PHP - Les sessions

Date de dernière mise à jour : 27/06/2007 à 19:36

Présentation

Les sessions en PHP permettent de sauvegarder des variables de page en page pendant une certaine durée prédéfinie par PHP (et modifiable bien entendu). L'avantage des sessions sur les variables de type $_GET et $_POST est double : outre l'absence de formulaires (et même de tout code HTML) pour gérer les sessions, ce système va vous permettre de transmettre des variables sur toutes les pages de votre site d'une manière transparente pour vous (en PHP) ainsi que pour l'utilisateur. Vous imaginez si un membre devait cliquer à chaque fois sur les pages qu'il visite sur un bouton de formulaire indiquant qu'il est connecté ? ça ne serait pas vraiment viable comme système. Les variables de session vont pouvoir mémoriser tout ce que vous souhaitez pendant la durée de connexion d'un visiteur. Sur ce site par exemple, lorsque vous êtes connecté, j'utilise le système de sessions pour savoir que vous êtes toujours là sans que vous ayez besoin de retaper votre mot de passe ouvotre nom d'utilisateur à chaque page consultée.

La technique est simple : chaque utilisateur ayant besoin des sessions se voit attribuer un identifiant unique appelé **ID de session**. Cet identifiant est stocké sur le poste de l'internaute sous forme d'un cookie ou transite via l'URL si l'option **session.use_trans_sid** est à 1 (ou On) dans le fichier **php.ini**. Utiliser la méthode du cookie est plus que recommandée : un identifiant de session dans l'URL empêche de retenir l'adresse d'une page simplement par le visiteur néophyte, cela rallonge donc l'adresse de votre page. Cela nuit également au référencement de votre page. En effet, l'identifiant étant unique, lorsque le robot du moteur de recherche scanne votre site, il ne retombera jamais sur la même URL plusieurs fois de suite. Vous pouvez donc dire adieu à votre référencement en procédant comme ça.

Il est important de toujours avoir à l'idée qu'un identifiant de session peut être dérobé par un pirate. Évitez donc autant que possible le transit par l'url des informations de session (qui peuvent généralement se retrouver dans des copier-collers de liens dans un forum par exemple).

Utilisation des sessions

Avant d'utiliser les sessions sur une page, on doit toujours utiliser la fonction **session_start()** placée avant tout envoi de code HTML, et donc généralement tout en haut de votre page PHP :

```
<?php     session_start
();
//doit être placé avant tout echo ou autre code html     //Vous pouvez commencer à utiliser les sessions après le session_start();
?>
```

Pourquoi session_start() doit-elle être placée de la sorte ? et bien parce que si PHP utilise les cookies pour repérer quel est l'id de session utilisé par l'internaute, il va écrire cet id de session dans un cookie. Or, le protocole HTTP fonctionne de telle sorte que les en-têtes (qui permettent de dire à votre navigateur "crée un cookie ayant tel nom et telle valeur") sont envoyés avant le premier caractère HTML transmis. Cela veut dire que dès que vous transmettrez un caractère HTML, les en-têtes seront envoyés et vous ne pourrez plus les modifier, vous ne pourrez donc plus écrire le cookie de session. En général, si vous faites une maladresse de ce style, vous vous retrouverez avec une erreur de headers already sent.

Les variables de session fonctionnent comme les variables classiques. Voici un exemple pour attribuer une valeur à une variable de session nommée **login** :

```
<?php     session_start
();
//doit être appelée avant toute utilisation des sessions
$_SESSION
[
'login'
]=
'valeur'
;
```

```
?>
```

Pour récupérer la valeur d'une variable de session (sur la même page ou sur une autre page, après un session_start()), il s'agit de la même procédure que pour une variable classique hormis l'ajout du session_start() tout en haut de la page :

```
<?php session_start
(); if(isset(
$_SESSION
[
'login'
])) { echo
$_SESSION
[
'login'
];
//affiche la valeur de $_SESSION['login'] qui a pu être attribué dans une autre page php
}
?>
```

Pour savoir si une variable de session existe, on procède de la même façon que pour les autres variables à savoir qu'on utilise la fonction isset().

Vous souhaiterez peut-être effacer une ou plusieurs variables de session, au même titre qu'une autre variable en PHP, il faut utiliser pour ça la fonction **unset()**.

Voici un petit exemple simple pour effacer la variable de session **login** que nous avons créée tout à l'heure (le membre ne pourra donc plus être considéré comme identifié) :

```
<?php session_start
(); if(isset(
$_SESSION
[
'login'
])) { unset(
$_SESSION
[
'login'
]); }
?>
```

Si vous souhaitez effacer toute la session d'un coup, vous pouvez utiliser la fonction **session_destroy()** pour détruire toutes les variables de session d'un visiteur. Cette fonction doit quand même avoir été initialisée par un session_start(), comme vous le feriez sur toutes vos pages.

Pour une plus grande sécurité, vous aurez peut-être envie de connaître l'id de session du visiteur, pour pouvoir effectuer divers traitements dessus ou encore pour l'enregistrer quelque part pour en garder une trace. PHP dispose d'une fonction destinée à cet usage, il s'agit de la fonction **session_id()**.

```
<?php session_start
(); echo
session_id
();
//Retourne l'identifiant de session
?>
```

Vous pouvez également modifier l'id de session courant en spécifiant une valeur dans la fonction **session_id()**, mais par contre il faudra cette fois-ci

que vous le fassiez avant le **session_start()** sinon ça ne fonctionnera pas.

```
<?php     session_id
(
md5
(
mt_rand
()));
session_start
();     echo
session_id
();
//Retourne l'identifiant de session
?>
```

Voilà, ce que vous venez de voir suffit amplement pour gérer un système de sessions sur votre site, pour réaliser un espace membres ou encore un panier en ligne. Comme quoi, des notions simples permettent souvent de faire des choses concrètes en PHP.

Chapitre suivant