

Chapitre 8

Cookies et sessions

●	8.1. Les cookies	281
●	8.2. Les sessions	310
●	8.3. Exercice	318

┌

┐

└

┘

Dans ce chapitre, nous allons nous intéresser aux différents moyens qui nous permettent de stocker de l'information chez l'internaute. Pour illustrer ces différentes techniques, nous mettrons en place une miniboutique.

8.1. Les cookies

En surfant sur le Web, vous avez pu vous apercevoir que certains sites étaient en mesure de vous "reconnaître". En revenant sur une boutique où vous avez déjà acheté, il n'est pas rare d'avoir des messages du type : "Bienvenue M. Dupont" ; "Vous avez acheté le produit X, nous vous proposons cette liste de produits qui peuvent vous intéresser" ; "Voici les derniers produits apparus dans notre boutique depuis votre dernière visite"...

Essayez maintenant de vous connecter depuis une autre machine ou un autre compte, et vous vous apercevrez que tous ces messages sont absents.

Ce genre de site utilise, en fait, une technique qui leur permet de stocker des informations dans votre ordinateur (et plus précisément dans votre compte), informations qui leur sont retransmises dès que vous naviguez sur leur site. Les données sont stockées dans de petits fichiers appelés "cookies". Ces cookies ne servent pas que les intérêts marketing des grandes boutiques du Web, ils vous permettent aussi :

- De ne pas avoir à rentrer systématiquement votre identifiant et votre mot de passe sur certains sites.
- De pouvoir disposer d'un système de panier très pratique dans les boutiques en ligne.



◀ **Fig. 8.1 :**
Suppression des cookies

Les cookies permettent donc de stocker tout type d'information et sont utilisés dans de très nombreux cas de figure. Souvent diabolisés, ils nous rendent finalement plus de services qu'ils nous desservent. Leur présence est, d'ailleurs, d'autant moins gênante

qu'ils peuvent être effacés à tout moment. Sous Windows, il suffit d'aller dans le menu **Options Internet** puis d'appuyer sur le bouton **Supprimer les cookies** pour tous les faire disparaître.

La perte d'espace disque est aussi un reproche, très souvent exagéré, que l'on fait aux cookies. Un cookie fait, en effet, en moyenne, entre 50 et 150 octets. Il vous faudrait stocker 10 millions de cookies pour perdre 1 Go d'espace disque.

● **Remarque Pas gênant, mais quand même...**

En pouvant identifier de manière unique les internautes, certains services tel que doubleclik.com (gestion de bannières publicitaires) ont constitué une base de données mondiale afin d'étudier les comportements des internautes. Ils savent où vous êtes allé, quand, combien de temps... Big Brother n'est alors plus très loin.

Aspects techniques

Plusieurs fois dans ce livre nous sommes intéressés à l'en-tête HTTP des pages web, notamment toutes les fois où nous avons dû utiliser la fonction `header()`. Les cookies sont, en fait, des données stockées en texte, qui sont transmises dans l'en tête HTTP des requêtes et des réponses :

Listing 8.1 : Exemple de directive HTTP permettant de créer le cookie moncookie
`Set-Cookie: moncookie=hello; path=/; expires Mon, 09-Dec-2002 13:46:00 GMT`

Listing 8.2 : Directive HTTP contenue dans la requête envoyée par un navigateur
`Cookie: moncookie=hello`

En PHP, la création d'un cookie est triviale. Elle ne nécessite que l'utilisation de la fonction `setcookie()`. Par défaut, cette fonction prend deux paramètres : le nom du cookie et sa valeur.

Produisons deux scripts, l'un pour créer le cookie et l'autre pour lire sa valeur :

Listing 8.3 : Le script `Cree_cookie.php3`

```
<?php
setcookie("moncookie","hello");
echo "cookie créé";
?>
```

Listing 8.4 : Le script `Lit_cookie.php3`

```
<?php
echo "valeur contenu dans mon cookie : $moncookie";
?>
```



◀ **Fig. 8.2 :**
Lecture d'un cookie

La lecture du cookie est donc encore plus simple : il suffit de lire la valeur contenue dans la variable qui porte le même nom que le cookie.

● **Remarque** Suppression

Pour supprimer un cookie en PHP, il suffit de supprimer le contenu du cookie. Si nous souhaitons supprimer le cookie `moncookie`, il nous suffit d'écrire `setcookie("moncookie")`.

Comme les cookies sont transmis par l'en tête HTTP, leur utilisation implique les mêmes contraintes que pour la fonction `header()` : interdiction absolue d'afficher quelque chose avant d'utiliser `setcookie()`.

Tout n'est quand même pas permis avec les cookies, voici une liste non exhaustive des limitations :

- Seul le site qui a créé le cookie peut y accéder.
- La taille d'un cookie doit être inférieure à 4 ko.
- Le nombre de cookies créés par un domaine donné est limité à 20.

La fonction `setcookie()` peut prendre d'autres paramètres :

- La date d'expiration : en secondes, la fonction PHP `time()` renvoyant la date actuelle en secondes.
- Le chemin : indique quelle partie du site peut avoir accès au cookie.
- Le domaine : indique quel domaine peut avoir accès au site.
- Un indicateur de sécurité : si sa valeur est 1, il indique que la valeur du cookie ne peut être transmise que si nous sommes en HTTS (HTTP sécurisé).

Étudions quelques exemples.

- Exemple 1 : modifions `creer_cookie.php3`.

```
<?php
setcookie("moncookie","hello",time()+5);
```

8 - Cookies et sessions

```
echo "cookie créé";  
?>
```

En appuyant sur le bouton rafraîchir ("reload") sur la page http://www.kernix/lit_cookie.php3, au bout de 5 secondes, le cookie disparaîtra.



◀ **Fig. 8.3 :**
Le cookie a expiré

● **Attention** Heure du serveur

Il est courant que l'heure du serveur web soit décalée par rapport à l'heure de votre machine. Cela peut fausser vos tests. Pour voir la date et l'heure du serveur web, utilisez la fonction `date` ("Y-m-d G:i:s").

■ Exemple 2 :

```
setcookie("moncookie","hello",time()+3600,"/",".kernix",1);
```

Le cookie est, cette fois, créé pour une heure. Tous les scripts situés sur le domaine `.kernix` peuvent y accéder, à la condition que les transmissions soient cryptées.

■ Exemple 3 :

```
setcookie("moncookie","hello");
```

En ne précisant pas de date d'expiration, nous créons cette fois un cookie de session. Le cookie existera tant que le navigateur restera ouvert. Dès sa fermeture, le cookie sera effacé.

Pour tester ce comportement, enchaînez les étapes suivantes :

- Appelez le script `http://www.kernix/creer_cookie.php3` (en ayant modifié son contenu au préalable !).
- Appelez le script `http://www.kernix/lit_cookie.php3` (vous pouvez le laisser ouvert aussi longtemps que vous le souhaitez).
- Fermez le navigateur.
- Appelez de nouveau le script `http://www.kernix/lit_cookie.php3` : le cookie a disparu.

Pour changer la valeur du cookie `moncookie`, il suffit d'appeler la fonction `setcookie()` en modifiant la valeur du cookie.

Application : la miniboutique FoxShop

Pour appliquer les concepts que nous venons de voir, nous allons développer une boutique extrêmement simple. Les cookies seront utilisés à plusieurs endroits :

- Pour la gestion du panier.
- Pour enregistrer le profil client et lui éviter de devoir le réécrire à chaque achat.

Pour ne pas mélanger les scripts de cette boutique avec notre applicatif de gestion d'école, nous allons créer le répertoire *boutique* sur notre compte distant, et y placer tous nos développements. Pour accéder à la page d'accueil de la boutique, l'URL sera donc : `http://www.kernix/boutique`.

Au niveau de la base de données, nous avons besoin de deux tables :

- *produit* : `idproduit`, référence produit, nom, prix, description.
- *commande* : `idcommande`, liste des produits, montant global, nom client, prénom client, adresse client, date.

Notre applicatif sera composé de deux parties.

1. Le front-office, qui permet à un internaute d'acheter en ligne. Fichiers :

- *index.php3* : page d'accueil de la boutique.
- *boutique.php3* : affichage du catalogue.
- *ajout_caddie.php3* : ajout d'un produit au panier.
- *voir_caddie.php3* : résumé de la commande, identification de paiement.
- *enregistre_commande.php3* : enregistrement de la commande.

2. Le back-office, qui permet de gérer les produits et les commandes. Fichiers :

- *adm_produits.php3* : script permettant l'ajout de produits.
- *adm_commandes.php3* : script listant les commandes.
- Plus *identification.inc.php3*, *variables.inc.php3*, *haut.inc.php3*, *bas.inc.php3*.

● **Conseil** Ce pourrait être le bon moment...

... de fermer le livre et d'essayer de réaliser ce petit applicatif. Si vous cherchez de votre côté, les progrès viendront, en effet, bien plus vite qu'en lisant le livre.

Le back-office

Les tables dont nous avons besoin peuvent être définies de la manière suivante :

Table *produit* :

```
CREATE TABLE produit (
  idproduit int(10) unsigned NOT NULL auto_increment,
  reference varchar(16) NOT NULL default'',
  nom varchar(16) NOT NULL default'',
  description tinytext NOT NULL,
  prix float(12,2) unsigned NOT NULL default'0.00',
  PRIMARY KEY (idproduit)
)
```

Table *commande* :

```
CREATE TABLE commande (
  idcommande int(10) unsigned NOT NULL auto_increment,
  produits varchar(64) NOT NULL default'',
  montant float(12,2) unsigned NOT NULL default'0.00',
  nom varchar(16) NOT NULL default'',
  prenom varchar(16) NOT NULL default'',
  adresse tinytext NOT NULL,
  date datetime NOT NULL default'0000-00-00 00:00:00',
  PRIMARY KEY (idcommande)
)
```

Listing 8.5: Le script `adm_produits.php3`

```
<?php
include("variables.inc.php3");
include("identification.inc.php3");
if ($enregistre == "oui")
{
  if (empty($reference) || empty($nom) || empty($prix)
      || empty($description))
    die("ERREUR : tous les champs doivent être remplis.");
  if (ereg("^[[:digit:]]+(\.[[:digit:]]+)?$", $prix) == 0)
    die("ERREUR : prix non valide.");

  $liendb = mysql_connect($bddserver, $bddlogin, $bddpassword);
  mysql_select_db ($bdd);

  $sql = "INSERT INTO $table_produit (reference, nom, prix, description)
  ➤ VALUES ('$reference', '$nom', '$prix', '$description')";
  mysql_query ($sql);
}
```

```

mysql_close($liendb);
}
include("haut.inc.php3");
?>
<p align=left> :: ajouter un produit au catalogue de la boutique</p>
<form action="<?php echo $PHP_SELF; ?>" method="post">
<input type="hidden" name="enregistre" value="oui">
<table width="98%">
<tr>
<td>référence</td><td><input type="text" name="reference"></td>
<td rowspan="3" valign="top">description</td><td rowspan="3">
<textarea name="description" cols="40" rows="8"></textarea></td>
</tr>
<tr>
<td>nom</td><td><input type="text" name="nom"></td>
</tr>
<tr>
<td>prix</td><td><input type="text" name="prix"></td>
</tr>
</table>
<br>
<input type="submit" value="enregistrer le produit">
</form>
<hr>
<p align="left"> :: les produits de la boutique</p>
<table width="98%" align="center" border="1">
<tr>
<td class="intitule" align="center">id</td>
<td class="intitule" align="center">référence</td>
<td class="intitule" align="center">nom</td>
<td class="intitule" align="center">prix €</td>
</tr>
<?php
$liendb = mysql_connect($bddserver, $bddlogin, $bddpassword);
mysql_select_db($bdd);
$sql = "SELECT * FROM $table produit";
$resultat = mysql_query ($sql);
if (mysql_num_rows($resultat) > 0)

```

8 - Cookies et sessions

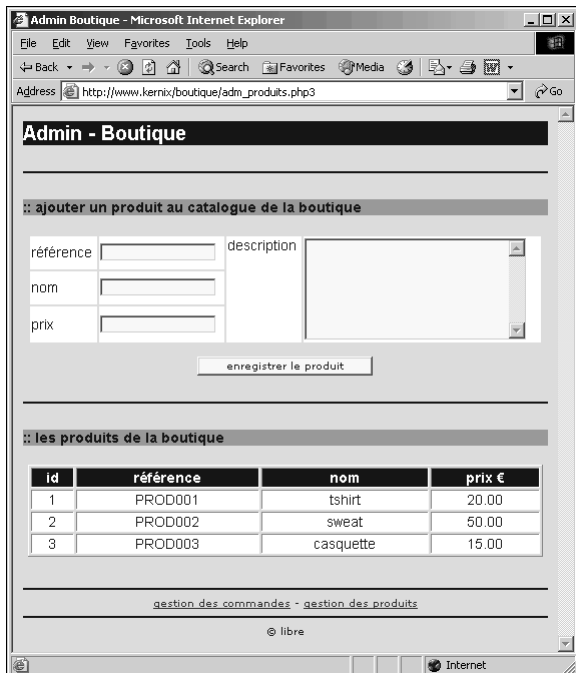
```
{
  while ($produit = mysql_fetch_array ($resultat))
  {
    $id = $produit['idproduit'];
    $reference = $produit['reference'];
    $nom = $produit['nom'];
    $prix = $produit['prix'];
    echo "<tr>";
    echo "<td align=center>$id</td>";
    echo "<td align=center>$reference</td>";
    echo "<td align=center>$nom</td>";
    echo "<td align=center>$prix</td>";
    echo "</tr>";
  }
}
else
{
  echo "<tr><td colspan=4 align=center>aucun produit</td></tr>";
}

echo "</table>";

mysql_close($liendb);

include("bas.inc.php3");

?>
```



◀ Fig. 8.4 :
Page d'administration
de la boutique