Objectifs

- ☞ Intéractions PHP/MySQL
- ☞ Interroger et mettre à jour une base de données via PHP.

Dans ce TP on va créer un petit clone simple de Twitter, que l'on va appeler "Gazouillis". La semaine prochaine, on rajoutera une vraie gestion des utilisateurs et des followers.

Manipulation [Récupérer les fichiers nécessaires au TP, et mettre en place la base de données]

Voici la procédure à suivre :

1. Ouvrir un terminal
2. Taper dans l'ordre les commandes suivante :
   (a) `cd public_html`
   (b) `wget http://pablo.rauzy.name/files/mon/tpweb/tp6.tgz`
   (c) `tar -xzf tp6.tgz`

Cela vous créera un dossier "gazouillis" dans votre dossier "public_html".
Ce dossier contient 6 fichiers : index.php, functions.php, user.php, actions.php, register.php, et gazouillis.sql.
Le fichier index.php permet d'envoyer un gazouillis et liste ceux qui sont dans la base. Le fichier user.php liste les gazouillis d'un utilisateur en particulier. Le fichier register.php contient le formulaire d'inscription d'un utilisateur. Le fichier actions.php gère les actions. Le fichier functions.php contient les fonctions que les autres fichiers utilisent.

Importez le fichier gazouillis.sql dans PHPMyAdmin pour créer les tables "gzl_users" et "gzl_messages" que l'on va utiliser pendant ce TP.
Lisez bien le fichier gazouillis.sql pour comprendre comment sont les tables.

Exercice 1 [Connexion et déconnexion à la base de données]

Le but de cet exercice est de permettre à actions.php, index.php et user.php de se connecter à la base de données.

1. Ouvrir le fichier actions.php et cherchez y les mentions de `À COMPLÉTER - exo 1`, il y en a 2.
2. À la première : utiliser les fonctions `mysql_connect` et `mysql_select_db` et pour ouvrir une connection à votre base de données MySQL. Le résultat de `mysql_connect` doit être mis dans la variable `$db`. Si la connexion ne fonctionne pas, utiliser les fonctions `die` et `mysql_error` pour signaler qu'il y a un problème et arrêter l'exécution du script.
3. À la seconde : utiliser la fonction `mysql_close` pour fermer la connexion à la base de données.
4. Une fois que vous avez la solution, copiez la dans index.php et user.php aux endroits où il y a `À COMPLÉTER - comme exo 1`.

La documentation des fonctions PHP se trouve à l'adresse suivante : `http://fr.php.net/`*nom_de_la_fonction*.

Le principe du fichier actions.php est d'appeler la fonction correspondant à l'action reçue en paramètre GET, et cette fonction renvoie soit `null` si tout va bien, soit une chaine explicant l'erreur. Si la fonction renvoie `null`, on retourne à l'accueil, sinon, on affiche une page avec le message d'erreur.
Dans le fichier functions.php, toutes les fonctions prennent un premier argument qui s'apelle $db et qui sera la connexion à la base de donnée de vous avez créé à l'exerice 1.

Exercice 2 [Création d'un utilisateur]

1. Dans le fichier register.php :
   (a) Créer un formulaire qui utilise la méthode POST et qui a pour action “actions.php ?action=create-user”.
   (b) Il doit comporter deux champs texte : “pseudo”, et “email”, ainsi qu’un bouton submit.
2. Dans le fichier functions.php, coder la fonction `create_user` :
   (a) Utiliser la fonction `mysql_query` pour faire une requête qui insert l’utilisateur dans la table “gzl_users”.
   (b) Si tout s’est bien passé, renvoyer `null`. En pratique quand on fait un INSERT et que ce n’est pas critique, on suppose que tout se passe bien... Attention à bien tester votre code quand même !

Exercice 3 [Création d’un gazouilli]

1. Dans le fichier functions.php, coder la fonction `get_user_id`. Cette fonction reçoit le pseudo en paramètre et doit retourner l’identifiant de l’utilisateur :
   (a) Utiliser les fonctions `mysql_query` et `mysql_fetch_assoc`.
   (b) Si l’utilisateur n’existe pas la fonction doit renvoyer `false`.
2. Dans le fichier functions.php, coder la fonction `create_gazouilli`. Cette fonction reçoit en paramètre le pseudo et le message :
   (a) Utiliser la fonction `get_user_id` que vous venez de coder pour récupérer l’id de l’utilisateur.
   (b) Si cette fonction ne renvoie pas un nombre (tester avec la fonction `is_numeric`) alors renvoyer une erreur “utilisateur inconnu”.
   (c) Sinon, utiliser la fonction `mysql_query` pour faire une requête qui insert le gazouilli dans la table “gzl_messages”. Utiliser la fonction `time` pour récupérer le timestamp courant.
   (d) Avant de faire l’insertion dans la base de données, n’oubliez pas de tester si le message n’est pas trop long (il doit faire moins de 140 caractères) avec la fonction `strlen`.
   (e) Si tout s’est bien passé, retournez `null` (oui c’est bizarre, mais souvenez vous qu’on retourne les erreurs, donc renvoyer `null` signifie “pas d’erreur !”).

Exercice 4 [Récupération des gazouillis]

1. Dans le fichier functions.php, coder la fonction `fetch_gazouillis`. Il s’agit de récupérer toutes les informations sur les gazouillis, et aussi le pseudo de l’utilisateur qui l’a écrit pour chaque gazouilli. Ensuite, on va mettre ça sous une forme pratique avant de le renvoyer :
   (a) Utiliser la fonction `mysql_query` pour faire la requête demandée (c’est à dire, récupérer tous les champs de la table gzl_messages et le champ “pseudo” de la table gzl_users). On veut que les gazouillis soient affichés du plus récent au plus ancien.
   (b) Créer une variable, disons `$gazouillis` et l’initialiser comme un tableau vide (fonction `array`).
   (c) La fonction `mysql_fetch_assoc` renverra `false` si il n’y a plus de résultat à récupérer dans la ressource renvoyée par `mysql_query`. Elle est donc utilisable comme condition dans une boucle `while` (si besoin, voir les exemples sur la documentation). L’idée ici est simplement de mettre dans le tableau `$gazouillis` tous les résultats.
   (d) À l’intérieur de la boucle `while` : pour rajouter une nouvelle valeur dans un tableau il suffit de ne pas préciser l’index (“`$foo[] = 'toto';`” rajoute “toto” à la fin du tableau `$foo`).
   (e) La fonction doit retourner le tableau de résultat.
2. Dans le fichier functions.php, coder la fonction `fetch_user_gazouillis`, c’est la même que la précédente sauf qu’elle prend en argument l’identifiant d’un utilisateur, et qu’elle ne doit récupérer que ses gazouillis à lui.
3. Dans le fichier functions.php, coder la fonction `get_user_pseudo`. Cette fonction reçoit l’identifiant en paramètre et doit retourner le pseudo de l’utilisateur :
   (a) Utiliser les fonctions `mysql_query` et `mysql_fetch_assoc`.
   (b) Si l’utilisateur n’existe pas la fonction doit renvoyer `false`.
   (c) Bon hé heu c’est la même que `get_user_id` de l’exercice 3 mais dans l’autre sens donc on copie colle on fait les petits changements qui vont bien et hop hop c’est fait !

Exercice 5 [Modification d'un utilisateur]

1. Dans user.php :
   (a) Créer un formulaire qui utilise la méthode POST et qui a pour action "actions.php ?action=edit-user&amp ;id=ID", où "ID" est remplacer par l'identifiant de l'utilisateur, comme c'est déjà fait dans le lien pour supprimer l'utilisateur.
   (b) Il doit comporter un champs texte : "new_pseudo", ainsi qu'un bouton submit.
2. Dans functions.php, coder la fonction `edit_user`. Celle-ci reçoit l'identifiant de l'utilisateur, ainsi que son nouveau pseudo.
   (a) Utiliser la fonction `mysql_query` pour mettre à jour la table `gzl_users` avec le nouveau pseudo de l'utilisateur.
   (b) N'oubliez pas de retourner `null`.

Exercice 6 [Suppression d'un gazouilli ou d'un utilisateur]

1. Dans functions.php, coder la fonction `delete_gazouilli`. Celle-ci reçoit l'identifiant du gazouilli à supprimer en paramètre.
   (a) Utiliser la fonction `mysql_query`.
   (b) N'oubliez pas de retourner `null`.
2. Dans functions.php, coder la fonction `delete_user`. Celle-ci reçoit l'identifiant de l'utilisateur à supprimer en paramètre.
   (a) Attention aux contraintes de clef étrangères dans la base !
   (b) Utiliser la fonction `mysql_query`.
   (c) N'oubliez pas de retourner `null`.