TP 3 : sessions, cookies

I. Introduction aux sessions

Le protocole http est sans état. Cela signifie qu'il n'y a aucune donnée conservée entre deux appels à un site donné. Il est cependant possible de créer une session lorsque l'utilisateur accède au site pour la première fois. Cela permet alors de stocker des données sur le serveur qui seront associées uniquement à cet utilisateur jusqu'à ce qu'il ferme son navigateur ou qu'il se déconnecte (le comportement peut varier en fonction de la configuration du serveur).

Le but de cet exercice est de découvrir le fonctionnement des sessions en PHP. Les explications détaillées seront abordées en cours prochainement. Pour « démarrer » une session, il faut utiliser la fonction `session_start`

L'appel à `session_start` doit être réalisé avec toute autre action, code HTML compris. Dans le cas contraire, un message d'erreur sera affiché.

Il est possible de stocker des données dans la session en utilisant la variable super-globale `$_SESSION` qui est un tableau associatif (exactement comme `$_GET` et `$_POST`). L'ensemble des données stockées lors de l'accès précédent est alors accessible.

1. Écrivez un script qui crée une page avec simplement un formulaire permettant à l'utilisateur de saisir un message dans une zone de texte (un `textarea`) avec un bouton pour valider.
2. Lorsque l'utilisateur valide le formulaire, le message est stocké en session. Lorsque la page est affichée, en plus du formulaire, tous les messages doivent apparaitre. Nous aurons donc besoin d'un tableau de messages en session.
3. Nous souhaitons maintenant autoriser uniquement un utilisateur logué à déposer des messages. Modifiez votre script de la manière suivante :
    a. Si l'utilisateur n'est pas logué, on affiche un formulaire de login (pseudo + mot de passe) ; le pseudo et le mot de passe seront stockés en dur dans le code PHP.
    b. Si l'utilisateur est logué, on affiche le formulaire permettant de saisir un message, ainsi que tous les messages saisis au préalable.

Vous ne devez utiliser qu'un seul script.

4. Nous souhaitons maintenant permettre à un utilisateur de pouvoir se déconnecter. Lorsqu'il est logué, nous ajoutons donc un formulaire avec un bouton « Déconnexion ». Lorsque l'utilisateur se déconnecte, les messages sont supprimés.
5. Que se passe-t-il si vous fermez le navigateur sans vous déconnecter et que vous l'ouvrez à nouveau ? Etes-vous toujours connecté ? Ce comportement est-il le même sous Chrome, Firefox et Bing ?
6. Que se passe-t-il si vous réalisez les actions suivantes :
    a. Exécutez le script dans un onglet de navigation privée ;
    b. Connectez-vous ;
    c. Fermez l'onglet ;
    d. Exécutez à nouveau le script dans un onglet de navigation privée.

II. Introduction aux cookies

Contrairement à une session, un cookie est une donnée (une chaîne de caractères) associée à l'utilisateur mais stockée sur le poste de celui-ci. La création du cookie se fait sur le serveur, en PHP par exemple, et la valeur est envoyée au client via le message de réponse http, dans l'en-tête. Le client (le navigateur) sauvegarde alors la valeur du cookie localement (le répertoire dépend du navigateur). Lorsque l'utilisateur parcourt les différentes pages d'un même site (domaine), la valeur du cookie est envoyée systématiquement par le navigateur et peut être récupérée et modifiée du côté du serveur.

Pour déposer un cookie, il faut utiliser la fonction `setCookie` et pour récupérer sa valeur, on utilise la variable super-globale `$_COOKIE`. Consultez le manuel PHP pour plus d'information sur la fonction `setCookie`.

1. La modification de la variable `$_COOKIE` n'a pas d'effet sur le cookie en lui-même. Elle contient uniquement les valeurs envoyées sur le client. Pour modifier la valeur d'un cookie, il faut donc utiliser obligatoirement la fonction `setCookie`.
2. Comme pour la session, l'appel à `setCookie` doit être réalisé avec toute autre action, code HTML compris.

1. Ecrivez un script qui vérifie si l'utilisateur a déjà visité le site ou non en utilisant les cookies. Quand l'utilisateur arrive sur le site pour la première fois, le script affiche : « Je ne vous connaissais pas, mais maintenant si !». Sinon, le script affiche « Je vous connais ». Utilisez une durée de validité pas trop longue (de quelques secondes) pour faciliter vos tests.
2. Créez deux autres scripts : un qui permet de supprimer le *cookie* (`supprime.php`) et un autre qui permet d'indiquer si le *cookie* existe ou non (`etat.php`). Ajoutez sur chaque script, un lien vers les deux autres scripts.
3. Que se passe-t-il lorsque le *cookie* est créé et que vous accédez au script `etat.php` via un onglet de navigation privée ? Peut-on créer des *cookies* dans un onglet de navigation privé ?
4. Qu'est-ce qu'un *cookie* de session ? Refaites les tests précédent en utilisant un *cookie* de session.
5. Nous souhaitons reprendre l'exercice précédent. Au lieu de stocker les messages en session, nous les stockons dans un *cookie*.