



Créer un espace membre pour son site

Par Mr. Black
et Neoterranos



www.siteduzero.com

*Licence Creative Commons 6 2.0
Dernière mise à jour le 30/04/2012*

Sommaire

Sommaire	2
Partager	1
Créer un espace membre pour son site	3
Partie 1 : Création d'un système simple	5
Introduction	5
Un espace membre ? Kékécé ?	5
Présentation de la première partie	5
Inscription	6
Connexion et déconnexion	6
Profil	6
Édition de profil et administration	6
Présentation de la seconde partie	7
Inscription 1/2 (le retour)	7
Inscription 2/2	7
Connexion	7
Profil	8
Édition d'un profil	8
La table SQL	8
Les bases du système	9
Des pages bien utiles	10
information.php	10
index.php	11
fonctions.php	13
config.php	20
D'autres pages à inclure	21
La page haut.php	21
colg.php	23
Page bas.php	23
Le design	24
Inscription	26
Avant de commencer	27
Le formulaire	27
Charte et système anti-robot	30
Si tu as lu le règlement, tu peux t'inscrire	31
Si tu arrives à me décoder, tu peux entrer	36
La validation et les mails	44
Validation du formulaire	45
Les fonctions	46
Validation proprement dite	49
Un petit mail ?	61
TP : connexion et déconnexion	65
Prérequis et explications	65
connexion.php	65
deconnexion.php	65
Correction	65
connexion.php	69
deconnexion.php	70
Et après ?	71
TP : site et profils	71
Exercice 1 - Création de statistiques	72
Les différents cas	72
Correction de l'exercice 1	74
Exercice 2 - Création de la page user.php	89
Correction de l'exercice 2	90



Créer un espace membre pour son site



Par

Neoterranos et



Mr. Black

Mise à jour : 30/04/2012

Difficulté : Intermédiaire



1 653 visites depuis 7 jours, classé 86/797

Bonjour amis Zér0s !! 😊

Dans ce big-tuto, nous allons créer ensemble un système d'espace pour les membres de votre site : grâce à cela, vous pourrez créer une communauté et gérer au mieux votre site. 😊

Au niveau des pré-requis, avoir suivi l'intégralité du cours PHP/MySQL ainsi que le cours XHTML/CSS de M@teo21 suffira amplement (et par la même occasion, c'est une base nécessaire). 😊

Cependant, nous utiliserons dans la partie II du JavaScript et les objets XHR (XMLHttpRequest), et la programmation orientée objets en partie III : si vous les connaissez, cela sera grandement utile, mais nous y reviendrons en temps voulu. 😊

Vous êtes prêts ? 🤖

Renseignements

- 28/08/2008 : décision de créer ce tutoriel, les premières lignes en sont tapées le soir même.
- 22/09/2008 : le tutoriel en est à la fin de son quatrième chapitre et proposé à la validation.
- 05/10/2008 : le tutoriel est refusé à la validation pour des problèmes au niveau de la page `information.php` ainsi que de la taille du chapitre IV, relecture et mise à niveau exigée.
- 22/10/2008 : le tutoriel est renvoyé à la validation après d'importantes modifications.
- 29/10/2008 : le tutoriel est accepté, mais il contient quelques fautes qui seront corrigées dans la prochaine édition, et deux-trois points sont à modifier.
- 29/10/2008 : correction de nombreuses fautes, modification de certains points, et coupure en deux du chapitre d'inscription. Rajout d'une sous-partie sur la fonction `mail()`.
- 20/11/2008 : après la zCorrection de `ptipilou` qui a encore permis de trouver quelques erreurs et de discuter de la lourdeur de certaines phrases le tuto va reprendre son développement.
- 23/11/2008 : en créant un chapitre intermédiaire avec le TP suivant, découverte d'une **erreur** sur `sqlquery()`, pensez à mettre à jour votre code, ajout d'une fonction `queries()`, donc suppression des `<?php global $queries;` et autres `<?php $queries++; ?>`. Pensez à regarder cela aussi. 😊



À cause d'un très fort manque de temps de notre part à tous les deux (Neoterranos et LkY), ainsi que d'autres raisons qui ont été exposées maintes fois sur le forum, nous avons décidé **d'arrêter totalement la création du tutoriel**. Dans l'état où il est, le tutoriel permet tout de même d'avoir les bases pour un espace membre simple, il ne tient qu'à vous de l'améliorer.

Je vous prie d'accepter nos excuses pour cela.

Après de multiples demandes issues de membres sans cesse refusées, j'ai pris la décision d'ajouter ce message au tuto :



- **Aucune archive contenant les fichiers créés dans ce tuto ne sera fournie**, pour la simple raison que les codes abordés via ce tuto y sont étudiés dans leur intégralité.



- Si vous rencontrez des erreurs, il est par ailleurs intéressant de chercher par vous-même, et vous pouvez ensuite demander de l'aide dans le topic du forum dédié à ce tuto, j'essaierai de répondre à vos questions.

Merci de votre compréhension.

Partie 1 : Création d'un système simple

Dans cette première partie, nous allons mettre en place tout le système basique. Les pages pour s'inscrire, se connecter, gérer son compte etc.

Il sera fonctionnel à la fin de cette partie et peut largement suffire, même si nous vous proposerons des améliorations importantes dans la seconde partie. 😊

Nous allons commencer en douceur, avec un chapitre introduisant ce tutoriel, vous y découvrirez comment nous allons procéder. Allons-y ! 😊

Introduction

Bonjour et bienvenue à tous dans le premier chapitre de ce tuto.

Pour commencer, nous allons vous expliquer ce que nous (LkY et Neoterranos) vous avons prévu dans ce tuto.

Si vous vous en fichez royalement, passez au chapitre suivant, mais nous vous recommandons tout de même de lire ce chapitre : il pose les bases du système et nous vous y donnerons la table SQL, c'est important ! 😊

Un espace membre ? Kékécé ?

C'est une bonne question, ça : qu'est-ce que c'est, un espace membre ?

Sur conseil de Ziame, nous avons décidé de faire une sous-partie pour vous expliquer ce qu'on entend par espace membre. L'espace membre est tout ce qui permet au membre de contribuer à un site web.

On peut donc en quelque sorte dire que le forum fait partie de l'espace membre, ou que la messagerie privée fait partie de l'espace membre.

Mais nous ne voulons pas refaire le monde, encore moins vous donner en un seul tuto de quoi faire tout un site web ; nous allons donc limiter cette définition à une partie restreinte : l'inscription, la connexion, l'affichage et la gestion de profils.

Comme nous ne voulons pas vous abandonner n'importe où, nous vous proposerons aussi une gestion des droits que vous pourrez même étendre à un forum ou à un système de news. 😊

Pour vous donner un point de comparaison avec le SdZ, nous traiterons des pages similaires à celles que vous allez voir sur ces images.

Je mets des miniatures car les grandes images prennent de la place : pour les voir en taille réelle, vous savez quoi faire. 😊

Et pour ceux qui savent pas, faut cliquer dessus. Ok je sors ==> []



Pour éviter des remarques futures (pis je suis content, je la sors en premier 🤪), non, nous ne vous apprendrons pas à faire un système de synthèse vocale. 😊

Bon, attaquons un descriptif des parties prévues. 😊

Présentation de la première partie

Avant toute chose, nous nous sommes demandés de quoi nous allons vous parler dans ce tuto, cela n'a pas été difficile : nous voulons créer un espace membre.

Pour avoir un espace membre, il faut déjà permettre aux membres de s'inscrire, de se connecter et de modifier leur profil, voire de consulter celui des autres (ça peut toujours servir 🤪).

Nous allons donc vous décrire ici tout ce que vous (oui, vous ; nous, on est des feignasses et on compte pas se fatiguer pour vous 🤪) allez faire dans la première partie.

Inscription

La première page du système proprement dit contiendra le formulaire d'inscription, la deuxième, la page de vérification. Comme ce sera le début, nous devons aussi vous faire coder d'autres fonctions qui nous serviront tout au long de cette partie. Le formulaire d'inscription contiendra les éléments suivants :

- pseudo ;
- mot de passe ;
- confirmation du mot de passe ;
- e-mail ;
- confirmation de l'e-mail ;
- date de naissance ;
- système de protection anti-robot.

Comme vous le voyez, ça fait déjà pas mal. Surtout le système anti-robot, c'est ce qui risque de nous prendre le plus de temps.



La liste ici ne vous présente pas toutes les options ; dans la partie *Profil*, nous comptons rajouter d'autres choses, vous verrez au moment voulu. 😊

La deuxième partie parlera du traitement du formulaire :

- vérification de la disponibilité du pseudo ;
- vérifications sur le mot de passe ;
- comparaison des deux champs mot de passe ;
- vérification de l'e-mail ;
- comparaison des deux e-mails ;
- vérification de la protection anti-robot ;
- envoi dans la base de données ou redirection selon le cas.

Dans cette page, nous toucherons déjà plus aux variables de sessions (et autres), car la redirection, pour être propre, se fera avec l'utilisation de sessions.

À noter que nous utiliserons aussi la fonction `mail` , si vous pouvez vous en servir.

Connexion et déconnexion

Ce chapitre traitera de la connexion.

Il y aura un formulaire demandant pseudo et mot de passe, ainsi qu'un bouton Connexion automatique qui utilisera les cookies.

La page d'arrivée vérifiera les informations et créera les cookies si la case a été cochée, une bonne occasion d'utiliser `setcookie()` . 😊

`deconnexion.php` fermera la session ouverte.

Profil

Cette page affichera au moyen d'une variable `get` un membre par son id ou par son pseudo.

Sur la page, on y verra différentes parties : pseudo, date d'inscription, âge si le membre veut le rendre public, l'e-mail s'il veut le rendre public aussi, adresses MSN, Yahoo, AIM, Skype, profession, localisation, biographie (comme sur le SdZ 😊) et un bel avatar pour être le plus beau ; il sera assez facile d'y rajouter d'autres choses, comme des liens forum (genre voir tous les messages de ce membre) et autres joyeusetés du genre.

Si on a les droits, on pourra éditer le profil (il y aura un lien) de la personne, et accéder aux informations privées (car pour l'administrateur, rien n'est privé 😊).

Édition de profil et administration

Sur cette page, il y aura un graaaaaaaaaaaaaaaaand formulaire (y a assez de a ? 🤪) qui permettra de modifier tout ça et peut-être plus encore :

- mot de passe ;
- e-mail ;
- date de naissance ;
- MSN ;
- Yahoo ;
- AIM ;
- localisation ;
- profession / occupation ;
- avatar ;
- biographie ;
- signature pour forum (nous sommes prévoyants 🤪) ;
- ~~nom de son chien...~~

Comme quoi il y a du boulot, car après, il faut aussi balancer tout ça dans la base de données, et plus il y a d'infos, plus les risques de plantage sont importants ; en plus, faut faire gaffe à la sécurité. 😊

C'est pour cela que nous aborderons ce point dans les derniers.

L'administration touchera à un système de droits. 😊

Présentation de la seconde partie

La seconde partie, comme il a été dit, prendra en charge des améliorations de ce système.

Elle touchera donc à des points plus complexes tels que JavaScript, les objets XMLHttpRequest, et des modifications des scripts déjà en place.

Une introduction vous parlera de JavaScript et d'AJAX.



Cette introduction n'aura pas pour but de vous apprendre tout ce qu'il faut savoir sur JavaScript ou sur les objets XHR, mais de poser des rappels et vous donner des liens vers de très bons tutoriels pour mieux comprendre comment tout cela marche.

Nous estimons cependant qu'elle est nécessaire car elle vous fera quitter les sphères PHP pour vous ouvrir à un autre langage et à une méthode qui utilise JS et PHP.

Ensuite, nous toucherons à notre espace membre proprement dit.

Inscription 1/2 (le retour)

Dans cette page, nous allons rendre dynamique un bon nombre de choses :

- vérification en temps réel de la disponibilité du pseudo (XHR) ;
- mot de passe correct (JavaScript) ;
- mot de passe et mot de passe de confirmation identiques (JavaScript) ;
- évaluation du mot de passe (Faible / Moyen / Fort) (JavaScript) ;
- vérification de l'e-mail (XHR) ;
- comparaison des deux e-mails (JavaScript) ;
- affichage du bouton Valider mon inscription seulement si le formulaire est bon (JavaScript) ;
- actualisation du captcha s'il est illisible (JavaScript).

Inscription 2/2

Il n'est pas encore certain que nous ayons à modifier cette page, mais c'est une éventualité que nous gardons à l'esprit. 😊

Connexion

De même que pour inscription 2/2.

Profil

A priori, ce n'est que de l'affichage, nous ne sommes pas censés y revenir, mais nous ajouterons un système de "parsing" avec les expressions régulières (ou avec DOM, cela dépendra de ma compréhension du système DOM quand je me pencherai dessus) pour faire un joli "bbcode" (ou zCode) qui donnera plus d'allure à la biographie du membre ou à sa signature forum, etc. 😊

D'autres systèmes verront peut-être le jour sur cette page aussi.

Édition d'un profil

Dans ce chapitre, nous allons créer un "bbcode" (ou zCode) à l'insertion, ainsi qu'un système d'aperçu utilisant XHR. Sur le papier c'est court, mais cette partie nous promet pas mal de travail. 🤖

La table SQL

Bien, avant d'attaquer l'espace membre proprement dit, nous allons créer la table SQL : sans cela, nous risquons d'avoir deux, voire trois soucis ! 🤖

Pour avoir un espace membre correct, il faut que vos membres puissent renseigner un certain nombre de champs avec leurs informations personnelles, comme leur adresse de messagerie instantanée, leur signature (si jamais votre site dispose d'un forum), etc.

La table sera donc assez conséquente, nous allons vous donner le code SQL pour créer la table nécessaire à ce tutoriel.



Selon l'organisation de votre site, vous aurez peut-être besoin d'avoir d'autres champs dans votre table membres, rien ne vous empêche d'en rajouter ! 😊

Voici le code de la table.

Code : SQL

```
CREATE TABLE `membres` (
  `membre_id` int(11) NOT NULL auto_increment,
  `membre_pseudo` varchar(32) NOT NULL,
  `membre_mdp` varchar(40) NOT NULL,
  `membre_mail` varchar(100) NOT NULL,
  `membre_inscription` bigint(20) NOT NULL,
  `membre_naissance` varchar(11) NOT NULL,
  `membre_msn` varchar(255) NOT NULL,
  `membre_yahoo` varchar(255) NOT NULL,
  `membre_aim` varchar(255) NOT NULL,
  `membre_localisation` varchar(255) NOT NULL,
  `membre_profession` varchar(255) NOT NULL,
  `membre_avatar` varchar(255) NOT NULL,
  `membre_biographie` text NOT NULL,
  `membre_signature` text NOT NULL,
  `membre_derniere_visite` bigint(20) NOT NULL,
  `membre_banni` int(11) NOT NULL default '0',
  PRIMARY KEY `membre_id` (`membre_id`),
  UNIQUE (`membre_pseudo`),
  UNIQUE (`membre_mail`)
) ENGINE=InnoDB CHARACTER SET utf8 COLLATE utf8_general_ci
AUTO_INCREMENT=1
```

Les noms des champs sont explicites, je n'aurai pas besoin de vous faire un dessin. 😊

Je voudrais juste faire une parenthèse sur trois champs : les champs `membre_mdp`, `membre_banni` et `membre_derniere_visite`.

- `membre_mdp` : il contiendra un md5, donc normalement 32 caractères suffisent ; j'en ai mis 40 pour ceux qui veulent

utiliser une fonction de hash plus grosse (pensez à adapter à votre besoin).

- `membre_banni` : si ce champ vaut 0, le membre n'est pas banni, il pourra donc se connecter. En revanche, si ce champ vaut 1, le membre est banni et il lui sera impossible de se connecter.
- `membre_derniere_visite` : ce champ sera mis à jour avec le timestamp de la dernière visite du membre.

C'est tout pour la table SQL, créez-la puis commençons ! 😊

Zzzzzz... Quoi ?! Vous dormez ?? Allez, debout tout de suite !

Hum... Maintenant que vous savez à quoi vous attendre, nous vous proposons de passer au chapitre suivant qui sera déjà plus intéressant.

Les bases du système

Avant de coder le système proprement dit, on a besoin de préparer un certain nombre de pages qui reviendront par la suite pas mal de fois. 😊

Le système n'est pas très compliqué, mais il est quand même nécessaire d'avoir une bonne organisation. Ce chapitre sert à cela, nous allons vous donner les fonctions, pages à inclure et les astuces que nous utiliserons tout au long de ce tutoriel. Bien sûr, vous n'êtes pas obligés de procéder comme nous, mais nous allons quand même vous donner un exemple d'organisation.

Des pages bien utiles

Après la théorie, un peu de pratique avant le système lui-même. 😊

Afin de ne pas devoir refaire plusieurs dizaines de fois les mêmes actions, nous allons ici créer des fonctions qui vont nous permettre de gagner du temps, et, en plus, vos scripts n'en seront que plus clairs ! 😊

Chacune des pages que nous vous proposerons contiendra un entête avec le nom de la page, et une courte description, plus quelques informations utiles.



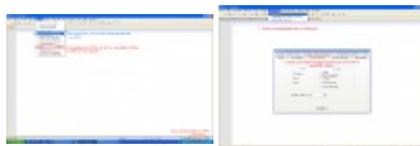
Je suppose que votre site est totalement vide, et que donc nous allons tout créer ensemble.

De plus, je vous informe que nous encoderons toutes nos pages, et les afficherons avec la norme UTF-8.

Dans Notepad++, l'encodage est indiqué en bas à droite (voir image ci-après) et il se modifie dans Format (voir image aussi).

Enfin, il est possible de paramétrer Notepad++ pour avoir toutes vos pages par défaut en UTF-8.

Je mets mes images en miniature car les gros formats prennent beaucoup de place : pour les voir en taille réelle, vous savez quoi faire. 😊



Bon, passons à la suite.

Dans ce chapitre, nous allons vous donner pas mal de codes tout faits car ils ne sont pas passionnants, et que c'est juste pour nous faciliter la vie ; dans les chapitres qui suivront, nous ne vous donnerons pas les codes mâchés, il faudra réfléchir un peu (même si nous donnons les solutions, si vous voulez apprendre, il n'y a pas de secret, il faut bosser, donc, ne recopiez pas bêtement comme des moutons).

information.php

À la racine du site ou de votre dossier de test, vous allez créer une page *information.php* qui va contenir ceci :

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page information.php

Gère les informations (page incluse).

Quelques indications : (Utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----
```

```

Liste des informations/erreurs :
-----
Erreur interne
-----
*/

if(!isset($informations))
{
    $informations = Array(/*Erreur*/
        true,
        'Erreur',
        'Une erreur interne est survenue...',
        '',
        ROOTPATH.'/index.php',
        3
    );
}

if($informations[0] === true) $type = 'erreur';
else $type = 'information';
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" >
    <head>
        <title><?php echo $informations[1]; ?> : <?php echo TITRESITE; ?
    </title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
    />
        <meta name="language" content="fr" />
        <meta http-equiv="Refresh" content="<?php echo $informations[5]; ?
    >;url=<?php echo $informations[4]; ?>">
        <link rel="stylesheet" title="Design" href="<?php echo ROOTPATH; ?
    >/design.css" type="text/css" media="screen" />
    </head>

    <body>
        <div id="info">
            <div id="<?php echo $type; ?>"><?php echo $informations[2]; ?>
            Redirection en cours...<br/>
            <a href="<?php echo $informations[4]; ?>">Cliquez ici si vous ne
            voulez pas attendre...</a><?php echo $informations[3]; ?></div>
        </div>
    </body>
</html>
<?php
unset($informations);
?>

```

Cette page sera incluse pour afficher des informations / erreurs éventuelles, elle reçoit un array `$informations`, et en affiche le contenu ; une erreur générique est mise en place au cas où. 😊

Ne vous souciez pas de `ROOTPATH` ni de `TITRESITE` pour le moment, nous allons y revenir. 😊

index.php

Bon et puisque nous avons attaqué la racine, continuons sur cette lancée : vous allez créer une page `index.php`.

Code : PHP

```

<?php
/*
Neoterranos & LkY
Page index.php

```

```

Index du site.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Aucune information/erreur
-----
*/

session_start();
header('Content-type: text/html; charset=utf-8');
include('includes/config.php');

/*****Actualisation de la session...*****/

include('includes/fonctions.php');
connexionbdd();
actualiser_session();

/*****Fin actualisation de session...*****/

/*****Entête et titre de page*****/

$titre = 'Inscription';

include('includes/haut.php'); //contient le doctype, et head.

/*****Fin entête et titre*****/
?>

<div id="colonne_gauche">
<?php
include('includes/colg.php');
?>
</div>

<div id="contenu">
<div id="map">
<a href="index.php">Accueil</a>
</div>

<h1>Bienvenue sur mon super site !</h1>
<p>Ce site parlera de ... et est ouvert à tous.
Cependant, faut payer pour <a
href="membres/inscription.php">s'inscrire</a> mouhahaha !

Le Webmaster
</p>
</div>

<?php
include('includes/bas.php');
mysql_close();
?>

```

Comme vous le devinez, nous allons très prochainement créer `haut.php`, `bas.php`, `colg.php`, `fonctions.php`, `config.php`, etc. 😊

Concernant `colg`, je me permets une parenthèse : dans le design que nous vous proposons, nous nous basons sur le design du SdZ qui a le mérite d'être très simple, joli et facile d'accès ; `colg` représente donc une colonne gauche - d'une largeur égale à 15 % de la largeur de la page - qui contient des informations ou d'autres choses qui ne changeront pas (qui seront les mêmes sur

toutes les pages).

Si cette idée ne vous convient pas, n'hésitez pas à supprimer cette colonne et à modifier le CSS en conséquence. 😊

fonctions.php

À la racine de votre site ou de votre dossier de test, créez un dossier nommé *includes*.

Dans ce dossier, créez une page que nous allons nommer *fonctions.php*, elle contiendra toutes nos fonctions faites maison. 😊

Nous allons d'abord créer une fonction SQL pour se faciliter la vie ; je la mets directement, car ce n'est pas difficile à comprendre 😊 (en plus, le code est commenté 😊).

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page fonctions.php

Contient quelques fonctions globales.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions donnÃ©es)

Liste des fonctions :
-----
sqlquery($requete,$number)
connexionbdd()
actualiser_session()
vider_cookie()
-----

Liste des informations/erreurs :
-----
Mot de passe de session incorrect
Mot de passe de cookie incorrect
L'id de cookie est incorrect
-----
*/

function sqlquery($requete, $number)
{
    $query = mysql_query($requete) or exit('Erreur SQL :
'.mysql_error().' Ligne : '.__LINE__'.'); //requête
    queries();

    /*
    Deux cas possibles ici :
    Soit on sait qu'on a qu'une seule entrée qui sera
    retournée par SQL, donc on met $number à 1
    Soit on ne sait pas combien seront retournées,
    on met alors $number à 2.
    */

    if($number == 1)
    {
        $query1 = mysql_fetch_assoc($query);
        mysql_free_result($query);
        /*mysql_free_result($query) libère le contenu de $query, je
        le fais par principe, mais c'est pas indispensable.*/
        return $query1;
    }

    else if($number == 2)
    {
        while($query1 = mysql_fetch_assoc($query))
        {
```

```

        $query2[] = $query1;
        /*On met $query1 qui est un array dans $query2 qui
        est un array. Ca fait un array d'arrays :o*/
    }
    mysql_free_result($query);
    return $query2;
}

else //Erreur
{
    exit('Argument de sqlquery non renseigné ou incorrect.');
```

Comme ça, nous économisons du temps, car nous n'avons plus à taper tout cela à chaque fois. 😊

Dans la fonction `sqlquery()`, nous utilisons une fonction nommée `queries()`, il faut donc la définir :

Code : PHP

```

<?php
function queries($num = 1)
{
    global $queries;
    $queries = $queries + intval($num);
}
?>
```

Nous allons maintenant réaliser une fonction pour nous connecter en une seule ligne (😊) à la base de données.

Code : PHP

```

<?php
function connexionbdd()
{
    //Définition des variables de connexion à la base de données
    $bd_nom_serveur='localhost';
    $bd_login='root';
    $bd_mot_de_passe='';
    $bd_nom_bd='espace_membre';

    //Connexion à la base de données
    mysql_connect($bd_nom_serveur, $bd_login, $bd_mot_de_passe);
    mysql_select_db($bd_nom_bd);
    mysql_query("set names 'utf8'");
}
?>
```

Si cette fonction est appelée, la connexion à la base de données se fait automatiquement.

La ligne `<?php mysql_query("set names 'utf8'"); ?>` définit l'encodage des transferts entre la base de données et les pages PHP (donc Apache).

On peut passer à `actualiser_session()`.

Code : PHP

```

<?php
function actualiser_session()
{
    if(isset($_SESSION['membre_id']) && intval($_SESSION['membre_id'])
    != 0) //Vérification id
    {
        //utilisation de la fonction sqlquery, on sait qu'on aura qu'un
```

```

résultat car l'id d'un membre est unique.
$return = sqlquery("SELECT membre_id, membre_pseudo, membre_mdp
FROM membres WHERE membre_id = ".intval($_SESSION['membre_id']), 1);

//Si la requête a un résultat (c'est-à-dire si l'id existe dans
la table membres)
if(isset($return['membre_pseudo']) && $return['membre_pseudo'] !=
'')
{
    if($_SESSION['membre_mdp'] != $return['membre_mdp'])
    {
        //Dehors vilain pas beau !
        $informations = Array(/*Mot de passe de session incorrect*/
            true,
            'Session invalide',
            'Le mot de passe de votre session est incorrect, vous devez
vous reconnecter.',
            '',
            'membres/connexion.php',
            3
        );
        require_once('../information.php');
        vider_cookie();
        session_destroy();
        exit();
    }

    else
    {
        //Validation de la session.
        $_SESSION['membre_id'] = $return['membre_id'];
        $_SESSION['membre_pseudo'] = $return['membre_pseudo'];
        $_SESSION['membre_mdp'] = $return['membre_mdp'];
    }
}

else //On vérifie les cookies et sinon pas de session
{
    if(isset($_COOKIE['membre_id']) && isset($_COOKIE['membre_mdp']))
    //S'il en manque un, pas de session.
    {
        if(intval($_COOKIE['membre_id']) != 0)
        {
            //idem qu'avec les $_SESSION
            $return = sqlquery("SELECT membre_id, membre_pseudo, membre_mdp
FROM membres WHERE membre_id = ".intval($_COOKIE['membre_id']), 1);

            if(isset($return['membre_pseudo']) && $return['membre_pseudo']
!= '')
            {
                if($_COOKIE['membre_mdp'] != $return['membre_mdp'])
                {
                    //Dehors vilain tout moche !
                    $informations = Array(/*Mot de passe de cookie incorrect*/
                        true,
                        'Mot de passe cookie erroné',
                        'Le mot de passe conservé sur votre cookie est incorrect
vous devez vous reconnecter.',
                        '',
                        'membres/connexion.php',
                        3
                    );
                    require_once('../information.php');
                    vider_cookie();
                    session_destroy();
                    exit();
                }
            }

            else

```

```

    {
        //Bienvenue :D
        $_SESSION['membre_id'] = $retour['membre_id'];
        $_SESSION['membre_pseudo'] = $retour['membre_pseudo'];
        $_SESSION['membre_mdp'] = $retour['membre_mdp'];
    }
}

else //cookie invalide, erreur plus suppression des cookies.
{
    $informations = Array(/*L'id de cookie est incorrect*/
        true,
        'Cookie invalide',
        'Le cookie conservant votre id est corrompu, il va donc
être détruit vous devez vous reconnecter.',
        '',
        'membres/connexion.php',
        3
    );
    require_once('../information.php');
    vider_cookie();
    session_destroy();
    exit();
}

else
{
    //Fonction de suppression de toutes les variables de cookie.
    if(isset($_SESSION['membre_id'])) unset($_SESSION['membre_id']);
    vider_cookie();
}
}

function vider_cookie()
{
    foreach($_COOKIE as $cle => $element)
    {
        setcookie($cle, '', time()-3600);
    }
}
?>

```

Explications

Cette fonction doit actualiser les sessions, donc d'abord vérifier s'il existe une session.

On vérifie la variable `$_SESSION['membre_id']`, car a priori, l'id d'un membre ne changera jamais, et seuls les membres connectés ont un id.

On vérifie que `$_SESSION['membre_id']` contient un nombre différent de 0 (via `intval()`), une fonction très intéressante qui retourne la valeur numérique entière d'une variable ; si la variable est une chaîne commençant par une lettre, on aura un 0, c'est ici ce qui nous intéresse 😊).

Si cette vérification se passe bien, on appelle alors SQL (et on incrémente au passage une variable globale `$queries` pour le nombre de requêtes) pour avoir les informations désirées et on vérifie que le pseudo retourné n'est pas vide, ce qui sert à deux choses : déjà à être sûrs que le membre existe, ensuite qu'il a un pseudo (normalement oui, mais vous n'êtes jamais à l'abri d'un bug).

Notez l'utilisation de `intval()` dans la requête SQL. `intval()` force la variable à être un nombre, il n'y a donc pas de risque pour la sécurité de la requête.



Pour plus d'informations sur ces histoires de sécurité dans les requêtes SQL, je vous invite à vous rendre [ici](#) (notez à quel point je suis feignant, je ne vous explique même pas ça 😊).

Bon, alors : si le membre a un pseudo, on compare les mots de passe en session et venant de la bdd, et si tout roule, on valide la

session du membre ; sinon, on inclut `information.php` et on affiche une erreur. 😊

Si le membre n'a pas de session, on va dans le `else`, on vérifie donc s'il a des cookies et s'ils sont valides, la procédure ressemble assez à celle des sessions, sauf que si le cookie id est incorrect, on génère une autre erreur.

Enfin, si le membre n'a pas de session valide, ni de cookie valide, on détruit la variable `$_SESSION['membre_id']` (si elle existait), et on détruit les cookies.

La fonction `vider_cookie()` est facile à comprendre, elle parcourt les cookies avec un `foreach` et les détruit tous (mouhahahahaha).

Voilà. 😊

Donc, dans `fonctions.php`, nous avons tout ça, normalement :

Secret (cliquez pour afficher)

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page fonctions.php

Contient quelques fonctions globales.

Quelques indications : (utiliser l'outil de recherche et
rechercher les mentions donnÃ©es)

Liste des fonctions :
-----
sqlquery($requete,$number)
connexionbdd()
actualiser_session()
vider_cookie()
-----

Liste des informations/erreurs :
-----
Mot de passe de session incorrect
Mot de passe de cookie incorrect
L'id de cookie est incorrect
-----
*/

function sqlquery($requete, $number)
{
    $query = mysql_query($requete) or exit('Erreur SQL :
'.mysql_error().' Ligne : '. __LINE__ .'.'); //requête
    queries();

    /*
    Deux cas possibles ici :
    Soit on sait qu'on a qu'une seule entrée qui sera
    retournée par SQL, donc on met $number à 1
    Soit on ne sait pas combien seront retournées,
    on met alors $number à 2.
    */

    if($number == 1)
    {
        $query1 = mysql_fetch_assoc($query);
        mysql_free_result($query);
        /*mysql_free_result($query) libère le contenu de $query, je
        le fais par principe, mais c'est pas indispensable.*/
        return $query1;
    }

    else if($number == 2)
```

```

{
    while($query1 = mysql_fetch_assoc($query))
    {
        $query2[] = $query1;
        /*On met $query1 qui est un array dans $query2 qui
est un array. Ca fait un array d'arrays :o*/
    }
    mysql_free_result($query);
    return $query2;
}

else //Erreur
{
    exit('Argument de sqlquery non renseigné ou incorrect.');
```

```

}

function queries($num = 1)
{
    global $queries;
    $queries = $queries + intval($num);
}

function connexionbdd()
{
    //Définition des variables de connexion à la base de données
    $bd_nom_serveur='localhost';
    $bd_login='root';
    $bd_mot_de_passe='';
    $bd_nom_bd='espace_membre';

    //Connexion à la base de données
    mysql_connect($bd_nom_serveur, $bd_login, $bd_mot_de_passe);
    mysql_select_db($bd_nom_bd);
    mysql_query("set names 'utf8'");
}

function actualiser_session()
{
    if(isset($_SESSION['membre_id']) &&
intval($_SESSION['membre_id']) != 0) //Vérification id
    {
        //utilisation de la fonction sqlquery, on sait qu'on aura qu'un
résultat car l'id d'un membre est unique.
        $retour = sqlquery("SELECT membre_id, membre_pseudo, membre_mdp
FROM membres WHERE membre_id = ".intval($_SESSION['membre_id']),
1);

        //Si la requête a un résultat (id est : si l'id existe dans la
table membres)
        if(isset($retour['membre_pseudo']) && $retour['membre_pseudo']
!= '')
        {
            if($_SESSION['membre_mdp'] != $retour['membre_mdp'])
            {
                //Dehors vilain pas beau !
                $informations = Array(/*Mot de passe de session incorrect*/
true,
'Session invalide',
'Le mot de passe de votre session est incorrect, vous
devez vous reconnecter.',
'',
'membres/connexion.php',
3
);
                require_once('../information.php');
                vider_cookie();
                session_destroy();
                exit();
            }
        }
    }
}

```

```

else
{
    //Validation de la session.
    $_SESSION['membre_id'] = $retour['membre_id'];
    $_SESSION['membre_pseudo'] = $retour['membre_pseudo'];
    $_SESSION['membre_mdp'] = $retour['membre_mdp'];
}
}
}

else //On vérifie les cookies et sinon pas de session
{
    if(isset($_COOKIE['membre_id']) &&
isset($_COOKIE['membre_mdp'])) //S'il en manque un, pas de
session.
    {
        if(intval($_COOKIE['membre_id']) != 0)
        {
            //idem qu'avec les $_SESSION
            $retour = sqlquery("SELECT membre_id, membre_pseudo,
membre_mdp FROM membres WHERE membre_id =
".intval($_COOKIE['membre_id']), 1);

            if(isset($retour['membre_pseudo']) && $retour['membre_pseudo']
!= '')
            {
                if($_COOKIE['membre_mdp'] != $retour['membre_mdp'])
                {
                    //Dehors vilain tout moche !
                    $informations = Array(/*Mot de passe de cookie incorrect*/
true,
'Mot de passe cookie erroné',
'Le mot de passe conservé sur votre cookie est
incorrect vous devez vous reconnecter.',
'',
'membres/connexion.php',
3
);
                    require_once('../information.php');
                    vider_cookie();
                    session_destroy();
                    exit();
                }

                else
                {
                    //Bienvenue :D
                    $_SESSION['membre_id'] = $retour['membre_id'];
                    $_SESSION['membre_pseudo'] = $retour['membre_pseudo'];
                    $_SESSION['membre_mdp'] = $retour['membre_mdp'];
                }
            }
        }

        else //cookie invalide, erreur plus suppression des cookies.
        {
            $informations = Array(/*L'id de cookie est incorrect*/
true,
'Cookie invalide',
'Le cookie conservant votre id est corrompu, il va donc
être détruit vous devez vous reconnecter.',
'',
'membres/connexion.php',
3
);
            require_once('../information.php');
            vider_cookie();
            session_destroy();
            exit();
        }
    }
}

```

```

    }
}

else
{
    //Fonction de suppression de toutes les variables de cookie.
    if(isset($_SESSION['membre_id']))
unset($_SESSION['membre_id']);
    vider_cookie();
}
}
}

function vider_cookie()
{
    foreach($_COOKIE as $cle => $element)
    {
        setcookie($cle, '', time()-3600);
    }
}
?>

```

config.php

Toujours dans notre dossier *includes*, nous allons créer un fichier *config.php*.

Code : PHP

```

<?php
/*
Neoterranos & LkY
Page config.php

Deux define et la variable de queries.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Aucune information/erreur
-----
*/

define('ROOTPATH', 'http://'.$_SERVER['HTTP_HOST'], true);
define('TITRESITE', 'Mon site de test', true);
$queries = 0;
?>

```

Explications

Je me suis demandé pendant un bout de temps comment j'allais m'organiser pour la gestion des liens des pages incluses. Plutôt que d'utiliser une fonction pour repérer où je suis et bosser totalement en relatif, j'ai pensé à un chemin absolu, mais ce genre de chemin a un défaut : si on change de nom de domaine, on doit le modifier ; du coup, j'ai voulu créer une variable de configuration `rootpath` dans un fichier de configuration et l'inclure partout où j'en ai besoin, pour n'avoir qu'une et une seule modification de NDD à faire.



Oui, mais pourquoi ne pas trouver une solution pour ne pas avoir à modifier ROOTPATH ?

J'y viens ; dans l'objectif de faire quelque chose sur lequel je revienne le moins souvent possible, j'ai (enfin, c'est LkY, mais on va dire que c'est moi 🤖) trouvé une variable serveur qui me convenait parfaitement : `$_SERVER['HTTP_HOST']`.

Essayez ce script en local, sur votre ftp, vous comprendrez l'intérêt de cette variable :

Code : PHP

```
<?php
echo $_SERVER['HTTP_HOST'];
?>
```

Il ne restait plus qu'à mettre le `http://` devant et j'avais mon lien absolu.



Pensez cependant à vérifier sur quoi pointe ROOTPATH : en effet, si vous êtes en local, il faudra peut-être modifier celui-ci pour qu'il pointe sur votre dossier de test !!

Exemple : mon ROOTPATH en local ressemble au code qui suit.

Code : PHP

```
<?php
define(ROOTPATH,
'http://'.$_SERVER['HTTP_HOST'].'/Site/Sdz/tuto/tuto-membres',
true);
?>
```

TITRESITE n'est pas très utile, et on pourrait la mettre ailleurs, mais j'ai décidé de la mettre là. 😊

`$queries` est créé sur chaque page appelant `config.php`, elle sert à enregistrer le nombre de requêtes effectuées sur une page (au passage : pensez à la portée des variables).

D'autres pages à inclure

Nous allons ensuite préparer d'autres pages (il y a déjà `fonctions.php`) à inclure au début de vos scripts : de cette manière, vos scripts seront plus propres, et puis si vous décidez de faire un changement, vous n'aurez pas à modifier toutes vos pages ! 😊

Nous mettrons ces pages dans le dossier `includes` précédemment créé.

La page haut.php

Cette page contiendra le code minimal HTML nécessaire à tout site Internet.

La voici. 😊

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page haut.php

Page incluse créant le doctype etc etc.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
```

```

Aucune fonction
-----

Liste des informations/erreurs :
-----
Aucune information/erreur
-----
*/
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" >
  <head>
    <?php
      /*****Vérification du titre...*****/

      if(isset($titre) && trim($titre) != '')
        $titre = $titre.' : '.TITRESITE;

      else
        $titre = TITRESITE;

      /*****Fin vérification titre...*****/
    ?>
    <title><?php echo $titre; ?></title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
  />
    <meta name="language" content="fr" />
    <link rel="stylesheet" title="Design" href="<?php echo ROOTPATH; ?
  >/design.css" type="text/css" media="screen" />
  </head>

  <body>
    <div id="banniere">
      <a href="<?php echo ROOTPATH; ?>/index.php"></a>
    </div>

    <div id="menu">
      <div id="menu_gauche">
        <!-- Vide, mettez-y les liens qui ne dépendent pas du statut
du membre (connecté ou non) -->
      </div>

      <div id="menu_droite">
        <?php
        if(isset($_SESSION['membre_id']))
        {
          ?>
          <a href="<?php echo ROOTPATH; ?>/membres/moncompte.php">Gérer
mon compte</a>   <a href="<?php echo ROOTPATH; ?
>/membres/deconnexion.php">Se déconnecter</a>
          <?php
          }

        else
        {
          ?>
          <a href="<?php echo ROOTPATH; ?
>/membres/inscription.php">Inscription</a>   <a href="<?php echo
ROOTPATH; ?>/membres/connexion.php">Connexion</a>
          <?php
          }
        }
      </div>
    </div>
  </div>

```

Rien de bien particulier ici, à part le titre, le menu et la bannière. 😊

Pour le titre (dans la balise `<title>`), on vérifie simplement que la variable existe : si oui, on l'affiche.

colg.php

Toujours dans le dossier *includes*, créez le fichier *colg.php* et mettez-y ce que vous voulez, il ne nous servira pas pour le moment. 😊

Pour ma part, j'y ai mis ça :

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page colg.php

La colonne de gauche de votre site.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Aucune information/erreur
-----
*/
?>
<div>Hello World !</div>
```

Page bas.php

Voici le code de la page *bas.php*, on affiche le footer (avec un copyright 😎) et on ferme les balises `body` et `html` .

Voici la page. 😊

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page bas.php

Pied de page.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----
```

```

Liste des informations/erreurs :
-----
Aucune information/erreur
-----
*/
?>
<div id="footer">
  Site réalisé par Neoterranos et LkY.<br/>
  <?php echo $queries; ?> requête(s).
</div>
</body>
</html>

```



Il se peut que nous rajoutions par la suite d'autres choses dans nos pages haut et bas, donc gardez-les sous la main !



Le design

Bien que je vous aie recommandé de faire un design pour votre système, vous en avez sûrement un pour votre site. Mais si vous faites ce tuto uniquement pour vous entraîner et apprendre comment mettre en place ce type de système (même s'il est toujours recommandé d'essayer avant de pomper les codes 🤖), je vous ai concocté un petit design simple et propre ! 😊

À la racine du site, créez *design.css* :

Code : CSS

```

body {
  background-color: #CDF0F0;
  width:auto;
  margin-left: 1%;
  margin-right: 1%;
  font-family: verdana, serif;
  font-size:12px
}

#banniere {
  height:auto;
  width:auto;
  margin-left: 0.25%;
}

#banniere a img {
  width: 99.5%;
  height: auto;
}

#menu {
  padding-top: 7px;
  padding-bottom: 3px;
  background-color: white;
  width: auto;
  height: 20px;
  border: 1px solid #2499cc;
}

#colonne_gauche {
  background-color: white;
  width:15%;
  margin-top: 5px;
  margin-bottom: 10px;
  padding:0;
  border: 1px solid #2499cc;
  float: left;
}

```

```
}

#menu_droite {
  text-align: right;
}

#contenu {
  background-color: white;
  margin-left: 16%;
  margin-bottom: 10px;
  margin-right: 0;
  margin-top: 5px;
  padding: 5px;
  width: auto;
  border: 1px solid #2499cc;
  min-height: 200px;
}

#footer {
  background-color: white;
  clear: both;
  margin: 0;
  padding: 5px;
  width: auto;
  text-align: center;
  border: 1px solid #2499cc;
}

#info {
  width: 40%;
  position: absolute;
  top: 42%;
  left: 30%;
  border: 1px solid black;
  background-color: #CFEDF4;
}

#erreur {
  text-align: center;
  color: #EE0808;
}

#information {
  text-align: center;
  color: #08EE08;
}

/* -----
----- */

h1 {
  text-align: center;
  text-decoration: underline;
  color: #3080D0;
  font-family: "trebuchet ms", verdana, sans-serif;
  letter-spacing: 1.2px;
  font-variant: small-caps;
  font-weight: bold;
  font-size: 22px;
  margin-top: 0;
}

a, a:visited {
  color: #4497e6;
  text-decoration: none;
}

a:hover, a:focus {
  text-decoration: underline;
  color: #76b4ee;
}
```

```
}  
  
a img {  
  border: none;  
}
```

Voilà ! 😊

On en a fini avec ce chapitre. 😊

Beaucoup de ces pages vont servir presque tout le temps, et certaines seront sujettes à des modifications fréquentes, donc ne les zappez pas.

Allez, on attaque l'inscription.

Inscription

Nous allons enfin pouvoir commencer à coder nos pages.

Ici ce n'est pas très dur, c'est tout le formulaire d'inscription, alors on y va. 😊

Avant de commencer

Avant de commencer à coder notre page, vous allez, à la racine de votre site ou de votre dossier de test, créer un dossier `membres`, et y créer la page `inscription.php` ; ensuite, nous allons réfléchir ensemble à ce que nous voulons y insérer.

Nous voulons un formulaire contenant tous les éléments suivants :

- pseudo ;
- mot de passe ;
- confirmation du mot de passe ;
- e-mail ;
- confirmation de l'e-mail ;
- date de naissance ;
- système de protection anti-robot.

On pourrait se limiter à cela, mais par principe, on veut un système qui marche bien, donc on va réfléchir encore plus loin (en fait, pas tant que ça quand même). 😊

Nous voulons que cette page serve à l'inscription ; or, un membre connecté ne peut prétendre vouloir s'inscrire.

On va donc limiter l'inscription aux membres non connectés.

La première idée qui nous viendrait, c'est de couper les sessions sur cette page en ne mettant pas de `session_start()` ; .

Cependant, pour une raison pratique, nous n'allons pas le faire, car nous allons utiliser les sessions en vue de pré-remplir les champs corrects si l'inscription échoue pour une raison quelconque, cela évitera au membre de tout retaper.

On va donc se baser sur une variable de session particulière qui n'existera pas si l'on n'est pas connecté :

```
$_SESSION['membre_id'] .
```

Donc, cette page est interdite aux membres connectés.

Par ailleurs, nous voulons éviter que les bugs puissent permettre à un membre connecté de s'inscrire, ou empêchent un membre non connecté de s'inscrire.

On va donc, avant de se pencher sur l'inscription, se pencher sur l'actualisation de session pour qu'il n'y ait plus que deux cas possibles :

1°) le membre a une session valide, il ne peut pas s'inscrire ;

2°) le membre n'a pas de session valide (il n'est pas connecté), il peut donc s'inscrire.

En fait, se pencher dessus, ça ne veut pas dire grand chose : on va utiliser la fonction mise en place précédemment, donc rien de stupéfiant.

On y va. 😊

Le formulaire

Ça y est ! Ça y est ! 😊

On commence !! 😊

Bien : pour permettre une plus grande clarté dans nos formulaires, nous allons utiliser la balise `<label>` avec la classe `float` que nous rajouterons dans notre CSS quand le formulaire sera fini.

On commence (enfin) à coder.

Vous allez, dans l'ordre, ouvrir les sessions, configurer avec un header l'encodage en UTF-8 et appeler en `include` le fichier de configuration.

En dehors de l'encodage, vous savez tous faire ce que je vous demande.

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page inscription.php
```

```

Permet de s'inscrire.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Aucune information/erreur
-----
*/

session_start();
header('Content-type: text/html; charset=utf-8');
include('../includes/config.php');
?>

```

Avant de faire quoi que ce soit, nous allons actualiser les sessions. Ces deux lignes de code sont assez faciles aussi. 😊

Code : PHP

```

<?php
/*****Actualisation de la session...*****/

include('../includes/fonctions.php');
connexionbdd();
actualiser_session();

/*****Fin actualisation de session...*****/
?>

```

Maintenant, je veux que vous fassiez ce que je vous avais annoncé juste avant : vous allez vérifier si le visiteur est connecté ; s'il l'est, vous l'envoyez purement et simplement sur l'index (on ne va pas leur tenir la jambe avec un message d'erreur. 😊).

Prenez votre temps, n'affichez le code (qui utilise une fonction que vous devriez connaître) que lorsque vous aurez fini. 😊

Secret (cliquez pour afficher)

Code : PHP

```

<?php
if(isset($_SESSION['membre_id']))
{
header('Location: '.ROOTPATH.'/index.php');
exit();
}
?>

```

Bon : on a viré les intrus ; et si on s'occupait de ceux qui veulent venir ? 😊

Code : PHP

```

<?php

```

```

/*****Entête et titre de page*****/
$titre = 'Inscription 1/2';
include('../includes/haut.php'); //contient le doctype, et head.
/*****Fin entête et titre*****/
?>

```



Faites attention à inclure `config.php` (ce que l'on a fait ici, en gros 😊) avant d'inclure `haut.php` car nous utilisons la constante `TITRESITE` dans `haut.php`, et si elle n'est pas définie... ça risque de ne pas aller ! 😊

Les trois fichiers étant inclus, nous pouvons commencer notre formulaire proprement dit. 😊

Commençons par la colonne de gauche :

Code : PHP

```

<!--Colonne gauche-->
<div id="colonne_gauche">
  <?php
  include('../includes/colg.php');
  ?>
</div>

```

Enfin, tapons notre formulaire (mais pas trop fort, parce qu'on va en avoir besoin). 😊

Ici vous allez donc bosser un peu : je vous rappelle que la liste des choses nécessaires est en haut, n'oubliez pas le *sitemap*, ni une petite présentation de votre formulaire d'inscription. 😊

Bon ben moi, j'vais me faire un café... 😊

Secret (cliquez pour afficher)

Code : HTML

```

<!--Contenu-->
<div id="contenu">
  <div id="map">
    <a href="../index.php">Accueil</a> => <a
href="inscription.php">Inscription 1/2</a>
  </div>

  <h1>Formulaire d'inscription</h1>
  <p>Bienvenue sur la page d'inscription de mon site !<br/>
  Merci de remplir ces champs pour continuer.</p>
  <form action="trait-inscription.php" method="post"
name="Inscription">
    <fieldset><legend>Identifiants</legend>
    <label for="pseudo" class="float">Pseudo :</label> <input
type="text" name="pseudo" id="pseudo" size="30" /> <em>(compris
entre 3 et 32 caractères)</em><br />
    <label for="mdp" class="float">Mot de passe :</label> <input
type="password" name="mdp" id="mdp" size="30" /> <em>(compris
entre 4 et 50 caractères)</em><br />
    <label for="mdp_verif" class="float">Mot de passe
(vérification) :</label> <input type="password" name="mdp_verif"
id="mdp_verif" size="30" /><br />
    <label for="mail" class="float">Mail :</label> <input
type="text" name="mail" id="mail" size="30" /> <br />

```

```

        <label for="mail_verif" class="float">Mail (vérification)
    :</label> <input type="text" name="mail_verif" id="mail_verif"
    size="30" /><br />
        <label for="date_naissance" class="float">Date de naissance
    :</label> <input type="text" name="date_naissance"
    id="date_naissance" size="30" /> <em>(format JJ/MM/AAAA)</em><br/>
        <div class="center"><input type="submit" value="Inscription"
    /></div>
    </fieldset>
</form>
</div>

```

Notre page est terminée ! Allez hop hop hop, on passe au traitement ! (ouuuh il a l'air pressé, lui... 🤔)



Mais... Mais.. Et le système anti-robot ?? Tu me prends pour un idiot ?

Rah zut ! Ils s'en sont rappelés !! 🤔 LkY, au secours, à l'aide !!!

Bon, ben puisque vous y tenez tant, nous allons attaquer ce problème dans la sous-partie suivante (faut bien se faire désirer, non ? 🤔).

Avant ça, n'oubliez pas que l'on a créé une dernière page, la page `bas.php`, il faut bien que l'on ferme nos pages, non ? 😊

Code : PHP

```

<!--bas-->
<?php
include ('../includes/bas.php');
mysql_close();
?>

```

Et voilà le travail, on peut maintenant sauvegarder notre page et afficher le tout ! 😊

Mais avant cela, n'oubliez pas de modifier votre CSS, hein ?

Code : CSS

```

.float {
display: block;
width: 200px;
float: left;
}

.center {
text-align: center;
}

```

Allez hop, à la suite (enfin, entre autres), le captcha !

Charte et système anti-robot

Bon : alors déjà, avant de nous lancer ensemble dans la création d'un système permettant de bloquer les robots publicitaires, j'aimerais clarifier deux points.

Premièrement, il n'y a pas de meilleure protection qu'une protection innovante ; les captchas, ce n'est pas mal, mais malheureusement, plus on avance, moins c'est bien car les gens qui créent des bots les améliorent pour contourner les captchas.

Ainsi, je vais vous proposer deux systèmes, que vous pourrez combiner, mais indépendants l'un de l'autre : un système de charte avec QCM à remplir, comme celui qu'il y a eu pendant un moment sur le SdZ, qui sera moins performant je pense, mais qui

bloquera les robots qui n'ont pas été programmés spécialement pour attaquer votre site ; le deuxième système sera un captcha, que je vous invite fortement à modifier comme il vous convient ; je le fais pour vous permettre de vous plonger dans GD et vous entraîner à la manipulation d'images.

Deuxièmement, il n'existe aucune protection incontournable, ne serait-ce que parce que derrière le bot qui vous attaque, il y a un humain pour le créer, donc quelqu'un qui pense et s'adapte, qui peut même, pour contourner le captcha, s'inscrire lui-même et ensuite filer les commandes au robot.

Dans cette logique, je vous invite à ne pas devenir paranoïaque, tout en laissant un niveau de protection correct. Ne vous enfermez pas dans des programmes flash ultra-sophistiqués où il faut faire trois cercles autour de trois carrés (j'ai dit sophistiqué ?), c'est complètement inutile...

Enfin, n'oubliez pas qu'il y a des mal-voyants et des non-voyants qui surfent comme ils peuvent sur le net, et qu'il est indispensable de penser à eux : donc, si captcha vous mettez, vous devez aussi mettre soit un captcha audio, soit un e-mail ou lien pour vous contacter de façon qu'ils puissent, eux aussi, s'inscrire chez vous.

Après ces belles paroles (en toute modestie, je m'épate parfois 🤪), lançons-nous dans le système charte / questions.

Si tu as lu le règlement, tu peux t'inscrire

Bon, au boulot (vous, pas moi 🤪) : vous allez me virer le bouton `submit` , me fermer le `fieldset` , m'en ouvrir un avec comme légende *charte du site et protection anti-robot*, m'inclure un fichier `charte.php` et, à la suite, mettre un système anti-robot (le formulaire, un message d'explication, et l'image avec comme source `captcha.php`) dont on s'occupera plus tard. Fermez enfin le `fieldset` et mettez le bouton `submit` , le tout en une minute et 25 secondes (et douze centièmes 🤪).



Eh eh oh ! Tu vas trop vite là ! Tu peux répéter ?

Ben quoi ? Ce n'est pas encore fini ?! Au boulot, tout de suite !

Voilà la correction :

Secret (cliquez pour afficher)

Code : PHP

```

<!-- Formulaire mis à jour : -->
<form action="trait-inscription.php" method="post"
name="Inscription">
  <fieldset><legend>Identifiants</legend>
  <label for="pseudo" class="float">Pseudo :</label> <input
type="text" name="pseudo" id="pseudo" size="30" /> <em>(compris
entre 3 et 32 caractères)</em><br />
  <label for="mdp" class="float">Mot de passe :</label> <input
type="password" name="mdp" id="mdp" size="30" /> <em>(compris
entre 4 et 50 caractères)</em><br />
  <label for="mdp_verif" class="float">Mot de passe
(vérification) :</label> <input type="password" name="mdp_verif"
id="mdp_verif" size="30" /><br />
  <label for="mail" class="float">Mail :</label> <input
type="text" name="mail" id="mail" size="30" /> <br />
  <label for="mail_verif" class="float">Mail (vérification)
:</label> <input type="text" name="mail_verif" id="mail_verif"
size="30" /><br />
  <label for="date_naissance" class="float">Date de naissance
:</label> <input type="text" name="date_naissance"
id="date_naissance" size="30" /> <em>(format JJ/MM/AAAA)</em><br/>
  </fieldset>
  <fieldset><legend>Charte du site et protection anti-
robot</legend>
  <?php
  include ('../includes/charte.php');
  ?>

  <h1>Système anti-robot :</h1>

```

```

<p>Qu'est-ce que c'est ?<br/>
  Pour lutter contre l'inscription non désirée de robots qui
  publient du contenu non désiré sur les sites web,
  nous avons décidé de mettre en place un système de sécurité
  .<br/>
  Aucun de ces systèmes n'est parfait, mais nous espérons que
  celui-ci, sans vous être inaccessible sera suffisant
  pour lutter contre ces robots.<br/>
  Il est possible que certaines fois, l'image soit trop dure à
  lire ; le cas échéant, actualisez la page jusqu'à avoir une image
  lisible.<br/>
  Si vous êtes dans l'incapacité de lire plusieurs images
  d'affilée, <a href="../contact.php">contactez-nous</a>, nous nous
  occuperons de votre inscription.</p>
  <label for="captcha" class="float">Entrez les 8 caractères
  (majuscules ou chiffres) contenus dans l'image :</label> <input
  type="text" name="captcha" id="captcha"><br/>
  
</fieldset>
<div class="center"><input type="submit" value="Inscription"
/></div>
</form>

```

Laissez tomber `captcha.php` et tout ce qui gravite autour pour le moment, occupons-nous de la charte.

Ah tenez, tant qu'à faire, moi je trouve que ces `input` sont bien laids. 😞

Vous allez (😞) Nous allons améliorer tout ça. 😊

Code : CSS

```

input[type="submit"], input[type="button"] {
  background-color: #CDF0F0;
  color: black;
  border: 1px outset black;
}

input[type="text"]:focus, input[type="password"]:focus {
  background-color: #CDF0F0;
  color: black;
}

```

Je ne pouvais pas vous demander de les connaître, ces propriétés CSS pour ces balises, mais voilà : elles existent. 😊

Sachez cependant qu'Internet Explorer a du mal avec l'indicateur `:focus` et que, du coup, nos champs d'entrée ne seront pas colorés quand on cliquera dessus avec IE 7 ou antérieur (mais FF et Opéra supportent très bien ça 😊), espérons qu'IE8 saura faire de même...).

Ici pas de mystère : on va créer `charte.php` dans les `includes` (si, si, j'vous jure, vous avez un dossier `includes`), et vous allez mettre ce que vous voulez.

Quant à moi, je vous propose ça : soyez indulgents, j'ai écrit ça tard hier et sans aucune inspiration. 🙄

Secret (cliquez pour afficher)

Code : PHP

```

<?php
/*
  Neoterranos & LkY

```

Page charte.php

Contient la charte et génère le QCM qui va avec.

Quelques indications : (utiliser l'outil de recherche et rechercher les mentions données)

Liste des fonctions :

Aucune fonction

Liste des informations/erreurs :

Aucune information/erreur

**/*

?>

```

<div id="charte">
  <h1>La charte de mon site </h1>
  <p>Ce site contient un espace membres, un forum, un système
de news ainsi que d'autres modules.<br/>
  En vous inscrivant, vous reconnaissez avoir pris
connaissance de cette charte et l'avoir acceptée.
  Si tel n'est pas le cas, vous ne pouvez vous
inscrire.</p><br/>

  <div class="chartecat">Règlement Général du site</div><br/>
  <p>Ce site et toutes ses pages sont soumis à la loi
française, par conséquent, il est interdit
d'y tenir des propos ou d'y publier du contenu illégal aux
yeux de la loi, sont considérés illégaux entres
autres les contenus suivants : contenu à caractère raciste,
contenu diffamatoire, contenu incitant à la haine,
à la violence, contenu expliquant comment pirater (i.e. : à
des fins néfastes ou non), contenu violant les droits
d'auteur.<br/>
  À cette liste non exhaustive vient s'ajouter l'interdiction
de publier du contenu à caractère sexuel.<br/>
  Cette liste étant non exhaustive, nous faisons appel à votre
bon sens pour discerner ce que vous pouvez publier
et / ou dire de ce que vous ne pouvez publier / dire.<br/>
  Les propos insultants, dégradants, agressifs ou tout
comportement néfaste à une ambiance correcte sur l'ensemble
du site sont interdits.<br/>
  Le thème de ce site n'est pas restreint, bien que les
technologies de l'information soient le thème principal, libre
à vous de parler de couture si ça vous chante, mais
n'espérez pas trouver autant d'adeptes de la couture que de la
programmation ici.<br/>
  Les forums sont un espace de discussion important pour un
site à caractère communautaire, surtout s'il est centré
sur l'informatique, mais c'est aussi un espace d'entraide,
par conséquent, n'hésitez pas à y poser vos questions si
vous en avez, cependant, pensez à faire une recherche avant
de poster une question, peut-être que la question a déjà été posée
par
un autre membre, et de plus, votre sujet devra avoir un
titre clair et concis.<br/>
  Partout sur le site, vous devrez écrire dans un français
correct, toute forme de langage SMS (abrégé) est interdite.<br/>
  Il est important de noter que pour votre confort, et le
nôtre, le forum est surveillé par une équipe de modération
bénévole, qui peut être amenée à sanctionner tout membre
enfrenant le règlement, ceci allant de l'avertissement
à l'interdiction d'accéder au site.<br/>
  La messagerie privée est, comme son nom l'indique, privée.
Cependant, vous acceptez l'idée que vous, ou votre / vos
interlocuteur(s) puisse(nt), à tout moment, demander à

```

```

l'équipe de modération du site de lire votre échange avec
lui / eux en cas de problème.<br/>
Vous reconnaissez que ce site est la propriété de son
créateur, qui est, par conséquent libre de faire ce
qu'il veut de celui-ci, tout en respectant le caractère
privé des informations que vous, ou tout autre membre, lui donnez
en vous
inscrivant et en utilisant le site.<br/>
Vous êtes donc propriétaire de votre compte et responsable
de celui-ci (ainsi que des propos tenus avec), vous pouvez
à tout moment demander sa suppression. Veuillez noter qu'à
aucun moment, l'équipe du site ne vous demandera votre mot
de passe.<br/>

Fin (ben ouais, vous y mettez ce que vous voulez dans votre
charte, mais à 2h du mat, moi ça me barbe de continuer ;
en plus, c'est pas terrible. ^^ )
</p>
</div>

```

Bon ok ça pue, mais au moins, on peut faire un QCM maintenant, et il ne tient qu'à vous de modifier tout ça. 🤔



J'ai mis en place la charte en HTML et non dans la BDD parce que j'estime qu'un texte comme celui-là n'a pas sa place en BDD : vous pouvez faire comme vous voulez, même créer `charte.code`, mettre tout le contenu formaté en bbcode et l'afficher, mais j'estime ça important de vous désolidariser de SQL quand il s'agit de textes très peu modifiés. Quant au QCM, j'ai préféré le mettre dans la page `charte` au lieu de le mettre en BDD : c'est complètement débile, car la modification est plus longue sur la page que dans une page d'administration, mais ici, on veut un système simple, on retouchera le QCM en partie II, et il finira dans la BDD, c'est promis. 🤔

D'ailleurs, en parlant de QCM, on va s'y attaquer tout de suite. 😊

Premièrement, notre array de questions / réponses justes / réponses fausses, vous pouvez le faire vous-mêmes ; je précise juste que j'ai fait un Array d'Arrays.

Je vous propose ça, mais chacun fait comme il veut. 😊

Secret (cliquez pour afficher)

Code : PHP

```

<div id="qcm">
  <h1>Questionnaire sur la charte </h1>
  <p>Pous nous assurer que vous avez lu le règlement, ou du
moins que vous avez du bon sens,
voici un questionnaire à remplir. (Le remplir signifie
accepter la charte.)
  </p>
<?php
/*QCM CHARTE*/
$questions = Array();
$questions[] = Array('A qui appartient ce site ?', 'Son créateur',
'Vous', 'L\'hébergeur web du site', 'À personne', 'Je ne sais
pas', 1);
$questions[] = Array('Que faut-il faire avant de poser une
question sur les forums ?', 'Rien', 'Demander à son voisin s\'il a
la réponse', 'Une recherche sur le site', 'Consulter Google', 'Je
ne sais pas', 3);
$questions[] = Array('Vous pouvez supprimer votre compte...',
'quand vous voulez', 'quand vous voulez en le demandant à un
administrateur', 'seulement le mardi', 'seulement si vous ne
respectez pas le règlement', 'Je ne sais pas', 2);
$questions[] = Array('Quel titre de sujet convient ?', 'PC en
carton plante tout le temps', 'Grrrrrrrrr!!!! Quelqu\'un a un
marteau ?', 'A l\'aiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiide !!',

```

```

'[Windows] Plantage système au démarrage', 'Je ne sais pas', 4);
$questions[] = Array('Qui surveille les forums ?', 'L'équipe de
modération', 'Les administrateurs', 'L'hébergeur web du site',
'Il se surveille tout seul', 'Je ne sais pas', 1);
$questions[] = Array('Quel sujet n'est pas légal ?', '[SONDAGE]
Vous lavez-vous souvent les cheveux ?', '[HOTMAIL] Contourner le
filtre anti-spam', 'La nouvelle loi Hadopi, la fin du P2P en
France ?', 'Topic flood !', 'Je ne sais pas', 2);
$questions[] = Array('Si quelqu'un vous demande votre mot de
passe...', 'il n'est pas membre de l'équipe du site', 'c'est
qu'il est curieux', 'vous lui donnez', 'vous lui demandez le sien
en échange', 'Je ne sais pas', 1);
$questions[] = Array('Quel langage est interdit sur ce site ?',
'L'anglais', 'L'allemand', 'Le braille', 'Le SMS', 'Je ne sais
pas', 4);
$questions[] = Array('En vous inscrivant vous reconnaissez', 'être
fan du site', 'aimer le chocolat', 'avoir accepté la charte',
'être bête', 'Je ne sais pas', 3);
$Tquestions = count($questions);
?>

```



Chaque sous-array contient en entrée 0 la question et en entrée 6, le numéro de l'entrée contenant la bonne réponse : c'est comme ça que ça marche dans ce système (que je reconnais non optimisé, la BDD serait mieux je suis d'accord, mais ici c'est comme ça, na !).

Bon, ben du coup, en vous basant sur votre QCM et moi sur le mien, je vous propose de sélectionner trois questions au hasard (utilisez `mt_rand()`) (ou plus si vous y tenez 🤪) que l'on va stocker dans un array qui servira à l'affichage (je rappelle que je suis conscient qu'il y a des méthodes plus sympas, mais celle-ci est accessible sans difficultés).

Secret (cliquez pour afficher)

Code : PHP

```

<?php
$locate = Array();
$i=1;

while($i<=3)
{
    $k = $questions[mt_rand(0,$Tquestions-1)];
    while(in_array($k, $locate))
    {
        $k = $questions[mt_rand(0,$Tquestions-1)];
    }
    $locate[] = $k;
    $i++;
}
?>

```

Et maintenant l'affichage, en n'oubliant pas de créer les variables de session `reponse1` , `reponse2` et `reponse3` qui contiennent le numéro de la réponse :

Code : PHP

```

<!--Affichage-->
<?php
$i=1;
while($i<=3)
{

```

```

    $_SESSION['reponse'.$i] = $locate[$i-1][6];
    ?>
    <span class="question"><?php echo $locate[$i-1][0]; ?
  </span><br/>
    <input type="radio" name="reponse<?php echo $i; ?>" value="1"
id="<?php echo $i; ?>1" /> <label for="<?php echo $i; ?>1"><?php
echo $locate[$i-1][1]; ?></label><br />
    <input type="radio" name="reponse<?php echo $i; ?>" value="2"
id="<?php echo $i; ?>2" /> <label for="<?php echo $i; ?>2"><?php
echo $locate[$i-1][2]; ?></label><br />
    <input type="radio" name="reponse<?php echo $i; ?>" value="3"
id="<?php echo $i; ?>3" /> <label for="<?php echo $i; ?>3"><?php
echo $locate[$i-1][3]; ?></label><br />
    <input type="radio" name="reponse<?php echo $i; ?>" value="4"
id="<?php echo $i; ?>4" /> <label for="<?php echo $i; ?>4"><?php
echo $locate[$i-1][4]; ?></label><br />
    <input type="radio" name="reponse<?php echo $i; ?>" value="5"
id="<?php echo $i; ?>5" /> <label for="<?php echo $i; ?>5"><?php
echo $locate[$i-1][5]; ?></label><br />
  <?php
    $i++;
  }
  ?>

</div>

```

Vous pouvez tester, notre QCM est fin prêt, et vous aurez tout le temps de le constater en l'essayant. 😊

N'oubliez pas : dans la partie II on fera mieux, mais là, j'ai voulu éviter la création d'une table supplémentaire dans la BDD, parce qu'on sait tous qu'il y a des gens qui n'aiment pas SQL dans le coin. 🤔

Si tu arrives à me décoder, tu peux entrer

Avant de nous attaquer aux captchas, trois choses.

Premièrement, il y a un tuto entièrement dédié aux [captchas](#) ici. 😊 Allez y faire un tour !

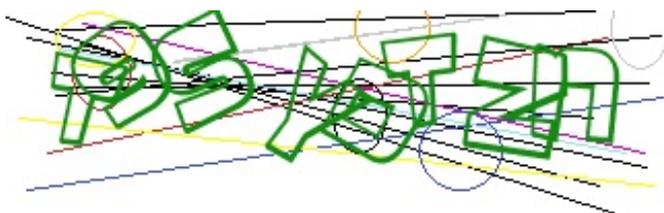
Deuxièmement, vous devriez peut-être revoir [le cours de M@teo sur GD](#), et troisièmement, vous devrez probablement vous fournir en polices.

Une police - petit rappel - est une apparence pour le texte : il en existe des milliers ; vous connaissez probablement arial, verdana, georgia, times new roman et d'autres, mais pour les captchas, elles ne sont pas top.

Aussi, pour vous fournir, je vous conseille de visiter [ce site](#) ou [celui-là](#) (que je ne connaissais pas, je l'ai trouvé dans le cours pour apprendre le langage C) et de privilégier les polices avec des caractères non remplis pour pouvoir faire de la superposition de caractères (entre autres, je sais pas si elle s'y trouve, mais pour [son captcha](#), Tracker a utilisé Basic Font il me semble, et ça rend plutôt bien ; malheureusement, l'image n'est plus affichée sur le sujet).

Pour ma part, je n'en ai pris qu'une : [baveuse3d](#).

Je vous conseille grandement d'en prendre plusieurs (donc, de ne pas faire comme moi), mais après avoir lu cette partie du chapitre, vous saurez faire plein de captchas. Pour ma part je suis arrivé à ça, et je ne suis pas trop mécontent de moi. 🤔



C'est parfois (souvent ?) un peu dur à lire, mais il semble que les caractères superposés soient un fléau pour les bots spammeurs ; donc, sans vous inciter à faire illisible ou comme moi (donc illisible 🤔), je vous invite à essayer de faire un captcha utilisant ce principe de superposition de lettres non remplies.

Téléchargez les polices qui vous intéressent (vous pouvez aussi, pour ceux qui utilisent Windows en trouver dans `C:\WINDOWS\Fonts`, mais l'option copier avec le clic droit de la souris est désactivée dans ce dossier, utilisez CTRL + C). Une

fois le plein de polices fait, créez dans le dossier [membres](#) le dossier [polices](#).

Secret (cliquez pour afficher)

Création du répertoire "polices" en cours...

10%...

40%...

90%...

99.1%...

99.2%...

99.8%...

99.85%...

...

Répertoire "polices" créé.

Waaaahhhh le bluff de fou, comment j't'ai trop eu 😊

Euhm. Huum, passons... 🤔

Bon c'est pas tout ça, mais si on parlait de GD et qu'on refaisait ensemble mon code (que je vous invite à modifier !!) ?

Une chose importante à savoir, c'est que, encore plus qu'en temps normal, PHP n'aime pas les erreurs dans la manipulation d'images.

Il est donc important de savoir faire le moins d'erreurs possible quand on touche à GD, parce que sinon, ça peut vite devenir un bordel sans nom et vous dégoûter des captchas à vie. 😞

Donc, créez dans votre dossier [membres](#) (toujours 🤔), le fichier [captcha.php](#).

Premièrement, on va ouvrir les sessions et déclarer notre image, c'est là qu'on va voir qui se rappelle encore du tuto de M@teo :

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page captcha.php

Génère un captcha.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Aucune information/erreur
-----
*/

session_start();
header ("Content-type: image/png");
$image = imagecreate(320, 100);
?>
```

Ensuite, on va déclarer une panoplie de couleurs via `imagecolorallocate()` .

Pour ma part, je veux un fond blanc, donc je dois déclarer \$blanc en premier. 🤔

Ici pas de secret : soit vous connaissez vos couleurs, soit vous ne les connaissez pas.

Code : PHP

```
<?php
$blanc = imagecolorallocate($image, 255, 255, 255);
$noir = imagecolorallocate($image, 0, 0, 0);
$gris = imagecolorallocate($image, 200,200,200);
$jaune = imagecolorallocate($image, 255, 255, 0);
$rouge = imagecolorallocate($image, 200, 39, 45);
$vert = imagecolorallocate($image, 45, 255, 39);
$cyan = imagecolorallocate($image, 0, 255, 255);
$magenta = imagecolorallocate($image, 200, 0, 200);
$orange = imagecolorallocate($image, 255, 160, 0);
$bleu = imagecolorallocate($image, 60, 75, 200);
$bleuclair = imagecolorallocate($image, 156, 227, 254);
$vertf = imagecolorallocate($image, 20, 140, 17);

/*$Anoir = imagecolorallocatealpha($image, 0, 0, 0, 80);
$Ajaune = imagecolorallocatealpha($image, 255, 255, 0, 80);
$Ablanc = imagecolorallocatealpha($image, 255, 255, 255, 80);
$Arouge = imagecolorallocatealpha($image, 200, 39, 45, 80);
$Avert = imagecolorallocatealpha($image, 45, 200, 39, 80);
$Acyan = imagecolorallocatealpha($image, 0, 255, 255, 80);
$Amagenta = imagecolorallocatealpha($image, 255, 0, 255, 80);
$Aorange = imagecolorallocatealpha($image, 255, 128, 0, 80);
$Ableu = imagecolorallocatealpha($image, 39, 45, 200, 80);
$Ableuclair = imagecolorallocatealpha($image, 156, 227, 254, 80);

$A2noir = imagecolorallocatealpha($image, 0, 0, 0, 25);
$A2jaune = imagecolorallocatealpha($image, 255, 255, 0, 25);
$A2blanc = imagecolorallocatealpha($image, 255, 255, 255, 25);
$A2rouge = imagecolorallocatealpha($image, 200, 39, 45, 25);
$A2vert = imagecolorallocatealpha($image, 45, 200, 39, 25);
$A2cyan = imagecolorallocatealpha($image, 0, 255, 255, 25);
$A2magenta = imagecolorallocatealpha($image, 255, 0, 255, 25);
$A2orange = imagecolorallocatealpha($image, 255, 128, 0, 25);
$A2bleu = imagecolorallocatealpha($image, 39, 45, 200, 25);
$A2bleuclair = imagecolorallocatealpha($image, 156, 227, 254, 25);*/
?>
```

Notez qu'en commentaire, il y a, à peu près, les mêmes couleurs déclarées avec des niveaux de transparence en plus (sauf le gris et le vert foncé qui sont venus plus tard, quand je me suis rendu compte que je n'utiliserai pas de couleur transparente 🤔).

Vous allez me faire un array des couleurs normales, et si vous voulez, deux autres des couleurs en commentaire (que vous penserez à dé-commenter si vous voulez pas avoir un bug 🤔).

Code : PHP

```
<?php
//Toutes les couleurs
$colors = Array($vert, $noir, $jaune, $blanc, $rouge, $cyan,
$magenta, $orange, $bleu, $bleuclair, $gris, $vertf);
$Tcolors = count($colors);

/*$colors2 = Array($Avert, $Anoir, $Ajaune, $Ablanc, $Arouge,
$Acyan, $Amagenta, $Aorange, $Ableu, $Ableuclair);
$Tcolors2 = count($colors2);
$colors3 = Array($A2vert, $A2noir, $A2jaune, $A2blanc, $A2rouge,
$A2cyan, $A2magenta, $A2orange, $A2bleu, $A2bleuclair);
$Tcolors3 = count($colors3);*/
```

Enfin, vous allez créer un array des couleurs lisibles, c'est-à-dire des couleurs que PHP peut utiliser pour colorer les caractères du captcha ; par exemple, dans ces couleurs, on ne mettra pas le jaune si on est sur fond blanc, parce que sinon, vous allez casser les yeux de beaucoup de monde.

Justement (quelle coïncidence 😊) je suis sur fond blanc, mon array va donc ressembler à ça :

Code : PHP

```
<?php
//couleurs autorisées pour les caractères
$Lcolors = Array($noir, $rouge, $magenta, $bleu, $vertf);
$TLcolors = count($Lcolors);
?>
```

Bon, nous avons tout ce que l'on veut comme couleurs ; maintenant, on va s'occuper de la police (ou des polices, tout dépend de vous), et des caractères autorisés.

Vous allez définir un array contenant les noms de vos polices, ainsi qu'une variable définissant les caractères autorisés (pensez à définir leur intervalle de taille avec des variables ; si vous avez le même intervalle pour toutes les polices, ce n'est pas nécessaire).

Code : PHP

```
<?php
$polices = Array('baveuse3d'); //Pensez à en rajouter !!
$Tpolices = count($polices);

//définition des caractères autorisés.
$carac = '0123456789ABCDEFGHIJKLMNPOQRSTUVWXYZ';
$Tcarac = strlen($carac);
?>
```

Maintenant, nous décorer le captcha avec lignes, ellipses et triangles colorés, dites-vous bien que cela ne gêne pas vraiment un bot, car souvent, un simple flou gaussien suffit à les "effacer" tout en gardant les lettres. Mais déjà avec ça, un captcha c'est moins vide, et puis n'oublions pas que nos caractères n'étant pas remplis, un flou gaussien peut compliquer leur lecture par une machine. 😊

Donc quatre opérations à faire.

Première opération, ajouter des lignes noires (et que des noires).

Deuxième opération, ajouter des lignes dont la couleur sera aléatoire.

Troisième opération, ajouter des ellipses dont la couleur sera aléatoire.

Quatrième opération, ajouter des triangles dont la couleur sera aléatoire.

La position de ces formes devra être pseudo-aléatoire : les lignes traverseront le captcha, les ellipses auront leur centre dans le captcha ; les triangles, eux, ben faites comme vous le sentez. 😊

Quelques rappels : on utilise `mt_rand()` pour les nombres aléatoires, les lignes se font avec la fonction `ImageLine()`, les ellipses avec `ImageEllipse()`, et les triangles avec `ImagePolygon()`.

Le nombre de chacune de ces formes sera aléatoire, mais restreint.

Maintenant c'est à vous de jouer, tout ce que je vous ai donné devrait vous suffire. 😊

Voici ce que vous devriez avoir (excepté niveau nombres, chacun ses notions en terme d'intervalle de nombre restreint 🤪 (que l'imbécile qui a mis pour `i` allant de 1 à 15524.3 se dénonce !)) :

Secret (cliquez pour afficher)

Code : PHP

```
<?php
```

```

//définition des lignes noires
$nb_lignes = mt_rand(3,7);
$i = 1;
while($i<=$nb_lignes)
{
    ImageLine($image, mt_rand(0,40), mt_rand(0,100), mt_rand(280,
320), mt_rand(0,100), $noir);
    $i++;
}

//définition des lignes colorées.
$nb_lignes = mt_rand(3,7);
$i = 1;
while($i<=$nb_lignes)
{
    ImageLine($image, mt_rand(0,40), mt_rand(0,100),
mt_rand(280,320), mt_rand(0,100), $colors[mt_rand(0,$Tcolors-1)]);
    $i++;
}

//définition des ellipses
$nb_ellipses = mt_rand(1,6);
$i = 1;
while($i<= $nb_ellipses)
{
    ImageEllipse($image, mt_rand(0,320), mt_rand(0,100),
25+mt_rand(0,15), 25+mt_rand(0,15),
$colors[mt_rand(0,$Tcolors-1)]);
    $i++;
}

//définition des triangles
$nb_triangles = mt_rand(1,6);
$i = 1;
while($i<=$nb_triangles)
{
    $array = Array(mt_rand(0,300), mt_rand(0,100), mt_rand(0,300),
mt_rand(0,100), mt_rand(0,300), mt_rand(0,100));
    ImagePolygon($image, $array, 3, $colors[mt_rand(0,$Tcolors-1)]);
    $i++;
}
?>

```

Si vous ne comprenez pas, je vous renvoie au cours de M@teo et à la doc PHP, c'est on ne peut plus clair. 😊

On en arrive aux caractères. Il y en aura 8 ; pensez à définir une constante d'écart de caractères (et à vous en servir), une couleur pour les caractères (la même pour tous, sinon ça sert à rien de profiter de la superposition), et à consulter la doc pour `imageTTFtext()` car vous allez vous servir de cette fonction pour l'affichage.

Comme M@teo n'en parle pas dans son tuto, on va l'étudier ensemble.

Citation : Doc PHP

```
array imageTTFtext (resource $image, float $size, float $angle, int $x, int $y, int $color, string $fontfile, string $text)
```

Dessine le texte fourni avec une police TrueType.

`$image` est votre image.

`$size` est la taille du caractère, c'est comme la propriété CSS `font-size` . Ici, on en veut des plutôt grands, on va donc utiliser `mt_rand(35, 45)` par exemple (ou bien vos constantes définies pour chaque police). Cependant, la taille de chaque caractère est propre à lui-même, donc dans la boucle le `mt_rand()` (pensez à le coordonner avec la police si besoin est). 😊

`$angle` définit l'angle d'affichage (qui a cru qu'il allait devoir faire des maths ? 😊), en degrés (que l'imbécile qui a dit Celsius ou Kelvin se dénonce !), donc incline le caractère (180 le retourne) ; ici encore une fois, on veut un intervalle d'angle, donc on va utiliser `mt_rand(-35, 35)` ; je vous conseille de garder ces valeurs, ou si vous les changez, de ne pas excéder -45,45, sinon ça devient lourd à lire.

Pareil, l'angle étant propre à chaque caractère, vous déclarerez `$angle` dans la boucle d'affichage.

`$x` définit l'abscisse horizontale du point de départ du caractère, `$y` l'ordonnée verticale (donc, si, il y a des maths 😊), ici, je vous laisse chercher seuls, faut pas tout vous mâcher non plus. 😊

`$color` est la couleur définie avant la boucle (vu qu'on veut la même pour tous les caractères), `$police` est la police, propre à chaque caractère (donc définie en boucle), choisie dans l'array `$polices`, et `$text` s'appellera ici `$lettre`, vu qu'on bossera lettre par lettre.

À vous de voir comment vous choisissez les lettres.

Voilà, vous avez tout ce qu'il vous faut pour finir votre captcha. 😊

On est partis. 😊

Une fois que vous avez fini, consultez ma correction :

Secret (cliquez pour afficher)

Déjà, on va définir la couleur des caractères, l'écart, et la session qui stockera le code du captcha (quoi, vous aviez oublié ? p):

Code : PHP

```
<?php
$aupifcolor = $Lcolors[mt_rand(0,$TLcolors-1)]; //la couleur des
caractères
$ecart = 300/10+4; //écart entre les caractères

$_SESSION['captcha'] = ''; //La voilà !! Enfin !!
?>
```

Maintenant on lance la boucle, on définit à chaque fois un caractère, sa taille, son angle, son ordonnée (l'abscisse n'est pas aléatoire), sa police... :

Code : PHP

```
<?php
$i = 0;
while($i <= 7)
{
    $lettre = $carac[mt_rand(0, $Tcarac-1)]; //choix de lettre
    $_SESSION['captcha'] .= $lettre; //stockage
    $taille = mt_rand(35,45); //taille
    $angle = mt_rand(-35,35); //angle
    $y = mt_rand(55, 60); //ordonnée
    $police = $polices[mt_rand(0, $Tpolices-1)]; //police :p

    imagettftext($image, $taille, $angle, $ecart*$i+15, $y,
    $aupifcolor, 'polices/'.$police.'.ttf', $lettre);
    $i++;
}

?>
```

Et on envoie l'image 😊 :

Code : PHP

```
<?php
imagepng($image); //On envoie et on prie :p
?>
```

Voilà : vous avez réussi votre captcha, et normalement, si vous allez voir votre page d'inscription, elle a maintenant de la gueule.



Bon, je vous remets le code `captcha.php` complet ici :

Secret (cliquez pour afficher)

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page captcha.php

Génère un captcha.

Quelques indications : (utiliser l'outil de recherche et
rechercher les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Aucune information/erreur
-----
*/

session_start();
header ("Content-type: image/png");
$image = imagecreate(320, 100);

$blanc = imagecolorallocate($image, 255, 255, 255);
$noir = imagecolorallocate($image, 0, 0, 0);
$gris = imagecolorallocate($image, 200, 200, 200);
$jaune = imagecolorallocate($image, 255, 255, 0);
$rouge = imagecolorallocate($image, 200, 39, 45);
$vert = imagecolorallocate($image, 45, 255, 39);
$cyan = imagecolorallocate($image, 0, 255, 255);
$magenta = imagecolorallocate($image, 200, 0, 200);
$orange = imagecolorallocate($image, 255, 160, 0);
$bleu = imagecolorallocate($image, 60, 75, 200);
$bleuclair = imagecolorallocate($image, 156, 227, 254);
$vertf = imagecolorallocate($image, 20, 140, 17);

/*$Anoir = imagecolorallocatealpha($image, 0, 0, 0, 80);
$Ajaune = imagecolorallocatealpha($image, 255, 255, 0, 80);
$Ablanc = imagecolorallocatealpha($image, 255, 255, 255, 80);
$Arouge = imagecolorallocatealpha($image, 200, 39, 45, 80);
$Avert = imagecolorallocatealpha($image, 45, 200, 39, 80);
$Acyan = imagecolorallocatealpha($image, 0, 255, 255, 80);
$Amagenta = imagecolorallocatealpha($image, 255, 0, 255, 80);
$Aorange = imagecolorallocatealpha($image, 255, 128, 0, 80);
$Ableu = imagecolorallocatealpha($image, 39, 45, 200, 80);
$Ableuclair = imagecolorallocatealpha($image, 156, 227, 254, 80);

$A2noir = imagecolorallocatealpha($image, 0, 0, 0, 25);
$A2jaune = imagecolorallocatealpha($image, 255, 255, 0, 25);
$A2blanc = imagecolorallocatealpha($image, 255, 255, 255, 25);
$A2rouge = imagecolorallocatealpha($image, 200, 39, 45, 25);
$A2vert = imagecolorallocatealpha($image, 45, 200, 39, 25);
$A2cyan = imagecolorallocatealpha($image, 0, 255, 255, 25);
$A2magenta = imagecolorallocatealpha($image, 255, 0, 255, 25);
$A2orange = imagecolorallocatealpha($image, 255, 128, 0, 25);
$A2bleu = imagecolorallocatealpha($image, 39, 45, 200, 25);
$A2bleuclair = imagecolorallocatealpha($image, 156, 227, 254,
```

```

25);*/

//Toutes les couleurs
$colors = Array($vert, $noir, $jaune, $blanc, $rouge, $cyan,
$magenta, $orange, $bleu, $bleuclair, $gris, $vertf);
$Tcolors = count($colors);

/*$colors2 = Array($Avert, $Anoir, $Ajaune, $Ablanc, $Arouge,
$Acyan, $Amagenta, $Aorange, $Ableu, $Ableuclair);
$Tcolors2 = count($colors2);
$colors3 = Array($A2vert, $A2noir, $A2jaune, $A2blanc, $A2rouge,
$A2cyan, $A2magenta, $A2orange, $A2bleu, $A2bleuclair);
$Tcolors3 = count($colors3);*/

//couleurs autorisées pour les caractères
$Lcolors = Array($noir, $rouge, $magenta, $bleu, $vertf);
$TLcolors = count($Lcolors);

$polices = Array('baveuse3d'); //Pensez à en rajouter !!
$Tpolices = count($polices);

//définition des caractères autorisés.
$carac = '0123456789ABCDEFGHIJKLMNPOQRSTUVWXYZ';
$Tcarac = strlen($carac);

//définition des lignes noires
$nb_lignes = mt_rand(3,7);
$i = 1;
while($i<=$nb_lignes)
{
    ImageLine($image, mt_rand(0,40), mt_rand(0,100), mt_rand(280,
320), mt_rand(0,100), $noir);
    $i++;
}

//définition des lignes colorées.
$nb_lignes = mt_rand(3,7);
$i = 1;
while($i<=$nb_lignes)
{
    ImageLine($image, mt_rand(0,40), mt_rand(0,100),
mt_rand(280,320), mt_rand(0,100), $colors[mt_rand(0,$Tcolors-1)]);
    $i++;
}

//définition des ellipses
$nb_ellipses = mt_rand(1,6);
$i = 1;
while($i<= $nb_ellipses)
{
    ImageEllipse($image, mt_rand(0,320), mt_rand(0,100),
25+mt_rand(0,15), 25+mt_rand(0,15),
$colors[mt_rand(0,$Tcolors-1)]);
    $i++;
}

//définition des triangles
$nb_triangles = mt_rand(1,6);
$i = 1;
while($i<=$nb_triangles)
{
    $array = Array(mt_rand(0,300), mt_rand(0,100), mt_rand(0,300),
mt_rand(0,100), mt_rand(0,300), mt_rand(0,100));
    ImagePolygon($image, $array, 3, $colors[mt_rand(0,$Tcolors-1)]);
    $i++;
}

$aupifcolor = $Lcolors[mt_rand(0,$TLcolors-1)]; //la couleur des
caractères

```

```
$ecart = 300/10+4; //écart entre les caractères

$_SESSION['captcha'] = ''; //La voilà !! Enfin !!

$i = 0;
while($i <= 7)
{
    $lettre = $carac[mt_rand(0, $Tcarac-1)]; //choix de lettre
    $_SESSION['captcha'] .= $lettre; //stockage
    $taille = mt_rand(35,45); //taille
    $angle = mt_rand(-35,35); //angle
    $y = mt_rand(55, 60); //ordonnée
    $police = $polices[mt_rand(0, $Tpolices-1)]; //police :p

    imagettftext($image, $taille, $angle, $ecart*$i+15, $y,
$aupifcolor, 'polices/'.$police.'.ttf', $lettre);
    $i++;
}

imagepng($image); //On envoie et on prie :p
?>
```

On a fini, le système anti-bot est enfin au point. 🤖 Il nous aura pris du temps, mais il marche, et, quand la page de validation sera finie, vous verrez que tout marche comme sur des roulettes !

Avant de passer à la validation d'ailleurs, je vous invite, si vous aimez vraiment GD et les captchas, à aller voir cette fonction qui pourra, si votre version de PHP et votre serveur le permettent, vous offrir des trucs sympas à rajouter : [imagefilter](#).
On a fini le chapitre !

Le formulaire est enfin prêt, passons à la deuxième partie de l'inscription. 😊

La validation et les mails

Bon, nous avons fait notre formulaire, et il marche, mais il faut encore pouvoir le valider et compléter l'inscription du membre, sinon bah... Cela ne sert pas à grand-chose. 😞

Validation du formulaire

La validation ne demande pas de grandes compétences en PHP, juste un peu de méthode.

Ce sera, avec la page d'édition de profil, la plus grosse de notre espace membres, car il y a beaucoup de vérifications à faire, encore que, j'ai converti pas mal de choses en fonctions, car elles resserviront pour la page d'édition.

Attaquons, sans plus attendre, la page de validation. 😊

Dans le dossier `membres`, créez le fichier `trait-inscription.php`.

Ajoutons-y ceci tout de suite :

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page trait-inscription.php

Permet de valider son inscription.

Quelques indications : (utiliser l'outil de recherche et rechercher
les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Déjà inscrit (en cas de bug...)
-----
*/

session_start();
header('Content-type: text/html; charset=utf-8');
include('../includes/config.php');

/*****Actualisation de la session...*****/

include('../includes/fonctions.php');
connexionbdd();
actualiser_session();

/*****Fin actualisation de session...*****/

if(isset($_SESSION['membre_id']))
{
    header('Location: '.ROOTPATH.'/index.php');
    exit();
}
?>
```

Là, je vais vous informer d'une idée que j'ai eue qui consiste à créer, si l'inscription est réussie, une variable de session `$_SESSION['inscrit']` et de la comparer à `$_POST['pseudo']`.

Dans quel but ? Imaginons que notre homme valide son inscription et que pour une raison ou une autre, le site plante avant la confirmation, ou que le navigateur du bonhomme plante ou un autre problème du genre.

En revenant, le membre va réessayer de s'inscrire avec le même pseudo, et la variable de session lui indiquera qu'il s'est déjà

inscrit et qu'il peut se connecter.

Bon : vous me direz que son utilité est contestable, mais c'est une idée que je propose ; je vous rappelle quand même que le système se veut le plus personnalisable possible, donc n'hésitez pas à modifier des trucs. Moi-même, si je m'en sers (faudrait déjà que je me décide à faire un site), je le modifierai probablement. 😊

Code : PHP

```
<?php
if($_SESSION['inscrit'] == $_POST['pseudo'] &&
trim($_POST['inscrit']) != '')
{
    $informations = Array(/*Déjà inscrit (en cas de bug...)* /
        true,
        'Vous êtes déjà inscrit',
        'Vous avez déjà complété une inscription avec le pseudo <span
class="pseudo">'.htmlspecialchars($_SESSION['inscrit'],
ENT_QUOTES).'</span>.',
        ' - <a href="'.ROOTPATH.'/index.php">Retourner à
l\'index</a>',
        ROOTPATH.'/membres/connexion.php',
        5
    );
    require_once('../information.php');
    exit();
}
?>
```

Une nouvelle classe fait son apparition, éditons donc le CSS. 😊

Code : CSS

```
.pseudo {
font-weight: bold;
}
```

Bon, stoppons le blabla : après cette formalité, on va devoir attaquer directement les fonctions avant de continuer la page d'inscription, car sinon, je sens que je vais vous perdre. 😊

Les fonctions

checkpseudo

La première fonction s'occupera du pseudo : elle devra vérifier dans l'ordre si le pseudo est vide, ensuite, s'il est trop court ou trop long, et enfin s'il est pris, sinon, c'est bon.

La fonction retournera un code pour chaque erreur, et le code ok si c'est bon.

Secret (cliquez pour afficher)

Code : PHP

```
<?php
function checkpseudo($pseudo)
{
    if($pseudo == '') return 'empty';
    else if(strlen($pseudo) < 3) return 'tooshort';
    else if(strlen($pseudo) > 32) return 'toolong';

    else
```

```

{
    $result = sqlquery("SELECT COUNT(*) AS nbr FROM membres WHERE
membre_pseudo = '".mysql_real_escape_string($pseudo)."'", 1);
    global $queries;
    $queries++;

    if($result['nbr'] > 0) return 'exists';
    else return 'ok';
}
}
?>

```

Vous avez maintenant déjà une idée de la tête qu'auront les autres fonctions. 😊

checkmdp

checkmdp devra vérifier si le mot de passe entré est ni trop long ni trop court, s'il contient au moins un chiffre et au moins une majuscule, sinon, tout est ok pour le mot de passe.

Secret ([cliquez pour afficher](#))

Code : PHP

```

<?php
function checkmdp ($mdp)
{
    if($mdp == '') return 'empty';
    else if(strlen($mdp) < 4) return 'tooshort';
    else if(strlen($mdp) > 50) return 'toolong';

    else
    {
        if(!preg_match('#[0-9]{1,}#', $mdp)) return 'nofigure';
        else if(!preg_match('#[A-Z]{1,}#', $mdp)) return 'noupcap';
        else return 'ok';
    }
}
?>

```

checkmdpS

Cette fonction devra vérifier que le mot de passe de vérification correspond au mot de passe (et même s'ils sont différents, les deux doivent être non vides pour être déclarés différents), sinon on retourne une erreur ; s'ils sont pareils, on le passe au même traitement que le mot de passe.

Secret ([cliquez pour afficher](#))

Code : PHP

```

<?php
function checkmdpS ($mdp, $mdp2)
{
    if($mdp != $mdp2 && $mdp != '' && $mdp2 != '') return
'different';
    else return checkmdp ($mdp);
}
?>

```

checkmail

Cette fonction devra vérifier le format du mail via *regex*, et si le mail n'est pas déjà pris. Je ne vous explique pas la *regex*, M@teo a fait exactement la même dans son tuto.

Secret ([cliquez pour afficher](#))

Code : PHP

```
<?php
function checkmail($email)
{
    if($email == '') return 'empty';
    else if(!preg_match('#^[a-z0-9._-]+@[a-z0-9._-]{2,}\.[a-z]{2,4}$#is', $email)) return 'isnt';

    else
    {
        $result = sqlquery("SELECT COUNT(*) AS nbr FROM membres WHERE
membre_mail = '".mysql_real_escape_string($email)."', 1);
        global $queries;
        $queries++;

        if($result['nbr'] > 0) return 'exists';
        else return 'ok';
    }
}
?>
```

checkmails

Cette fonction fait comme checkmdpS.

Secret ([cliquez pour afficher](#))

Code : PHP

```
<?php
function checkmails($email, $email2)
{
    if($email != $email2 && $email != '' && $email2 != '') return
'different';
    else return 'ok';
}
?>
```

birthdate

Cette fonction est un peu plus longue que les autres.

Elle devra vérifier le format de la date de naissance (deux/), l'âge de l'utilisateur (vous voulez vraiment un enfant de 4 ans sur votre site ? 🤪), l'existence de la date, enfin valider celle-ci si c'est bon.

Secret ([cliquez pour afficher](#))

Code : PHP

```
<?php
function birthdate($date)
```

```

{
    if($date == '') return 'empty';

    else if(substr_count($date, '/') != 2) return 'format';
    else
    {
        $DATE = explode('/', $date);
        if(date('Y') - $DATE[2] <= 4) return 'tooyoung';
        else if(date('Y') - $DATE[2] >= 135) return 'toooold';

        else if($DATE[2]%4 == 0)
        {
            $maxdays = Array('31', '29', '31', '30', '31', '30', '31',
'31', '30', '31', '30', '31');
            if($DATE[0] > $maxdays[$DATE[1]-1]) return 'invalid';
            else return 'ok';
        }

        else
        {
            $maxdays = Array('31', '28', '31', '30', '31', '30', '31',
'31', '30', '31', '30', '31');
            if($DATE[0] > $maxdays[$DATE[1]-1]) return 'invalid';
            else return 'ok';
        }
    }
}
?>

```

vidersession

Durant toute l'inscription, on va mettre dans `$_SESSION` plein de variables car si la validation n'est pas correcte, on renvoie le visiteur sur le formulaire d'inscription, on affiche à nouveau les erreurs que l'on a rencontrées, et on pré-remplit les champs qui étaient corrects.

Mais si l'inscription se passe bien, juste avant de créer `$_SESSION['inscrit']` on va vider toute la session sans `session_destroy()`, car sinon on peut pas créer `$_SESSION['inscrit']`.

Un simple foreach suffit.

Secret (cliquez pour afficher)

Code : PHP

```

<?php
function vidersession()
{
    foreach($_SESSION as $cle => $element)
    {
        unset($_SESSION[$cle]);
    }
}
?>

```

Voilà, nos fonctions sont prêtes, passons à la suite. 😊

Nota : une autre solution consiste à mettre toutes ces variables temporaires dans une variable `$_SESSION['temp']`. On aura donc `$_SESSION['temp']['form_pseudo']` etc.

Vous n'aurez alors qu'à faire un `unset()` sur `$_SESSION['temp']` pour vider le formulaire temporaire.

Validation proprement dite

Bon : on a fini les fonctions, revenons à [trait-inscription.php](#). 😊

Maintenant que les fonctions sont prêtes, il suffit pour chaque champ du formulaire de vérifier s'il existe (`isset`), de supprimer les espaces en début et fin de chaîne (`trim`), de les passer par leurs fonctions respectives ; enfin, de mettre les bonnes erreurs en session et de créer une variable `$_SESSION['erreurs']` qui recense le nombre d'erreurs.

On y va, c'est très facile : essayez d'abord sans regarder mon code. 😊

Secret (cliquez pour afficher)

Code : PHP

```
<?php
/*Code anti-pompage sans lecture :p*/
exit('Pour ceux qui recopient bêtement ce code sans avoir essayé,
sachez que vous êtes bien
bêtes, le but de ce tuto est de vous entraîner à comprendre et à
coder en PHP. Tci
il n\'y a rien de dur, c\'est un simple traitement de
l\'information fournie par des fonctions
et de la définition de variables, alors pomper ce code sans
essayer tout seul, c\'est vraiment
pas une attitude de Zér0 :o)');

/*****Étude du bazar envoyé*****/

$_SESSION['erreurs'] = 0;

//Pseudo
if(isset($_POST['pseudo']))
{
    $pseudo = trim($_POST['pseudo']);
    $pseudo_result = checkpseudo($pseudo);
    if($pseudo_result == 'tooshort')
    {
        $_SESSION['pseudo_info'] = '<span class="erreur">Le pseudo
        '.htmlspecialchars($pseudo, ENT_QUOTES).' est trop court, vous
        devez en choisir un plus long (minimum 3
        caractères).</span><br/>';
        $_SESSION['form_pseudo'] = '';
        $_SESSION['erreurs']++;
    }

    else if($pseudo_result == 'toolong')
    {
        $_SESSION['pseudo_info'] = '<span class="erreur">Le pseudo
        '.htmlspecialchars($pseudo, ENT_QUOTES).' est trop long, vous
        devez en choisir un plus court (maximum 32
        caractères).</span><br/>';
        $_SESSION['form_pseudo'] = '';
        $_SESSION['erreurs']++;
    }

    else if($pseudo_result == 'exists')
    {
        $_SESSION['pseudo_info'] = '<span class="erreur">Le pseudo
        '.htmlspecialchars($pseudo, ENT_QUOTES).' est déjà pris,
        choisissez-en un autre.</span><br/>';
        $_SESSION['form_pseudo'] = '';
        $_SESSION['erreurs']++;
    }

    else if($pseudo_result == 'ok')
    {
        $_SESSION['pseudo_info'] = '';
        $_SESSION['form_pseudo'] = $pseudo;
    }
}
```

```
else if($pseudo_result == 'empty')
{
    $_SESSION['pseudo_info'] = '<span class="erreur">Vous n\'avez
pas entré de pseudo.</span><br/>';
    $_SESSION['form_pseudo'] = '';
    $_SESSION['erreurs']++;
}
}

else
{
    header('Location: ../index.php');
    exit();
}

//Mot de passe
if(isset($_POST['mdp']))
{
    $mdp = trim($_POST['mdp']);
    $mdp_result = checkmdp($mdp, '');
    if($mdp_result == 'tooshort')
    {
        $_SESSION['mdp_info'] = '<span class="erreur">Le mot de passe
entré est trop court, changez-en pour un plus long (minimum 4
caractères).</span><br/>';
        $_SESSION['form_mdp'] = '';
        $_SESSION['erreurs']++;
    }

    else if($mdp_result == 'toolong')
    {
        $_SESSION['mdp_info'] = '<span class="erreur">Le mot de passe
entré est trop long, changez-en pour un plus court. (maximum 50
caractères)</span><br/>';
        $_SESSION['form_mdp'] = '';
        $_SESSION['erreurs']++;
    }

    else if($mdp_result == 'nofigure')
    {
        $_SESSION['mdp_info'] = '<span class="erreur">Votre mot de passe
doit contenir au moins un chiffre.</span><br/>';
        $_SESSION['form_mdp'] = '';
        $_SESSION['erreurs']++;
    }

    else if($mdp_result == 'noupcap')
    {
        $_SESSION['mdp_info'] = '<span class="erreur">Votre mot de passe
doit contenir au moins une majuscule.</span><br/>';
        $_SESSION['form_mdp'] = '';
        $_SESSION['erreurs']++;
    }

    else if($mdp_result == 'ok')
    {
        $_SESSION['mdp_info'] = '';
        $_SESSION['form_mdp'] = $mdp;
    }

    else if($mdp_result == 'empty')
    {
        $_SESSION['mdp_info'] = '<span class="erreur">Vous n\'avez pas
entré de mot de passe.</span><br/>';
        $_SESSION['form_mdp'] = '';
        $_SESSION['erreurs']++;
    }
}
}
```

```
else
{
    header('Location: ../index.php');
    exit();
}

//Mot de passe suite
if(isset($_POST['mdp_verif']))
{
    $mdp_verif = trim($_POST['mdp_verif']);
    $mdp_verif_result = checkmdpS($mdp_verif, $mdp);
    if($mdp_verif_result == 'different')
    {
        $_SESSION['mdp_verif_info'] = '<span class="erreur">Le mot de
passe de vérification diffère du mot de passe.</span><br/>';
        $_SESSION['form_mdp_verif'] = '';
        $_SESSION['erreurs']++;
        if(isset($_SESSION['form_mdp'])) unset($_SESSION['form_mdp']);
    }

    else
    {
        if($mdp_verif_result == 'ok')
        {
            $_SESSION['form_mdp_verif'] = $mdp_verif;
            $_SESSION['mdp_verif_info'] = '';
        }

        else
        {
            $_SESSION['mdp_verif_info'] = str_replace('passe', 'passe de
vérification', $_SESSION['mdp_info']);
            $_SESSION['form_mdp_verif'] = '';
            $_SESSION['erreurs']++;
        }
    }
}

else
{
    header('Location: ../index.php');
    exit();
}

//mail
if(isset($_POST['mail']))
{
    $mail = trim($_POST['mail']);
    $mail_result = checkmail($mail);
    if($mail_result == 'isnt')
    {
        $_SESSION['mail_info'] = '<span class="erreur">Le mail
'.htmlspecialchars($mail, ENT_QUOTES).' n\'est pas
valide.</span><br/>';
        $_SESSION['form_mail'] = '';
        $_SESSION['erreurs']++;
    }

    else if($mail_result == 'exists')
    {
        $_SESSION['mail_info'] = '<span class="erreur">Le mail
'.htmlspecialchars($mail, ENT_QUOTES).' est déjà pris, <a
href="../contact.php">contactez-nous</a> si vous pensez à une
erreur.</span><br/>';
        $_SESSION['form_mail'] = '';
        $_SESSION['erreurs']++;
    }

    else if($mail_result == 'ok')
```

```
{
    $_SESSION['mail_info'] = '';
    $_SESSION['form_mail'] = $mail;
}

else if($mail_result == 'empty')
{
    $_SESSION['mail_info'] = '<span class="erreur">Vous n\'avez pas
entré de mail.</span><br/>';
    $_SESSION['form_mail'] = '';
    $_SESSION['erreurs']++;
}
}

else
{
    header('Location: ../index.php');
    exit();
}

//mail suite
if(isset($_POST['mail_verif']))
{
    $mail_verif = trim($_POST['mail_verif']);
    $mail_verif_result = checkmailS($mail_verif, $mail);
    if($mail_verif_result == 'different')
    {
        $_SESSION['mail_verif_info'] = '<span class="erreur">Le mail de
vérification diffère du mail.</span><br/>';
        $_SESSION['form_mail_verif'] = '';
        $_SESSION['erreurs']++;
    }

    else
    {
        if($mail_result == 'ok')
        {
            $_SESSION['mail_verif_info'] = '';
            $_SESSION['form_mail_verif'] = $mail_verif;
        }

        else
        {
            $_SESSION['mail_verif_info'] = str_replace(' mail', ' mail de
vérification', $_SESSION['mail_info']);
            $_SESSION['form_mail_verif'] = '';
            $_SESSION['erreurs']++;
        }
    }
}

else
{
    header('Location: ../index.php');
    exit();
}

//date de naissance
if(isset($_POST['date_naissance']))
{
    $date_naissance = trim($_POST['date_naissance']);
    $date_naissance_result = birthdate($date_naissance);
    if($date_naissance_result == 'format')
    {
        $_SESSION['date_naissance_info'] = '<span class="erreur">Date de
naissance au mauvais format ou invalide.</span><br/>';
        $_SESSION['form_date_naissance'] = '';
        $_SESSION['erreurs']++;
    }
}
```

```
else if($date_naissance_result == 'tooyoung')
{
    $_SESSION['date_naissance_info'] = '<span
class="erreur">Agagagougougou areuh ? (Vous êtes trop jeune pour
vous inscrire ici.)</span><br/>';
    $_SESSION['form_date_naissance'] = '';
    $_SESSION['erreurs']++;
}

else if($date_naissance_result == 'toold')
{
    $_SESSION['date_naissance_info'] = '<span class="erreur">Plus de
135 ans ? Mouais...</span><br/>';
    $_SESSION['form_date_naissance'] = '';
    $_SESSION['erreurs']++;
}

else if($date_naissance_result == 'invalid')
{
    $_SESSION['date_naissance_info'] = '<span class="erreur">Le
'.htmlspecialchars($date_naissance, ENT_QUOTES).' n\'existe
pas.</span><br/>';
    $_SESSION['form_date_naissance'] = '';
    $_SESSION['erreurs']++;
}

else if($date_naissance_result == 'ok')
{
    $_SESSION['date_naissance_info'] = '';
    $_SESSION['form_date_naissance'] = $date_naissance;
}

else if($date_naissance_result == 'empty')
{
    $_SESSION['date_naissance_info'] = '<span class="erreur">Vous
n\'avez pas entré de date de naissance.</span><br/>';
    $_SESSION['form_date_naissance'] = '';
    $_SESSION['erreurs']++;
}
}

else
{
    header('Location: ../index.php');
    exit();
}

//qcm
if($_SESSION['reponse1'] == $_POST['reponse1'] &&
$_SESSION['reponse2'] == $_POST['reponse2'] &&
$_SESSION['reponse3'] == $_POST['reponse3'] &&
isset($_POST['reponse1']) && isset($_POST['reponse2']) &&
isset($_POST['reponse3']))
{
    $_SESSION['qcm_info'] = '';
}

else
{
    $_SESSION['qcm_info'] = '<span class="erreur">Au moins une des
réponses au QCM charte est fausse.</span><br/>';
    $_SESSION['erreurs']++;
}

//captcha
if($_POST['captcha'] == $_SESSION['captcha'] &&
isset($_POST['captcha']) && isset($_SESSION['captcha']))
{
    $_SESSION['captcha_info'] = '';
}
```

```

    }

    else
    {
        $_SESSION['captcha_info'] = '<span class="erreur">Vous n\'avez
pas recopié correctement le contenu de l\'image.</span><br/>';
        $_SESSION['erreurs']++;
    }

    unset($_SESSION['reponse1'], $_SESSION['reponse2'],
$_SESSION['reponse3']);
    unset($_SESSION['captcha']);

    /*****Fin étude*****/
?>

```

Voilà, le traitement à proprement parler est fini ; on va maintenant afficher le titre de la page et faire une insertion du tout si tout s'est bien passé. 😊

Code : PHP

```

<?php
/*****Entête et titre de page*****/
if($_SESSION['erreurs'] > 0) $titre = 'Erreur : Inscription 2/2';
else $titre = 'Inscription 2/2';

include('../includes/haut.php'); //contient le doctype, et head.

/*****Fin entête et titre*****/
?>
<div id="colonne_gauche">
<?php
include('../includes/colg.php');
?>
</div>

<div id="contenu">
<div id="map">
<!-- Absence de lien à Inscription 2/2 volontaire -->
<a href="../index.php">Accueil</a> => Inscription 2/2
</div>

```

À présent, nous allons attaquer un subtil détail que je n'ai pas encore énoncé.

S'il n'y a pas d'erreur, on va définir une requête et tenter de l'appliquer.

Seulement, imaginons qu'une fraction de secondes après la vérification du pseudo pour un visiteur X, un visiteur Y s'inscrive avec le même pseudo : la vérification du pseudo dans le traitement indique qu'il est libre, mais le visiteur X envoie alors la requête et prend le pseudo.

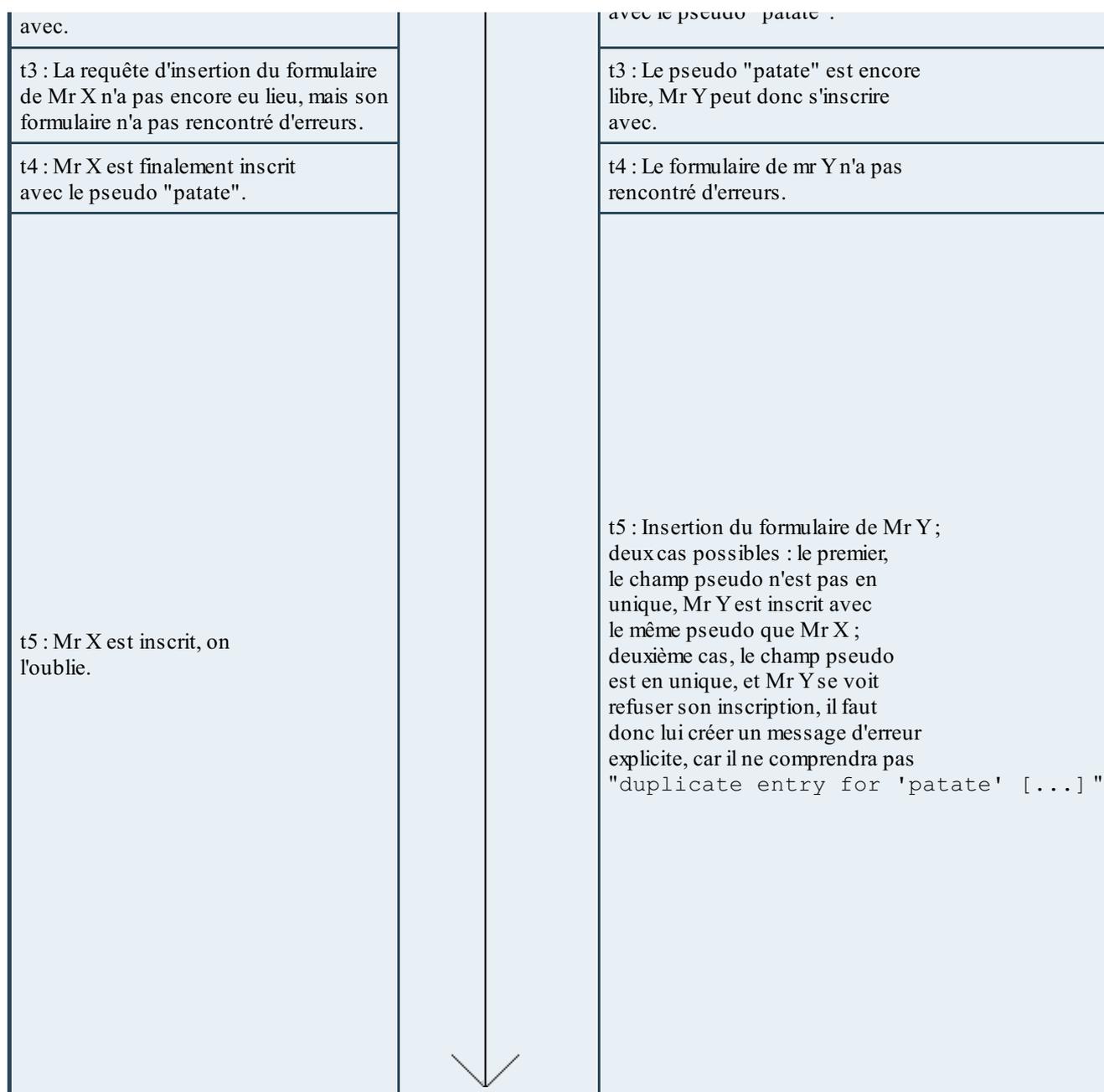
Le visiteur Y va alors envoyer la requête à son tour et là, deux possibilités.

1°) Vous n'aviez pas prévu ça, et on a deux utilisateurs ayant le même pseudo sur votre site, ce qui est fort gênant.

2°) Vous aviez prévu ça et le visiteur Y se fait refuser car le champ pseudo est configuré en UNIQUE, mais il ne sait pas pourquoi il s'est fait jeter !

Voici un schéma pour vous aider à comprendre :

t1 : Mr X valide son formulaire avec le pseudo "patate".	Axe du temps :	t1 : Mr Y n'a pas encore validé.
t2 : Le pseudo "patate" est libre, Mr X peut donc s'inscrire		t2 : Mr Y valide son formulaire avec le pseudo "patate".



Vous avez compris, je veux travailler le cas où on l'empêche de s'inscrire, mais on lui explique pourquoi l'inscription est refusée. Ça va être un peu nouveau pour vous, on va faire appel à une fonction sur `mysql_error()` que vous ne connaissez pas tous : `stripos`.

`stripos` recherche la première occurrence d'une chaîne dans une autre et retourne sa position (qui peut valoir 0 si le premier caractère de l'occurrence est le premier de la chaîne), `stripos` est insensible à la casse contrairement à `strpos`.

Code : PHP

```
<?php
echo stripos('abcdefg', 'ab'); //echo 0, car a est le premier
caractère de abcdefg;
echo stripos('abcdefg', 'gh'); //FALSE
echo stripos('abcdefg', 'd'); //echo 3, car d est le 4e caractère
de abcdefg
?>
```

Il est important de noter que la fonction peut retourner 0, donc il faudra utiliser `===` pour tester l'égalité et `!==` pour tester l'inégalité.

Les opérateurs `===`/`!==` marchent comme `==`/`!=` sauf qu'ils vérifient aussi si les variables sont de même type ou non.

Le meilleur exemple est avec 0 ou 1 et respectivement FALSE et TRUE.

Si vous testez l'égalité entre 0 et FALSE, vous trouverez que celle-ci est vraie ; or, 0 est un entier, et FALSE est un booléen, c'est l'une des caractéristique de PHP, c'est un langage faiblement typé. Du coup, on dit que 0 est un équivalent de FALSE en PHP.

Seulement, on a parfois besoin que l'égalité entre 0 et FALSE soit fausse, on utilisera donc `===` qui teste aussi le type.

Exemple :

Code : PHP

```
<?php
if(0 == FALSE) echo 'Oooohhhh! 0 == False !!<br/>';
else echo 'Oh non, 0 != False :(<br/>';
if(0 === FALSE) echo 'Aahhhhhh! 0 === False !!<br/>';
else echo 'Oh, 0 !== False :x'; //On sent l'inspiration :p
?>
```

Retour de ce code :

Citation : Pas de titre

```
Oooohhhh! 0 == False !!
Oh, 0 !== False :x
```

Compris ?

Dans le code d'erreur SQL, si le problème vient d'un champ unique, le nom qu'on a tenté d'insérer est affiché : donc, avec stripos, on va chercher le pseudo et / ou le mail dans l'erreur.

Allons-y. 😊

Code : PHP

```
<!--Test des erreurs et envoi-->
<?php
if($_SESSION['erreurs'] == 0)
{
$insertion = "INSERT INTO membres VALUES (NULL,
'".mysql_real_escape_string($pseudo)."',
'".md5($mdp)."', '".mysql_real_escape_string($mail)."',
'.time()."', '".mysql_real_escape_string($date_naissance)."',
'.'.time()."', 0)";

if(mysql_query($insertion))
{
$queryes++;
vidersession();
$_SESSION['inscrit'] = $pseudo;
/*informe qu'il s'est déjà inscrit s'il actualise, si son
navigateur
bugue avant l'affichage de la page et qu'il recharge la page,
etc.*/
?>
<h1>Inscription validée !</h1>
<p>Nous vous remercions de vous être inscrit sur notre site,
votre inscription a été validée !<br/>
Vous pouvez vous connecter avec vos identifiants <a
href="connexion.php">ici</a>.
</p>
<?php
}
```

```

else
{
    if(stripos(mysql_error(), $_SESSION['form_pseudo']) !== FALSE)
// recherche du pseudo
    {
        unset($_SESSION['form_pseudo']);
        $_SESSION['pseudo_info'] = '<span class="erreur">Le pseudo
'.htmlspecialchars($pseudo, ENT_QUOTES).' est déjà pris, choisissez-
en un autre.</span><br/>';
        $_SESSION['erreurs']++;
    }

    if(stripos(mysql_error(), $_SESSION['form_mail']) !== FALSE)
//recherche du mail
    {
        unset($_SESSION['form_mail']);
        unset($_SESSION['form_mail_verif']);
        $_SESSION['mail_info'] = '<span class="erreur">Le mail
'.htmlspecialchars($mail, ENT_QUOTES).' est déjà pris, <a
href=" ../contact.php">contactez-nous</a> si vous pensez à une
erreur.</span><br/>';
        $_SESSION['mail_verif_info'] = str_replace('mail', 'mail de
vérification', $_SESSION['mail_info']);
        $_SESSION['erreurs']++;
        $_SESSION['erreurs']++;
    }

    if($_SESSION['erreurs'] == 0)
    {
        $sqlbug = true; //plantage SQL.
        $_SESSION['erreurs']++;
    }
}
}
?>

```

Rien de choquant ici : si vous aviez compris le fonctionnement de stripos et que vous aviez vu comment on marchait, vous saviez faire ça.

Enfin, s'il y a des erreurs, on utilise pas else if, car \$_SESSION['erreurs'] a pu devenir supérieur à 0 dans le if, donc on fait une nouvelle condition. 😊

Code : PHP

```

<?php
if($_SESSION['erreurs'] > 0)
{
    if($_SESSION['erreurs'] == 1) $_SESSION['nb_erreurs'] = '<span
class="erreur">Il y a eu 1 erreur.</span><br/>';
    else $_SESSION['nb_erreurs'] = '<span class="erreur">Il y a eu
' . $_SESSION['erreurs'] . ' erreurs.</span><br/>';
    ?>
<h1>Inscription non validée.</h1>
<p>Vous avez rempli le formulaire d'inscription du site et nous
vous en remercions, cependant, nous n'avons
pas pu valider votre inscription, en voici les raisons :<br/>
<?php
echo $_SESSION['nb_erreurs'];
echo $_SESSION['pseudo_info'];
echo $_SESSION['mdp_info'];
echo $_SESSION['mdp_verif_info'];
echo $_SESSION['mail_info'];
echo $_SESSION['mail_verif_info'];
echo $_SESSION['date_naissance_info'];
echo $_SESSION['qcm_info'];

```

```

    echo $_SESSION['captcha_info'];

    if($sqlbug !== true)
    {
    ?>
    Nous vous proposons donc de revenir à la page précédente pour
    corriger les erreurs. (Attention, que vous
    l'ayez correctement remplie ou non, la partie sur la charte et
    l'image est à refaire intégralement.)</p>
    <div class="center"><a href="inscription.php">Retour</a></div>
    <?php
    }

    else
    {
    ?>
    Une erreur est survenue dans la base de données, votre formulaire
    semble ne pas contenir d'erreurs, donc
    il est possible que le problème vienne de notre côté, réessayez
    de vous inscrire ou contactez-nous.</p>

    <div class="center"><a href="inscription.php">Retenter une
    inscription</a> - <a href=" ../contact.php">Contactez-nous</a></div>
    <?php
    }
    }
    ?>
</div>

<?php
include (' ../includes/bas.php ');
?>
<!--fin-->

```

Voilà, `trait-inscription.php` est fini : si vous avez tenté d'en coder une majeure partie vous-mêmes sans pomper le code, alors vous êtes gagnants ; les autres, ben... vous n'êtes pas de vrais Zéro0s. 🤪

Maintenant, vous avez vu qu'on a utilisé des sessions et qu'on propose de revenir à `inscription.php` si l'inscription échoue. Ce n'est pas pour rien, les sessions : on va modifier `inscription.php` pour qu'elle affiche à nouveau les erreurs qu'il y a eu si l'entier `$_SESSION['erreurs']` est supérieur à 0 et pour pré-remplir les champs corrects. 😊

Pour le coup, je vous donne le code directement, car il n'y a rien de technique, c'est juste de la gestion d'affichage. 😊

Secret (cliquez pour afficher)

Code : PHP

```

<div id="contenu">
  <div id="map">
    <a href=" ../index.php">Accueil</a> => <a
href="inscription.php">Inscription 1/2</a>
  </div>

  <?php
  if($_SESSION['erreurs'] > 0)
  {
  ?>
  <div class="border-red">
  <h1>Note :</h1>
  <p>
  Lors de votre dernière tentative d'inscription, des erreurs
  sont survenues, en voici la liste :<br/>
  <?php
  echo $_SESSION['nb_erreurs'];
  echo $_SESSION['pseudo_info'];

```

```

    echo $_SESSION['mdp_info'];
    echo $_SESSION['mdp_verif_info'];
    echo $_SESSION['mail_info'];
    echo $_SESSION['mail_verif_info'];
    echo $_SESSION['date_naissance_info'];
    echo $_SESSION['qcm_info'];
    echo $_SESSION['captcha_info'];
    ?>
    Nous vous avons pré-rempli les champs qui étaient
corrects.<br/>
    Note : la partie QCM et image est entièrement à refaire, que
vous vous soyez trompé ou non.
</p>
</div>
<?php
}
?>

<h1>Formulaire d'inscription</h1>
<p>Bienvenue sur la page d'inscription de mon site !<br/>
Merci de remplir ces champs pour continuer.</p>
<form action="trait-inscription.php" method="post"
name="Inscription">
    <fieldset><legend>Identifiants</legend>
        <label for="pseudo" class="float">Pseudo :</label> <input
type="text" name="pseudo" id="pseudo" size="30" value="<?php
if($_SESSION['pseudo_info'] == '') echo
htmlspecialchars($_SESSION['form_pseudo'], ENT_QUOTES) ; ?>" />
<em>(compris entre 3 et 32 caractères)</em><br />
        <label for="mdp" class="float">Mot de passe :</label> <input
type="password" name="mdp" id="mdp" size="30" value="<?php
if($_SESSION['mdp_info'] == '') echo
htmlspecialchars($_SESSION['form_mdp'], ENT_QUOTES) ; ?>" />
<em>(compris entre 4 et 50 caractères)</em><br />
        <label for="mdp_verif" class="float">Mot de passe
(vérification) :</label> <input type="password" name="mdp_verif"
id="mdp_verif" size="30" value="<?php
if($_SESSION['mdp_verif_info'] == '') echo
htmlspecialchars($_SESSION['form_mdp_verif'], ENT_QUOTES) ; ?>"
/><br />
        <label for="mail" class="float">Mail :</label> <input
type="text" name="mail" id="mail" size="30" value="<?php
if($_SESSION['mail_info'] == '') echo
htmlspecialchars($_SESSION['form_mail'], ENT_QUOTES) ; ?>" /> <br
/>
        <label for="mail_verif" class="float">Mail (vérification)
:</label> <input type="text" name="mail_verif" id="mail_verif"
size="30" value="<?php if($_SESSION['mail_verif_info'] == '') echo
htmlspecialchars($_SESSION['form_mail_verif'], ENT_QUOTES) ; ?>"
/><br />
        <label for="date_naissance" class="float">Date de naissance
:</label> <input type="text" name="date_naissance"
id="date_naissance" size="30" value="<?php
if($_SESSION['date_naissance_info'] == '') echo
htmlspecialchars($_SESSION['form_date_naissance'], ENT_QUOTES) ; ?
>" /> <em>(format JJ/MM/AAAA)</em><br/>
    </fieldset>
    <fieldset><legend>Charte du site et protection anti-
robot</legend>
    <?php
    include('../includes/charte.php');
    ?>

    <h1>Système anti robots :</h1>

    <p>Qu'est-ce que c'est ?<br/>
    Pour lutter contre l'inscription non désirée de robots qui
publient du contenu non désiré sur les sites web,
nous avons décidé de mettre en place un système de sécurité
.<br/>

```

Aucun de ces systèmes n'est parfait, mais nous espérons que celui-ci, sans vous être inaccessible sera suffisant pour lutter contre ces robots.

Il est possible que certaine fois, l'image soit trop dure à lire, le cas échéant, actualisez la page jusqu'à avoir une image lisible.

Si vous êtes dans l'incapacité de lire plusieurs images d'affilée, contactez-nous, nous nous occuperons de votre inscription.</p>

```
<label for="captcha" class="float">Entrez les 8 caractères (majuscules ou chiffres) contenus dans l'image :</label> <input type="text" name="captcha" id="captcha"><br/>
```

```

</fieldset>
<div class="center"><input type="submit" value="Inscription" /></div>
</form>
</div>
```

Voilà, c'était pas bien dur. 🤪

Eh bien vous savez quoi ? Le traitement du formulaire est bel et bien fini, cette fois ! 😊

Un petit mail ?

Bon, bon, bon... 🤪

On a déjà fait pas mal de choses, mais il faut encore envoyer un mail au membre qui vient de s'inscrire. 😊

Pour cela, il faut avoir un serveur qui le permette ; normalement, vous n'avez rien à faire pour que la fonction marche.

Il y a un [tutoriel](#) du SdZ qui propose d'étudier la fonction mail , je vous le recommande chaudement !

Cependant, on ne sait jamais : si par hasard vous voulez vérifier si vous ne pouvez pas l'activer, consultez [la documentation](#).

Un bon exemple de l'utilisation de la fonction mail est fait dans [ce tutoriel](#).

Nous allons ici étudier cette fonction et certaines de ses possibilités.

Comme nous le dit la [documentation PHP](#), la fonction mail () se décrit ainsi :

Citation : Doc

```
bool mail(string $to, string $subject, string $message[, string $additional_headers[, string $additional_parameters]])
```

Elle prend donc en paramètre obligatoire un destinataire (surpris ? 🤪), un sujet, un message, et c'est tout.

Code : PHP

```
<?php
$to = 'neoterranos@siteduzero.com'; //je m'y crois un peu trop, moi
o_o
$subject = 'Merci !!!';
$message = 'Merci le SdZ. :)';

mail($to, $subject, $message); //marche
?>
```

Ce petit bout de code marche sans soucis, mais évitez de le faire : vous n'aurez jamais de réponse. 🤪

Cependant, la fonction `mail` telle quelle n'est pas terrible.

Le message est restreint : en effet, on ne peut envoyer que des textes contenant 70 caractères par ligne, chaque ligne étant séparée par un `\n`.

Il y a de plus une histoire avec les `.` dans le message...

Bref, avec la fonction `mail`, on ne peut rien faire... 😞

Heureusement qu'il y a des headers 😞 (en fait, on peut faire des choses avec `mail`, mais c'est mieux de passer outre les impositions vraiment lourdes).

En effet, en paramètre additionnel (donc facultatif), vous pouvez fournir des en-têtes.

Imaginons que, par un grand hasard, vous ayez, sans vous en rendre compte (tout cela n'est que pure coïncidence), composé ce code :

Code : PHP

```
<?php
$to = 'neoterranos@siteduzero.com'; //je m'y crois un peu trop, moi
o_o
$subject = 'Merci !!!';
$message = '<html>
<head>
<title>J\'vous kiffe de trop :D</title>
</head>
<body>
<p>Loin de moi l\'envie fortuite de vous importuner avec mes<br/>
discours aussi inutiles qu\'endormants, mais je trouve que vous
êtes<br/>
tout simplement génial !!</p>
<p>I love you !! (De tout mon body).</p>
</body>
</html>';

$headers = 'MIME-Version: 1.0' . "\r\n";
$headers .= 'Content-type: text/html; charset=utf-8' . "\r\n";

// En-têtes additionnels
$headers .= 'From: "Mon super site" <contact@supersite.com>' .
"\r\n";
$headers .= 'Cc: "Contact" <contact@supersite.com>' . "\r\n";
$headers .= 'Bcc: "Contact" <contact@supersite.com>' . "\r\n";
$headers .= 'Reply-To: "Members" <membres@supersite.com>' . "\r\n";

$mail = mail($to, $subject, $message, $headers); //marche

if($mail) echo 'ok';
else echo 'erreur.';
?>
```

Eh bien certes vous enverrez un mail ridicule et personne ne le lira, mais normalement, il arrivera en un seul morceau. 😞



Oh my god, on peut envoyer des mails en HTML !!

En effet, la fonction `mail` peut formater des mails en HTML, c'est ce qui va nous servir par la suite ; avec cette fonction, vous pourrez même faire de la mise en forme, envoyer des pièces jointes (images, etc.), vous donnerez ainsi des rendus agréables à lire.

Nous ici, on veut envoyer un mail à notre cher inscrit.

Il faut donc créer une fonction `inscription_mail()`, elle sera très basique, mais en partie II, nous vous proposerons d'envoyer des pièces jointes avec les mails, pour faire des belles newsletters comme le SdZ en fait. 😊

Sans plus attendre :

Code : PHP

```
<?php
function inscription_mail($mail, $pseudo, $passe)
{
    $to = $mail;
    $subject = 'Inscription sur '.TITRESITE.' - '.$pseudo;
?>
```

On définit le message. 😊

Code : PHP

```
<?php
    $message = '<html>
<head>
<title></title>
</head>

<body>
<div>Bienvenue sur '.TITRESITE.' !<br/>
Vous avez complété une inscription avec le pseudo
'.htmlspecialchars($pseudo, ENT_QUOTES).' à l\'instant.<br/>
Votre mot de passe est : '.htmlspecialchars($passe,
ENT_QUOTES).'<br/>
Veillez à le garder secret et à ne pas l\'oublier.<br/><br/>

En vous remerciant.<br/><br/>
Moi - Wembaster de '.TITRESITE.'
</body>
</html>';
?>
```

Ensuite viennent les headers additionnels (cherchez avant de lire le code) :

Secret (cliquez pour afficher)**Code : PHP**

```
<?php
//headers principaux.
$headers = 'MIME-Version: 1.0' . "\r\n";
$headers .= 'Content-type: text/html; charset=utf-8' . "\r\n";
//headers supplémentaires
$headers .= 'From: "Mon super site" <contact@supersite.com>' .
"\r\n";
$headers .= 'Cc: "Duplicata" <duplicata@supersite.com>' . "\r\n";
$headers .= 'Reply-To: "Membres" <membres@supersite.com>' .
"\r\n";
?>
```

Enfin, l'envoi (idem, facile). 😊

Secret (cliquez pour afficher)**Code : PHP**

```
<?php
$mail = mail($to, $subject, $message, $headers); //marche
```

```

if($mail) return true;
return false;
}
?>

```

Notre fonction est fin prête.

Il ne reste plus qu'à l'implémenter dans `trait_inscription.php`. 😊

Je vous laisse faire. 😊

Secret (cliquez pour afficher)

Code : PHP

```

<?php
    if(mysql_query($insertion))
    {
        if(inscription_mail($mail, $pseudo, $mdp)) $sent = 'Un mail
de confirmation vous a été envoyé.';
        else $sent = 'Un mail de confirmation devait être envoyé,
mais son envoi a échoué, vous êtes cependant bien inscrit.';
        vidersion();
        $_SESSION['inscrit'] = $pseudo;
        /*informe qu'il s'est déjà inscrit s'il actualise, si son
navigateur
bugue avant l'affichage de la page et qu'il recharge la page,
etc.*/
    }
    ?>
    <h1>Inscription validée !</h1>
    <p>Nous vous remercions de vous être inscrit sur notre site,
votre inscription a été validée !<br/>
    Vous pouvez vous connecter avec vos identifiants <a
href="connexion.php">ici</a><br/>.
    <?php echo $sent; ?>

```

On crée une variable `sent` qui contient le statut d'envoi ; si tout est bon, on affiche un message informatif, sinon on affiche une erreur (qui n'annule pas l'inscription).

Voilà : le plus dur dans la fonction `mail` , c'est de comprendre comment elle marche ; le reste, c'est du gâteau. 🍷

Pfiou, l'inscription est enfin finie...

J'espère que vous êtes contents de vous. 😊

Voilà, le prochain chapitre sera un TP, vous y créerez une page de connexion / déconnexion.

Prêts ? 😊

TP : connexion et déconnexion

Dans ce chapitre, nous allons voir comment nous connecter et nous déconnecter. Nous allons également faire un petit système utilisant les cookies pour faire une connexion automatique sans avoir besoin de rentrer les informations de connexion.

Vous verrez, ce n'est vraiment pas compliqué, en plus, ce n'est même pas long. 😊

C'est parti !

Prérequis et explications

Bon : avant de nous lancer dans notre premier TP ensemble, je vais résumer ce que j'attends de vous, et comment je vois les choses. 😊

connexion.php

Cette page, comme vous vous en doutez, contiendra le formulaire de connexion au site. Je voudrais que le formulaire de connexion et la page validant la connexion ne fassent justement qu'une seule et même page. Par conséquent, je vous invite à utiliser un champ `hidden` et à tester sa valeur pour savoir si vous êtes en phase validation ou non. 😊

Cette page vérifiera si le visiteur est connecté ou non, si le pseudo est bon, si le mot de passe est bon, s'il existe des doublons dans votre base de données (normalement c'est impossible, mais on est sûr de rien).

Si tout roule, on valide la connexion et on crée les variables de session. Sinon, on utilise `information.php`. 😊



Si le mot de passe est erroné, je veux que la page se rappelle du pseudo entré par le visiteur.

deconnexion.php

Cette page détruit sessions et cookies et appelle `information.php`. Rien de plus simple. 😊

Vous n'avez pas besoin de conseils, vous pouvez très bien y arriver tout seuls. 😊

Allez, c'est parti !

Correction

Bon, ben nous y voilà. 😊

Je vous donne les corrections, et j'expliquerai deux ou trois choses qui me semblent importantes. 😊

membres / connexion.php

Secret (cliquez pour afficher)

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page connexion.php

Permet de se connecter au site.

Quelques indications : (Utiliser l'outil de recherche et
rechercher les mentions données)
```

```

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Membre qui essaie de se connecter alors qu'il l'est déjà
Vous êtes bien connecté
Erreur de mot de passe
Erreur de pseudo doublon (normalement impossible)
Pseudo inconnu
-----
*/

session_start();
header('Content-type: text/html; charset=utf-8');
include('../includes/config.php');

/******Actualisation de la session...******/

include('../includes/fonctions.php');
connexionbdd();
actualiser_session();

/******Fin actualisation de session...******/

if(isset($_SESSION['membre_id']))
{
    $informations = Array(/*Membre qui essaie de se connecter alors
qu'il l'est déjà*/
        true,
        'Vous êtes déjà connecté',
        'Vous êtes déjà connecté avec le pseudo <span
class="pseudo">'.htmlspecialchars($_SESSION['membre_pseudo'],
ENT_QUOTES).'</span>.',
        ' - <a href="'.ROOTPATH.'/membres/deconnexion.php">Se
déconnecter</a>',
        ROOTPATH.'/index.php',
        5
    );

    require_once('../information.php');
    exit();
}

if($_POST['validate'] != 'ok')
{
/******Entête et titre de page******/

$title = 'Connexion';

include('../includes/haut.php'); //contient le doctype, et head.

/******Fin entête et titre******/
?>
    <div id="colonne_gauche">
    <?php
    include('../includes/colg.php');
    ?>
    </div>

    <div id="contenu">
        <div id="map">
            <a href="../index.php">Accueil</a> => <a
href="connexion.php">Connexion</a>
        </div>

        <h1>Formulaire de connexion</h1>

```

```

    <p>Pour vous connecter, indiquez votre pseudo et votre mot de
    passe.<br/>
    Vous pouvez aussi cocher l'option "Me connecter automatiquement
    à mon
    prochain passage." pour laisser une trace sur votre ordinateur
    pour être
    connecté automatiquement.<br/>
    Ce système de trace est basé sur les cookies, ce sont des
    petits fichiers
    contenant votre numéro d'identification ainsi qu'une version
    cryptée de votre
    mot de passe. Ces fichiers ne peuvent en aucun cas endommager
    votre ordinateur,
    ni l'affecter d'aucune façons, vous pourrez les supprimer à
    tout moment dans
    les options de votre navigateur.</p>

    <form name="connexion" id="connexion" method="post"
    action="connexion.php">
        <fieldset><legend>Connexion</legend>
        <label for="pseudo" class="float">Pseudo :</label> <input
        type="text" name="pseudo" id="pseudo" value="<?php
        if(isset($_SESSION['connexion_pseudo'])) echo
        $_SESSION['connexion_pseudo']; ?>"><br/>
        <label for="mdp" class="float">Passe :</label> <input
        type="password" name="mdp" id="mdp"><br/>
        <input type="hidden" name="validate" id="validate"
        value="ok"/>
        <input type="checkbox" name="cookie" id="cookie"/> <label
        for="cookie">Me connecter automatiquement à mon prochain
        passage.</label><br/>
        <div class="center"><input type="submit" value="Connexion"
        /></div>
    </fieldset>
    </form>

    <h1>Options</h1>
    <p><a href="inscription.php">Je ne suis pas inscrit !</a><br/>
    <a href="moncompte.php?action=reset">J'ai oublié mon mot de
    passe !</a>
    </p>
    <?php
}

else
{
    $result = sqlquery("SELECT COUNT(membre_id) AS nbr, membre_id,
    membre_pseudo, membre_mdp FROM membres WHERE
    membre_pseudo = '".mysql_real_escape_string($_POST['pseudo'])."'
    GROUP BY membre_id", 1);

    if($result['nbr'] == 1)
    {
        if(md5($_POST['mdp']) == $result['membre_mdp'])
        {
            $_SESSION['membre_id'] = $result['membre_id'];
            $_SESSION['membre_pseudo'] = $result['membre_pseudo'];
            $_SESSION['membre_mdp'] = $result['membre_mdp'];
            unset($_SESSION['connexion_pseudo']);

            if(isset($_POST['cookie']) && $_POST['cookie'] == 'on')
            {
                setcookie('membre_id', $result['membre_id'],
                time()+365*24*3600);
                setcookie('membre_mdp', $result['membre_mdp'],
                time()+365*24*3600);
            }

            $informations = Array(/*Vous êtes bien connecté*/
                false,

```

```

        'Connexion réussie',
        'Vous êtes désormais connecté avec le pseudo <span
class="pseudo">'.htmlspecialchars($_SESSION['membre_pseudo'],
ENT_QUOTES).'</span>.',
    ),
    ),
    ROOTPATH.'/index.php',
    3
);
require_once('../information.php');
exit();
}

else
{
    $_SESSION['connexion_pseudo'] = $_POST['pseudo'];
    $informations = Array(/*Erreur de mot de passe*/
        true,
        'Mauvais mot de passe',
        'Vous avez fourni un mot de passe incorrect.',
        ' - <a href="'.ROOTPATH.'/index.php">Index</a>',
        ROOTPATH.'/membres/connexion.php',
        3
    );
    require_once('../information.php');
    exit();
}

else if($result['nbr'] > 1)
{
    $informations = Array(/*Erreur de pseudo doublon
(normalement impossible)*/
        true,
        'Doublon',
        'Deux membres ou plus ont le même pseudo, contactez un
administrateur pour régler le problème.',
        ' - <a href="'.ROOTPATH.'/index.php">Index</a>',
        ROOTPATH.'/contact.php',
        3
    );
    require_once('../information.php');
    exit();
}

else
{
    $informations = Array(/*Pseudo inconnu*/
        true,
        'Pseudo inconnu',
        'Le pseudo <span
class="pseudo">'.htmlspecialchars($_POST['pseudo'],
ENT_QUOTES).'</span> n\'existe pas dans notre base de données.
Vous avez probablement fait une erreur.',
        ' - <a href="'.ROOTPATH.'/index.php">Index</a>',
        ROOTPATH.'/membres/connexion.php',
        5
    );
    require_once('../information.php');
    exit();
}
?>
</div>

<?php
include('../includes/bas.php');
mysql_close();
?>

```

membres / deconnexion.php

Secret (cliquez pour afficher)

Code : PHP

```

<?php
/*
Neoterranos & LkY
Page deconnexion.php

Permet de se déconnecter du site.

Quelques indications : (Utiliser l'outil de recherche et
rechercher les mentions données)

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations/erreurs :
-----
Déconnexion
-----
*/

session_start();
include('../includes/config.php');
include('../includes/fonctions.php');
connexionbdd();
mysql_query("DELETE FROM connectes WHERE connectes_id =
".$_SESSION['membre_id'] or exit(mysql_error());
vider_cookie();
session_destroy();

$informations = Array(/*Déconnexion*/
    false,
    'Déconnexion',
    'Vous êtes à présent déconnecté.',
    ' - <a href="'.ROOTPATH.'/membres/connexion.php">Se
connecter</a>',
    ROOTPATH.'/index.php',
    5
);

require_once('../information.php');
exit();
?>

```

connexion.php

Déjà, ma manière de faire qu'un formulaire et sa validation soient sur la même page consiste à utiliser une variable `post` qui se trouvera dans un `input` caché du formulaire. C'est une méthode ; on pourrait aussi vérifier un des champs du formulaire, mais pour des raisons pratiques, je ne le fais pas : comme ça, si je modifie le formulaire, je ne touche pas à ce champ et la page reste valable.

Niveau SQL, beaucoup de monde fait une requête type `SELECT * WHERE pseudo=$_POST['pseudo'] AND passe=$_POST['passe']` : je n'utilise pas cette méthode, non pas parce qu'elle est moins bien, juste parce que j'ai l'habitude de celle que je vous propose. En plus, ça me permettait de vous montrer comment sélectionner des données et faire un `COUNT()` en même temps ; en effet, essayez en retirant `GROUP BY membre_id`, vous aurez droit à une belle erreur.

À vrai dire, on se fiche pas mal de pouvoir faire un `count` et un `select` ici, mais bon, comme ça, vous le saurez à l'avenir. 😊

L'option *se souvenir de moi* est tout ce qu'il y a de plus banal.

Bref, cette correction vous révèle certains de mes tics en PHP, pas forcément mieux ni même bien, mais ce sont les miens : ce sont des automatismes, et je vous invite à les modifier s'ils ne vous plaisent pas, et à me dire en commentaire si pour une raison ou pour une autre, vous estimez que je devrais éditer ces codes pour utiliser des méthodes différentes.

Pour en revenir aux actions de la page, elle fait bien tout ce que je vous ai raconté :

Code : PHP

```
<?php
if(isset($_SESSION['membre_id']))
{
    $informations = Array(/*Membre qui essaie de se connecter alors
qu'il l'est déjà*/
        true,
        'Vous êtes déjà connecté',
        'Vous êtes déjà connecté avec le pseudo <span
class="pseudo">'.htmlspecialchars($_SESSION['membre_pseudo'],
ENT_QUOTES).'</span>.',
        ' - <a href="'.ROOTPATH.'/membres/deconnexion.php">Se
déconnecter</a>',
        ROOTPATH.'/index.php',
        5
    );

    require_once('../information.php');
    exit();
}
?>
```

Ici on vérifie le statut du visiteur, on pourrait le renvoyer bêtement à l'index, mais contrairement à l'inscription, la connexion est assez fréquente, il suffit que l'utilisateur ait le lien vers la connexion en historique et il se fait refouler sans précisions, contrairement à l'inscription, je mets un message.

Code : PHP

```
<?php
if($_POST['validate'] != 'ok')
?>
```

Voilà ma variable cachée en action. 😊

Code : PHP

```
<label for="pseudo" class="float">Pseudo :</label> <input
type="text" name="pseudo" id="pseudo" value="<?php
if(isset($_SESSION['connexion_pseudo'])) echo
$_SESSION['connexion_pseudo']; ?>"/><br/>
```

Ici la variable `souvenir` .

Le reste c'est du test de pseudo, mot de passe et affichage d'erreurs. 😊

deconnexion.php

Ici, rien de compliqué non plus : je vide les cookies, j'efface les sessions, et j'affiche un message pour confirmer que le membre

s'est déconnecté. 😊

Et après ?

Comme vous l'avez remarqué, ce n'était vraiment pas difficile. 😊

Ce chapitre était en fait vraiment simple, le but était de vous donner un peu de boulot, faudrait pas que je sois le seul à bosser. 😊

Sachez cependant que le prochain chapitre sera un TP peut-être un peu plus difficile : il faudra réfléchir et s'y retrouver entre les statistiques [stats.php](#) et l'affichage des profils [user.php](#).

Avant de nous quitter, je vous propose cependant quelques améliorations pour le système de connexion / déconnexion.

1°) Quand tout est bon, on valide, et un beau message (de préférence centré) apparaît pour dire au membre qu'il est connecté. À l'inverse, quand le mot de passe est mauvais, ou quand le pseudo est inconnu, on affiche aussi une erreur avec [information.php](#). Ça a des avantages (clarté), mais aussi des inconvénients (obligé de recharger la page [connexion.php](#)).

Pourquoi ne pas incorporer les messages d'erreur dans la page [connexion.php](#) ?

Il faut retravailler le code pour mettre un `div` avant le formulaire qui affichera les erreurs éventuelles.

2°) La déconnexion c'est sympa, mais il se peut (je ne sais pas) que le membre ne veuille pas se déconnecter, mais juste réinitialiser ses cookies ou les effacer.

Pourquoi ne pas mettre une page déconnexion avec les trois options et utiliser une variable `get` pour effectuer l'action voulue par l'utilisateur ?

Pareil, ça a ses plus et ses moins : l'utilisateur impatient dira que ça sert à rien, l'utilisateur pointilleux trouvera ça génial. 😊

3°) Et si, au lieu de faire une page [connexion](#) et une page [deconnexion](#), vous faisiez une seule page ([statut.php](#) par exemple) qui gère le tout (encore des variables `get`) ?

Là pour le coup, l'inconvénient est la gestion des cas, l'avantage est d'éviter d'avoir des pages avec seulement 15 lignes de code, et puis certains préfèrent comme ça. 😊

4°) Vous pourriez rajouter une redirection de [trait-inscription.php](#) vers [connexion.php](#) avec un `timer` de 5 à 10 secondes.

Voilà, ce sont quelques idées, juste pour vous rappeler que c'est vous qui personnalisez votre système, pas moi. 😊

Ce chapitre TP touche à sa fin.

Pour ceux qui n'ont pas réussi, ce n'est pas grave, vous ferez mieux dans pas longtemps. 😊

Je vous annonce que le prochain chapitre est aussi un TP, car il reprend essentiellement des points abordés chapitre 3.

Allons-y. 😊

TP : site et profils

Comme prévu, ce chapitre est aussi un TP ; cependant, il sera sensiblement plus long que le précédent, je vais donc le découper en deux parties. Une première portera sur la mise en place de statistiques (même si le terme est assez exagéré pour ce que nous allons faire), et de la page de visualisation des profils, tandis que la seconde partie portera sur l'édition de son profil ou de celui d'un membre (administration oblige). 😊

Pour ce TP, vous n'aurez besoin que des connaissances acquises jusque-là, nous pouvons donc y aller sans plus tarder.

Exercice 1 - Création de statistiques

Bon, nous sommes d'accord : le mot statistiques est assez ambigu, on ne va pas faire des choses ultra-poussées comme le nombre de visiteurs uniques par mois, le nombre de pages lues, le nombre de Français qui parcourent votre site et j'en passe ; cela, on y réfléchira peut-être en partie II, mais pour le moment, on se cantonne à des choses très simples. 😊

Ici, je veux donner une utilité à `colg.php`, je vais donc vous montrer ce que j'y ai mis, et vous allez essayer de me créer les pages et fonctions nécessaires pour faire marcher tout ce bazar !

Code : PHP

```
<div id="statistiques"><h1>Site</h1>
    <a href="<?php echo ROOTPATH; ?
>/stats.php?see=nb_membres"><?php echo $num1 = get('nb_membres');
if($num1 <= 1) echo ' membre inscrit'; else echo ' membres
inscrits'; ?></a><br/>
    <a href="<?php echo ROOTPATH; ?
>/stats.php?see=connectes"><?php echo $num2 = get('connectes');
if($num2 <= 1) echo ' visiteur'; else echo ' visiteurs' ?></a><br/>
    <a href="<?php echo ROOTPATH; ?
>/stats.php?see=passed">Dernières visites</a><br/>
</div>
```

Voilà, c'est pas bien gros, mais ça va vous occuper quelques temps. 😊

Détaillons ce que nous voulons.

À la racine de votre site (ou espace de test), créez le fichier `stats.php`.

Ce fichier recevra une variable `$_GET['see']` qui lui permettra de savoir quoi faire.

Les différents cas

nb_membres.php

Si la variable `$_GET['see']` vaut `nb_membres`, alors `stats.php` inclura `stats/nb_membres.php` qui lui affichera une liste de tous les membres inscrits, avec leur pseudo et dates d'inscription (mis en page).

Nous essayerons d'améliorer cette liste en partie II.

connectes.php

Si la variable `$_GET['see']` vaut `connectes`, alors `stats.php` inclura la page `stats/connectes.php` qui dépendra elle-même d'une variable `$_GET['a']`.

La page `connectes.php` devra afficher une liste des membres inscrits connectés si `$_GET['a']` diffère de 1 ou 2.

Elle montrera une liste des invités connectés si `$_GET['a']` vaut 2, enfin elle proposera une liste complète si `$_GET['a']` vaut 1.



Mais... Mais ?! Tu veux qu'on la sorte d'où, la liste des connectés ?!

Ah tiens, excellente question !

Il est temps de créer une deuxième table SQL !

Code : SQL

```

CREATE TABLE `connectes` (
  `connectes_id` int(11) NOT NULL, --id du membre connecté, vaut -1
  si le membre est un invité.
  `connectes_ip` varchar(16) NOT NULL, --l'ip du connecté, fera
  appel à une fonction getIp().
  `connectes_membre` varchar(16) NOT NULL, --est-ce que le connecté
  est un membre ? Vaut 1 si oui, vaut l'adresse IP du connecté sinon.
  `connectes_actualisation` int(11) NOT NULL, --dernière
  actualisation
  UNIQUE KEY `membre_id` (`connectes_id`,`connectes_membre`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
*/

```

Bien : maintenant, nous allons donc devoir faire des modifications à `actualiser_session()` !
 Premièrement, à chaque fois que le membre actualise une page, sa session est vérifiée et validée à nouveau, il faut donc qu'à chaque fois que le membre actualise une page, on mette à jour le champ `membre_derniere_visite` avec un code dans ce goût :

Code : PHP

```

<?php
    mysql_query("UPDATE membres SET membre_derniere_visite =
    ".time()." WHERE membre_id = ".$_SESSION['membre_id']) or
    exit(mysql_error());
    queries();
?>

```

Enfin, il va falloir mettre à jour la table des connectés à chaque actualisation de page ; mettez ce code juste avant de fermer la fonction :

Code : PHP

```

<?php
    if(isset($_SESSION['membre_id'])) $id =
    $_SESSION['membre_id'];
    else $id = -1;
    updateConnectes($id);
?>

```

Vous avez donc une fonction supplémentaire à définir.

Alors là j'en vois venir certains : à quoi sert le champ `connectes_actualisation` dans la table `connectes` si l'on met à jour les deux à chaque fois ?

Eh bien la table des connectés est débarrassée des champs où l'actualisation dépasse les 300 secondes (5 minutes) : donc si on se base dessus, on ne peut pas savoir quand un membre est passé pour la dernière fois.

A contrario, si on se base sur le champ de la table `membres`, comment sait-on quand un membre est connecté ou non, et comment recenser les invités ? Au final, nous ne gagnerions rien à ne pas utiliser deux champs au lieu d'un.

Enfin, la fonction `getIp()` est définie ainsi :

Code : PHP

```

<?php
function getIp()
{
    if(isset($_SERVER['HTTP_X_FORWARDED_FOR'])) return
    $_SERVER['HTTP_X_FORWARDED_FOR'];
    else return $_SERVER['REMOTE_ADDR'];
}

```

```
?>
```

Elle permet de mieux identifier quelqu'un qui passerait par un proxy transparent, par exemple, celui de son université ou autre. Certains hébergeurs gratuits (olympic network, par exemple) utilisent un proxy intermédiaire pour sécuriser leur infrastructure, si bien que pour avoir les IP des membres, j'ai été obligé de mettre cette fonction en place. Cette fonction ne peut en aucun cas retourner l'IP d'un membre qui passerait par un proxy visant à le rendre anonyme ; un proxy, je vous le rappelle, permet de passer par un ordinateur avant d'envoyer une requête à votre serveur. Si certains ne sont même pas conscients de passer par un proxy (exemple d'une université, d'une bibliothèque), d'autres le font pour pouvoir avoir accès à des données réservées à des habitants d'un pays précis ou pour pirater en toute tranquillité (ou presque... :o)). Sans lancer un débat sur le bien-fondé des proxies, il peut être utile d'utiliser cette fonction.

Bref, pour obtenir l'IP de quelqu'un, il ne vous reste plus qu'à faire :

Code : PHP

```
<?php
$ip = getIp();
?>
```

Pratique. 😊

Bien, nous en avons, je pense, fini avec `connectes.php`.

`passed.php`

Si la variable `$_GET['see']` vaut `passed`, alors `stats.php` inclura la page `stats/passed.php`.

Cette page, au premier abord similaire à `connectes.php`, est directement tirée du SdZ (un peu comme toutes les mises en page que je vous propose, l'objectif étant d'avoir un joli rendu). 😊

Elle affichera une liste exhaustive des membres avec leur pseudo, leur date de dernière connexion, ainsi que leur statut actuel (connecté ou non).

Elle ne changera donc pas dans la partie II.

`Autre cas : index.php`

Si la variable `$_GET['see']` a une autre valeur, alors `stats.php` inclura la page `stats/index.php`.

Cet index des stats contiendra une liste des pages de statistiques accessibles (3), ainsi que d'autres pages de statistiques que vous auriez pu mettre en place vous-mêmes.

Qu'on soit bien clairs : cette page n'est pas indispensable, vous pouvez mettre une redirection si vous le préférez. 😊



Les pages `nb_membres.php`, `connectes.php`, `passed.php` et `index.php` sont toutes les quatre dans un dossier `stats` à la racine de votre site (ou dossier de tests), sinon ma correction diffèrera de votre système !

Bien, bien, maintenant que nous avons planté le décor, je pense qu'il est temps pour vous d'entrer en scène : pendant ce temps, je vais aller mettre mon costume de prof.

Bonne chance !

Correction de l'exercice 1

Alors, ça bosse dur ? 😊

Voilà sans plus attendre une correction détaillée de l'exercice. 😊

`colg.php` (je la remets au cas où)

Code : PHP

```

<?php
/*
Neoterranos & LkY
Page colg.php

La colonne de gauche de votre site (page incluse).

Quelques indications (utiliser l'outil de recherche et rechercher
les mentions données) :

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations / erreurs :
-----
Aucune information / erreur
-----
*/
?>

<div id="statistiques"><h1>Site</h1>
    <a href="<?php echo ROOTPATH; ?
>/stats.php?see=nb_membres"><?php echo $num1 = get('nb_membres');
if($num1 <= 1) echo ' membre inscrit'; else echo ' membres
inscrits'; ?></a><br/>
    <a href="<?php echo ROOTPATH; ?
>/stats.php?see=connectes"><?php echo $num2 = get('connectes');
if($num2 <= 1) echo ' visiteur'; else echo ' visiteurs' ?></a><br/>
    <a href="<?php echo ROOTPATH; ?
>/stats.php?see=passed">Dernières visites</a><br/>
</div>

```

On définit donc get () .

Code : PHP

```

<?php
function get($type) //je ne me suis pas foulé. :D
{
    if($type == 'nb_membres')
    {
        $count = sqlquery("SELECT COUNT(*) AS nbr FROM
membres", 1);
        return $count['nbr'];
    }

    else if($type == 'connectes')
    {
        $count = sqlquery("SELECT COUNT(*) AS nbr FROM
connectes", 1);
        return $count['nbr'];
    }

    else
    {
        return 0;
    }
}
?>

```

On définit aussi une fonction de mise en page de la date (inspirée de la gestion des dates du SdZ) :

Code : PHP

```

<?php
function mepd($date)
{
    if(intval($date) == 0) return $date;

    $stampon = time();
    $diff = $stampon - $date;

    $dateDay = date('d', $date);
    $stamponDay = date('d', $stampon);
    $diffDay = $stamponDay - $dateDay;

    if($diff < 60 && $diffDay == 0)
    {
        return 'Il y a '.$diff.'s';
    }

    else if($diff < 600 && $diffDay == 0)
    {
        return 'Il y a '.floor($diff/60).'m et
'.floor($diff%60).'s';
    }

    else if($diff < 3600 && $diffDay == 0)
    {
        return 'Il y a '.floor($diff/60).'m';
    }

    else if($diff < 7200 && $diffDay == 0)
    {
        return 'Il y a '.floor($diff/3600).'h et
'.floor(($diff%3600)/60).'m';
    }

    else if($diff < 24*3600 && $diffDay == 0)
    {
        return 'Aujourd\'hui à '.date('H\hi', $date);
    }

    else if($diff < 48*3600 && $diffDay == 1)
    {
        return 'Hier à '.date('H\hi', $date);
    }

    else
    {
        return 'Le '.date('d/m/Y', $date).' à '.date('h\hi',
$date).'.';
    }
}
?>

```

Vous remarquerez que j'ai vraiment un problème avec les switch, je ne les aime vraiment pas. 🤔

C'est notre point de départ, à partir d'ici, vous avez normalement créé `stats.php` :

Code : PHP

```

<?php
/*
Neoterranos & LkY
Page stats.php

Statistiques du site.

Quelques indications (utiliser l'outil de recherche et rechercher
les mentions données) :

```

```

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations / erreurs :
-----
$_GET['see'] contient des caractères invalides (tentative de hack ?)
-----
*/

session_start();
header('Content-type: text/html; charset=utf-8');
include('includes/config.php');

/*****Actualisation de la session...*****/

include('includes/fonctions.php');
connexionbdd();
actualiser_session();

/*****Fin actualisation de session...*****/

if($_GET['see'] == '' || !isset($_GET['see']))
{
    include('stats/index.php');
}

else
{
    if(strpos($_GET['see'], '.') !== FALSE ||
strpos($_GET['see'], ':') !== FALSE || strpos($_GET['see'], 'http')
!== FALSE) //$_GET['see'] contient des caractères invalides
(tentative de hack ?)
    {
        include('stats/index.php');
    }

    else if(file_exists('stats/'.$_GET['see'].'.php'))
    {
        include('stats/'.$_GET['see'].'.php');
    }

    else
    {
        include('stats/index.php');
    }
}

include('includes/bas.php');
mysql_close();
?>

```

Voilà, j'ai procédé ainsi pour inclure autant de fichiers que voulu, cette méthode est sécurisée. Il n'y a, a priori, rien que vous ne puissiez comprendre. 😊

Passons à stats/nb_connectes.php

Comme vous l'avez vu, nous n'avons absolument pas donné de nom à notre page, le nom sera défini dans la page incluse :

Code : PHP

```

<?php
/*
Neoterranos & LkY

```

```

Page nb_membres.php

Affiche une liste complète des membres.

Quelques indications (utiliser l'outil de recherche et rechercher
les mentions données) :

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations / erreurs :
-----
Aucune information / erreur
-----
*/

/*****En-tête et titre de page*****/

$titre = 'Liste des membres du site';

include('includes/haut.php'); //contient le doctype, et head.

/*****Fin en-tête et titre*****/
?>
    <div id="colonne_gauche">
    <?php
    include('includes/colg.php');
    ?>
    </div>

    <div id="contenu">
        <div id="map">
            <a href="index.php">Accueil</a> =>
<a href="stats.php?see=nb_membres">Liste des membres</a>
        </div>

```

Jusqu'ici, rien de bien nouveau. 😊

Maintenant, nous allons, via une requête, récupérer tous les membres, les classer par ID (SQL le fait très bien), et les afficher dans un tableau, je pense d'ailleurs qu'il peut être utile de revoir le [chapitre de M@teo sur les tableaux](#) (cours XHTML/CSS) :

Code : PHP

```

<?php
    $membre_query = sqlquery("SELECT membre_id,
membre_pseudo, membre_inscription
FROM membres
ORDER BY membre_id ASC", 2); //ORDER BY s'occupe du tri !
    $i = 0;
    ?>
        <div class="membre_liste">
            <table>
                <thead>
                    <th>N° du
membre</th>
                    <th>Pseudonyme</th>
                    <th>Date
d'inscription</th>
                </thead>
                <tfoot>
                    <th>N° du
membre</th>
                    <th>Pseudonyme</th>
                    <th>Date

```

```
d'inscription</th>
```

```
</tfoot>
```

Ici, on a préparé la tête et le pied du tableau, n'oubliez pas qu'on les met avant la balise `<tbody>` !



Mais tu ne fais pas de `limit` dans ta requête SQL ?

Non, pour le moment, on n'en aura pas besoin, mais effectivement, quand on a 113 326 membres inscrits sur son site, et une requête à rallonge avec des jointures pour savoir si le membre est dans tel ou tel groupe (notez que le SdZ semble avoir une façon plus simple de faire : max un groupe pour un membre, donc peut-être pas de requête à rallonge), cela peut devenir utile. 😊
Cependant, on aura besoin de définir une fonction page dans ce cas-là, qui gèrera la position de la page visualisée par rapport à la première et à la dernière page, donc on verra cela plus tard, ce n'est pas indispensable tout de suite.

Affichons maintenant les membres :

Code : PHP

```
<tbody>
    <?php
    while(isset($membre_query[$i]))
    {
        echo '<tr class="ligne_' . ($i%2) . '">
    <td>'. $membre_query[$i]['membre_id'] . '</td>
    <td><a href="membres/user.php?
    id=' . $membre_query[$i]['membre_id'] . '">'. htmlspecialchars($membre_query[$i]['men
    ENT_QUOTES) . '</a></td>
    <td>' . mepd($membre_query[$i]['membre_inscription']) . '</td>
    </tr>

    ';
        $i++;
    }

    if($i == 0) echo '<tr><td colspan="3">Pas de membre trouvé.</td>
    ?>
    </tbody>
    </table>
    </div>
</div>
```

Et un peu de CSS :

Code : CSS

```
table {
    margin: auto;
    margin-top: 10px;
    border: 1px solid black;
    text-align: center;
    border-collapse: collapse;
}

th {
    border: 1px solid black;
    text-align: center;
}

td {
    border: 1px solid black;
    text-align: center;
}
```

```
.ligne_0 {
    background-color: #dfe5f5;
}

.ligne_1 {
    background-color: #edf1f8;
}
```

Voilà ! Votre page `nb_membres` est finie !
Vous pouvez l'admirer avec joie !

Passons à l'affichage des connectés, et donc la création de fonctions supplémentaires.

actualiser_session() :

Code : PHP

```
<?php
function actualiser_session()
{
    if(isset($_SESSION['membre_id']) &&
intval($_SESSION['membre_id']) != 0) //Vérification id
    {
        //utilisation de la fonction sqlquery, on sait
        qu'on aura qu'un résultat car l'id d'un membre est unique.
        $retour = sqlquery("SELECT membre_id, membre_pseudo,
membre_mdp FROM membres WHERE membre_id =
".$_SESSION['membre_id'], 1);

        //Si la requête a un résultat (id est : si l'id
        existe dans la table membres)
        if(isset($retour['membre_pseudo']) &&
$retour['membre_pseudo'] != '')
        {
            if($_SESSION['membre_mdp'] !=
$retour['membre_mdp'])
            {
                //Dehors vilain pas beau !
                $informations = Array(/*Mot de passe
de session incorrect*/

                true,

                'Session invalide',

                'Le mot de passe de votre session est incorrect, vous devez vous
reconnecter.',

                '',

                'membres/connexion.php',

                3

            );

            require_once('../information.php');
            vider_cookie();
            session_destroy();
            exit();
        }

        else
        {
            //Validation de la session.
            $_SESSION['membre_id'] =
$_SESSION['membre_id'];
            $_SESSION['membre_pseudo'] =
```

```

$retour['membre_pseudo'];
                                $_SESSION['membre_mdp'] =
$retour['membre_mdp'];
                                mysql_query("UPDATE membres
SET membre_derniere_visite = ".time()." WHERE membre_id =
"$_SESSION['membre_id']) or exit(mysql_error());
                                queries();
                                }
                                }
                                }

else //On vérifie les cookies et sinon pas de session
{
    if(isset($_COOKIE['membre_id']) &&
isset($_COOKIE['membre_mdp'])) //S'il en manque un, pas de session.
    {
        if(intval($_COOKIE['membre_id']) != 0)
        {
            //idem qu'avec les $_SESSION
            $retour = sqlquery("SELECT
membre_id, membre_pseudo, membre_mdp FROM membres WHERE membre_id =
".intval($_COOKIE['membre_id']), 1);

            if(isset($retour['membre_pseudo'])
&& $retour['membre_pseudo'] != '')
            {
                if($_COOKIE['membre_mdp'] !=
$retour['membre_mdp'])
                {
                    //Dehors vilain tout
                    $informations =
Array(/*Mot de passe de cookie incorrect*/
true,
'Mot de passe cookie erroné',
'Le mot de passe conservé sur votre cookie est incorrect, vous devez
vous reconnecter.',
'',
'membres/connexion.php',
3
);
                require_once('../information.php');
                                vider_cookie();
                                session_destroy();
                                exit();
                                }
                                else
                                {
                                    //Bienvenue :D

$_SESSION['membre_id'] = $retour['membre_id'];
$_SESSION['membre_pseudo'] = $retour['membre_pseudo'];
$_SESSION['membre_mdp'] = $retour['membre_mdp'];
                                mysql_query("UPDATE
membres SET membre_derniere_visite = ".time()." WHERE membre_id =
"$_SESSION['membre_id']) or exit(mysql_error());
                                queries();
                                }
                                }
}

```

```

    }

    else //cookie invalide, erreur plus
    suppression des cookies.
    {
        $informations = Array(/*L'id de
    cookie est incorrect*/
true,
    'Cookie invalide',
    'Le cookie conservant votre id est corrompu, il va donc être
    détruit, vous devez vous reconnecter.',
    '',
    'membres/connexion.php',
3
);

        require_once('../information.php');
        vider_cookie();
        session_destroy();
        exit();
    }
}

else
{
    //Fonction de suppression de toutes les
    variables de cookie.
    if(isset($_SESSION['membre_id']))
    unset($_SESSION['membre_id']);
    vider_cookie();
}

if(isset($_SESSION['membre_id'])) $id =
$_SESSION['membre_id'];
else $id = -1;
updateConnectes($id);
}
?>

```

updateConnectes() et getIp() :

Code : PHP

```

<?php
function updateConnectes($id)
{
    $ip = getIp();
    if($id != -1)
    {
        $id = $_SESSION['membre_id'];
        $additionnal = 1; //la variable à mettre dans
connectes_membre
    }

    else
    {
        $additionnal = $ip;
    }

    mysql_query("DELETE FROM connectes WHERE

```

```

connectes_actualisation < ".(time()-300) or exit(mysql_error());
//MàJ générale des connectés
mysql_query("INSERT INTO connectes VALUES(".$id.",
'".$ip."', '".$$additionnal."', ".time().")
ON DUPLICATE KEY UPDATE connectes_actualisation=".$time().",
connectes_ip='".$ip."'") or exit(mysql_error()); //tiens, tiens, ON
DUPLICATE... :o
        queries(2);
    }

function getIp()
{
    if(isset($_SERVER['HTTP_X_FORWARDED_FOR'])) return
$_SERVER['HTTP_X_FORWARDED_FOR'];
    else return $_SERVER['REMOTE_ADDR'];
}
?>

```

Quelques explications s'imposent. 😊

Je vous ai fait utiliser un champ `connectes_membre`, dans lequel on insère ici `$additionnal` : à quoi ce champ sert-il ? Si vous avez bien regardé la structure de la table des connectés, vous avez dû remarquer ceci :

Code : SQL

```

UNIQUE KEY `membre_id` (`connectes_id`,`connectes_membre`)

```

Cette ligne place une condition d'unicité sur deux champs en même temps : `connectes_id` et `connectes_membre`. Une autre façon de voir la ligne SQL que j'ai citée, c'est de l'interpréter en français : "Si, au moment d'une insertion, la valeur de `connectes_id` et la valeur de `connectes_membre` sont déjà **toutes les deux prises sur une même ligne**, retourner une erreur."

L'intérêt est en effet de permettre à plusieurs visiteurs d'être présents en même temps, donc de comptabiliser chaque IP comme un visiteur, tout en ne permettant à un membre inscrit de n'être comptabilisé qu'une fois, donc que son IP ne soit pas une condition d'unicité, mais cette fois son id et son statut de membre !

Donc si le membre ayant l'*id 1* est connecté, le champ `connectes_id` vaut 1, et le champ `connectes_membre` vaut 1 aussi, s'il actualise, on lui refuse l'insertion d'une nouvelle ligne, car elle transgresse la condition d'unicité ; par contre, pour les invités, le champ `connectes_membre` valant leur IP, on peut avoir sans problèmes plusieurs invités connectés en même temps pour un peu qu'ils aient des IP différents. 😊

Cependant, un problème se pose si un membre ou un invité déjà enregistré dans la table `connectes` actualise une page, l'insertion étant refusée, il faut donc faire un UPDATE. 😊

C'est ici que `ON DUPLICATE KEY UPDATE` intervient.

Plutôt que de faire une nouvelle requête pour mettre à jour la connexion, on appose cette condition en fin de requête d'insertion et les champs à mettre à jour.

Cela revient à dire en français : "Si l'insertion transgresse la condition d'unicité, mettre à jour le champ `connectes_actualisation` à la valeur `time()`."

Ainsi, on condense deux requêtes en une. 😊

C'est vrai qu'à première vue, ce n'est pas évident ; j'ai essayé de vous expliquer cela le plus simplement possible, mais je sais déjà que mes explications ne conviendront pas à certains, sur la condition d'unicité je ne peux rien faire de plus. Par contre, pour `ON DUPLICATE KEY UPDATE`, vous pouvez vous rendre [ici](#) (attention, page en anglais). 😊

On peut enfin attaquer `connectes.php`

Code : PHP

```

<?php
/*

```

```

Neoterranos & LkY
Page connectes.php

Affiche une liste complète des membres connectes.

Quelques indications (utiliser l'outil de recherche et rechercher
les mentions données) :

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations / erreurs :
-----
Aucune information / erreur
-----
*/

/*****En-tête et titre de page*****/

$titre = 'Liste des connectés';

include('includes/haut.php'); //contient le doctype, et head.

/*****Fin en-tête et titre*****/
?>
    <div id="colonne_gauche">
    <?php
    include('includes/colg.php');
    ?>
    </div>

    <div id="contenu">
        <div id="map">
            <a href="index.php">Accueil</a> =>
<a href="stats.php?see=connectes&a=<?php echo intval($_GET['a']); ?
>">Liste des connectés</a>
        </div>

        <h1>Liste des connectés</h1>

```

Ensuite, vient le moment de s'occuper du menu, et donc de la requête SQL à effectuer en fonction des liens sélectionnés. Moi je ne m'embête pas, je fais une condition if/else if, un switch est aussi possible. 😊

Code : PHP

```

<?php
    if($_GET['a'] == 1)
    {
        $id = '';
    }

    else if($_GET['a'] == 2)
    {
        $id = ' AND connectes_id = -1';
    }

    else
    {
        $id = ' AND connectes_id <> -1';
    }

    $membre_query = sqlquery("SELECT membre_id,
membre pseudo, connectes actualisation

```

```
FROM connectes
LEFT JOIN membres ON membre_id = connectes_id
WHERE connectes_actualisation > ".(time()-300).$id."
ORDER BY connectes_actualisation DESC", 2);
?>
```

C'est un peu du bidouillage, mais je préfère cela plutôt que de monter trois requêtes différentes. 😊

Passons au mini-menu : comme on veut qu'il soit centré, on va le prendre dans un div avec la propriété CSS qui convient :

Code : PHP

```
<!--menu/-->
<div class="center">
  <?php
    if($_GET['a'] == 1 ||
$_GET['a'] == 2)
    {
      echo '<a
href="stats.php?see=connectes">Voir seulement les membres
connectés</a>';
      $avt = 1;
    }
    if($_GET['a'] != 2)
    {
      if($avt == 1) echo '
- ';
      echo '<a
href="stats.php?see=connectes&a=2">Voir seulement les invités
connectés</a>';
      $avt = 1;
    }
    if($_GET['a'] != 1)
    {
      if($avt == 1) echo '
- ';
      echo '<a
href="stats.php?see=connectes&a=1">Voir les membres et les invités
connectés</a>';
    }
  ?>
</div>
```

Ici aussi, nous n'affichons que les menus proposant une page différente de celle que l'on visualise actuellement, donc on s'arrange pour rendre cela joli, même s'il faut du coup "bidouiller" un peu (je n'aime pas le mot bidouiller dans ce cas-là, car ce n'est pas vraiment du bidouillage, quelle que soit la méthode employée, le principe est le même, sauf si l'on affiche les trois menus, là au moins on ne s'ennuie pas, même pas besoin de PHP).

Il ne reste plus qu'à mettre le tableau en place, avec des liens pointant sur les membres, mais pas sur les invités :

Code : PHP

```
<!--tableau/-->
<table>
  <thead>
    <th>N° du membre</th>
    <th>Pseudonyme</th>
    <th>Dernière connexion</th>
  </thead>
  <tfoot>
    <th>N° du membre</th>
    <th>Pseudonyme</th>
    <th>Dernière connexion</th>
```

```

                                </tfoot>

                                <tbody>

<?php
$i = 0;
while(isset($membre_query[$i]))
{
    if($membre_query[$i]['membre_id'] != '')
    {
        $lien = '<a href="membres/user.php?
id='.$membre_query[$i]['membre_id'].'">';
        $lien2 = '</a>';
    }
    else
    {
        $lien = '';
        $lien2 = '';
    }

    if($membre_query[$i]['membre_id'] == '') //un invité
n'a pas de lien, ni d'id.
    {
        $membre_query[$i]['membre_id'] = 'Non
renseigné'; //
        $membre_query[$i]['membre_pseudo'] = 'Invité
';
    }

    echo '<tr class="ligne_' . ($i%2) . '">
<td>'.$membre_query[$i]['membre_id'].'</td>
<td>'.$lien.htmlspecialchars($membre_query[$i]['membre_pseudo'],
ENT_QUOTES).$lien2.'</td>
<td>'.mepd($membre_query[$i]['connectes_actualisation']).'</td>
</tr>

';
        $i++;
    }

    if($i == 0) echo '<tr><td colspan="3">Pas de membre
trouvé.</td></tr>';
    ?>
                                </tbody>
                                </table>
                                </div>

```

Voilà pour `connectes.php`. 😊

passed.php

Allez, on attaque `passed.php`, elle se passe d'explications, elle est très similaire aux précédentes.

Code : PHP

```

<?php
/*
Neoterranos & LkY
Page passed.php

Affiche une liste complète des dates de visite des membres.

Quelques indications (utiliser l'outil de recherche et rechercher les mentions c

Liste des fonctions :
-----

```

```

Aucune fonction
-----

Liste des informations / erreurs :
-----
Aucune information / erreur
-----
*/

/*****En-tête et titre de page*****/

$titre = 'Liste des dernières visites';

include('includes/haut.php'); //contient le doctype, et head.

/*****Fin en-tête et titre*****/
?>
    <div id="colonne_gauche">
    <?php
    include('includes/colg.php');
    ?>
    </div>

    <div id="contenu">
        <div id="map">
            <a href="index.php">Accueil</a> => <a href="stat
see=passed">Liste des visites</a>
        </div>

        <h1>Liste des visites</h1>

        <?php
        $membre_query = sqlquery("SELECT membre_id, membre_pseudo,
membre_derniere_visite, connectes_id
FROM membres
LEFT JOIN connectes
ON membre_id = connectes_id
ORDER BY membre_derniere_visite DESC", 2);
        $i = 0;
        ?>
            <div class="membre_liste">
                <table>
                    <thead>
                        <th>N° du membre</th>
                        <th>Pseudonyme</th>
                        <th>Dernière connexion</th>
                        <th>Statut</th>
                    </thead>

                    <tfoot>
                        <th>N° du membre</th>
                        <th>Pseudonyme</th>
                        <th>Dernière connexion</th>
                        <th>Statut</th>
                    </tfoot>

                    <tbody>
                        <?php
                        while(isset($membre_query[$i]))
                        {
                            if($membre_query[$i]['connectes_id'] ==
$membre_query[$i]['membre_id']) //gestion des statuts de connexion
                            {
                                $statut = '<span class="actif">Connecté</span>';
                            }

                            else
                            {
                                $statut = '<span class="inactif">Déconnecté</spa

```

```

        }

        echo '<tr class="ligne_' . ($i%2) . '">
<td>' . $membre_query[$i]['membre_id'] . '</td>
<td><a href="membres/user.php?
id=' . $membre_query[$i]['membre_id'] . '">' . htmlspecialchars($membre_query[$i]['men
ENT_QUOTES) . '</a></td>
<td>' . mepd($membre_query[$i]['membre_derniere_visite']) . '</td>
<td>' . $statut . '</td>
</tr>

';

        $i++;
    }

    if($i == 0) echo '<tr><td colspan="4">Pas de membre trouvé.</td>
?>
</tbody>
</table>
</div>
</div>

```

Ici, on ne gère pas les invités. 🤖

Il ne reste plus qu'à mettre l'index des stats :

Code : PHP

```

<?php
/*
Neoterranos & LkY
Page index.php

Index des statistiques du site (page incluse).

Quelques indications (utiliser l'outil de recherche et rechercher
les mentions données) :

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations / erreurs :
-----
Aucune information / erreur
-----
*/

/*****En-tête et titre de page*****/

$titre = 'Statistiques du site';

include('includes/haut.php'); //contient le doctype, et head.

/*****Fin en-tête et titre*****/
?>
        <div id="colonne_gauche">
        <?php
        include('includes/colg.php');
        ?>
        </div>

        <div id="contenu">
            <div id="map">

```

```

                                <a href="index.php">Accueil</a> =>
<a href="stats.php">Statistiques</a>
                                </div>

                                <h1>Statistiques</h1>

                                Bienvenue sur la page des statistiques du
site.<br/>
                                Ici, vous pourrez voir les statistiques
concernant les membres, les forums, les news, etc., etc. :)<br/>
                                Bonne visite !

                                <h2>Membres</h2>
                                <div class="center">
                                    -> <a href="stats.php?
see=nb_membres">Il y a <?php echo $num1; if($num1 <= 1) echo '
membre inscrit'; else echo ' membres inscrits'; ?>.</a><br/>
                                    -> <a href="stats.php?
see=connectes">Il y a <?php echo $num2; if($num2 <= 1) echo '
visiteur'; else echo ' visiteurs' ?>.</a><br/>
                                    -> <a href="stats.php?
see=passed">Voir les dernières visites de chaque membre.</a><br/>
                                </div>
                                </div>

```

Notez juste que je réutilise les variables définies dans `colg.php` ; si vous supprimez le bloc sur les stats dans cette page, alors vous devrez les définir ici.

On définit enfin trois nouvelles propriétés CSS :

Code : CSS

```

.actif {
    color: blue;
    font-weight: blod;
}

.inactif {
    color: red;
    font-weight: bold;
}

h2 {
    text-align:center;
    text-decoration: underline;
    color: #507090;
    font-family:"trebuchet ms", verdana, sans-serif;
    letter-spacing:1.1px;
    font-variant:small-caps;
    font-weight:bold;
    font-size:19px;
    margin-top:0;
}

```

Voilà, j'espère que vous avez tout compris. 😊

Passons à l'exercice II !

Exercice 2 - Création de la page `user.php`

Bien, les statistiques sont finies, et vous avez vu que l'on y a mis un lien vers une page `user.php` (qui se trouvera dans le dossier `membres`).

Il s'agit donc pour vous de la créer. 😊

Le Site du Zéro a une façon d'afficher les profils que je trouve particulièrement réussie, je vous suggère d'essayer de faire la même.

Il y a donc plusieurs choses que je veux y voir :

- le titre de la page doit contenir le pseudo du membre ;
- le statut de connexion du membre doit être visible ;
- sa date de naissance doit être visible ;
- son âge (calculé avec la date de naissance) doit être visible ;
- l'e-mail, le MSN, le yahoo ou toute autre messagerie doit être affichée par image.

Précisons le dernier point

Comme vous le savez, les adresses e-mail des membres sont a priori des données privées ; pour le moment, dans le cadre de notre système simple, on ne propose rien au membre pour le cacher, donc les robots spammeurs n'ont qu'à venir, lire les mails, et spammer les membres.

Le fait de les afficher par image est censé (je dis bien *censé*) leur compliquer la tâche.

Ici, ce n'est pas le cas, car je vais vous demander de créer un fichier `image.php` qui affiche bêtement le texte qu'on lui file en get, donc forcément, le mail sera dans le code source, donc lisible par un bot.

L'objectif ici est juste de vous sensibiliser, on verra comment régler ce problème efficacement dans le TP suivant, en créant ces images et en les enregistrant dans un dossier spécifique lors de la modification du profil d'un membre ou de son inscription.



Une idée qui pourrait vous venir à l'esprit est de créer `image.php` et de la faire marcher avec deux GET ; un premier prendrait le pseudo, le deuxième le type de mail (email, msn, yahoo, gmail ou autre...), le souci est que cela imposerait une requête par image, lourd pour afficher un simple profil.

On modifiera donc dans le prochain TP la page `user.php`, la page `image.php`, la page `trait-inscription.php`. 😊

Pour le moment on s'en fiche pas mal. 😊

Pour le reste, il s'agit d'un bête affichage de données provenant de la BDD, vous savez faire ça depuis maintenant quelques temps. 😊

Bonne chance !

Correction de l'exercice 2

Alors, c'est fini ?

On peut donc corriger votre travail. 😊

On suivra la page en respectant les points que je vous avais donnés.

Tout d'abord, le pseudo doit figurer dans le titre de la page `user.php`, cela implique de faire `sqlquery()` avant d'inclure `haut.php` et de définir le titre.

D'abord, lançons notre page :

Code : PHP

```
<?php
/*
Neoterranos & LkY
Page index.php

Affichage de profil.

Quelques indications (utiliser l'outil de recherche et rechercher les mentions c

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations / erreurs :
-----
```

```

Le membre n'existe pas
-----
*/

session_start();
header('Content-type: text/html; charset=utf-8');
include('../includes/config.php');

/*****Actualisation de la session...*****/

include('../includes/fonctions.php');
connexionbdd();
actualiser_session();

/*****Fin actualisation de session...*****/

```

Maintenant, passons à la gestion avant affichage :

Code : PHP

```

<?php
/*****Gestion avant affichage...*****/
if($_GET['id'] == '') //s'il n'y a pas d'id, on affiche la page du membre connecté
{
    if(isset($_SESSION['membre_id'])) $id = $_SESSION['membre_id'];
    else $id = -1;
}

else $id = $_GET['id'];

$profil = sqlquery("SELECT * FROM membres
LEFT JOIN connectes
ON connectes_id = membre_id
WHERE membre_id=".intval($id), 1);
if($profil['membre_id'] == '' || $id == -1)
{
    $informations = Array(/*L'id de cookie est incorrect*/
        true,
        'Page membre inconnue',
        'Ce membre n\'existe pas.',
        '',
        '../index.php',
        3
    );
    require_once('../information.php');
    exit();
}
/*****Fin gestion avant affichage*****/
?>

```

Voilà, ce n'était pas plus dur que cela ; maintenant, on peut afficher `haut.php`, le titre, etc. 😊

Code : PHP

```

<?php
/*****En-tête et titre de page*****/

$title = 'Membre : '.htmlspecialchars($profil['membre_pseudo'], ENT_QUOTES).';

include('../includes/haut.php'); //contient le doctype, et head.

/*****Fin en-tête et titre*****/
?>

```

```

<div id="colonne_gauche">
<?php
include ('../includes/colg.php');
?>
</div>

```

La partie assez habituelle de toute page de notre site est prête, passons au contenu. 😊

Code : PHP

```

<!--contenu/-->
<div id="contenu">
<div id="map">
<a href="../index.php">Accueil</a> => <a href="u
id=?php echo intval($profil['membre_id']); ?>">Profil de <?
php echo htmlspecialchars($profil['membre_pseudo'], ENT_QUOTES); ?></a>
</div>

<h1>Profil de <?
php echo htmlspecialchars($profil['membre_pseudo'], ENT_QUOTES); ?></h1>

<div class="profil_cellule_float">
<h2>Informations générales</h2>

<div class="avatar">
<?php
if($profil['membre_avatar'] == '')
{
    echo 'Pas d\'avatar';
}
?>
</div>

```

Le sitemap porte aussi le nom du bonhomme, de même que le titre de la page. 😊

Quant à l'avatar, pour le moment, à moins que vous n'ayez votre propre gestion des uploads, c'est pas vraiment le point fort de notre page. 😞

Ensuite, on fait une liste des informations principales sur le membre ; je dis bien *sur le membre*, pas sur sa personne, donc son pseudo, sa date d'inscription, son dernier passage sur le site, le groupe dans lequel il est (pour le moment aucun, a priori, on se basera donc sur membre_banni), et s'il est connecté actuellement...

Code : PHP

```

<!--liste/-->
<ul>
<li><b>Pseudo :</b> <?
php echo htmlspecialchars($profil['membre_pseudo'], ENT_QUOTES); ?></li>
<li><b>Date d'inscription :</b> <?
php echo mepd($profil['membre_inscription']); ?></li>
<li><b>Groupe :</b> <?
php if($profil['banni'] == 1) echo 'bannis'; else echo 'membres'; ?></li>
<li><b>Dernier passage :</b> <?
php echo mepd($profil['membre_derniere_visite']); ?></li>
<li><b>Statut :</b>
<?php
if($profil['connectes_id'] == $profil['m
{
?>
<span class="actif"><?

```

```

        <span class="actif"><?php
php echo htmlspecialchars($profil['membre_pseudo'], ENT_QUOTES); ?> est connecté
        <?php
        }
        else
        {
        ?>
        <span class="inactif"><?php
php echo htmlspecialchars($profil['membre_pseudo'], ENT_QUOTES); ?> est déconnecté</span></li>
        <?php
        }
        ?>
    </ul>
</div>

```

Comme vous le voyez, le point numéro 2 est respecté. 😊

Tout de suite après, on ajoute des informations, sur la personne du membre cette fois, donc la date de naissance, l'âge, la ville, la profession, mais pas l'e-mail, c'est un peu à part. 😊

Code : PHP

```

<!--infos complémentaires!-->
    <div class="profil_cellule">
        <h2>Informations complémentaires</h2>
        <ul>
            <li><b>Date de naissance :</b> <?php ech
        </li>
            <li><b>Âge :</b> <?php echo age($profil|
            <?php
            if(trim($profil['membre_profession']) !=
            {
            ?>
            <li><b>Profession :</b> <?
php echo htmlspecialchars($profil['membre_profession'], ENT_QUOTES); ?></li>
            <?php
            }
            if(trim($profil['membre_localisation'])
            {
            ?>
            <li><b>Ville :</b> <?
php echo htmlspecialchars($profil['membre_localisation'], ENT_QUOTES); ?></li>
            <?php
            }
            ?>
        </ul>
    </div>

```

Voilà : on a bien la date de naissance et l'âge du membre. 😊

D'ailleurs, occupons-nous de la fonction **age**, il s'agit d'un calcul simple :

Code : PHP

```

<?php
function age($birth)
{
    $DATE = intArray(explode('/', $birth));
    $stry = mktime(0, 0, 0, $DATE[1], $DATE[0], date('Y'));
    if(time() >= $stry) $age = date('Y') - $DATE[2];
}

```

```

        else $age = date('Y') - $DATE[2] - 1;

        return $age;
    }
    ?>

```

Tiens donc, voilà une nouvelle fonction : `intArray()` !

Ce n'est pas une fonction native, elle est juste là pour sécuriser un peu plus la date de naissance :

Code : PHP

```

<?php
function intArray($Array)
{
    foreach($Array as $cle => $element)
    {
        $Array[$cle] = intval($element);
    }

    return $Array;
}
?>

```

Passons maintenant à son activité sur le site (quand je vous dis que je me suis vraiment inspiré du SdZ, même si je fais ça moins bien...), cette case sera vide, nous n'avons pas de forum, de système de news, ou même de tutoriels. 🤪

Code : HTML

```

<!--activité/-->

<div class="profil_cellule_float">
    <h2><?php echo htmlspecialchars($profil['membre_p
    Si vous avez un forum, ou des news, ou autre chose
</div>

```

Enfin, passons aux moyens de contacter le membre :

Code : PHP

```

<!--contact/-->

<div class="profil_cellule">
    <h2>Comment le joindre</h2>
    <!--
    L'adresse e-mail est généralement une information
    nous ne la protégeons pas, mais très vite nous
    //-->
    S'il y a un module de MP sur votre site, ou si v

    <h3>Messageries</h3>
    <ul>
        <li><b>E-Mail :</b> </li>
        <?php
        if(trim($profil['membre_msn']) != '')
        {
            ?>
            <li><b>MSN :</b> </li>
        <?php
        }

        if(trim($profil['membre_yahoo']) != '')

```

```

        {
        ?>
        <li><b>Yahoo :</b> </li>
        <?php
        }

        if(trim($profil['membre_aim']) != '')
        {
        ?>
        <li><b>Aim :</b> </li>
        <?php
        }
        ?>
    </ul>
</div>

```

On rajoute la biographie, la signature, et on clôt la page.

Code : PHP

```

<!--signature et fin/-->
        <div class="profil_ligne">
            <h2>Signature</h2>
            <?
php echo htmlspecialchars($profil['membre_signature'], ENT_QUOTES); ?>
        </div>

        <div class="profil_ligne">
            <h2>Biographie</h2>
            <?
php echo htmlspecialchars($profil['membre_biographie'], ENT_QUOTES); ?
        >
        </div>
    </div>

    <?php
    include('../includes/bas.php');
    mysql_close();
    ?>

```

On peut enfin passer à `image.php`.

Code : PHP

```

<?php
/*
Neoterranos & LkY
Page image.php

Génère une image d'un texte passé en GET['tex'].

Quelques indications (utiliser l'outil de recherche et rechercher les mentions c

Liste des fonctions :
-----
Aucune fonction
-----

Liste des informations / erreurs :

```

```

-----
Aucune information / erreur
-----
*/

header ("Content-type: image/png");
$count = strlen($_GET['tex']);
$image = imagecreate($count*9, 18);

if(function_exists(imagecolorallocatealpha))
{
    $blanc = imagecolorallocatealpha($image, 255, 255, 255, 127);
}

else
{
    $blanc = imagecolorallocate($image, 255, 255, 255, 127);
}
$noir = imagecolorallocate($image, 0, 0, 0);

imagestring($image, 5, 1, 1, htmlspecialchars($_GET['tex'], ENT_QUOTES), $noir);
imagepng($image);
?>

```

Le plus dur dans cette page a été de définir la taille de l'image souhaitée, ni trop grosse, ni trop petite, avec la police que j'ai choisie, chaque caractère fait 9 pixels de large : ainsi, nous sommes tranquilles.

La fonction `function_exists()` vérifie qu'une fonction existe bien dans votre version de PHP.

Nous avons enfin notre dernier point respecté.

Voilà un profil qui ressemble à celui du SdZ comme deux gouttes d'eau se ressemblent, ce n'est pas par manque d'imagination que je vous propose cela, mais simplement parce que je sais que ce mode d'affichage plaît à pas mal de monde. 😊

Passons au CSS : pour le moment, cette pauvre page souffre un peu de son design. 🙄

Code : CSS

```

.profil_cellule_float {
    float: left;
    width: 47%;
    border: 1px solid #CCDFEF;
    max-height: 30%;
    margin-bottom: 1%;
    margin-left: 1%;
    margin-right: 1%;
    height: 250px;
}

.profil_cellule {
    margin-left: 49%;
    margin-bottom: 1%;
    margin-right: 1%;
    width: 47%;
    border: 1px solid #CCDFEF;
    height: 250px;
}

.profil_ligne {
    width: 95%;
    margin-left: 1%;
    border: 1px solid #CCDFEF;
    margin-bottom: 1%;
}

.avatar {
    margin-left: 10%;
    float: left;
}

```

```
        min-width: 150px;
        min-height: 130px;
    }
    h3 {
        text-decoration: underline;
        color: #507080;
        font-family: "trebuchet ms", verdana, sans-serif;
        font-variant: small-caps;
        font-weight: bold;
        font-size: 17px;
    }
```

Vous l'avez constaté, sur cette page, la réelle difficulté devrait être le CSS, et encore, c'est pour cela que je ne m'étends pas en explications, je ne voudrais pas que vous croyiez que je vous prends pour des idiots. 😊

Bien, vous avez enfin des listes de vos membres, des connectés, etc., elles sont assez vides, car le système est pour le moment basique, c'est à vous de le modifier ; mais dans la partie II, nous les remplirons un peu plus, quant à la visualisation de profils, vous voyez que ce n'était pas si difficile, nous verrons en partie II comment sécuriser certaines informations telles que le mail ou la date de naissance.

Sur ce, attaquons l'édition de profil !

Dans la deuxième partie, nous allons approfondir tout cela pour créer un système encore plus sympa. 😊

Le tutoriel n'est pas fini ! 😊

Nous continuons d'écrire les prochains chapitres pendant que vous lisez ces lignes.

Vous pouvez suivre l'avancement du tutoriel sur [ce topic](#). 😊

Vous pouvez aussi y poster vos suggestions.

À bientôt !

Neoterranos et LkY