Compte rendu de TP de MySQL

Coralie Pierrat • Décembre 2009 • 3EI

MySQL est un SGBD pour Système de Gestion de Base de Données. Il s'agit d'une des logiciels les plus utilisés dans le monde pour ce type d'applications. L'objectif de ces travaux pratiques est de nous faire découvrir les bases de ce système mais également de travailler sur le langage PHP. Le langage php permet de créer des sites Internet dynamiques. L'application pratique réside en la création d'un formulaire de carte de visite en ligne où différents utilisateurs pourraient remplir les champs demandés. Il est ensuite nécessaire de récupérer les informations saisies et de les répertorier dans une base de données. C'est alors que le logiciel MySQL prend toute sa valeur.

L'adresse de mon site Internet est :
http://mosieur.free.fr/2009/pierratc/formulaire/form.php
Afin d'y accéder, le nom d'utilisateur est coralie et le mot de passe requis est pingouin.

1) Les outils utilisés:

Afin de stocker tous les documents m'appartenant, il a été nécessaire d'utiliser un compte d'hébergement fourni par free. Ainsi il est possible de créer des pages Internet. Pour ma part, j'ai utilisé le compte mosieur, avec le compte pierratc. J'ai tout d'abord créé une page html très basique afin de comprendre les bases. Afin de charger les différents fichiers, nous avons eu recours au logiciel Fetch.

Ensuite, il a fallu éditer les différentes pages, pour cela TextWrangler s'est avéré utile. Il s'agit d'un éditeur de texte très performant pour la création de code puisqu'il ajoute des couleurs aux balises, conférant ainsi une meilleure lisibilité.

Enfin, la visualisation de notre travail final a été réalisée grâce au navigateur Firefox.

2)Protection du site:

Afin de limiter l'accès du formulaire uniquement à certaines personnes, nous avons créé un fichier intitulé interdit. Celui-ci contient le document .htpassword où sont contenus le nom d'utilisateur et le mot de passe s'y rattachant. Ensuite, nous avons inséré un document .htaccess, qui requiert un nom d'utilisateur valide. Notre site Internet est ainsi sécurisé.

3)Création du formulaire en ligne:

La création du formulaire en ligne se fait par le biais d'un fichier .PHP. Le code contient des commentaires expliquant les différentes étapes de création. Il est nécessaire de mettre en page notre formulaire. Pour cela, il faut se référer à une feuille de style (.css). Nous avons également organiser les différents champs de notre formulaire sous forme de tableau afin qu'ils soient alignés. Pour cela nous avons eu recours à la balise table ainsi qu'aux divers <tr><td> à l'intérieur du code.

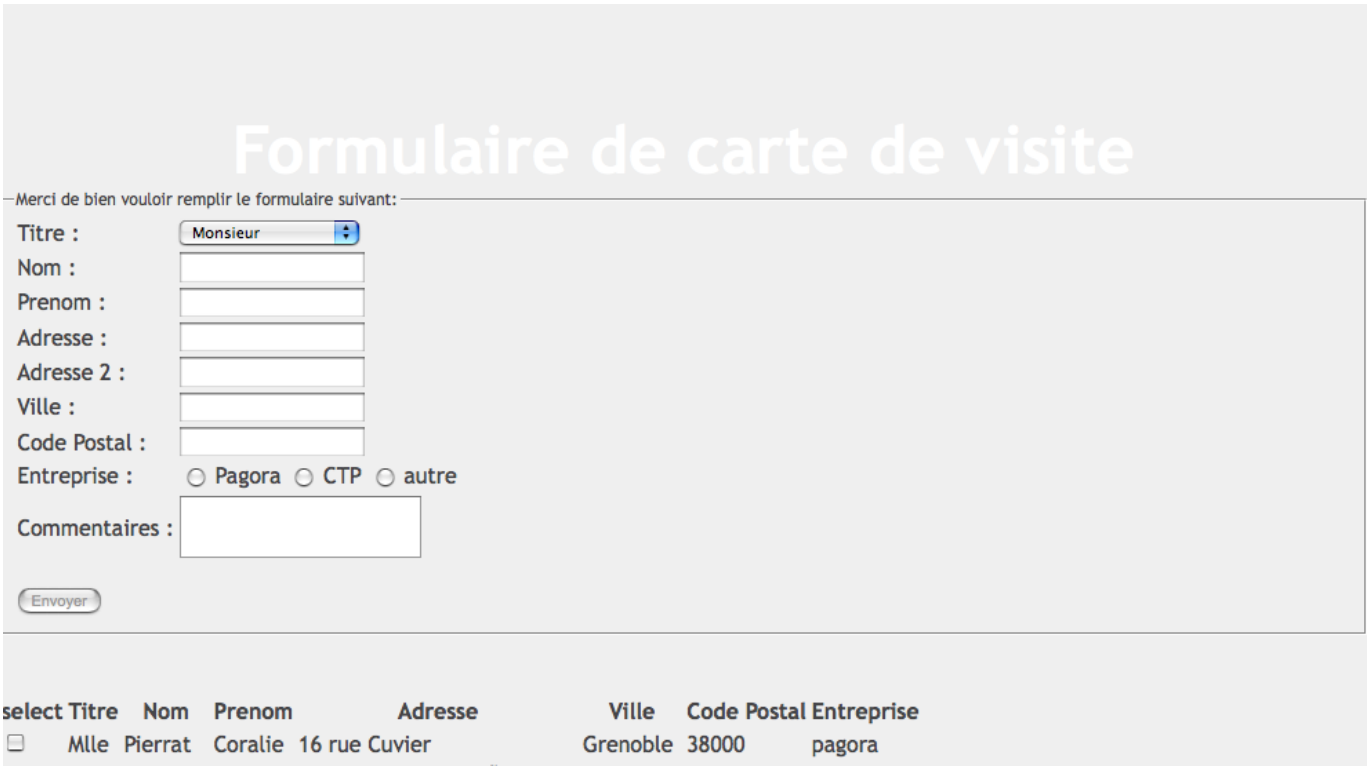
Le formulaire est codé en langage HTML mais php sera utilisé pour une partie, notamment dans la récupération des données saisies et la transmission vers la base de données.

Dans notre formulaire, nous créant un champ (fieldset), qui comportera une légende. A l'intérieur de celui-ci se trouveront plusieurs champs. Certains seront des zones de texte à remplir, notamment pour le nom, le prénom ou l'adresse ou des aires de texte comme pour les commentaires éventuels. D'autres pourront être des boutons de type radio correspondant à des cases à cocher (ici pour le choix de l'entreprise). Enfin, nous aurons recours à une liste déroulante pour le choix du titre.

Afin de valider les informations que l'utilisateur a entrées, un bouton «envoyer» se situe en bas de page. Le langage PHP permet d'utiliser deux méthodes pour renseigner les variables: get ou post. Nous avons opté pour cette dernière grâce à laquelle les variables ne sont pas dans l'url lorsque les données sont envoyées.

Afin d'insérer les valeurs récupérées avec la méthode POSt, on utilise la fonction mysql_query.

Voici un aperçu de mon formulaire en ligne :



4)Récupération des données saisies grâce à MySQL

Afin d'utiliser les données saisies dans le formulaire, on utilise le site phpmyadmin. Grâce à celui-ci, nous pouvons créer une table de base de données. Cette dernière nous permettra de récupérer les données insérées dans les formulaires et de les traiter. Nous créons une table qu'il est nécessaire de structurer. Celle-ci contient les informations le nom des champs à remplir, le type de valeur que l'utilisateur entrera (chiffre, date, heure, texte,...) ainsi que la taille maximale pouvant être contenue. Un champ permet de numéroter progressivement les envois.

Les requêtes dans MySQL s'effectuent grâce à 4 opérations de base que sont l'union, l'intersection, la différence et le produit cartésien.

Voici un aperçu de la structure de la table puis des informations récupérées (ongletaffichage):

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
☐	index	int(11)			Non	*aucune*	auto_increment	
☐	date	datetime			Non	*aucune*		
☐	titre	enum('M.','Mme','Mlle')	utf8_general_ci		Oui	*NULL*		
☐	nom	text	utf8_general_ci		Oui	*NULL*		
☐	prenom	text	utf8_general_ci		Oui	*NULL*		
☐	adresse	text	utf8_general_ci		Oui	*NULL*		
☐	adresse2	text	utf8_general_ci		Oui	*NULL*		
☐	ville	text	utf8_general_ci		Oui	*NULL*		
☐	codepostal	text	utf8_general_ci		Oui	*NULL*		
☐	entreprise	text	utf8_general_ci		Oui	*NULL*		
☐	description	text	utf8_general_ci		Oui	*NULL*		

Tout cocher / Tout décocher *Pour la sélection :*

		index	date	titre	nom	prenom	adresse	adresse2	ville	codepostal	entreprise	description
☐	✎ ✕	1	2009-11-13 09:31:00	Mlle	Pierrat	Coralie	16 rue Cuvier	*NULL*	Grenoble	38000	pagora	*NULL*
☐	✎ ✕	2	2009-11-13 09:33:07	M.	Maréchal	Vincent	11 rue Maurice Grignoux	Le Rabot	Grenoble	38000	Pagora	*NULL*
☐	✎ ✕	16	2009-11-13 11:35:55		COULOM	Romain	Mas Morer av de l'aéroport		Rivesaltes	66600	pagora	Ca marche bien, enfin on dirait, quand j'ai voulu ...

Tout cocher / Tout décocher *Pour la sélection :*

Afficher : 30 enregistrement(s) à partir de l'enregistrement n° 0

en mode horizontal et répéter les en-têtes à chaque groupe de 100

5) Conclusion.

Nous obtenons, grâce à la combinaison de ces divers logiciels deux fichiers qui communiquent entre eux. L'utilisateur enregistre ses informations via le formulaire en ligne puis l'administrateur peut les récupérer dans sa table.

Cette initiation à l'utilisation du langage php ainsi qu'au logiciel MySQL nous permet de découvrir les bases de la réalisation de pages internet dynamiques. Nous avons pu approfondir les notions de programmations en html que nous avions reçues l'an passé et enrichir nos connaissances dans le domaine très complexe et diversifié de l'encodage de site web.

Carcone Fiona
3EI
24/11/09

Création d'un formulaire

Dans ce rapport, nous allons expliciter les étapes à réaliser pour créer un formulaire et enregistrer les informations rentrées dans une base de donnée.

1) Sécurité

Tout d'abord, nous avons créé un fichier sécurité à l'aide de .htaccess et .htpassword afin de sécuriser l'accès au fichier source.
L'identifiant et le mot de passe sont les suivants : **fanny:calzone**

Ensuite, un fichier erreur a été créé afin de prévenir, par des messages d'erreur, l'opérateur. Selon le type d'erreur les codes sont différents :

```
$erreurs['404']="Page non trouvée";
$erreurs['401']="Page non autorisée";
$erreurs['500']="Erreur serveur"
```

2) Création du formulaire

Nous pouvons maintenant commencer la création du formulaire qui se trouve à l'adresse http://licpro.pagora.free.fr/2009/carconef/Carte%20de%20visite/formulaire.php

Dans le body, on créé une balise <form> qui va nous servir à définir le début du chemin. Ensuite on met le titre de notre page. Les effets de styles seront fait plus tard avec l'aide d'un fichier css.
La balise <fieldset> nous a permis de créer un cadre autour de notre texte.
Lors de l'écriture du code html de notre formulaire, nous avons utilisé deux méthodes différentes. La méthode GET nous a permis dans un premier temps de voir toutes les informations, rentrées dans le formulaire, dans l'adresse URL. La méthode POST permet de cacher les informations rentrées.
Les zones de saisie du formulaire ont été placées dans un tableau afin d'avoir un bon alignement entre chaque ligne. De plus, pour créer ces zones de saisies, il faut utiliser la balise input type="text". Il a également été mis en place une sécurité qui consiste à sauvegarder les données rentrées dans le formulaire précédent. Cela permet de retrouver ses informations si l'utilisateur à fait une erreur de manipulation.
Ce formulaire a aussi permis de voir les codes pour créer des boutons (le input submit permet de créer un bouton dans notre page) ou des listes déroulantes.

Notre formulaire est alors fonctionnel. Il reste cependant à l'améliorer au niveau de l'esthétique. Pour cela, il faut faire un fichier de style css.
Dans celui-ci, nous avons précisé la police souhaitée et la couleur du fond de la page web.
Afin de donner un aspect plus symétrique, le titre principal a été centré dans la page.

3) Création de la base de donnée

Pour créer cette base, nous sommes allé sur le site http://sql.free.fr. De là, nous avons créer pas à pas les éléments de notre table.

Champ	Type
Index	INT
Date	DATETIME
Titre	ENUM
Nom	VARCHAR
Prénom	VARCHAR
Adresse	VARCHAR
Ville	VARCHAR
Code postal	VARCHAR
Entreprise	ENUM
Commentaires	TEXT

Photo 1

Dans un premier temps il faut rentrer les noms de nos différents champs en fonctions des noms utilisés dans le formulaire (pour des questions pratiques). Nous pouvons remarquer sur la *photo 1* que les deux premiers champs ont été ajoutés par rapport au formulaire. La ligne Index permet d'incrémenter automatiquement les informations reçues du formulaire, et la ligne date permet de sauvegarder la date à laquelle le formulaire a été rempli.
Ensuite on rempli les colonnes. La colonne type permet de définir le type de données auquel doit s'attendre à recevoir la table. Par exemple pour des textes assez courts, on prendra VARCHAR (limite le nombre de caractères à 255, ce qui est largement suffisant pour les informations demandées). ENUM est utilisé quand on a le choix qu'entre un certain nombre de réponse.

Taille/Valeurs*1	Défaut2	Interclassement	Attributs	Null	Index	A_I	Commentaires
	aucune			☐	UNIQUE	☑	
	aucune	utf8_general_ci		☐	---	☐	
	NULL	utf8_general_ci		☑	---	☐	
30	NULL	utf8_general_ci		☑	---	☐	
30	NULL	utf8_general_ci		☑	---	☐	
50	NULL	utf8_general_ci		☑	---	☐	
30	NULL	utf8_general_ci		☑	---	☐	
15	NULL	utf8_general_ci		☑	---	☐	
	NULL			☑	---	☐	
	NULL	utf8_general_ci		☑	---	☐	

Photo 2

Enfin, TEXT est utilisé pour des textes plus longs. La colonne taille/valeur permet de limiter manuellement le nombre de caractères.

Une fois la base de donnée créée, il faut s'assurer que les informations rentrées dans le formulaire sont bien stockées dedans. Pour cela on rajoute une ligne de code dans le fichier php du formulaire à l'aide de la balise mysql_query.
Une fois le formulaire rempli, on vérifie que les informations sont bien enregistrées dans la base de données.

			Index	Date	Titre	Nom	Prenom	Adresse	Adresse 2	Ville	Code_postal	Entreprise	Description
☐	✎	✕	1	2009-11-13 09:22:55	Mlle	Calzone	Fanny	5 rue du Paradis	*NULL*	grenoble	38000	CTP	un peu folle
☐	✎	✕	5	2009-11-13 09:28:58	Mlle	Da Silva	Aurélie	15 rue du Portos	*NULL*	grenoble	38000	Pagora	un peu poilue
☐	✎	✕	3	2009-11-13 09:25:58	M	olivier	cyril	33 rue du plus beau	*NULL*	st chamas	13000	Pagora	CFA en papeterie
☐	✎	✕	4	2009-11-13 09:22:00	Mme	Coletta	Maria	Via sparuro, 7	*NULL*	cervaro	64313	CTP	vit en italie
☐	✎	✕	6	2009-11-13 09:32:00	M	Carcone	Sébastien	4, rue du rital	*NULL*	paris	75015	CTP	Grand frère qui travaille dans les musées
☐	✎	✕	7	2009-11-13 10:25:25	Mme	Coletta	Maria	Via sparuro, 7	*NULL*	cervaro	64313	CTP	vit en italie
☐	✎	✕	14	2009-11-13 11:04:12	Mme	calzone	fannia	4 rue brahim		grenoble	38000	CTP	coucou

Photo 3

On remarque que notre travail fonctionne.

Romaric CHERBONNEL 3EI

Compte Rendu TP MySQL

0 – Introduction

A travers les quelques séances que nous avons passé à travailler sur le PHP et sur les bases de données, nous avons eu pour objectif la réalisation d'un formulaire sous forme de page internet, où des personnes pourraient renseigner des informations les concernant. L'objectif étant par la suite de récupérer ces données dans une base de données MySQL afin de pouvoir les traiter et les utiliser. Pour ce faire, nous avons utilisé l'éditeur HTML 'TextWrangler', le navigateur internet 'Firefox', le logiciel de transfert FTP 'Fetch', ainsi qu'un compte Free permettant de stocker jusqu'à 10Go de données, et gérant les bases de données MySQL.
L'adresse de ce site perso est : http://romaricche04.free.fr/cherbonr/.

1 - Index.html

Dans un premier temps, nous allons créer un répertoire appelé 'cherbonr' où nous allons placer tout les fichiers et dossiers relatifs au site internet. Puis, afin de pouvoir afficher une page lorsque nous nous connectons au site, nous créons un fichier 'index.html' qui sera affiché par défaut à l'adresse url suivante : http://romaricche04.free.fr/cherbonr/.

Dans ce fichier html, nous allons retrouver la structure classique composée du <HEAD> puis du <BODY>. Nous insérons un titre dans le <HEAD> à l'aide de la balise <TITLE>, ainsi qu'une balise <META> qui indiquera le type du fichier. Le tout étant placé à l'intérieur d'une balise <HTML>. Enfin, afin de vérifier le bon fonctionnement du fichier, on insert un petit texte dans le <BODY>.

2 – Autoindex, .htaccess, et .htpasswd

Une fois cela réalisé, nous avons créé un dossier 'autoindex' placé à la racine du site, où nous avons placé deux fichiers TXT : HEADER.txt et README.txt. Le but est ici de pouvoir accéder aux fichiers placés dans ce répertoire, sans passer par une page HTML. Ainsi le HEADER.txt permet d'afficher un petit texte en entête, tandis que le README.txt s'affiche comme un commentaire sous la liste des dossiers et fichiers.
De plus afin de protéger l'accès à cette partie du site, nous avons créé un fichier .htaccess que nous avons placé dans ce répertoire. Ce fichier comporte les instructions suivantes :

ErrorDocument 404 cherbonr/http/erreur.html
PerlSetVar AuthFile cherbonr/http/interdit/.htpasswd
AuthName "Acces restreint dans cette zone : montrez votre patte blanche..."
AuthType Basic
require valid-user

La première ligne renvoi à la page erreur.html dans le cas où l'adresse url rentrée ne serait pas correcte. La seconde ligne quant à elle indique où se trouve la liste des utilisateurs autorisés, ainsi que les mots de passe associés à chacun d'entre eux. Enfin les autres lignes indiquent le type de sécurité mis en place pour cette page.
Les utilisateurs recensés et autorisés à accéder à cette partie du site sont indiqués dans le .htpasswd qui se trouve lui dans le dossier cherbonr\http\interdit\ :

romaric:bernard
bfricottin:clochett

La mise en forme dans ce fichier étant la suivante : nom_utilisateur:mot_de_passe
Enfin, afin de protéger ce fichier, et que personne ne puisse le voir ni le modifier, on place dans ce même répertoire le .htaccess suivant :

deny from all

Enfin, dans un souci de compatibilité des encodages UTF-8, on place un .htaccess à la racine du site, dans le dossier 'cherbonr' où l'on insert les lignes suivantes :

adddefaultcharset utf-8
errordocument 404 cherbonr/http/erreur.php

2 – Erreur.php

Comme nous l'avons vu précédemment, nous avons utiliser une page d'erreur personnalisée en cas de problème rencontré par le serveur (page non trouvée, erreur d'authentification…). Nous allons maintenant voir comment cette page située dans le répertoire cherbonr\http\ a été construite.

Dans un premier temps, on remarque que c'est une page html des plus basique avec un <HEAD> et un <BODY>. Mais intéressons nous plus particulièrement au <BODY> :
Tout d'abord, on a inséré un « FATAL ERROR » pour faire peur à l'utilisateur, qui est la seule partie statique de la page, le reste étant en PHP.
La première chose a remarquer est la façon dont on signale le passage au PHP, en effet pour cela il faut insérer la commande suivante : '<?php' le 'php' étant facultatif, et pour terminer, insérer celle-ci : ' ?>'.

Concentrons nous à présent sur le contenu de cette partie en PHP. Nous avons d'abord appelé la fonction 'echo' qui permet l'affichage d'un texte (ici « Coucou c'est moi »), puis on la termine par un ' ;'. On notera la présence du '\n' qui applique un retour à la ligne.
Ensuite, on dresse une liste non exhaustive d'erreurs possibles en leur attribuant le numéro correspondant grâce à la fonction 'array' :

```
$erreur=array(
                    '404' => "Page non trouvée",
                    '401' => "Page non autorisée",
                    '500' => "Erreur serveur",
                    );
```

et on en appelle une par la fonction 'echo' (ici la 401).
Ensuite on va afficher l'erreur retournée par le serveur grâce à la ligne suivante :

```
echo "<p>Le message d'erreur est :
".$erreur[$_SERVER['REDIRECT_STATUS']]."</p>\n";
```

Par la suite, on va afficher toutes les erreurs listées précédemment, grâce à la fonction 'echo' couplée à une boucle 'foreach'.
Enfin, afin d'avoir une vision globale des informations retournées par le serveur, nous allons créer une table dans laquelle on va afficher ces informations :

```
<table>

    <?php
        foreach($_SERVER as $k => $v){
        echo "<tr><td><tt>$k</tt></td><td><i>$v</i></td></tr>\n";
        }
    ?>
</table>
```

La page d'erreur ainsi terminée (mais non réaliste), nous allons à présent nous intéresser à la création d'un formulaire en ligne.

3 – Form.php

Pour cette page HTML, nous allons créer un <HEAD>, un <BODY>, ainsi que les divisions suivantes 'container', 'logo', 'titre', 'cadre', et 'footer'.
Pour commencer, intéressons nous au <HEAD>, qui comporte une balise <META> permettant de définir la racine du site (ici http://romaricche04.free.fr/cherbonr). On trouve ensuite une balise <TITLE> classique, et une balise <LINK> permettant de définir une feuille de style .css et de la localiser.

Le <BODY> contiendra la division 'container' dans laquelle on insèrera la division 'logo', 'titre', 'cadre', et 'footer'. Dans 'logo' nous allons mettre un lien vers cette page afin de la recharger. Le 'titre' quant à lui contient comme son nom l'indique le titre (« Formulaire d'inscription »). Enfin le 'cadre' contiendra le corps du formulaire. Nous allons commencer

par créer un champ ‘fieldset’ avec une ‘legend’, dans lequel nous allons insérer une table avec plusieurs champs à remplir. Tout d’abord, une liste déroulante pour le Titre avec 3 valeurs (Madame, Mademoiselle, Monsieur), puis des zones de texte pour le Nom, Prénom, Adresse, Adresse2, Ville, et Code Postal, des boutons de type radio pour l’Entreprise, et enfin un ‘textarea’ pour la Description. Nous créons finalement un bouton ‘Envoyer’ pour soumettre les valeurs.

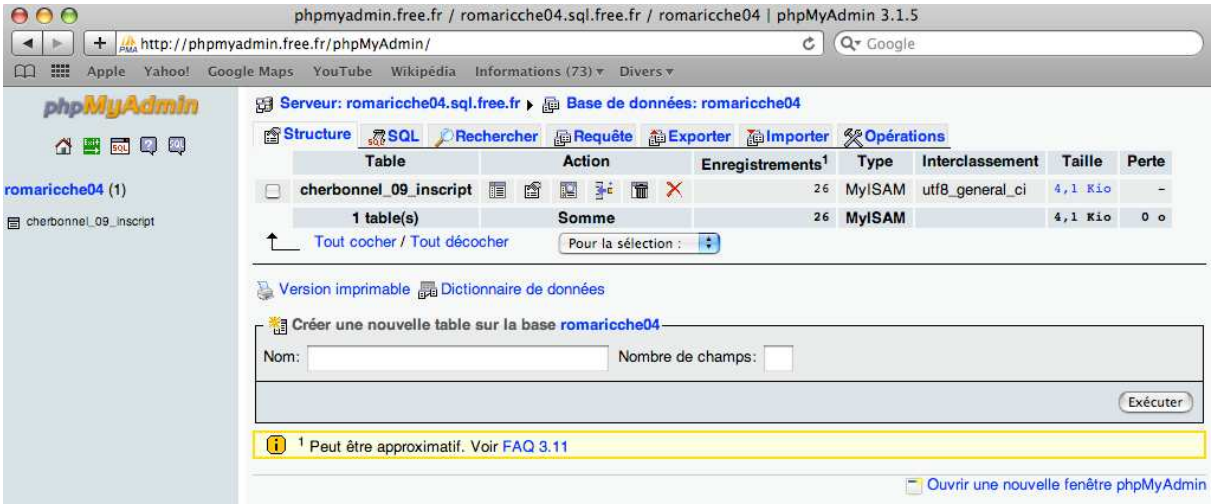
Il va maintenant falloir envoyer ces informations au serveur de base de données MySQL Free. Pour cela, nous exécutons la fonction PHP ‘require_once’ qui permet de récupérer les informations de connexion contenues dans le fichier ‘connexion.php’ qui, étant donnée le caractère confidentiel des informations contenues, est placé dans le répertoire ‘cherbonr\http\interdit\’. Ce fichier recense les paramètres de connexion à http://phpadmin.free.fr/, ainsi que le nom de la base de données et du codage des caractères.
De plus, grâce à ‘mysql_query’ on va insérer les valeurs récupérées avec la méthode POST, dans les champs de la base de données.

4 – Style.css

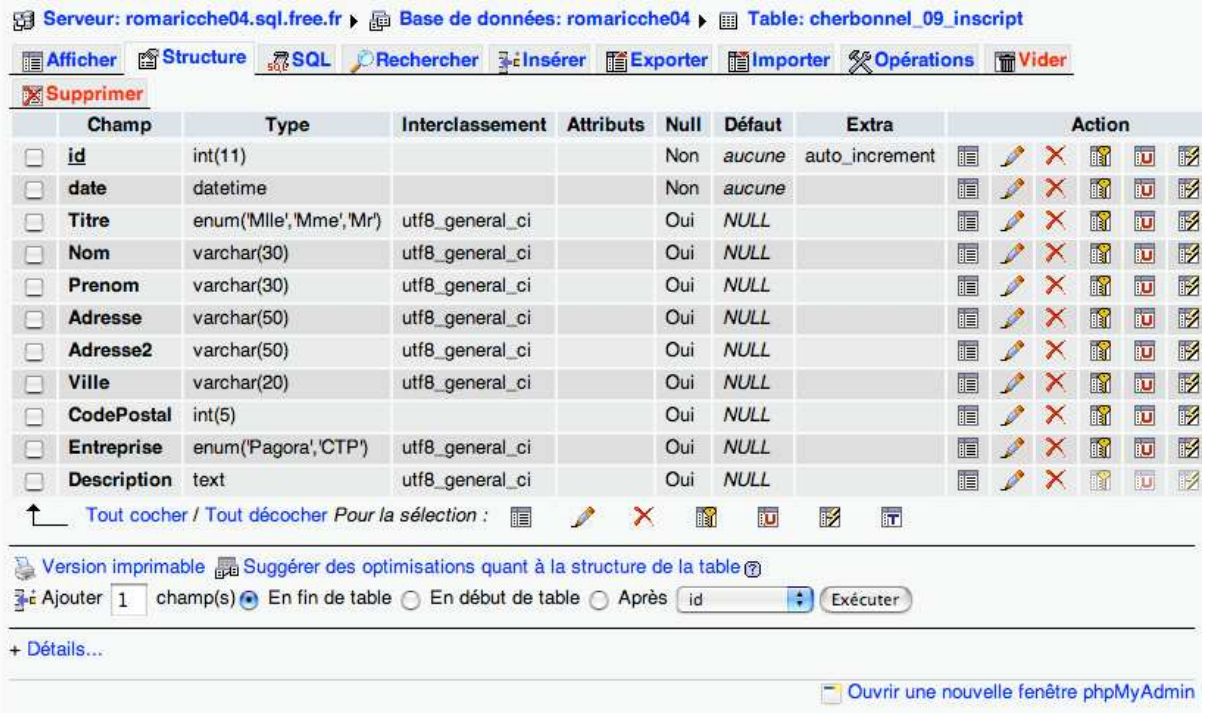
Afin de donner un design agréable à la page du formulaire, nous avons créé une feuille de style avec un style de police propre à chaque division, des images d’arrière plan, ainsi qu’une disposition adaptée.

5 – http://myphpadmin.free.fr/

Afin de récupérer, trier, et utiliser les données récoltées, il faut créer une table de base de données sur le site ‘http://myphpadmin.free.fr/.



Nous allons donc créer un champ par information recueillie en attribuant un type de donnée (Texte, Nombre,…) à chacun, ainsi que la taille en octets à réserver.



On pourra ensuite visualiser les informations recueillies auprès des différentes personnes inscrites dans le tableau suivant disponible sous l'onglet 'Affichage' :

6 – Conclusion

Ce TP fut une bonne occasion de découvrir quelques rudiments de base de programmation en PHP, ainsi que de se remémorer les notions de HTML vues en deuxième année. De plus, la gestion de bases de données est une compétence utilisable dans plusieurs domaines industriels et personnels.

CORROYER Elsa

Compte rendu des séances de TP de MySQL

But de l'ensemble des séances :

Au cours des différentes séances, nous avons travaillé sur le langage PHP et sur la création de bases de données. Pour cela, nous avons réaliser un formulaire par l'intermédiaire d'une page internet où les personnes pouvaient remplir les différents champs. Cependant, il a fallu ensuite s'occuper de récupérer et de stocker ses valeurs. C'est par l'intermédiaire d'une base de données Mysql que nous avons fait cela car elle permet de traiter et d'utiliser les informations.

Pour réaliser l'ensemble de ses actions, nous avons utilisé plusieurs outils :

- TextWrangler : éditeur de texte, il reconnaît les différents langages utilisés, met les balises en couleur ce qui permet d'avoir une vision claire du code.
- Le navigateur internet Firefox
- Le logiciel Fetch : qui permet de charger des fichiers
- Nous avons aussi créé des pages internet qui sont hébergées par Free, et c'est aussi free qui nous a permis de gérer notre base de données Mysql.

L'adresse de mon site internet est http://louphock.free.fr/2009/corroyee/carte/index.php

1. Création des dossiers pour une bonne gestion des fichiers

Au départ, nous avons tout d'abord créer le dossier corroyee dans lequel j'ai pu stocker tous les documents relatifs à mes pages web. Nous avons ensuite tout d'abord créer un premier document index.html, qui nous a permis de nous revoir les notions de bases concernant ce langage. En effet, ce fichier comporte les balises de structures classiques :

-<HEAD >, où nous avons inséré le titre de l'onglet

-<BODY >, où nous avons inséré un petit texte pour vérifier que notre fichier était correct.

Des balises HTML encadre le contenu du fichier.

2. Erreur .php

Nous avons créé une page d'erreur en cas de problème rencontré par le serveur (en cas par exemple de page non trouvée). Cette page est située dans corroyee\http, et nous allons détailler son contenu :

```
<HTML>
<HEAD>
<META HTTP-EQUIV="content-type" CONTENT="text/html; charset=windows-1252">
<TITLE>Elsa is not a geek</TITLE>
<link rel="shortcut icon" href="/favicon.ico?200809111205" />
</HEAD>
<BODY>
```

Cette première partie de code est celle d'une page HTML classique.

Nous allons maintenant regarder le contenu du body :

- un « fatal error » est inséré, il a pour but d'effrayer l'utilisateur
- ensuite nous utilisons le langage php. Celui doit être compris entre les balises suivantes < ? php, ?>, liste des fonctions php utilisées :
    - la fonction echo, qui fait apparitre dans ce cas « coucou c'est moi », qui doit se terminer par ; et aussi le message d'errueur
    - le retour à la ligne se fait par \n en php
    - la fonction array, qui permet de lister les erreurs possibles

```
$erreurs=array(
                '404' => "page non trouvée",
                '401' => "page non autorisée",
                '402' => "reformuler votre paiement",
                '403' => "accès interdit",
                '500' => "erreur interne du serveur",
                );
```

    - On utilise aussi içi le $, servant pour les variables en php.
    - On utilise aussi la fonction echo, avec une boucle foreach, pour permettre de faire afficher toutes les erreurs listées dans le array.
    - Pour avoir une vision claire et structurée des informations retournées par le serveur en cas de problèmes, nous allons créer une table.

```
<table>
<?php
        foreach ($_SERVER as $k => $v) {
        echo "<tr><td><tt>$k</tt></td><td><i>$v</i></td/</tr>\n";
        }
?>
</table>
```

3. Le formulaire en ligne

La base de ce formulaire en ligne est le fichier index.php. Ce fichier est situé dans le dossier corroyee\carte.

Voici les différents éléments importants de ce fichier :

<HEAD>
<META>, on définie la racine du site : <meta base="http://louphock.free.fr/2009/monsite"/>
<TITLE>, on donne un titre à notre page web
<LINK>, on peut faire un lien avec une feuille de style css pour mettre en page le formulaire

<BODY>
<h2> permet de donner un titre au formulaire
<fieldset> permet de séparer le formulaire en différents champs
<legend> permet de nommer le cadre du formulaire
<table> on utilise une table pour pouvoir inscrire les valeurs

Les différents éléments de la table ne seront pas tous à renseignés de la même manière. En effet, pour le titre on a une liste déroulante (3 valeurs : madame, mademoiselle, monsieur), de simple zone de texte pour Nom , Prénom, Adresse, Adresse 2, Ville, code postal). Pour le nom de l'entreprise, on a créé un bouton type radio. Un cadre pour des commentaires libres est aussi ajouté pour une description. Enfin, un bouton envoyé permet de soumettre les valeurs.
Pour renseigner les variables, on peut dans le php utiliser get ou post. Nous avons choisi d'utiliser post car cela permet de ne pas avoir les variables dans l'url lorsque l'on envoie les données.
On utilise aussi <? foreach ($_POST as $k=>$v) echo "
<tt>$k</tt> $v\n"; ?> pour afficher lorsque l'envoi est effectué les informations renseignées.
La fonction pre peut aussi être utilisée par les programmeurs car elle permet d'afficher toutes les données, de lister le contenu de ce qui est envoyé par le formulaire.

Lorsque les informations ont été renseignées, il faut les envoyer au serveur de base de données MySql de Free. Ce ci est effectué grâce à la fonction php require_once qui permet de récupérer les informations contenues dans le fichier 'connexion.php'. ces informations étant confidentielles, elles sont placées dans le répertoire interdit du site 'corroyee\http\interdit'. Ce fichier contient :
mysql_connect('localhost','louphock','clochett'); ce sont les paramètres pour se connecter à http://phpmyadmin.free.fr/phpMyAdmin/
mysql_select_db('corroyee_inscription'); le nom de la base de données où il faut renseignées les informations
mysql_query("set names utf8"); permet d'envoyer une requête au serveur MySql, c'est-à-dire lui signale qu 'il doit inscrire les informations dans corroyee_inscription. On signale aussi l'encodage utilisé.

4. – http://myphpadmin.free.fr/

Ce site nous permet de créer une table de base de données pour pouvoir utiliser les informations collectées via le site de formulaire en ligne.

Nous avons donc créer une base de données, celle-ci comporte tous les champs qui doivent être inscrit dans le formulaire, ainsi qu'il champ pour numéroté chaque envoi. Il faut pour chaque champ préciser le type de donnée (date, nombre texte), ainsi que la taille d'octets à réserver pour chaque valeur.

Voici le code source de la table créer par phpmyadmin

```
CREATE TABLE IF NOT EXISTS `corroyee_inscription` (
  `id` int(11) NOT NULL auto_increment,
  `date` datetime NOT NULL,
  `titre` enum('Mme','Mlle','M') default NULL,
  `nom` varchar(30) default NULL,
  `prenom` varchar(20) default NULL,
  `adresse` text,
  `adresse2` text,
  `ville` text,
  `codepostal` int(10) default NULL,
  `entreprise` enum('Pagora','CTP') default NULL,
```

```
  `description` text,
  PRIMARY KEY  (`id`),
  KEY `id` (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;
```

Voici la structure

Afficher | Structure | SQL | Rechercher | Insérer | Exporter | Importer | Opérations | Vider | Supprimer

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
☐	id	int(11)			Non	*aucune*	auto_increment	
☐	date	datetime			Non	*aucune*		
☐	titre	enum('Mme','Mlle','M')	utf8_general_ci		Oui	*NULL*		
☐	nom	varchar(30)	utf8_general_ci		Oui	*NULL*		
☐	prenom	varchar(20)	utf8_general_ci		Oui	*NULL*		
☐	adresse	text	utf8_general_ci		Oui	*NULL*		
☐	adresse2	text	utf8_general_ci		Oui	*NULL*		
☐	ville	text	utf8_general_ci		Oui	*NULL*		
☐	codepostal	int(10)			Oui	*NULL*		
☐	entreprise	enum('Pagora','CTP')	utf8_general_ci		Oui	*NULL*		
☐	description	text	utf8_general_ci		Oui	*NULL*		

On peut ensuite visualiser l'ensemble des informations recueillies dans l'onglet affichage :

+ Options

	id	date	titre	nom	prenom	adresse	adresse2	ville	codepostal	entreprise	description
☐	1	2009-11-13 09:14:24	M	MARECHAL	Vincent	le rabot	*NULL*	Grenoble	38000	Pagora	vive a chicha
☐	2	2009-11-13 09:17:57	Mlle	CORROYER	Elsa	56 avenue jeanne d'ARC	*NULL*	Grenoble	38100	Pagora	stage Tetra Pak
☐	6	2009-11-13 09:23:10	Mlle	PIERRAT	Coralie	16 rue Cuvier	*NULL*	Grenoble	38000	Pagora	la légende des bisounours !!!
☐	7	2009-11-13 09:26:15	M	TONDELIER	Alan	3 rue Paul Janet	*NULL*	Grenoble	38000	CTP	je suis un geek
☐	8	2009-11-13 09:32:36	Mme	CARCONE	Fiona	rue du tram	*NULL*	saint martin d'heres	38000	Pagora	*NULL*
☐	9	2009-11-13 09:14:24	M	MARECHAL	Vincent	le rabot	*NULL*	Grenoble	38000	Pagora	vive a chicha
☐	10	2009-11-13 09:14:24	M		Vincent	le rabot	*NULL*	Grenoble	38000	Pagora	vive a chicha
☐	11	2009-11-13 09:14:24	M		Vincent	le rabot	*NULL*	Grenoble	38000	Pagora	vive a chicha
☐	12	2009-11-13 09:14:24	M		Vincent	le rabot	*NULL*	Grenoble	38000	Pagora	vive a chicha
☐	14	2009-11-13 09:14:24	M	carlos	renÃ©	rue de la papeterie	rue de l'imprimerie	saint martin d'hÃ¨res	38420	Pagora	
☐	17	2009-11-13 09:14:24							0		
☐	18	2009-11-13 09:14:24							0		

Nous avons aussi modifié notre base de données grâce aux fonctions suivantes :

- INSERT INTO
- DELETE FROM
- UPDATE

Nous n'avons pas exploité notre base de données, cependant nous avons appris que les requêtes s'effectuaient grâce à 4 opérations :

- union
- intersection
- différence
- produit cartésien

Conclusion :

Ce TP nous a permis de revoir les bases concernant le langage html vu en deuxième année. Nous avons aussi découvert la programmation en php, la gestion des fichiers par internet et le fonctionnement d'une base de données.

MySQL

Introduction

MySQL est un serveur et système de gestion de base de données. On peut accéder à ces bases de données à l'aide de plusieurs langages de programmation. Le langage utilisé au cours des travaux pratiques est PHP.
L'objectif de ce TP est de créer un formulaire, sous la forme d'une carte de visite, dans lequel les 'clients' (professionnels de Pagora et du CTP) pourront entrer des informations personnelles. Ces informations seront ensuite répertoriées dans une table de données à partir du formulaire. Le formulaire se présentera de la façon suivante :

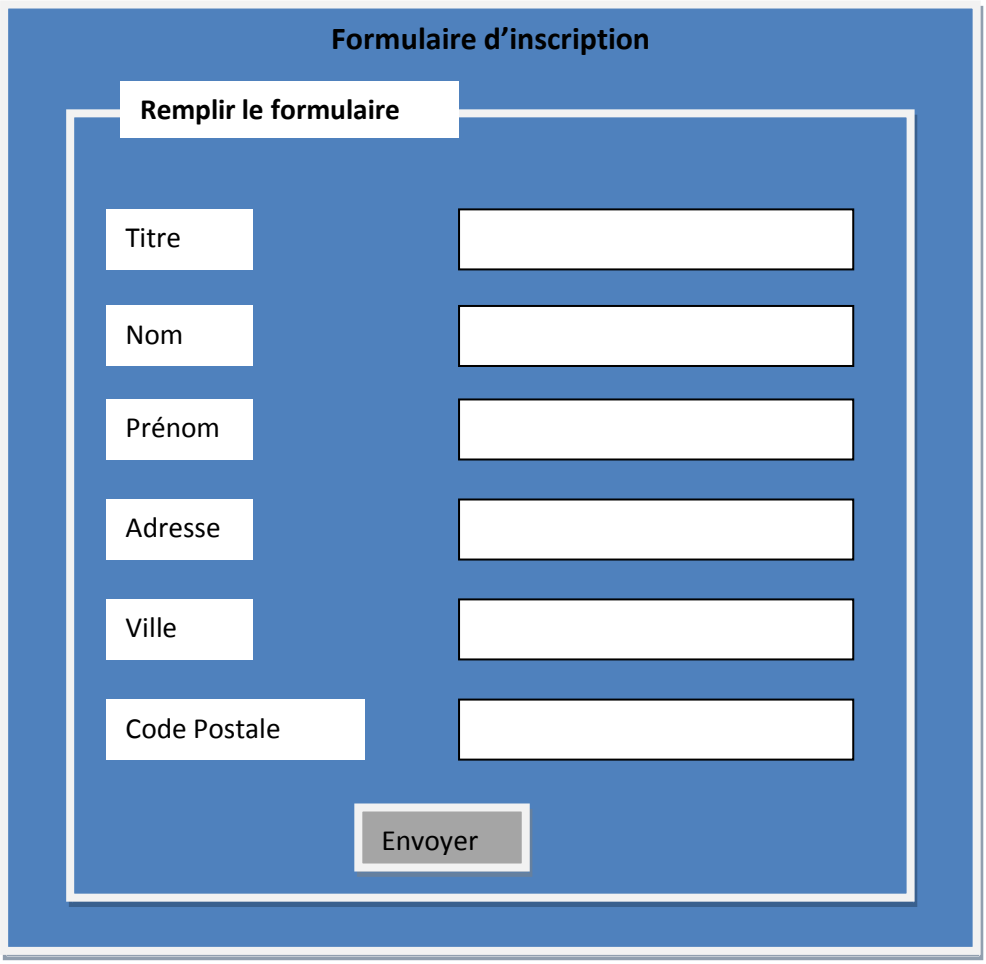


Schéma 1

Le formulaire

Le compte sur free est *licpro.pagora* et le mot de passe est pagora. L'adresse pour accéder au formulaire est la suivante : http://licpro.pagora.free.fr/2009/dasilvaa/formulaire/formulaire.php . Dans le code du formulaire, nous pouvons voir deux sortes de balises. Les balises <xxxxx> indiquent que le langage employé est du html, et les balises < ?xxx ?> que l'on utilise du langage php.
Le formulaire commence en html. On retrouvera alors les balises suivantes dans l'ordre énoncé ci-dessous :

```
<HTML>
<HEAD>
        <title> Un formulaire </title>
</HEAD>
<BODY>
</BODY>
</HTML>
```

En HTML, les formulaires sont délimités par une balise FORM, localisée dans le BODY. Grâce à cette balise, on peut faire apparaître, sur notre page web, des boutons, une liste à choix multiples ainsi que des champs et zones de saisies. Cette balise contient obligatoirement les attributs suivants : METHOD et ACTION. L'attribut METHOD permet de choisir la façon selon laquelle les réponses seront transmises : POST ou GET. C'est deux éléments seront définit ultérieurement. L'attribut ACTION, quant à lui, indique l'adresse (l'url) qui va recevoir les réponses. On peut voir apparaître ci-dessous l'attribut NAME afin de donner un nom au formulaire.

```
<form action = "<? echo $_SERVER['PHP_SELF']?>" method = "POST" name = "inscription">
```

La balise H2 définit la taille du titre 'Formulaire d'inscription'.
La balise FIELDSET sert à regrouper plusieurs éléments dans un cadre. Il permet de faciliter la lecture et l'utilisation du formulaire, notamment si plusieurs formulaires se succèdent. La balise LEGEND contenu dans le FIELDSET sert à donner un titre à l'encadré, soit au regroupement des différents éléments.

Dans cet encadré, nous avons choisit de faire apparaître les éléments suivants : Titre, Nom, Prénom, Adresse, Adresse_2, Ville, Code Postal, Entreprise, Poste, Description du poste et un bouton 'Envoyer'.
Le Titre correspond au choix de M. Mme ou Melle. Nous avons ainsi créé un menu déroulant. Pour ce faire, on utilise la balise SELECT (à l'intérieur de la balise FORM).

```
<select name="Titre">
   <option value :"Mme" <? if($_POST['Titre']=='Mme') echo '"selected"' ?> > Madame </option>
    <option value :"Mlle" <? if($_POST['Titre']=='Mlle') echo '"selected"' ?> > Mademoiselle </option>
   <option value :"M." <? if($_POST['Titre']=='M.') echo '"selected"' ?> > Monsieur </option>
</select>
```

Dans cette balise SELECT, la balise OPTION permet d'éditer les différents éléments du menu déroulant.

Image 1

Ainsi, si un des clients utilise le formulaire et choisit dans la liste déroulante « Madame », la valeur sélectionnée par défaut est Mme.
Pour les éléments Nom, Prénom, Adresse, Adresse_2, Ville, Code Postal, et Poste, on a créé une ligne de saisie grâce à la balise INPUT.

```
<input type="text" name ="Nom" value="<?echo$_POST['Nom']?>"/>
```

Image 2

L'attribut TYPE permet de déterminer le type du champ. Dans notre cas, on a choisit 'text'. Ce qui permet de rentrer du texte dans la zone de saisie. L'attribut NAME permet de nommer le champ.
Pour l'élément Entreprise, nous avons choisit de créer un bouton proposant plusieurs choix dont un seul uniquement peut être coché. Pour cela nous avons utilisé également la balise INPUT, et choisit le TYPE 'radio' dans notre balise INPUT.

```
<input type="radio" name ="Entreprise" value="Pagora" <? If ($_POST['Entreprise']=='Pagora') echo
'"checked"' ?> /> Pagora
 <input type="radio" name ="Entreprise" value="CTP" <? If ($_POST['Entreprise']=='CTP') echo
'"checked"' ?> /> CTP
```

Image 3

Pour l'élément Description, la balise TEXTAREA a été utilisée dans le but de créer une zone de saisie contenant le nombre de lignes et le nombre de caractère par lignes.

```
<textarea name="Description" col="80" row="4"><?=$_POST['Description']?></textarea>
```

Image 4

Pour la création du bouton envoyer, on utilise la balise INPUT avec le TYPE 'submit'.

```
<input type="submit"  value="envoyer"  name="Faire"/>
```

Image 5

Pour rendre le formulaire plus attrayant, nous lui avons attribué une feuille de style css (dans le HEAD), et présenter sous forme d'un tableau à l'aide des balise TABLE, TD, et TR dans le body. En ce qui concerne la mise en page et la photo ajoutée, vous trouverez des commentaires dans le code.

Les méthodes GET & POST

Les méthodes GET et POST permettent de transférer des données d'une page à une autre. Chacune possède des avantages et des inconvénients.

Dans un premier temps, nous avons utilisé la méthode GET. Cette méthode, simple d'utilisation, pose des problèmes de sécurité et de longueur de chaînes de caractères (limitées). En effet, les informations sont inscrites dans la barre de navigation, ce qui induit que n'importe qui peut les visualiser.

Suites à ses constations, nous avons préféré utiliser la méthode POST. Cette dernière permet d'envoyer une quantité d'informations plus importante et n'affiche pas les informations à la suite de l'url contrairement à la méthode GET.

Lorsque nous avons remplacé GET par POST dans la ligne suivante du code

```
<form action = "<? echo $_SERVER['PHP_SELF']?>" method = "POST" name = "inscription">
```

un message d'erreur est apparu, et on a constaté que les INPUT ne fonctionnaient plus. On a alors remplacé tous les GET par POST.

La table de données

À l'aide du site http://sql.free.fr , nous avons créé une table de données contenant les champs souhaités.

Identifiant : licpro.pagora
Mot de passe : pagora

Lors de la création de la table, on nomme notre table et on indique le nombre de champs souhaités. Pour ma part, le nombre de champs est de 11. Nous nommons ensuite les différents champs comme on l'a fait précédemment dans le formulaire. On obtient alors ceci :

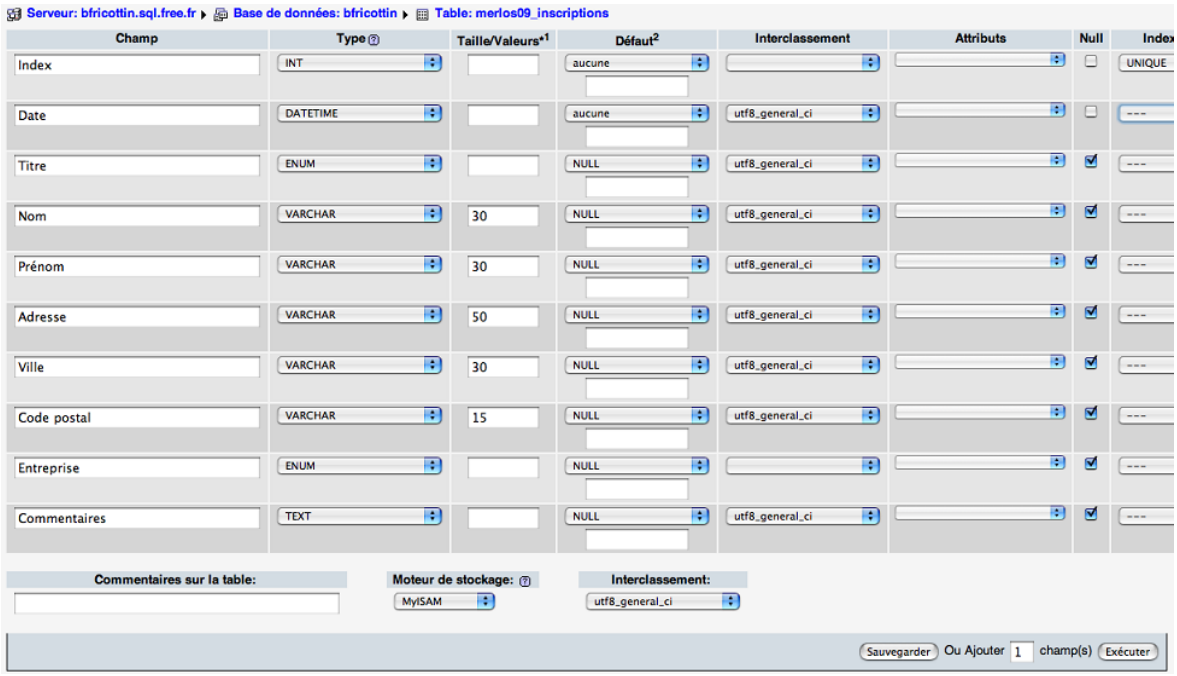


Image 6

Dans l'image ci-dessus, plusieurs colonnes apparaissent. La première indique les différents champs. La seconde donne une indication sur la nature du champ. En effet pour le champ 'index' qui permet d'incrémenter (choisir auto_incrementation) et de savoir le nombre de personnes ayant rempli le formulaire, on choisit le type 'integer' (entier dont les limites sont très larges). Pour le champ 'Titre', on préfère le type 'Enum' car on a créé une liste déroulante lors de la conception du formulaire. Dans le cadre de notre formulaire, la majorité des informations sont courtes et variables, les chaînes de caractères sont courtes. On choisit donc le type 'varchar' pour des chaînes variables avec un maximum de 255 caractères. Pour des informations plus condensées, on utilise le type 'text'.

Une fois la table créée, on peut remarquer en haut de la page divers onglets :

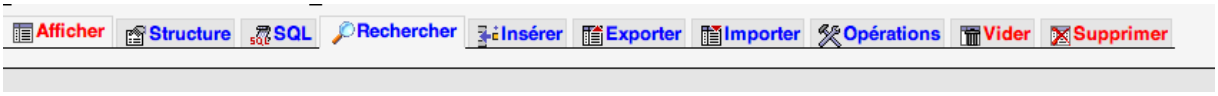


Image 7

Lors du TP, nous avons essayé plusieurs manipulations, comme l'insertion d'une personne, la suppression d'une autre afin de prendre en main ce nouvel outil. Vous pourrez découvrir dans le fichier 2009/dasilvaa/table.sql, les différentes manipulation effectuées ainsi que le code associé

Le lien entre le formulaire et la table de données

Pour entrer les données dans la table, on utilise la fonction REQUIRE du langage php. Cette dernière permet d'appeler le fichier de connection.
Ce fichier de connection se situe dans *2009/dasilvaa/http/interdit/conection.php*, il permet de se connecter à notre base de données. Dans ce dernier, nous avons indiqué à l'aide de *mysql_connect* l'hôte et le mot de passe, à l'aide *mysql_select_db* la base de données sélectionnée, *mysql_query envoie* une requête au serveur. On revient ensuite dans le codu formulaire. La fonction REQUIRE apparaît de nouveau et est suivie de ONCE. Ces lignes de code servent pour appeler le fichier de connection afin de se connecter à la base de données (indique où se trouve cette base), et informe sur le fait qu'il est nécessaire de se connecter qu'une seule fois.

```
<? require_once('../http/interdit/connection.php');
mysql_query("INSERT INTO `licpro_pagora`.`dasilvaa_09_formulaire` (`index`, `date`, `Titre`, `Nom`, `Prenom`, `Adresse`, `Adresse_2`, `Ville`, `CodePostale`, `Entreprise`, `Poste`, `Description`) VALUES (NULL, NOW(), 'Mlle', '".$_POST['Nom']."', '".$_POST['Prénom']."', '".$_POST['Adresse']."', '".$_POST['Adresse_2']."', '".$_POST['Ville']."', '".$_POST['CodePostale']."', '".$_POST['Entreprise']."', '".$_POST['Poste']."', '".$_POST['Description']."');" ); ?>
```

Conclusion

Ce TP permet de manipuler à nouveau le langage html, de se familiariser avec le langage PHP. Il introduit la notion de documents dynamiques et d'interaction entre différents éléments. Ce TP donne les bases nécessaires pour approfondir ces notions dans un cadre ultérieur.

JEAN Quentin
3EI
2009-2010

Rapport TD MySQL

Introduction :

MySQL est un système de gestion de base de données. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels. Ce cours va donc nous permettre de nous familiariser avec cet outil, l'objectif du TP étant la réalisation d'un formulaire sous forme de page Internet. Ce formulaire permettra de récupérer des données dans une base de données MySQL et de les traiter par la suite. Nous avons utilisé plusieurs logiciels dans le cadre de ce cours : L'éditeur HTLM « TextWrangler », le navigateur Internet « Mozilla Firefox », le logiciel de transfert FTP « Fetch ». De plus nous avons utilisé un compte Free afin d'héberger notre site, dont l'adresse est la suivante : http://louphock.free.fr/2009/jeanq/autoindex

Index.html :

Pour commencons nous créons un dossier qui correspond au dossier racine du site : à l'intérieur de ce dossier nous placerons tous les dossiers et fichiers constituants notre site Internet. Nous créons ensuite le fichier « index.php », qui correspond à la page affiché par défaut lors d'une connexion à l'adresse http://louphock.free.fr/2009/jeanq

La structure de ce fichier est constitué des balises classique de l'HTLM, c'est-à-dire <HEAD> , <BODY>, <TITTLE> et d'une balise <META> indiquant la nature du fichier.

Autoindex, .htaccess, et .htpasswd :

Après avoir réalisé les premières étapes, nous avons créé le dossier « autoindex » dans la racine du site. Dans ce dossier nous avons créé deux fichier TXT : le fichier HEADER.txt et le fichier README.txt. Le fichier HEADER.txt a pour but l'affichage d'un texte en entête, alors que le REAME.txt permet l'affichage d'un commentaire sous la liste des dossiers.

Dans le même dossier autoindex, nous créons le fichier .htacces : ce fichier permet de protéger l'accès de ce dossier. On y retrouve les instructions suivantes :

```
PerlSetVar AuthFile /2009/jeanq/http/interdit/.htpassword
AuthName "Acces privé"
AuthType Basic
<limit GET POST>
require user Quentin
```

```
#require valid-user
</limit>
```

La première ligne renvoi à la liste des utilisateurs autorisés et leur mot de passe associé. Les lignes suivantes correspondent au type de sécurité utilisé pour cette page.
La liste d'autorisation d'accès et les mots de passe se trouvent dans le fichier .htpassword. On trouve ce fichier dans le dossier jeanq\http\interdit. On y retrouve :

Quentin:Quentin

Cela indique que seulement l'utilisateur Quentin est autorisé pour l'accès et que son mot de passe est Quentin. En effet les données dans ce fichier sont rentré de la manière suivante :
nom_utilisateur:mot_de_passe

Dans l'optique de protéger ce fichier, nous plaçons dans ce répertoire le fichier .htacces :
deny from all

Erreur.php :

En cas de problème (erreur d'adresse, page introuvable …) nous avons créé une page d'erreur (erreur.php). Cette page se trouve dans le dossier jeanq\http\.

Cette page comporte des balises classiques d'une page html, c'est-à-dire un <HEAD> et un <BODY>. Le passage en php est signalé par la commande « < ? »php (le php étant facultatif). Le fin d'un passage php est signalé par « ?> ».

La première fonction php utilisée et la fonction echo, qui nous permet d'afficher le texte « Coucou c'est moi » puis de « Quentin » (qui est rentré dans la variable $nom). Le \n permet un retour à la ligne. On termine chaque ligne de code par un « ; ».

```
<?php
echo "\r".'<P>Coucou'." c'est moi";
$nom=' Quentin';
echo "$nom </P>\n";
```

On réalise ensuite la liste des erreurs avec leurs numéros attribués à l'aide de la fonction array.

```
$erreurs=array(
        '404'=>"Page non trouvÃ©e",
        '403'=>"Demande d'identification...",
        '401'=>"Page non autorisÃ©e",
        '500'=>"Erreur Serveur",
        );
```

A l'aide de la fonction echo on appel ensuite l'erreur 401 : echo $erreurs['401'];

On affiche ensuite le message d'erreur correspondant à cette erreur :

```
echo "<P><tt>".$_SERVER['REDIRECT_STATUS']." :</tt> le message d'erreur est: <i>
".$erreurs[$_SERVER['REDIRECT_STATUS']]." </i></P>\n";
```

A l'aide de la fonction boucle foreach et de la fonction echo, nous affichos toutes les erreurs de la liste d'erreur :

```
foreach($erreurs as $k => $v) {
echo "<P>le message d'erreur pour l'erreur numÃ©ro <tt>$k</tt> est <i>$v</i></P>\n";
}
```

Nous mettons toutes ces erreurs dans une table :

```
<table>
<?php
foreach($_SERVER as $k => $v){
echo "<tr><td><tt>$k</tt></td><td><i>$v</i></td></tr>\n";
$prenom='Quentin';
$Quentin='Jean';
echo"${$prenom}";
?>
</table>
```

Form.php :

Nous avons créé un formulaire « form.php » dans le dossier jeanq/formulaire/

Dans ce fichier php nous créons une balise <HEAD> , une balise <BODY>, une balise <META>, une balise <TITLLE> ainsi qu'une balise <link> permettant d'associer une feuille de style.css et de localiser son emplacement.

Nous créons un champ 'fieldset' avec une 'legend', dans lequel nous insérons une table avec plusieurs champs à remplir correspondant aux champs de notre formulaire :

- une liste déroulante pour le Titre avec 3 valeurs (Madame, Mademoiselle, Monsieur),
- des zones de texte pour le Nom, Prénom, Adresse, Adresse2, Ville, et Code Postal,
- des boutons de type radio pour l'Entreprise
- un « textarea » pour la Description.
- Un bouton « Envoyer » pour envoyer les valeurs

Pour envoyer les informations d'un formulaire rempli au serveur de base de données MySQL Free nous utilisons la fonction php « require_once » qui permet de récupérer les informations de connexion contenues dans le fichier « connexion.php » :

<? require_once('../http/interdit/connexion.php');

Ce fichier recense les paramètres de connexion à http://louphock.free.fr/ et le nom de la base de données.

Nous utilisons ensuite la fonction « mysql_query » pour insérer les valeurs récupérées avec la méthode POST, dans les champs de la base de données.

mysql_query ("INSERT INTO `louphock`.`jean_09_inscript` (`id` ,`date` ,`titre` ,`Nom` ,`Prenom` ,`Adresse` ,`Adresse 2` ,`Ville` ,`CodePostal` ,`Entreprise` ,`Description`)

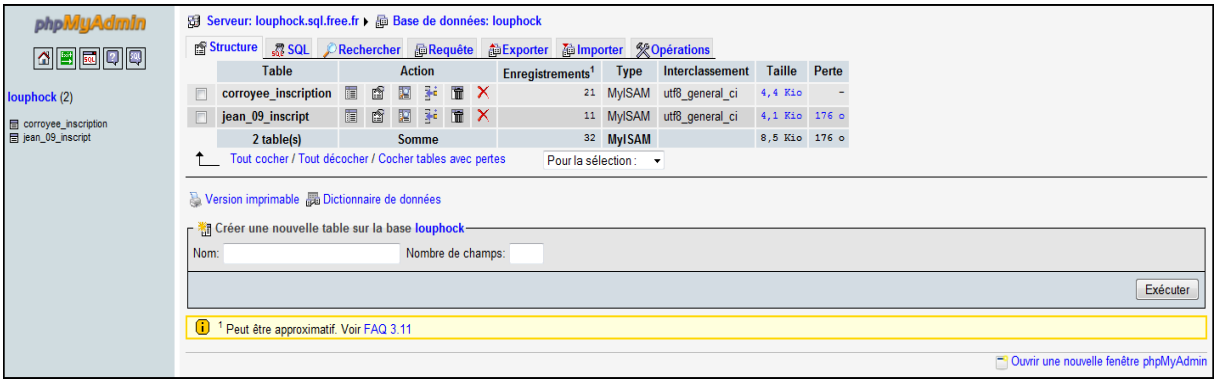
VALUES (NULL , NOW(), '".$_POST['Titre']."', '".$_POST['Nom']."', '".$_POST['PrÃ©nom']."', '".$_POST['Adresse']."', '".$_POST['Adresse 2']."', '".$_POST['Ville']."', '".$_POST['CodePostal']."', '".$_POST['Entreprise']."', '".$_POST['Description']."');");

Pour l'interface du site nous avons crée un fichier css que nous avons placé dans la racine du site :

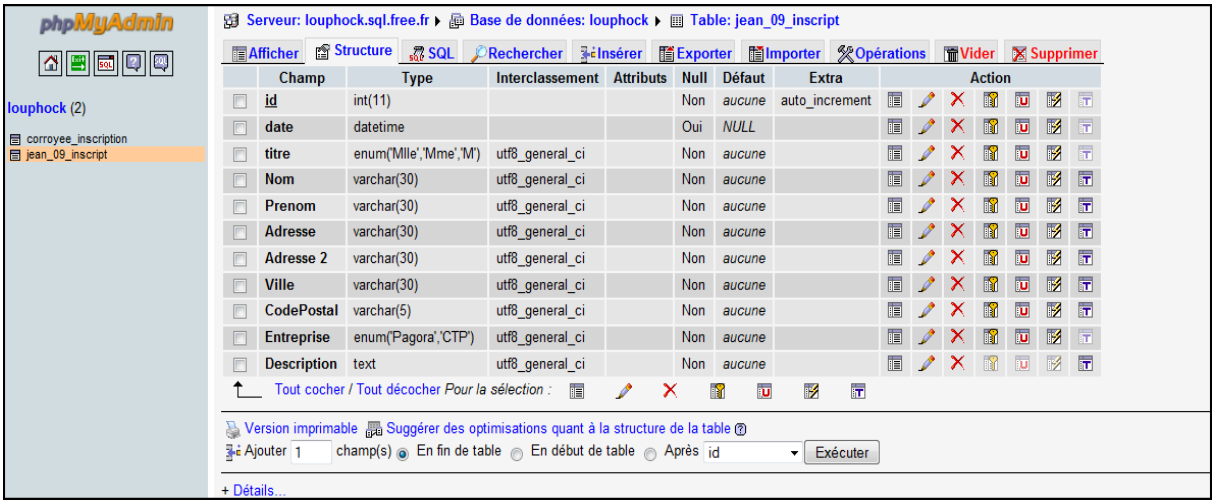
```
body { background-color:red; }
h1 { background-color:#ddeeff; border:solid 1px #888844; }
table { border-collapse:collapse; }
td, th { padding:4; border:solid 1px #888844; }
.err { color:red; }
td:first-child, tt { background-color:#ffbbbb; }
td:last-child, i { background-color:#ffccaa; }
```

phpmyphpadmin :

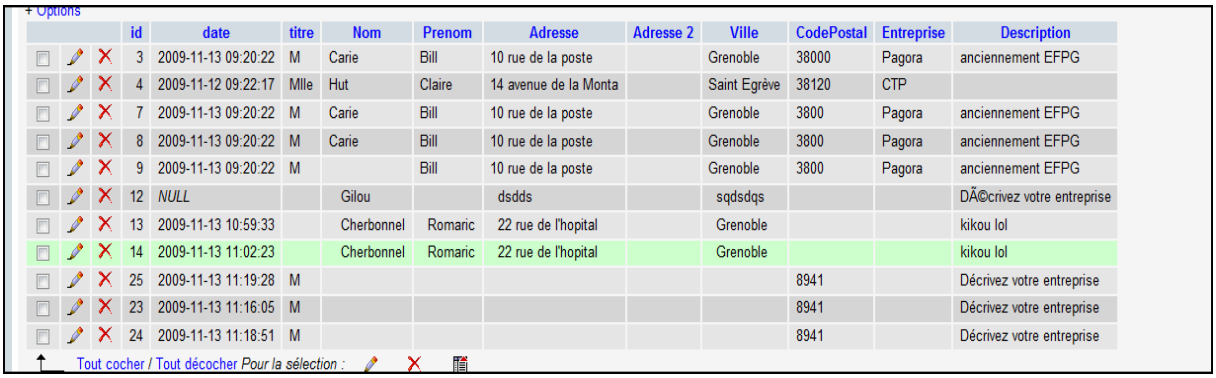
Nous utilisons le site phpmyadmin dans le but de créer la table de base de données qui nous permettra par la suite de récolter et d'utiliser les données provenant de formulaires envoyés.



L'étape suivante est la création de la structure de notre table, avec l'ensemble des informations à remplir, le type de donnée ainsi que leurs tailles en octets.



Ce site permet bien entendu de visualiser l'ensemble des informations envoyées après remplissage du formulaire.



Conclusion :

En plus de nous faire découvrir les premières notions de la programmation php, ce cours a permis de revoir les notions HTML. L'utilisation de nombreux logiciels et la création concrète de pages web fut très formateur.

Rapport SQL

Vincent Maréchal 3EI3 - 2009

Objectifs

Les objectifs de ce cours ont été de découvrir les outils de web et la mise en ligne de fichiers sur un server ftp. De revoir les bases en html CSS et de les appronfondir avec le dynamisme et la puissance du php associé aux bases de données SQL.

I. Informations sur le compte

Le site se trouve à cette adresse:
http://mosieur.free.fr/2009/marechav/
Le formulaire se trouve à cette adresse :
http://mosieur.free.fr/2009/marechav/formulaire/form.php
La page erreur.php se trouve à cette adresse :
http://mosieur.free.fr/2009/marechav/erreur.php

Caractéristiques du compte free.
Login : mosieur
mdp : clochett
Nom de la bdd : mosieur
Nom de la table : marechav_bdd

Accès au dossiers sécurisé
login : vincent
mdp : test

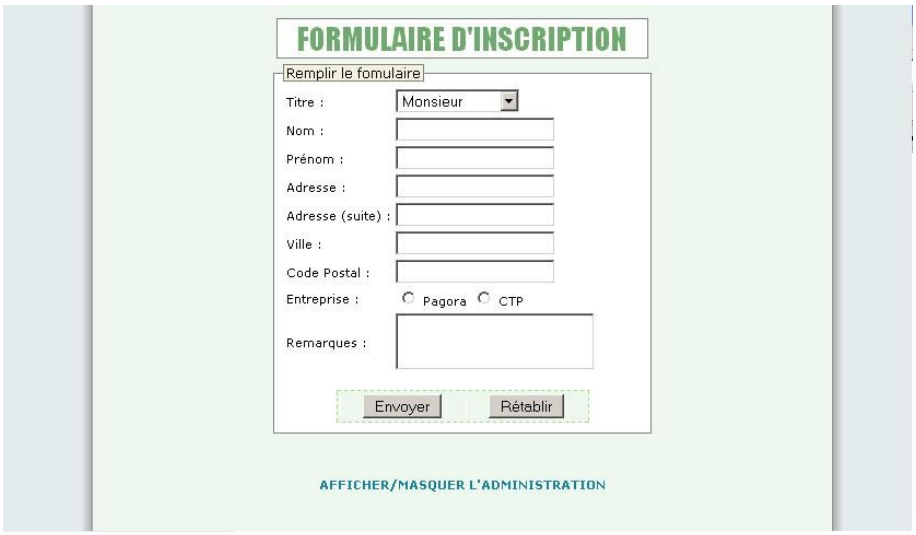
Voici l'aborescence du site :
/2009/marechav/

- **formulaire/**
  - *form.php*
- **http/**
  - **interdit/**
    - *connexion.php*
    - *.htpassword*
- **images/**
  - *bckgform.gif*
- *erreur.php*
- *index.htm*
- *style.css*
- *.htaccess*

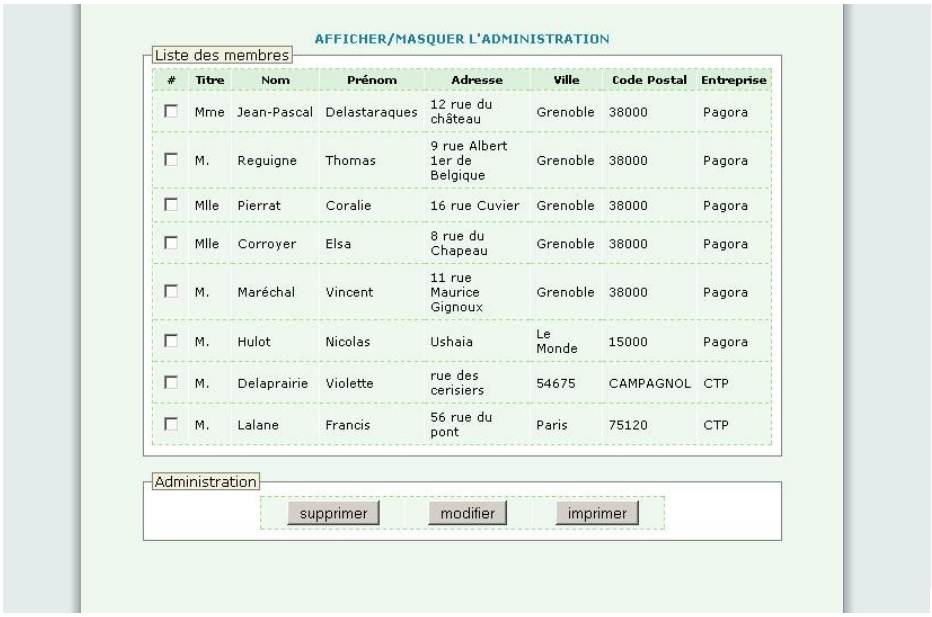
II. Création du formulaire

a. Architecture

La création du formulaire se fait sur une base en html mise en page CSS. Le principe est d'organiser le formulaire en 2 parties. Une première partie dite de saisie où l'utilisateur pourra entrer les donner et les envoyer à la base de données.



Une seconde qui affiche le contenu de la base dans le but de pouvoir effectuer quelques opération de modération (suppression/modification d'entrée et impression d'une liste selective d'entrée).*



*Notez que ce point a été aboutie dans l'architecture mais pas totalement dans le code. Même si l'affichage est effectif ces fonctions de sont pas actives (manque de temps pour les appronfondir en démarche personnel, mais le projet m'intéresse).

b. Les balises <form>
Il existe plusieurs methode pour générer une requête via le formulaire. On a le type POST et le type GET. POST e,vpo des dp,,ées stpclées da,s me cprŝ de ma requête, tandis que GET correspond à des données codées dans l'URL et séparées de l'adresse du script avec des ?.

Le formulaire utilisé est définit ici de type POST :
<form action="<? echo $_SERVER['PHP_SELF']?>" method="POST" name="incription">
(voir plus bas pour la signification du code php)

Voici un aperçu sommaire des différentes possibilités données par les balises forme dans la création d'un fomulaire.

Un formulaire est une interfance liant zone de saisie et boutons d'envoie, pour que les actions entre ces deux types d'éléments soient liées il faut que le tout soit contenu entre les balsies <form></form>.

> Champ de type input
Voici le code type :
<input name="XXX" value="YYY" type="ZZZ">

Chaque élément de saisie comporte 3 paramètres. On lui affilie un nom (pour pouvoir le rappeler par la suite par exemple), une valeur (on s'en servira pour conserver la saisie après validation), et un type.
Il existe plusieurs types d'input :
- text (champ de saisie classique)
- radio (bouton de selection)
- checkbox (cases à cocher)
- submit (bouton de validation)
- password (permet de créer un champ visuellement crypté lors de la saisie)
- reset (créer un bouton qui rétablie le formulaire)

> Champ de type button
Equivant à un Input de type submit sauf qu'ici il est possible de transformer n'importe quel élement en "bouton de validation" (texte, image etc...)

Voici le code type:
<button name="XXX" type="Submit"> <img src="images/image.gif alt"bouton de validation"> </button>

> Liste déroulantes
Les éléments de la listes sont contenus dans des balises <option> qui sont eux même tous contenu entre les balises <select></select>. On peut rajouter l'attribut *selected*="*selected*" pour qu'une des soit affichée en premier.

Voici le code type:
<select name="XXX">
<option selected="selected" value="YY1">Valeur1</option>
<option value="YY2">Valeur2</option>
<option value="YY3">Valeur3</option>
</select>

> Zone de saisie sur plusieurs lignes
Si l'on veut pouvoir spécifier la taille du champ de saisie à travers les paramètres rows et cols. Utile pour les champs avec beacoup de texte.

Voici le code type :

<textarea name="XXX" rows="4" cols="40" > Voici le texte affiché </textarea>

c. CSS
Le formulaire tient sa mise en page d'un fichier CSS unique situé à la racine du site. J'ai essayé de mettre a profit les connaissances en CSS apprises lors du précédent cours de DNS. Je vous invite à ouvrir le fichier style.css si vous souhaitez en savoir plus sur ce point.

III. Gestion et sécurité

d. Mot de passe
Pour assurer la connexion avec la base de donnée, il sera présent sur le server un fichier de connexion contenant les codes de connexion à la base. Pour protéger ces informations il est nécessaire de restreindre les accès et de les protéger par un mode de passe. Pour ce faire, un listing d'utilisateur et leur mot de passe est stocké dans le fichier ".htpassword" sous la forme d'une liste :

user1:pass1
user2:pass2
user3:pass3
etc..

Ensuite, il suffit de placer un .htaccess à la racine du dossier que l'on désire protéger et d'y inserer le code suivant (qui fait bien sûr le lien avec le .htpassword):

```
PerlSetVar AuthFile /2009/marechav/http/interdit/.htpassword
AuthName "Acces restreint dans cette zone : montrez votre patte blanche..."
AuthType Basic
<limit GET POST>
                require user vincent
                #require valid-user
</limit>
```

e. Redirection des pages d'erreur.

En remplissant le .htaccess avec les lignes suivantes :

```
ErrorDocument 404 /2009/marechav/erreur.php
ErrorDocument 401 /2009/marechav/erreur.php
ErrorDocument 500 /2009/marechav/erreur.php
AddDefaultCharset utf-8
```

On indique que si une des ces 3 erreurs se passent, il renvoie la page «erreur.php».

IV. Création de la base de données (BDD)

a. Architecture

La base de données ici est «mosieur»
Une nouvelle table appelée «marechav_bdd» a été créer.

La table possèdes les champs suivant :
Index, date, titre, nom, prenom, adresse, adresse2, ville, codepostal, entreprise, description.

Il faut spécifier la nature des données dans chaque champ (pour la plupart du «text» en uft8) et ne pas oublier de créer une table index autoincrémentée.

b. Possibilités

Avec une table de données et les requêtes SQL associées, les possibilités sont multiples !

D'une part on peut enrichir cette table via un formulaire par exemple. On peut aussi ajouter de nouveaux champs, modifier/supprimer les entrées existantes. Mais là où il est intéressant de travailler avec une bdd c'est dans la manière d'utiliser les informations.
On peut ainsi les afficher à volontée et à la demande pour mettre en page un site web tout comme créer des fichiers clients personnalisés par exemple.
Pour ce faire il existe une multitudes de requêtes SQL parmi lesquelles SELECT, DELETE, ORDER, INSERT ou encore UPDATE etc...

V. Le PHP pour faire le lien entre le formulaire et la BDD

a. Inscription dans la base de donnée

La requête du formulaire est de type post et pour le définir comme attribut action dans la balise <form>, on utilise les script php qui va recherche l'url de la page à travers la variable d'environnement PHP_SELF :
```
<? echo $_SERVER['PHP_SELF']?>
```

Pour lire et/ou écrire dans la bdd il faut dire au script de se «connecter» à cette bdd via un fichier «connexion.php» dans lequel est inscrit les login et mot de passe du server SQL.
```
<? require_once('../http/interdit/connexion.php'); ?>
```

Le code qui suit lance la requête SQL quand on valide le formulaire. La condition if() sert à eviter que la requête soit envoyée à chaque fois que l'on charge la page :

```
if(isset($_POST['nom']) and $_POST['nom']!=""){
mysql_query("INSERT INTO `mosieur`.`marechav_bdd` (`index` , `date` , `titre` , `nom` , `prenom` , `adresse` , `adresse2` , `ville` , `codepostal` , `entreprise` , `description`)
VALUES (NULL , NOW( ) , '".$_POST['titre']."', '".$_POST['nom']."', '".$_POST['prenom']."', '".$_POST['adresse']."', '".$_POST['adresse2']."', '".$_POST['ville']."', '".$_POST['codepostal']."', '".$_POST['entreprise']."', '".$_POST['description']."');");
}
?>
```
Cette requête se décompose en deux phrases. Dans un premier temps INSERT INTO (insérer dans) puis on liste les champ de la table qui sont concernés par cette insertion. Ensuite VALUES qui sont les valeurs qui devront être insérer dans les champs cités plus haut.

Typiquement cette requête a été récupéré d'un copié/collé de requête SQL de phpmyadmin.

b. Conserver la saisie
Par défaut, lorsque l'on valide le formulaire, les champs saisies se réinitialisent. Pour conserver ces valeurs ont utilise des scripts php qui vont chercher dans les données stockées dans la requête POST.

Par exemple, pour les input de type «text» on utilise le code suivant pour l'attribut «value»:
```
value="<? echo @$_POST['nom'];?>"
```

Pour le cas des fenêtres déroulantes (titre) on utilise la fonction if afin d'écrire «selected» parmi les attributs si l'option vient d'être envoyée :
```
<? if($_POST['titre']=='M.') echo 'selected'; ?>
```

De manière similaire pour les boutons type radio, on met une condition pour écrire «checked» dans les attributs de l'input :
```
<? if($_POST['entreprise']=='CTP') echo 'checked'; ?>
```

c. Afficher le contenu de la base de donnée
Dans ce code, je stocke le contenu de ma requête dans une variable $sql qui va elle même être transformer dans un tableau grâce au fetch_array.
J'affiche ensuite, à la demande, les éléments de mon tableau via la fonction echo de php.

Le tout avec une syntaxe de tableau html et le tour est joué !

```
<?
$sql=mysql_query("SELECT * FROM `marechav_bdd` LIMIT 0 , 30");
while ($table=mysql_fetch_array($sql)){

echo "<tr>"."<td><input type='checkbox' name='".$_table['nom']."'/></td>"."<td>".
$table['titre']."</td>"."<td>".$table['nom']."</td>"."<td>".
$table['prenom']."</td>"."<td>".$table['adresse']."</td>"."<td>".
$table['ville']."</td>"."<td>".$table['codepostal']."</td>"."<td>".
$table['entreprise']."</td>"."</tr>\n";
}
?>
```

Conclusion

Ces séances m'ont énormement apportées car j'avais des notions mais véritablement aucunes connaissances en matières de base de données pour le web. J'ai pu appliquer mes précédentes connaissances en html/CSS pour construire un site avec une réellement utilitée derrière.

Depuis, je continu l'aventure en codant moi-même un projet de site (simple) géré par l'intermédiaire de table et de requêtes via formulaires... l'aventure contenue!

MERLOS Emilie - 2Ei Pagora

Compte rendu du TP de MySQL

Objectifs :

Les Objectifs de ce TP sont de découvrir le langage de programmation PHP, en complément des bases acquises précédemment en html. Ceci pour permettre de créer notamment des sites internet dynamiques.
Ceci sera complété par la découverte de la création d'une base de donnés en MySQL.

La découverte de ces deux outils de programmation que son PHP et MySQL s'est faite, tout au long de ce TP par l'application à un cas concret : la réalisation d'un formulaire en html et PHP permettant de rentrer diverses données. Ces données sont par la suite envoyées vers une base de données MySQL pour être stockées.

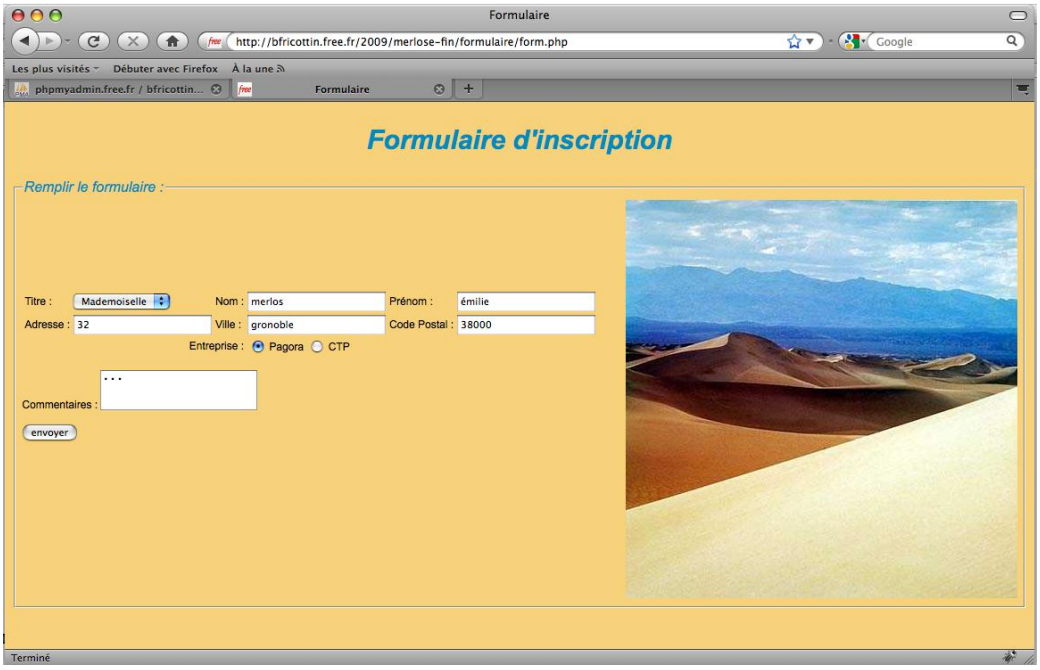
I - Informations générales

Afin de travailler à distance, comme c'est le cas dans la plupart des cas lorsque l'on travaille sur Internet, nous avons réalisé notre travail sur un compte Free, permettant de stocker nos données à distance sur un serveur (bfricottin.free.fr).

Nous avons créé un dossier dans lequel nous avons placé les différents fichiers que nous avons créés.
Il est ensuite important, pour des sites accessibles sur internet d'en assurer la sécurité. Pour cela, nous avons créé différents dossiers .htaccess et .htpassword afin de placer un mot de passe et d'effectuer des protection des dossiers par ce mot de passe.

II - Réalisation du formulaire en html

Nous allons écrire le code source du formulaire avec le logiciel TextWrangler, en langage html.
Ce formulaire comportera diverses informations, et divers champs que l'utilisateur devra compléter.
Il se présente sous la forme suivante :



FORMULAIRE D'INSCRIPTION

En ce qui concerne le code, le formulaire est délimité par les balises <FORM> qui définissent le formulaire et permettront au client de saisir interactivement des données.

Nous avons commencé par créer les différents champs que l'on souhaite voir apparaitre sur le formulaire, et la forme sous laquelle l'utilisateur devra rentrer les valeurs :

- Menu déroulant, pour le choix du titre (Mr, Mme, Melle)

Il correspond à la balise « select ». Elle est suivie de balises « option » contenant les différentes valeurs présentes dans la liste.

- Case pour rentrer des données, pour le nom, prénom, adresse, ville, code postal, commentaires…

Ceci permet à l'utilisateur de rentrer la chaine de caractère qu'il désire.
Il est réalisé par la balise « input type=text ».

- Boutons de choix, pour le choix de l'entreprise. L'utilisateur cliquera sur un des boutons pour choisir l'option qu'il désire. Les autres choix seront décochés dans ce cas. On appelle cela des radio boutons.

On utilise ici la balise « input type=radio » . Il faudra créer autant de balise qu'il y a d'options possibles. Car chaque balise contiendra le nom d'une option.

(voir les lignes 57 à 114) pour les détails du code, sur le fichier form.php

Nous avons ensuite réalisé le « design » du formulaire en liant au fichier précédent une feuille de style en CSS. Cette feuille nous permet ainsi de définir la couleur de fond, ainsi que la police, grandeur des caractères par exemple.

Ceci sera complété par la création d'un tableau dans lequel on mettra toutes nos données afin d'aligner tous les champs du formulaire (présence des balises <tr> et <td> dans le code source html).

Cependant, en l'état, ce formulaire est fixe, les données rentrées par l'utilisateur ne peuvent pas être utilisées. Ce formulaire sera donc complété par la suite.

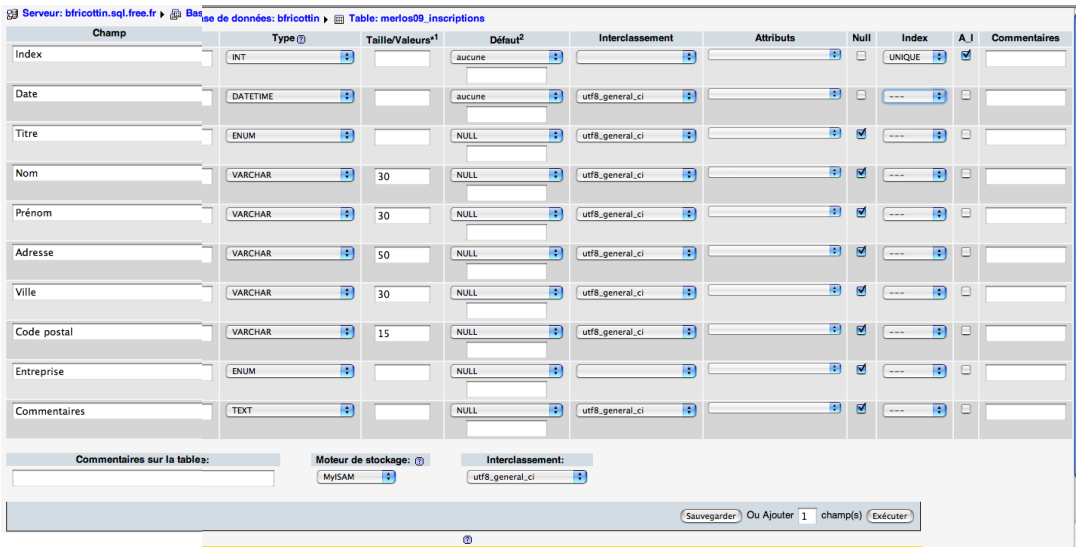
III - Réalisation de la base de données

Afin de compiler les informations que le formulaire nous permettra de récupérer, nous allons créer une base de données.
Pour cela nous utiliserons phpMyadmin, une fonctionnalité de Free permettant l'élaboration de bases de données en MySQL.

PhpMyadmin permet de créer sa base de données d'une manière simplifiée, à l'aide de scripts. Nous utiliserons cette fonctionnalité pour construire notre base de données.

La première étape est de créer une nouvelle table contenant les différents champs du formulaire, ainsi qu'un index et la date et heure de l'enregistrement.

L'étape suivante consiste à choisir le type de données que contiendra chaque colonne du tableau (entier, chaine de caractères…)



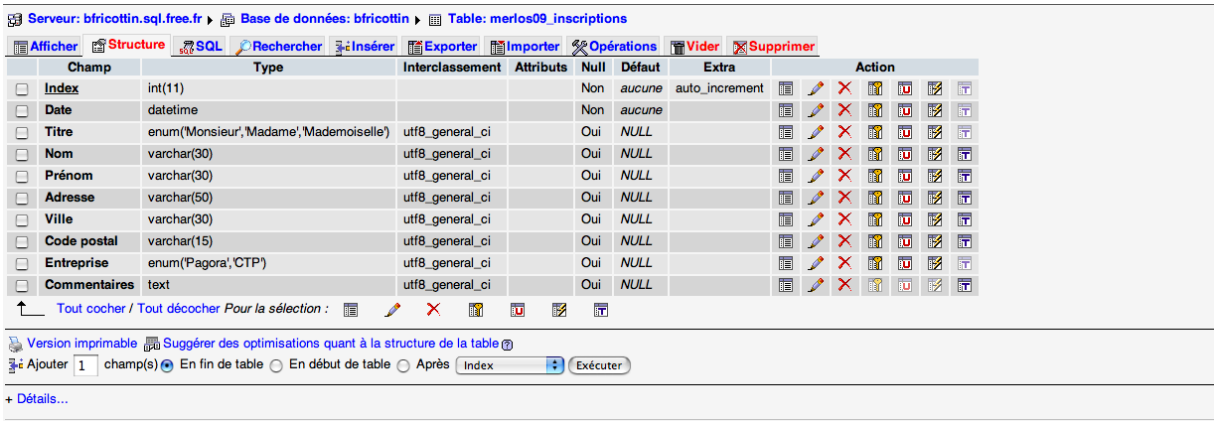
REALISATION DE LA BASE DE DONNEES : DETERMINATION DES DIFFERENTS CHAMPS

On choisit donc ici le nom du champs, le type de données qui vont être rentrées, la taille des chaînes de caractères, si on est dans le cas de chaînes de caractères.

On détermine ensuite la valeur qui sera attribuée par défaut dans le cas où aucune valeur n'est rentrée par l'utilisateur. On mettra rien ici dans le cas ou on choisit NULL.
La colonne interclassement nous permet d'informer la table du mode de codage des caractères qu'elle doit utiliser.

Dans le cas de l'index automatique que l'on a utilisé (première ligne), on devra choisir la case « UNIQUE » pour avoir toujours un indice différent. De même on devra cocher la case A_I (automatic index) pour incrémenter automatiquement l'index à chaque nouvelle donnée rentrée.
Dans le cas de la date, on utilisera une fonction prédéfinie qui donnera la date et l'heure automatiquement.

On aura donc, au final un tableau avec les caractéristiques suivantes :


Serveur: bfricottin.sql.free.fr ▸ Base de données: bfricottin ▸ Table: merlos09_inscriptions

Afficher | Structure | SQL | Rechercher | Insérer | Exporter | Importer | Opérations | Vider | Supprimer

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
☐	Index	int(11)			Non	aucune	auto_increment	
☐	Date	datetime			Non	aucune		
☐	Titre	enum('Monsieur','Madame','Mademoiselle')	utf8_general_ci		Oui	NULL		
☐	Nom	varchar(30)	utf8_general_ci		Oui	NULL		
☐	Prénom	varchar(30)	utf8_general_ci		Oui	NULL		
☐	Adresse	varchar(50)	utf8_general_ci		Oui	NULL		
☐	Ville	varchar(30)	utf8_general_ci		Oui	NULL		
☐	Code postal	varchar(15)	utf8_general_ci		Oui	NULL		
☐	Entreprise	enum('Pagora','CTP')	utf8_general_ci		Oui	NULL		
☐	Commentaires	text	utf8_general_ci		Oui	NULL		

Tout cocher / Tout décocher *Pour la sélection :*

Version imprimable | Suggérer des optimisations quant à la structure de la table

Ajouter 1 champ(s) ◉ En fin de table ○ En début de table ○ Après Index Exécuter

+ Détails...

RECAPITULATIF DES CARACTERISTIQUES DE LA BASE DE DONNEES

On peut aussi avoir accès au code ayant permis la réalisation de ce tableau :

```
CREATE TABLE  `bfricottin`.`merlos09_inscriptions` (
`Index` INT NOT NULL AUTO_INCREMENT ,
`Date` DATETIME NOT NULL ,
`Titre` ENUM(  "Monsieur",  "Madame",  "Mademoiselle" ) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL ,
`Nom` VARCHAR( 30 ) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL ,
`Pr�nom` VARCHAR( 30 ) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL ,
`Adresse` VARCHAR( 50 ) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL ,
`Ville` VARCHAR( 30 ) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL ,
`Code postal` VARCHAR( 15 ) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL ,
`Entreprise` ENUM(  "Pagora",  "CTP" ) NULL DEFAULT NULL ,
`Commentaires` TEXT CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL ,
PRIMARY KEY (  `Index` ) ,
UNIQUE (`Index`)
ENGINE = MYISAM CHARACTER SET utf8 COLLATE utf8_general_ci;
```

Sur ce tableau, on pourra ensuite effectuer plusieurs manipulations telles que effacer des données, rentrer manuellement des données, rechercher des données.
On pourra aussi avoir accès aux codes php correspondant à chacune des requêtes précédentes.

Par exemple dans le cas de suppression d'une ligne :

```
DELETE FROM `bfricottin`.`merlos09_inscriptions` WHERE `merlos09_inscriptions`.`Index` = 3 LIMIT 1
```

Dans le cas de l'insertion d'une ligne :

```
1 enregistrement(s) inséré(s).
Identifiant de l'enregistrement inséré : 1

INSERT INTO `bfricottin`.`merlos09_inscriptions` (
`Index` ,
`Date` ,
`Titre` ,
`Nom` ,
`Prénom` ,
`Adresse` ,
`Ville` ,
`Code postal` ,
`Entreprise` ,
`Commentaires`
)
VALUES (
NULL , '2009-11-13 09:17:48', 'Monsieur', 'Martin', 'Robert', '23 rue de la papeterie', 'Grenoble', '38000', 'CTP',
NULL
);
```

Dans le cas de la recherche de tous les « Monsieurs » du tableau :

```
Affichage des enregistrements 0 - 0 (1 total, Traitement en 0.0008 sec.)
SELECT *
FROM `merlos09_inscriptions`
WHERE `Titre` = 'Monsieur'
LIMIT 0 , 30
```

Maintenant que notre formulaire est construit, il faut modifier le formulaire en html, afin de pouvoir transférer les données rentrées par les utilisateur dans le formulaire, et envoyer ces informations vers la base de données.
Ceci nous permettra d'avoir un interface avec le client sans que celui-ci n'ait pas accès à la base de données. Ceci évite des destructions ou problèmes dans cette base de données.

IV – Modifications du formulaire en PHP

Afin que le formulaire envoie, vers la base de données, les informations rentrées sur le formulaire, nous devons utiliser le langage de programmation PHP.

Il se reconnait par des balises différentes des balises html : elles sont délimitées par : < ? ….. ?>.

Deux méthodes sont possibles pour transmettre les données en php : la méthode GET et la méthode POST. Nous avons choisi la méthode POST qui a un avantage majeur par rapport à la méthode GET : elle n'inscrit pas dans l'url les valeurs rentrées dans le formulaire, ce qui est utile, surtout dans le cas de la présence de mots de passe, qui seraient alors visibles de tous.

Le langage PHP sera utilisé dans le formulaire en HTML pour deux actions : récupérer les données rentrées par le client dans le formulaire et les transmettre ensuite à la base de données.

Dans le cas de la récupération de données, on va inclure le PHP directement dans le code en html :
Il faut tout d'abord indiquer que les données qui devront être échangées se trouvent dans le formulaire lui-même. Ceci est indiqué dans la fonction « action » avec
="<?= $_SERVER['PHP_SELF'] ?>"

Ainsi le code pour commencer notre formulaire en php sera : (avec « name » définissant le nom du formulaire)

```
<FORM action="<?= $_SERVER['PHP_SELF'] ?>" method="POST" name="inscription">
```

On va maintenant, pour chaque champs rempli par l'utilisateur, aller chercher les valeurs et les mémoriser afin de les envoyer à la base de données par la suite. Pour cela, on remplacera toutes les valeurs « value » rentrées en html par un code en php permettant d'aller chercher la valeur désirée :
Par exemple pour le prénom :
Prénom : <input type="text" name="prénom" value="<?= $_POST['prénom']?>"/>
La valeur récupérée sera celle contenue dans la balise « prénom ».

Une fois les informations récupérées, il faut les envoyer vers la base de données que l'on a créée.
Pour cela, on va insérer un autre code en php que l'on placera en début de code :

Par la suite, on va déterminer les informations qui vont être envoyées à la base de données.

Tout d'abord, il faut effectuer la connexion avec la base de données MySQL. Pour cela, on va placer les différentes informations nécessaires pour cette connexion dans un fichier protégé appelé « connexion.php »
Cette connexion ne sera effectuée qu'une fois, car ce n'est pas la peine d'effectuer cette opération une fois que la connexion est déjà établie. On utilise pour réaliser cela la requête « require_once », suivie du chemin pour atteindre le fichier contenant les paramètres de connexion.

```
require_once('../http/interdit/connexion.php');
```

On effectue ensuite la requête permettant le transfert des données vers la base de données. On utilise dans ce cas la requête « mysql_query »
On indique ensuite le nom de la table concernée, puis les différents champs à remplir, dans l'ordre.
La dernière étape consiste à indiquer le chemin d'accès aux informations à rentrer dans la base de données. Ceci sera effectué dans « VALUES ».
La méthode de transmission sera la méthode POST. L'ordre des champs indiqués doit être le même que le précédent (celui des champs à remplir).
On obtient ainsi le code suivant :

```
if (!mysql_query("INSERT
```

```
INTO `bfricottin`.`merlos09_inscriptions`
(`Index` ,`Date` ,`Titre` ,`Nom` ,`Prenom` ,`Adresse` ,`Ville` , `Code postal` ,`Entreprise` ,`Commentaires`)
VALUES (NULL , NOW(), '".$_POST['titre']."', '".$_POST['nom']."', '".$_POST['prénom']."', '".$_POST['adresse']."',
'".$_POST['ville']."', '".$_POST['codepostal']."', '".$_POST['entreprise']."', '".$_POST['commentaire']."');", $link))
{
echo "<p>erreur SQL : ".mysql_errno($link).": ".mysql_error($link)."</p> \n";}
?>
```

On obtient finalement deux documents communiquant entre eux, et permettant une interface avec un utilisateur.

Conclusion :

On a pu découvrir tout au long de ce TP les bases de la programmation en php, et la combinaison de celle-ci avec le langage html, vu l'an dernier.

L'application de cette programmation à un cas concret nous a permis de bien voir les fonctionnalités de chaque langage, ainsi que leurs spécificités.

REVILLION Joris

3EI

Décembre 2009

Compte rendu de PHP MySQL : création d'un formulaire de base de données

Objectifs :

Le principal objectif de ce cours est de découvrir et de nous familiariser avec le langage de script PHP qui est couramment utilisé, associé à du code HTML, pour la réalisation de page web dynamiques. Le PHP est un langage interprété et exécuté « coté serveur ». En effet, les balises PHP contiennent des informations et des requêtes qui sont interprétées par le serveur web, ce serveur va ensuite générer et envoyer la page demandé au format HTML. Cette page sera ensuite interprétée par le navigateur du client. Ce langage utilisé avec un système de gestion de bases de données tel que MySQL devient un outil très puissant qui permet de réaliser des formulaires et de gérer des bases de données sur internet.

Dans le cadre de ce cours, nous avons donc réalisé un formulaire en PHP qui permet à un utilisateur de fournir quelques informations. Ces informations pouvant ensuite être classées et enregistrées dans une base de données sur internet. Les différents fichiers réalisés durant les séances étant conservés sur un site free : **http://bfricottin.free.fr/2009/revillij/**

Travail réalisé :

Réalisation du formulaire en HTML

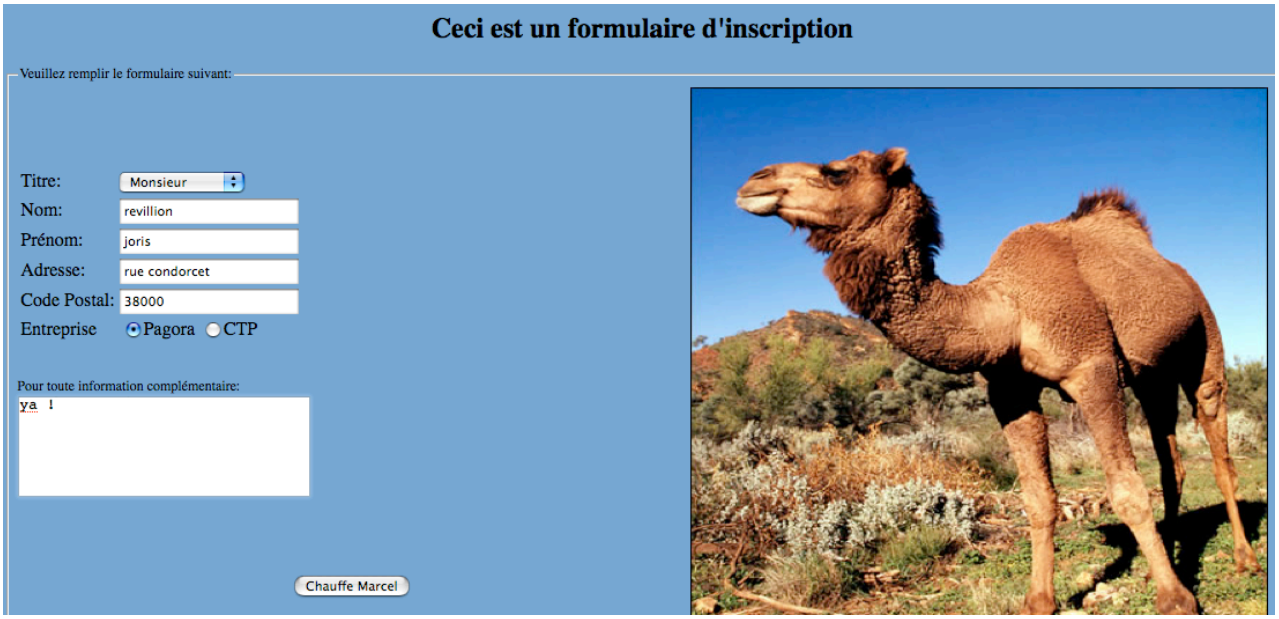
Dans un premier temps, nous avons réalisé, à l'aide de l'éditeur de texte TextWrangler notre formulaire de saisie uniquement en code HTML. Celui-ci permettait à l'utilisateur de saisir un certain nombre de renseignement dans des champs prévus à cet effet : son nom, prénom, adresse, code postal, titre, son entreprise et enfin un champ libre permettant de fournir toute information complémentaire. Ce fichier contient donc l'ensemble des balises nécessaires à toutes les pages en HTML : **<head>**, **<title>**, **<body>**, **<html>**. Elle contient également un lien vers un fichier css qui

permet de définir la mise en page (police de caractère, couleur de fond …) : <link type="text/css" href="../style.css" rel="stylesheet"/>.

Une image a également été insérée afin d'égayer quelque peu la page web : <img src="image.jpg" width="500" height="500" align="right">

Enfin l'utilisation d'un tableau (balises **<tr>** et **<td>**) permet d'organiser et d'aligner correctement les différents éléments.

Le principal défaut de ce questionnaire est qu'il est statique, les informations fournies par l'utilisateur ne peuvent pas être utilisées ou conservées, son intérêt est donc extrêmement limité. Il a donc été nécessaire d'avoir recours au langage PHP afin de rendre ce formulaire dynamique et fonctionnel.



Réalisation du formulaire en PHP

Comme nous l'avons déjà évoqué en introduction, les scripts PHP permettent de faire exécuter des actions au serveur web et peuvent donc être très utiles pour réaliser notre formulaire.

Les balises PHP se reconnaissent facilement car elles sont de type **< ?php** …. **?>**

Dans un premier temps il faut créer une balise **form** qui va permettre de créer l'ensemble des autres balises utilisées dan notre formulaire : **input**, **select**, **textarea**…

<form action="<?= $_SERVER['PHP_SELF']?>" method="POST" name="inscription">

Dans cette balise, **action** fait référence à la page qui est appelée lorsque l'utilisateur a envoyé son formulaire d'inscription. Ici, la page s'appelle elle-même ce qui permet de remplir plusieurs formulaires à la suite.

method est la façon dont est envoyé le formulaire, la méthode Post est celle qui est le plus couramment utilisé pour réaliser du PHP. Elle est plus utilisée que la méthode **get** car elle présente l'avantage de ne pas inscrire les valeurs des variables dans l'url de la page, cela permet ainsi de cacher un éventuel mot de passe ou d'autres données sensibles.

On crée en suite une balise **fieldset** qui va permettre de regrouper l'ensemble des éléments qui constituent notre formulaire. Nous allons décrire comment ont été réalisés les différents champs qui permettent à l'utilisateur de remplir le questionnaire.

Réalisation de zones de texte :

Dans ce formulaire certains renseignements sont des chaines de caractères simples qui sont directement renseignés par l'utilisateur pour donner son nom et son prénom par exemple. En HTML on utilise pour cela des balises input de type **text**.

Exemple :

Nom: <input type="text" name="nom" value="<? echo @$_POST['nom'] ?>"/>

Name permet de nommer la zone de texte et donc de générer une variable correspondante, ici la variable s'appelle **nom.**

Value corresponds à la valeur de la variable c'est-à-dire ce que l'utilisateur a renseigné dans ce champ. Ici, la commande **echo @$_POST['nom']** est une balise PHP qui permet d'afficher dans la case l'information qui a été fournie lors de l'envoie du précédent formulaire et qui est stockée dans la variable **nom**. Lorsqu'il s'agit du premier formulaire, aucune valeur n'a encore été attribuée à cette variable, la case est donc vide.

Création d'une liste déroulante :

D'autres renseignements peuvent nécessiter que l'utilisateur choisisse entre plusieurs possibilités prédéfinies. On utilisera pour cela des listes déroulantes.

Afin de réaliser une liste déroulante, on peut utiliser la balise **select.** Ici la variable corresponde est le **titre** et elle peut prendre plusieurs valeurs qui sont **Mme**, **Mlle** et **M** que l'on a définit par des balises **option**.

Exemple :

Titre: <select name="Titre">

<option value="Mme">**Madame**</option>

<option value="Mlle">**Mademoiselle**</option>

<option value="M" selected= "selected">**Monsieur**</option>

</select>

On a fait en sorte que la valeur par défaut qui est affichée soit celle de **Monsieur**, pour cela il faut utiliser la commande **selected=.**

Boutons de choix :

Le langage en PHP permet également de réaliser des « boutons » pour lesquels l'utilisateur devra cocher lui même la bonne proposition, on utilise pour cela la commande **input** de type **radio**.

Exemple :

```
Entreprise <input type="radio" name="Entreprise" value="Pagora"<? If
($_POST['Entreprise']=='Pagora')echo'checked'?>/> Pagora

<input type="radio" name="Entreprise" value="CTP"<? If
($_POST['Entreprise']=='CTP')echo'checked'?>/> CTP
```

Dans le cas décrit, l'utilisateur a le choix entre 2 entreprises : **Pagora** et le **CTP**. On peut également avoir recours à la commande **if** dans une balise PHP qui permet de cocher par défaut le choix qui a été réalisé précédemment. Le recourt à ce type de bouton est pertinent lorsque les choix ne peuvent pas être multiples, ici l'utilisateur devra faire un choix entre **Pagora** et le **CTP** et non pas cocher les deux champs. Il n'y a donc qu'une seule variable **entreprise** qui peut prendre deux valeurs selon le choix de l'utilisateur. Pour réaliser des choix multiples on utiliserait plutôt des cases à cocher de type **checkbox** qui n'ont pas été utilisées dans le présent formulaire.

Création d'une zone de texte libre:

Un formulaire peut nécessiter une zone de texte assez grande afin que l'utilisateur puisse y ajouter toute remarque qui lui semble pertinente de fournir dans le cadre du questionnaire. Pour cela on peut utiliser une balise **textarea** en définissant ses dimensions. La chaine de caractère qui est remplie dans cette case sera elle aussi une variable dont la valeur sera conservée sous le nom de **description.**

Exemple :

```
<textarea name="description" rows="5" cols="30"> <?= $_POST['description']?>
</textarea>
```

On retrouve une balise PHP **$_POST** qui permet de pré remplir le champ avec la valeur précédente de la variable.

Envoi du formulaire :

Lorsque l'utilisateur a terminé de remplir son questionnaire, celui-ci va envoyer le formulaire en cliquant sur un bouton de validation. Ce bouton est réalisé avec une balise **input** de type **submit.** Cela permet donc de définir une action à réaliser : envoyer les valeurs des différentes variables au serveur.

Exemple :

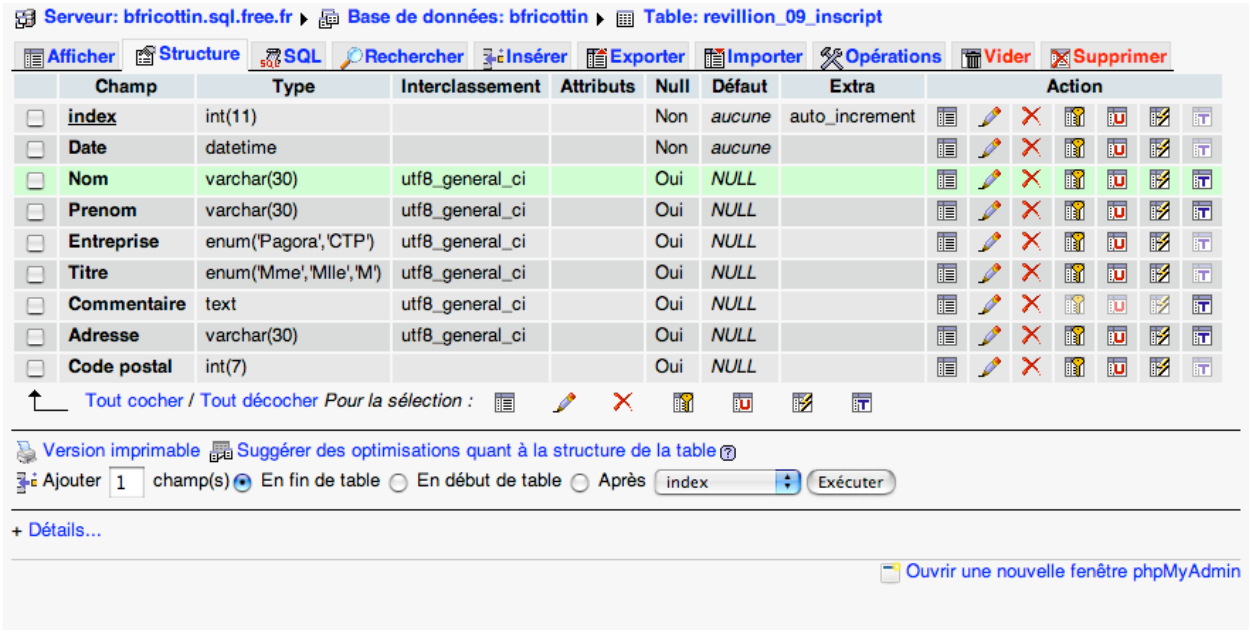
```
<input type="submit" value="Chauffe Marcel" name="faire" />
```

Enregistrement des données dans une base de données :

Le travail qui vient d'être réalisé permet donc de remplir convenablement le formulaire et d'envoyer un certain nombre d'élément au serveur qui peut à son tour nous les renvoyer grâce aux balises **echo**. Il reste cependant une étape très importante à réaliser : permettre l'enregistrement et la conservation à long terme de ces éléments afin de pouvoir les exploiter par la suite.

Pour cela on va utiliser le site http://phpmyadmin.free.fr/ qui va nous permettre de créer une base de données en MySQL. La première étape de ce travail est de créer une table qui contiendra l'ensemble des réponses fouines à chaque validation du formulaire. Dans cette table on va donc retrouver l'ensemble des variables qui correspondent au formulaire en **php** réalisé précédemment : nom, prénom, entreprise, adresse … Chaque élément de cette table va être définit en fonction du type de données qu'il va contenir. Ainsi pour les champs qui ne peuvent contenir que certaine valeurs (par exemple ceux définit par des listes déroulantes ou des boutons), on choisira la fonction **enum** en spécifiant les valeurs possibles, alors que pour les champs qui n'ont pas de valeur prédéfinies on utilisera par exemple **varchar(30)**. On peut également définir ou non des valeurs par défaut si l'utilisateur ne rempli pas un champ, ou encore la possibilité de laisser un champ vide. Il faut également bien spécifier le type de codage utilisé (ici **utf-8**) afin d'éviter les problèmes que peuvent causer les caractères accentués et caractères spéciaux par exemple.

Pour une meilleure utilisation de cette table il peut être utile de connaitre la date et l'heure précise du remplissage du formulaire. Un champ approprié a donc été créé dans cette table qui permet de réaliser cette opération automatiquement. Un dernier champ **Index** a également été créé afin de numéroter automatiquement les formulaires, l'incrémentation de ce champ est automatique (pour cela il faut cocher la case **AI** lors de la création de la table).



Après avoir créé cette table nommée **revillion_09_inscript**, il faut faire en sorte que les valeurs des variables du formulaire soient bien redirigée et enregistrés dans cette table. Pour cela il faut créer une requête SQL dans le formulaire lui-même avec la commande **INSERT INTO**.

```
<? require_once('../http/interdit/connexion.php');

$sql = " INSERT INTO `bfricottin`.`revillion_09_inscript`

(`index`, `Date`, `Nom`, `Prenom`, `Entreprise`, `Titre`, `Commentaire`, `Adresse`, `Code Postal`)

VALUES ( NULL , NOW( ), '".$_POST['nom']."', '".$_POST['prenom']."',
'".$_POST['Entreprise']."', '".$_POST['Titre']."', '".$_POST['description']."',
'".$_POST['adresse']."', '".$_POST['Codepost']."' ); ";

mysql_query($sql); echo "\n<p> sql = <tt>".$sql."</tt></p>";

?>
```

Les différentes opérations concernant la gestion de cette table (suppression/ajout de ligne ou de champs, sélection de certaines données…) peuvent être réalisées directement en mode graphique sur le site http://phpmyadmin.free.fr/ mais peuvent également être réalisées grâce à des requêtes sql. Quelque exemples de ces requêtes ont été conservés dans le fichier **table.sql** afin d'en connaître la structure.

Conclusion :

Le travail qui a été réalisé au cours des différentes séances nous a donc permit de découvrir de manière concrète le mode de fonctionnement des bases de données réalisées en PHP MySQL sur internet. Les principales difficultés rencontrées proviennent d'erreurs de syntaxe lors de la création du code en HTML ou en PHP qui peuvent rendre les actions désirées totalement inactives.

TONDELIER Alan
3E Grenoble INP-Pagora

MySQL

Réalisation d'un mini-site

Compte Rendu

Décembre 2009

Avant propos

Dans ce compte rendu nous allons aborder la gestion d'une base de données SQL via un site internet, le compte rendu se concentrera principalement sur l'aspect lecture/écriture /modification ; ainsi nous ne détaillerons pas le code HTML et CSS, ces deux langages ayant été vus lors des cours de « documents numériques structurés ».

Le site de gestion des cartes de visite est disponible à l'adresse :
http://agserver3.free.fr/pagora/cdv

Le login associé est le suivant :
User : agserver3
Pass : agserver

On rappel l'adresse du FTP :
ftpperso.free.fr
Et de phpmyadmin :
sql.free.fr
Nom de la base de donnée :
pag_inscription

Introduction

Durant ce cours, nous avons réalisé un site internet permettant la réalisation « simplifiée » de cartes de visites électroniques. Ce site permet donc de saisir les cartes de visite, de les visualiser, de modifier leur contenu ainsi que leur suppression.

Nous allons présenter le site dans sa globalité, puis nous nous arrêterons plus particulièrement sur les étapes d'écriture, de lecture et de modification de la base de données.

Table des matières

1. Généralité sur les « langages d'internet » : HTML & PHP

Une page internet classique peut être définie comme le regroupement de deux parties : une partie statique qui reste la même (par exemple un pied de page) et une partie dynamique dont le contenu est variable selon ce qui est saisi par le webmestre (les billets sur un blog) ou par les utilisateurs (les commentaires d'un billet).

1.1. Le contenu statique : langage HTML

Lorsque l'on navigue sur internet, l'ensemble des pages rencontrées sont représentées dans un langage à balises strictement défini : l'Hypertext Markup Language ou HTML. Ce langage permet donc d'afficher les différents éléments d'une page internet, par exemple le code :

```
<html>
<body>

<p>Hello world</p>

</body>
</html>
```

Affichera un paragraphe dont le contenu sera « Hello world ». Il existe un grand nombre de balises dédiées à la mise en page et à l'insertion d'éléments ; la balise <img /> permet par exemple d'insérer une image dans notre page.

Les limites de l'HTML se font cependant sentir, en effet le code d'une page HTML est figé pour modifier le texte de notre exemple, il faudrait ouvrir le fichier html et remplacer le texte « à la main ».

1.2. Le contenu dynamique : langage PHP et base de donnée MySQL

Afin de rendre les pages internet moins rigides les développeurs ont mis au point des langages informatiques permettant de modifier le contenu d'un site internet de manière simple et pratique. Le langage le plus utilisé pour arriver à ces fins est le langage PHP (Hypertext Preprocessor).

Le fonctionnement du langage PHP est résumé par le schéma suivant :

Schéma de fonctionnement d'une page en PHP – Source : wikipedia

Ainsi le PHP va en réalité générer du code HTML afin de rendre la page lisible par le navigateur internet. Reprenons l'exemple vu précédemment :

```
<html>
<body>

<p>Hello world</p>

</body>
</html>
```

Ce code html deviendrait alors en PHP :

```
<html>
<body>

<p><?php echo "Hello world" ?></p>

</body>
</html>
```

Lors du chargement de la page, le langage PHP va être interprété et le serveur Apache (le moteur de conversion en quelque sorte) va transformer ce morceau de code de façon à faire apparaitre « Hello world ».

Cependant l'intérêt d'un tel morceau de code est limité, en effet il n'y a guère de changement avec le code HTML, il est tout aussi nécessaire de modifier le code à la main afin d'afficher « Hello world ».

La solution pour afficher du texte dynamique est de faire appel à une base de donnée, en effet dans cette base de donnée nous allons pouvoir stocker des informations (via un

formulaire en HTML) et les extraire de façon sélective. Nous allons utiliser une base de donnée MySQL, format de base de donnée la plus rependue sur l'internet.

2. Structure et opérations sur la base de donnée

Dans cette seconde partie nous allons nous intéresser aux parties création, lecture et écriture de la base de données MySQL utilisée pour notre site de gestion de cartes de visites.

Il faut noter que pour toutes les opérations sur la base de données à partir du site internet, il est nécessaire de se connecter à la base de donnée. Ceci est effectué par le fichier connect.php appelé en début de fichier index.php.

2.1. Création de la base de donnée

La création de la base de donnée a été réalisée via l'interface phpmyadmin (sql.free.fr), cette interface permet de créer facilement une base de donnée et les tables associées :

Vue des tables de la base de donnée pag_inscription

A noter qu'il aurait été possible de créer la base de donnée via une requête SQL de type :

```
CREATE TABLE 'pag_inscription'
(Nom char(50),
Prénom char(50)
[…]
)
```

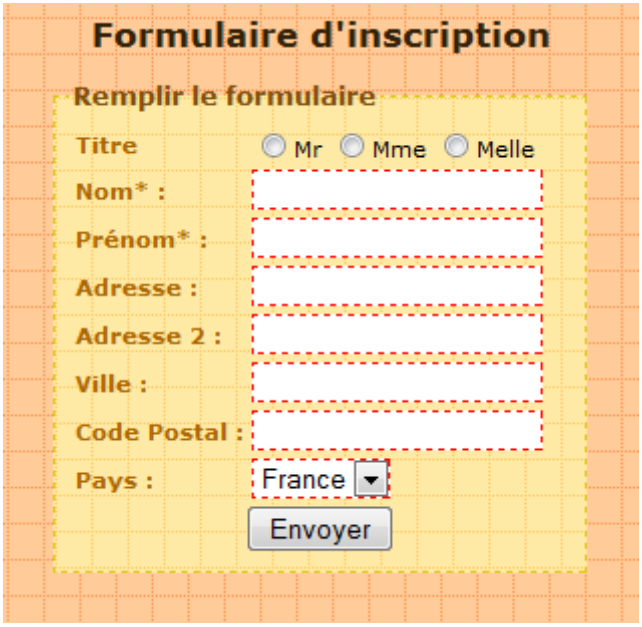
Cette requête pouvant être lancée telle quelle dans l'onglet SQL de phpmyadmin ou bien exécutée grâce au PHP via un mysql_query($texte_requete). Notre base de donnée une fois créée possède 10 champs :

id : Un index automatique, permettant d'identifier l'enregistrement.
date : date de création de l'enregistrement, automatique également.
Les 8 autres champs sont des valeurs saisies par l'utilisateur lors de la création d'une carte de visite.

2.2. Ecriture dans la base de donnée

Nous allons maintenant nous intéresser à l'insertion d'informations dans la base de donnée via notre site internet et donc grâce au code PHP.

Pour insérer des données nous avons réalisé un formulaire en HTML « Créer une nouvelle carte de visite »



Formulaire de saisies dans la base de donnée

Lorsque l'on rempli notre formulaire, les données sont envoyées dans la page actuelle :

```
<form action="<? echo $_SERVER['PHP_SELF'] ?>" method="POST"
name="inscription" id="formulaire" >
```

Chaque champ du formulaire possède un identifiant :

```
<input class="champ" type="text" name="prenom" value="<? echo
$_POST['prenom'] ?>"/>
```

Ainsi la valeur entrée dans le champ prénom sera stockée dans la variable $_POST['prenom'].
Il suffit ensuite de faire un test en PHP qui dit que si les variables sont envoyées alors on stocke les valeurs dans un nouvel enregistrement de notre base de donnée en prenant soin d'éviter les failles (protection des apostrophes par des slashes) :

```
if(isset($_POST['nom']) && isset($_POST['prenom'])) {
        $nom = addslashes($_POST['nom']);
        $prenom = addslashes($_POST['prenom']);
        $titre = addslashes($_POST['titre']);
        $adresse = addslashes($_POST['adresse']);
        $adresse2 = addslashes($_POST['adresse2']);
        $ville = addslashes($_POST['ville']);
        $codepostal = addslashes($_POST['codepostal']);
        $pays = addslashes($_POST['pays']);
}
```

On envoie ensuite les variables dans la base de donnée grâce à une requête SQL INSERT :

```
$requete = "INSERT INTO `agserver3`.`pag_inscription` (`id`
,`date` ,`titre` ,`nom` ,`prenom` ,`adresse1` ,`adresse2`
,`codepostal` ,`ville` ,`pays`) VALUES (NULL , NOW( ) ,
'".$titre."', '".$nom."', '".$prenom."', '".$adresse."',
'".$adresse2."', '".$codepostal."', '".$ville."', '".$pays."')";
mysql_query($requete) ;
```

Se référer au code complet pour l'ensemble des informations.

Ainsi on a ajouté un nouvel enregistrement, il est possible d'en rajouter un très grand nombre. Nous allons nous intéresser maintenant à la lecture de ces enregistrements.

2.3. Lecture des enregistrements de la base de donnée

Dans la page « Liste des cartes de visites » nous avons la liste complète des différentes cartes de visites qui ont été ajoutées dans la base de donnée. Pour afficher ces différentes cartes de visites nous faisons une nouvelle requête SQL :

```
$sql = "SELECT * FROM `pag_inscription`"
```

Cette requête selectionne l'ensemble des valeurs dans la table inscription. Nous faisons une boucle jusqu'à ce que l'on ai affiché le dernier enregistrement :

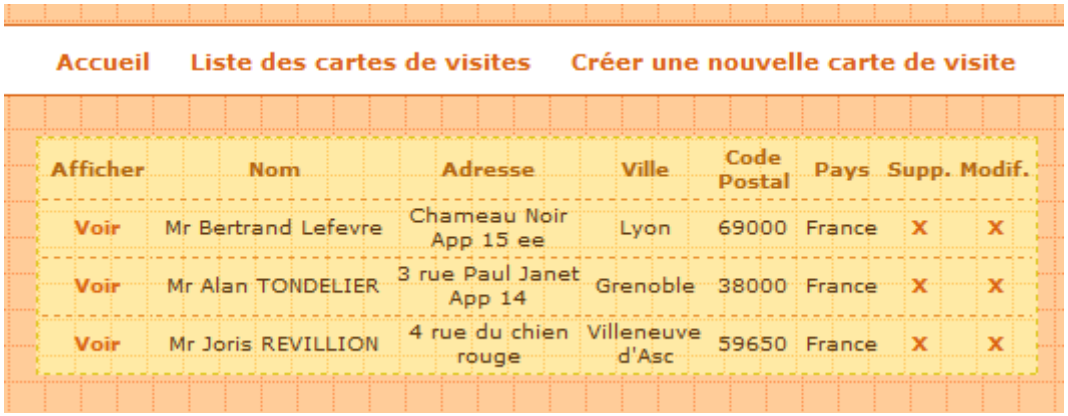
```
$req=mysql_query($sql) or die("erreur de connexion au serveur") ;
while($table=mysql_fetch_array($req)){
```

Puis on peut afficher l'ensemble des enregistrement en faisant appel aux données passées dans le tableau $table :

```
[...]
<td><?php echo stripslashes($table['titre'])."
".stripslashes($table['prenom'])." ".stripslashes($table['nom']);
?></td>

<td><?php echo stripslashes($table['adresse1'])."
".stripslashes($table['adresse2']); ?></td>
[...]
```

Résultat final :



Listing des cartes de visites

2.4. Suppression et modification d'un enregistrement

Il est possible d'effectuer d'autres opérations sur les différents enregistrement, la capture d'écran ci-dessus montre qu'il est possible de supprimer un enregistrement.

La suppression d'un enregistrement se fait via une requête SQL DELETE en prenant soin de préciser l'id de l'enregistrement à supprimer :

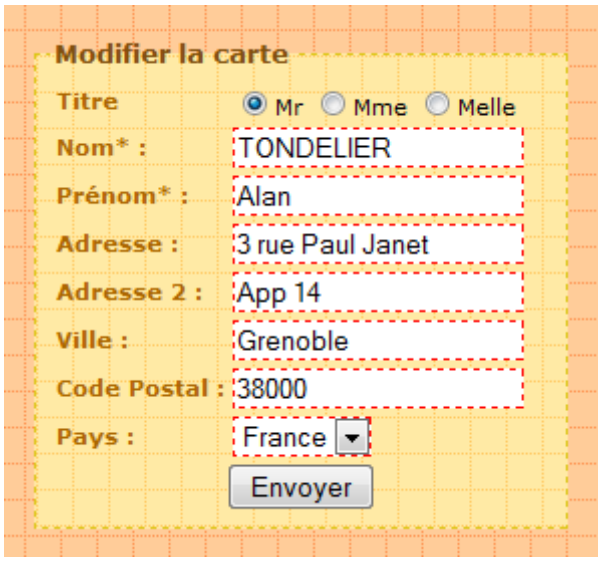
```
$del="DELETE FROM `agserver3`.`pag_inscription` WHERE
`pag_inscription`.`id` = '".$_GET['del']."'";
```

L'id est passé via l'URL, c'est-à-dire que dans le cas suivant :
http://agserver3.free.fr/pagora/cdv/index.php?menu=1&**del=25**
on supprimera alors l'enregistrement possédant l'id 25.

La démarche pour modifier un enregistrement est assez proche de celle d'insertion, à la différence qu'on doit écrire nos informations à la place d'autres existantes.
Le tout se gère via la requête SQL UPDATE. On commence tout d'abord par extraire les données actuelles de l'enregistrement :

```
  $sql2 = "SELECT * FROM `pag_inscription` WHERE
`id`=".$_GET['mod'];
```

Le passage de l'id se fait de la même manière que pour la suppression. On affiche ensuite dans le formulaire de modification les valeurs actuelles :



Formulaire de modification, chaque champ à pour valeur : <? echo $mod['champ'] ?>

NB : le formulaire possède deux champs cachés, pour transmettre l'id de l'enregistrement et pour spécifier que c'est une modification.

Une fois les modifications effectuées celles-ci sont envoyées à la base de données en remplacement des anciennes valeurs :

```
"UPDATE `agserver3`.`pag_inscription` SET `nom` = '".$nom."',
`prenom` = '".$prenom."', `titre` = '".$titre."', `adresse1` =
'".$adresse."', `adresse2` = '".$adresse2."', `ville` =
'".$ville."', `codepostal` = '".$codepostal."', `pays` =
'".$pays."' WHERE `pag_inscription`.`id` =".$_POST['id'];
```

3. Discussion sur l'intérêt des bases de données MySQL et du PHP.

Cet exercice ne fait qu'effleurer la puissance du couple PHP/MySQL, en effet avec ces instruments il est possible de réaliser des sites complexes et complets allant des forums aux sites marchants. Bien qu'aujourd'hui les différents moyens de réaliser des sites dynamiques se fassent concurrence (ASP/.NET, JSP ...) le PHP/MySQL reste une solution simple et gratuite pour développer des solutions sur internet.

Mais ce n'est pas tout, certaines entreprises (Gascogne PAPER par exemple) reposent sur des solutions en intranet basées sur le PHP pour les statistiques machines (en revanche je pense que la base de donnée doit être une base Oracle).

Pour conclure, la connaissance de ces langages et de la gestion d'une base de donnée, permet de mettre en place des solutions peu couteuse afin de se faire connaitre sur l'internet, mais aussi de proposer des solutions en interne, comme l'amélioration de la production grâce à un regroupement de l'information de façon structurée, exploitable facilement et de manière extrêmement flexible.