<?php
include 'session.php';
if($_SESSION['login'] == 'ok') {
 echo 'vous êtes bien connecté';
}
else {
 die('pas en session');
}

?>
```

5. Les variables d'environnement

Utilisation du tableau $_ENV pour récupérer les variables d'environnement : en voici quelques unes d'utiles.

Les variables d'environnement

```
<?php
echo 'Nombre de process actifs=>';
echo $_ENV['NUMBER_OF_PROCESSORS'].'
';
echo 'Système d\'exploitation=>';
echo $_ENV['OS'].'
';
echo 'Chemin du répertoire temporaire=>';
echo $_ENV['TMP'].'
';
echo 'Chemin du profil utilisateur=>';
echo $_ENV['USERPROFILE'].'
';
?>
```

6. Les variables serveurs

Utilisation du tableau $_SERVER pour récupérer les variables serveur : voici quelques exemples.

Les variables serveur

```
<?php
echo 'Chemin du script courant=>';
echo $_SERVER['PHP_SELF'].'
';
echo 'Nom du serveur=>';
echo $_SERVER['SERVER_NAME'].'
';
echo 'Variables passées au script=>';
echo $_SERVER['QUERY_STRING'].'
';
echo 'Document root=>';
echo $_SERVER['DOCUMENT_ROOT'].'
';
echo 'Référant=>';
echo $_SERVER['HTTP_REFERER'].'
';
echo 'Adresse ip de l\'utilisateur=>';
echo $_SERVER['REMOTE_ADDR'].'
';
?>
```

Vous pouvez avoir la liste des variables serveurs et d'environnement en créant le petit fichier PHP ci−dessous et en l'exécutant.

```
<?php
phpinfo();
?>
```

7. Récupérer facilement les variables

Pour récupèrer facilement les variables, vous pouvez utiliser la fonction PHP extract. Elle va exporter votre tableau associatif et créer une variable pour chaque clé du tableau.

```
extract($_POST,EXTR_OVERWRITE);
```

Cette fonction va créer une variable pour chaque clé du tableau associatif $_POST.Si on a :

- $_POST['nom']
- $_POST['prenom']
- $_POST['age']

La fonction extract() va créer les variables suivantes :

- $nom
- $prenom
- $age

Le second argument sert à gérer les collision de variables. Donc il sert à dire ce que l'on fait si une variable existe déjà. Par défaut les variables qui existent déjà seront écrasées.

Type	signification
EXTR_OVERWRITE	Ecrase les variables existantes
EXTR_SKIP	N'écrase pas les variables existantes
EXTR_PREFIX_SAME	Si une variable existe déjà, une nouvelle variable est créée avec un préfixe donné en 3ème argument à la fonction
EXTR_PREFIX_ALL	Crée de nouvelles variables avec le préfixe passé en 3ème argument pour toutes les clés du tableau
EXTR_PREFIX_INVALID	Crée de nouvelles variables avec le préfixe passé en 3ème argument pour les noms de variable invalides (par exemple $1)

Voilà un petit exemple qui montre comment fonctionne la fonction extract. La variable "$nom" existant déjà, une nouvelle variable va être créée "$new_nom".

```
<?php
$nom = 'blabla';
$_POST['nom'] = 'jenny';

extract($_POST,EXTR_PREFIX_SAME,'new');

echo 'variable "nom" =>'.$nom.'<br>';
echo 'variable "new_nom" =>'.$new_nom.'<br>';
?>
```

Résultat de cet exemple :

```
variable "nom" =>blabla
variable "new_nom" =>jenny
```

8. Conclusion

Avec ces quelques petits exemples, l'utilisation des tableaux associatifs ne devrait plus avoir de secrets pour vous !

Gloups... une bouée.. le capitaine se noie... flyingcow