PHP 5° PARTIE:

- 1. Introduction
- 2. Les *cookies* et PHP
- 3. Restrictions
- 4. La fonction setcookie()
- 5. Exemples simples
- 6. Travaux pratiques

1. Introduction

Un *cookie* (parfois appelé témoin) est une petite pièce d'information que les scripts peuvent stocker sur une machine côté client. Un *cookie* peut être créé sur la machine d'un utilisateur en envoyant un en-tête HTTP contenant les données au format suivant :

```
Set-Cookie: NOM=VALEUR; [expires=DATE;]
[path=CHEMIN;] [domain=NOM DOMAINE;] [secure]
```

Cet en-tête conduit à la création d'un cookie appelé **NOM** avec la valeur **VALEUR**. Tous les autres paramètres sont optionnels.

Lorsqu'un navigateur se connecte sur une URL, il commence par rechercher et examiner les *cookies* stockés localement. S'il détecte des *cookies* affectés à l'URL avec laquelle est établie la connexion, il les envoie en retour au serveur.

Les *cookies* sont un mécanisme d'enregistrement d'informations sur le client, et de lecture de ces informations.

2. Les cookies et PHP

PHP supporte les *cookies* de manière transparente. Ce système permet notamment d'authentifier et de suivre les visiteurs. Vous pouvez envoyer un *cookie* avec la commande setcookie() ou manuellement avec la commande header() et la syntaxe donné plus haut.

Les *cookies* font partie de l'en-tête **HTTP**, ce qui impose que setcookie() soit appelée avant tout affichage de texte. Ce sont les mêmes limitations que pour header().

Tous les *cookies* qui sont envoyés au client seront automatiquement retournés au script PHP, et transformés en variables, exactement comme pour GET et POST. Si vous souhaitez affecter plusieurs valeurs à un seul *cookie*, ajoutez [] au nom du *cookie*.

3. Restrictions

- Les cookies sont des fichiers que le serveur écrit sur l'ordinateur client et qui permettent à un script de sauvegarder des variables liées à l'utilisateur.
- Plusieurs risques sont liés à l'utilisation des cookies :
 - ❖ Le client peut refuser les *cookies* (A chaque visite vous risquez de lui dire "Bonjour, c'est la première fois que vous visitez ce site"...)
 - ❖ Le client peut avoir effacé les *cookies*, d'un jour à l'autre vous perdez les informations
 - Le client peut utiliser un navigateur qui n'accepte pas les *cookies* (situation très rare actuellement)

4. Fonction setcookie()

L'envoie du *cookie* se fait par la fonction :

```
int setcookie(string name, string value, int expire, string path,
string domain, int secure);
```

Les paramètres de la fonction setcookie():

- name est le nom du *cookie*, c'est une chaîne de caractères privée de ":","," et " ".
- value est la valeur du cookie
- expire est la date d'expiration du *cookie* (retourné par time ou mktime), une fois la date atteinte le *cookie* est supprimé ou invalidé, s'il n'est pas spécifié, le *cookie* expire à la fin de la session.
- path est le répertoire dans lequel le *cookie* est valide. Les sous répertoires sont également acceptés.
- domain est le domaine pour lequel le *cookie* est associé (ex: www.tvtsii.net!)
- secure indique que le *cookie* ne doit être transmis que par une connexion sécurisée HTTPS

Remarques:

- Seul le paramètre name est obligatoire.
- S'il n'y a que name de défini alors ce cookie est supprimé.

5. Exemple simple (I)

```
<?php // fichier : cookie.php</pre>
define (EXP HR, 3600); // Délai d'expiration : une heure
$compte = "tv"; // un compte fictif
setcookie("compte", $compte, time() + EXP HR); // envoie du cookie
echo "<a href=\"suite.php\">suite.php</a>";
?>
<?php // fichier : suite.php</pre>
# Le cookie est-il défini ?
isset($ COOKIE["compte"]) or die("Le cookie n'est pas dé fini");
// on récupère le cookie
// $cookie = $HTTP COOKIE VARS["compte"]; // anciennement
$cookie = $ COOKIE['compte'];
# Affichage du contenu du cookie
echo "<PRE>\n". var dump($cookie) ."<br>Ou alors : $compte<br />"; echo
"</PRE>\n";
?>
```

On peut aussi essayer avec un phpinfo().

5. Exemple simple (II)

```
<?php // fichier : cookies1.php</pre>
setcookie( "cookie[3]", "cookie3" );
setcookie( "cookie[2]", "cookie2" );
setcookie( "cookie[1]", "cookie1" );
Header("Location: cookies2.php");
?>
<?php // fichier : cookies2.php</pre>
echo "Contenu du cookie :<br />";
if(isset($cookie))
        while(list($name,$value) = each($cookie))
                echo "$name = $value<br />\n";
```

6. Travaux pratiques

- 1. Tester les deux exemples du cours et répondre aux questions suivantes :
 - ❖ Où est-il stocké ?
 - Quel est son nom et son extension ?
 - Que contient-il ?
- 1. Réaliser une application de gestion de panier utilisant les cookies.

Ce script permet de gérer les produits suivants :

Livre PHP: 21 euros

Livre MySQL: 9 euros

Livre HTML: 7 euros

Livre Javascript : 12 euros

Le premier script marche.php servira à mettre à 0 les données cookies et redirigera vers le script remplir.php.

Le script remplir.php est l'interface principal de l'application qui permet les opérations suivantes :

- ❖ Ajouter un livre (lien vers remplir.php)
- Vider le panier (lien vers marche.php)
- Calculer le total (lien vers total.php)

Ce script met à jour le contenu du panier si des livres sont ajoutés et vérifie si un panier existe (sinon il renvoie vers marche.php)

Le script total.php lit le contenu du panier, calcule et affiche le coût total. Il affichera les liens suivants : Modifier son panier et Vider son panier.

<u>Remarque</u>: attention, ce n'est pas une bonne idée pour un vrai panier car quelqu'un qui refuserait les <u>cookies</u> ne pourrait pas utiliser ce script.