

Parcours de formation SIN

**Déploiement d’une interface de communication web**

Module SIN413: Programmation WEB HTML, PHP, SQL

Durée : 3h

Objectifs à atteindre : Acquérir les compétences indispensables au développement de sites internet en PHP. Savoir manipuler les données d'une base de données relationnelle à partir de l'application.

Niveau des connaissances envisageable : Première

Pré requis : - Connaitre la structure générale d'une page web, tableaux, liens, principe de fonctionnement..

- Connaitre les éléments de programmation procédurale (langage C de préférence), algorithmique de base.

Logiciels utilisés : - VMware

- Linux ubuntu

- Serveur apache2

- Serveur MySQL

- phpmyadmin

Connexion à internet obligatoire.

Outils mobilisés, le cas échéant : tous les tutoriels disponibles sur internet

Webographie :

<http://www.siteduzero.com/tutoriel-3-13484-votre-premiere-page-xhtml.html> *(Par M@teo21, licence CC)*

<http://www.siteduzero.com/tutoriel-3-14668-concevez-votre-site-web-avec-php-et-mysql.html> *(Par M@teo21, licence CC)*

<http://www.php.net/manual/fr/> *(Copyright © 2001-2011 The PHP Group)*

<http://doc.ubuntu-fr.org/programmation> *(licence CC BY-SA v3.0)*

Illustrations :

Copies d’écran Linux Ubuntu – Firefox – phpMyAdmin – M. Silanus.

# Le langage HTML

Toute page Web comprend une base de langage HTML. Il s'agit d'un langage de balisage qui définit essentiellement la structure de la page web (titres, tableaux, paragraphes, etc…). Il permet en outre de créer des liens d'un document à un autre ou d'un endroit d'un document à un autre endroit du même document.

Le langage HTML est interprété au niveau du client par un navigateur web (Internet Explorer, Mozilla Firefox, Google chrome, Opéra, …). Il n’est donc pas nécessaire d’utiliser un serveur web pour lire une page HTML.

## Premier exemple

Dans un éditeur de texte (Gedit sous Ubuntu ou notepad++ sous Windows), ouvrez le fichier *test.html* du dossier *html*.

|  |
| --- |
| <HTML>  Ceci est une balise. Elles sont encadrées par < et >  <HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/>  <TITLE>Page Test</TITLE>  </HEAD>  <BODY BGCOLOR="#FF9966">  <H1><CENTER><B>Bonjour &agrave; tous!</B></CENTER></H1>  <H2><CENTER><B>Bienvenue</B></CENTER></H2>  <P>Vous &ecirc;tes ici sur une <B>page de test....</B></P>  <P>Pour faire une recherche, <A HREF="http://www.google.fr"><B><I>cliquez ici!</I></B></A></P>  </BODY>  </HTML> |

Testez l’affichage de la page en double-cliquant sur l’icone du fichier. Vous devriez obtenir le résultat suivant :

URL : file

|  |
| --- |
| <H1><CENTER><B>Bonjour &agrave; tous!</B></CENTER></H1>  <A  HREF="http://www.google.fr"> <B><I>cliquez ici!</I></B>  </A>  <BODY BGCOLOR="#FF9966">  <HEAD>  <TITLE>Page Test</TITLE>  </HEAD> |

Le navigateur interprète les différentes balises et produit le rendu graphique.

Ainsi, <BODY BGCOLOR="#FF9966"> indique la couleur de fond du corps de la page, <H1><CENTER><B> indique que le texte qui suit (Bonjour à tous !) sera du premier niveau de titre (<H1>), centré (<center>) et en gras (<B>).

Une page HTML peut s'accompagner de feuilles de style en cascade (*CSS : Cascading Style Sheets - Feuilles de style en cascade*) qui sont des styles qui se chargent de la mise en forme de la page. Le fait de dissocier le formatage du contenu permet de faciliterle ré habillage du site. Il suffira de modifier les styles définis dans le fichier CSS pour impacter toutes les pages HTML du site.

La ligne <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/> permet au navigateur d’identifier le jeu de caractères utilisé.

UTF-8 est le jeu de caractère universel, qui permet de coder à peut près tous les caractères connus de toutes les langues.

Il faut donc veiller à ce que le fichier HTML soit bien écrit en utilisant ce jeu de caractères sous peine de voir les caractères spéciaux non codés selon la norme HTML (&ecirc; pour ê par exemple) de manière curieuse et illisible (Ã© ou � par exemple). Si vous utilisez Gedit sous Ubuntu, choisissez le bon format lors de l’enregistrement, si vous êtes sous Windows et que vous utilisez Notepad++, choisissez *Format → Encoder en UTF-8*.

Lorsqu’un fichier HTML est exécuté localement, la barre d’adresse indique **file://**. Lorsqu’il est hébergé sur un serveur, elle indique **http://**.

## Structure d’une page HTML

Un document HTML commence par la balise <HTML> et finit par la balise </HTML>. Il contient également un en-tête décrivant le titre de la fenêtre du navigateur ou de l’onglet, puis un corps dans lequel se trouve le contenu de la page.

L'en-tête est délimité par les balises <HEAD> et </HEAD>. Le corps est délimité par les balises <BODY> et </BODY>.

|  |
| --- |
| <HTML>  <HEAD>  <TITLE>Titre de la page</TITLE>  </HEAD>  <BODY>  Contenu de la page  </BODY>  </HTML> |

## Principales balises HTML

On trouvera sur internet de nombreux sites répertoriant les balises HTML de manière plus exhaustive.

La liste ci-après ne représente que les quelques balises qu’il me semble, indispensable de connaitre.

### Structure de la page HTML

|  |  |
| --- | --- |
| **Balises** | **Définitions** |
| <HTML>...</HTML> | Début et fin de fichier Html |
| <HEAD>...</HEAD> | Zone d'en-tête d'un fichier Html |
| <TITLE>...</TITLE> | Titre affiché par le browser (élément de HEAD) |
| <BODY>...</BODY> | Début et fin du corps du fichier Html |
| <BODY bgcolor="#XXXXXX"> | Couleur d'arrière-plan (en hexadécimal) |
| <BODY background="xyz.gif"> | Image d'arrière-plan |

### Mise en forme des caractères

|  |  |
| --- | --- |
| **Balises** | **Définitions** |
| <B>...</B> | Texte en gras |
| <I>...</I> | Texte en italique |
| <FONT color="#XXXXXX">...</FONT> | Texte en couleur où XXXXXX est une valeur hexadécimale |
| <FONT size=X>...</FONT> | Taille des caractères où X est une valeur de 1 à 7 |

### Mise en forme du texte

|  |  |
| --- | --- |
| **Balises** | **Définitions** |
| <!--...--> | Commentaire ignoré par le navigateur |
| <BR> | Retour à la ligne |
| <CENTER>...</CENTER> | Centre tout élément compris dans la balise |
| <P>...</P> | Nouveau paragraphe |
| <P align=center>...</P> | Paragraphe centré. La balise align accepte left et right |

### Les liens hypertextes ou ancrages

|  |  |
| --- | --- |
| **Balises** | **Définitions** |
| <A href="http://...">...</A> | Lien vers une page Web |
| <A href="mailto:...">...</A> | Lien vers une adresse Email |
| <A name="xyz">...</A> | Définition d'une ancre |
| <A href="xyz">...</A> | Lien vers une ancre |

### Listes

|  |  |
| --- | --- |
| **Balises** | **Définitions** |
| <UL>  <LI> ... </LI>  <LI> ... </LI>  </UL> | Liste à puces  Elément de la liste |
| <OL>  <LI> ... </LI>  <LI> ... </LI>  </OL> | Liste numérotée  Elément de la liste |
| <HR> | Barre horizontale |

### Images

|  |  |
| --- | --- |
| **Balises** | **Définitions** |
| <IMG src="xyz.gif"> | Insertion d'une image. |
| <IMG ... width=x height=y> | Mise à l'échelle de l'image en pixels |
| <IMG ... alt="votre texte"> | Texte alternatif lorsque l'image n'est pas affichée |
| <IMG ... align=bottom> | Alignement : bottom, middle, top, center, left, right |

### Tableau

|  |  |
| --- | --- |
| **Balises** | **Définitions** |
| <TABLE>...</TABLE> | Définition d'un tableau |
| <TABLE width=x> | Largeur du tableau en pixels ou en % (width=x) |
| <TABLE border=x> | Largeur de la bordure |
| <TABLE cellpadding=x> | Espace entre la bordure et le texte |
| <TABLE cellspacing=x> | Epaisseur du trait entre les cellules |
| <TR>...</TR> | Ligne du tableau |
| <TD>...</TD> | Cellule du tableau |

### Les formulaires

|  |  |
| --- | --- |
| **Balises** | **Définitions** |
| < FORM>...</FORM> | Définition d'un formulaire |
| <FORM METHOD=x> | Méthode de soumission des données au serveur (METHOD=POST ou METHOD=GET) |
| <FORM ACTION=x> | URL du script de traitement des données du formulaire. |
| <input type="text" /> | Définition d’une zone de saisie de texte. |
| <input type="password" /> | Définition d’une zone de saisie de mot de passe (caractères cachés). |
| <input type="submit" /> | Définition d’un bouton de soumission de formulaire. |
| <input type="checkbox" /> | Définition d’une case à cocher (cases carrées, plusieurs choix possibles). |
| <input type="radio" /> | Définition de cases d’options à cocher (cases rondes, un seul choix possible) |
| <input type="reset" /> | Définition d’un bouton de remise à zéro (effacement) d’un formulaire. |
| <input type="hidden" /> | Définition d’un champ caché. |
| <textarea> | Définition d’une zone de texte multi-lignes. |
| <select name="pays">  <option value="c1">Choix 1</option>  <option value="c2">Choix 2</option>  <option value="c3">Choix 3</option>  </select> | Définition d’une liste déroulante.  name : nom de la liste  option : entrée dans la liste  value : valeur renvoyée par le formulalire  >...< : chaine affichée dans la liste |

### Exercice

Créez une page HTML de présentation qui contient vos nom, prénom, discipline enseignée, établissement d’exercice, ville et une photo (image de taille maxi 100ko) ou adaptez la *page homer\_s.html* à votre propre situation.

Ajoutez un lien vers google.fr pour faire une recherche et un lien vers le site de votre établissement.

## Les limites du HTML

Une page web créée en HTML est totalement statique et n'offre aucune possibilité d'interaction avec le visiteur. Chaque page doit être créée au préalable et doit être manuellement mise à jour.

Pour rendre une page web plus dynamique, il est possible d'y insérer des objets grâce aux balises <object>, <img>, <applet> ou <script>. Les objets peuvent être des animations flash, des vidéos, des programmes joints à la page web et exécutés soit dans le navigateur (*scripts java, active X*), soit par l’ordinateur client (*applets java*).

Les objets ne fournissent pas toutes les solutions aux besoins d'interactivité et de dynamisme des sites et peuvent parfois lourds à mettre en place.

Il faut alors avoir recours à la programmation coté serveur.

Plusieurs langages sont alors disponibles, nous étudierons uniquement ici le PHP que l’on peut associer à une base de données relationnelle pour fournir plus de puissance à nos applications.

# Le langage PHP

PHP est un langage de script interprété côté serveur (comme les scripts CGI, ASP, ...).

Sa syntaxe est voisine de celle du C.

Le serveur interprète le code PHP et génère du code HTML transmis au client. Il offre donc un haut niveau de sécurité car le code originel n’est jamais transmis au client, contrairement aux programmes exécutés côté client.

PHP est distribué sous licence GNU GPL, ce qui permet de l’utiliser librement et gratuitement. De nombreux développeurs ont profité de l'Open Source pour développer des modules prêts à l'usage et paramétrables à volonté.

Les modules serveurs de PHP sont implémentés, entre autre, sur les serveurs Apache. Ce type de serveur est généralement installé dans les distributions de linux. Sous Windows, il est possible d’installer un tel serveur à l’aide du logiciel Wampserver (<http://www.wampserver.com/download.php>). La solution serveur web de Microsoft (IIS) est aussi en mesure d’interpréter les scripts PHP.

PHP travaille généralement avec une base de données MySQL. Elle est installée par défaut avec Wampserver. Sous linux, il faut procéder à son installation.

Les fichiers correspondants à cette partie se trouvent dans le dossier *PHP*.

## Documentation

Il existe sur internet de très nombreux sites traitant du langage PHP. Le manuel de référence en français est consultable à l’URL suivante : <http://www.php.net/manual/fr/> .

## Installation des serveurs apache et mysql

### Linux Ubuntu

En mode root, saisissez les commandes suivantes pour procéder aux installations :

|  |
| --- |
| # apt-get install apache2  # apt-get install mysql-server-5.1  # apt-get install phpmyadmin |

### Windows

Téléchargez et installez Wampserver : <http://www.wampserver.com/download.php>

### Vérification du fonctionnement

Sur l’ordinateur (réel ou virtuel) où l’installation a été faite, ouvrez un navigateur et saisissez l’URL : <http://localhost> pour tester le fonctionnement du serveur apache et <http://localhost/phpmyadmin> pour tester la connexion au serveur de base de données mysql.

**Dans la suite du document, c’est un serveur sous linux Ubuntu dans une machine virtuelle qui sera utilisé.**

## Premiers scripts

Le site web par défaut du serveur apache se trouve dans le dossier */var/www*.

### Vérification du fonctionnement de php

Dans le dossier du site web par défaut, créez un fichier nommé *test.php* :

|  |
| --- |
| # cd /var/www  # gedit test.php |

Dans l’éditeur de texte qui s’est ouvert, copier le code suivant :

|  |
| --- |
| <?php  // Affiche toutes les informations  phpinfo();  ?> |

Ouvrez la page *test.php* dans un navigateur : <http://localhost/test.php>

|  |
| --- |
| Ubuntu-2011-02-26-00-12-32  Généré par la fonction phpinfo(); |

Si votre navigateur affiche une page comme celle-ci-dessus, PHP est correctement interprété par le serveur.

**Remarque : Le code PHP est encadré par les balises <?php ou simplement <? et ?>.**

**En dehors de ces balises, le code est transmis tel quel au client. On peut donc inclure du PHP dans du code HTML.**

### Afficher le contenu d’une variable

Toutes les variables doivent être précédées du signe $. Les variables en langage PHP peuvent être de trois types : scalaires, tableaux, tableaux associatifs.

Ouvrez le fichier *variables.php* et copiez le dans le dossier par défaut du serveur web. Ouvrez-le avec un éditeur de texte (Gedit par exemple, attention aux permissions, vous devrez être *root* pour copier et modifier le fichier.) :

|  |
| --- |
| <html>  <head> <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/> </head>  <body>    </h1><center>Bonjour  Déclaration et initialisation de variables  **<?php**  **$aujourdhui = date("d/m/Y");**  **$prenom="mettre ici votre prénom";**  **$a = 5;**  echo permet d’afficher à l’écran le contenu d’une variable ou le texte entre " ".  Le . (point) est l’opérateur de concaténation.  **$b = 10;**  **echo $prenom;**  **echo "<br>nous sommes le ".$aujourdhui;**  **?>**  </center></h1>  Opérations sur des variables numériques  Petits cacluls : <br>  **<?php**  Commentaires sur une ligne  /\* … \*/ : commentaires sur plusieurs lignes  **$somme = $a + $b;**  **$produit = $a \* $b;**  **$div = $a / $b;**  **echo "a + b = ".$somme;**  **echo "<br>a \* b = ".$produit; // affiche a \* b = 50**  **echo "<br>a / b = ".$div;**  **?>**  </body></html> |

Testez le résultat dans votre navigateur : <http://localhost/variables.php>

|  |
| --- |
| Ubuntu-2011-02-26-01-02-17 |

Remarques :

* En PHP, on peut utiliser " ou ' pour définir une chaine de caractères mais l’affichage sera différent si on utilise des variables :

|  |
| --- |
| echo "a + b = $somme"; //Affiche : a + b = 15  echo 'a + b = $somme'; //Affiche : a + b = $somme |

* On peut introduire du code php dans une page HTML où l’on veut. Mais il est pratique courante de n’écrire en HTML uniquement le début et la fin de la page. Tout le corps de la page est codé en PHP.

## Les instructions de base

### Les conditions

* **if (condition) {action ;} else {actions ;}**

|  |
| --- |
| $age = 15 ;  $majorite = 18 ;  if($age>=$majorite)  {  echo'Vous êtes majeur' ;  }  else  {  echo'Vous êtes mineur' ;  } |

Pour enchainer les tests, on peur utiliser l’instruction elseif.

* **switch(condition)   
  {  
   case valeur1 :   
   action ;   
   break ;   
   …   
   case valeurn :   
   action ;   
   break ;  
   default :   
   action ;   
   break ;  
  }**

|  |
| --- |
| $nombre=4; //Modifier la valeur de $nombre  switch($nombre)  {  case 1:  echo "\$nombre est égal à 1";  break;  case 2:  echo "\$nombre est égal à 2";  break;  case 3:  echo "\$nombre est égal à 3";  break;  default:  echo "\$nombre est inférieur à 1 ou supérieur à 3";  break;    } |

Remarque : Le caractère \ est, comme en C/C++, le caractère d’échappement. Il indique à l’interpréteur de commandes PHP de ne pas tenir compte du caractère qui suit. Le $ ne sera pas interprété mais afficher.

De même, dans la commande :

echo '**Il est l**'heure';

Provoque une erreur à cause de l’apostrophe qui ferme prématurément la chaine de caractères.

La bonne écriture est :

echo '**Il est l\**'**heure**'; ou echo "**Il est l**'**heure**";

### Les boucles

* **for(compteur; condition; évolution)**

**{**

**action ;**

**}**

|  |
| --- |
| for($compt = 1; $compt <= 10 ; $compt++ )  {  echo "$compt / ";  } |

* **while(condition)**

**{**

**action ;**

**}**

|  |
| --- |
| $compt = 1;  while($compt < 10 )  {  echo "$compt / ";  $compt++ ;  } |

* **do**

**{**

**action ;**

**}**

**while(condition)**

|  |
| --- |
| $compt = 10;  do  {  echo "$compt / ";  $compt++ ;  }  while($compt < 10 ) ; |

* **foreach(tableau as [clef =>] valeur)**

**{**

**action ;**

**}**

|  |
| --- |
| $a = array(  "un" => 1,  "deux" => 2,  "trois" => 3,  "dix-sept" => 17  );  foreach ($a as $k => $v)  {  echo "\$a[$k]=>$v.\n";  }  //affiche $a[un]=>1. $a[deux]=>2. $a[trois]=>3. $a[dix-sept]=>17. |

## Interaction avec l’utilisateur

Elle se fait principalement au moyen d’un formulaire HTML qui permet de transmettre les variables au script php.

### Exemple minimal

Fichier *formulaire\_post.php*

|  |
| --- |
| <HTML>  Si la variable valider n’est pas créée : affichage du formulaire.  <HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/>  <TITLE>formulaire post</TITLE>  </HEAD>  <BODY>  **<?php**  **if(!isset($\_POST["valider"]))**  **{**  La variable valider est créée lorsque l’utilisateur appui sur le bouton valider.  Traitement du formulaire  **echo'<p>Quel est ton prénom :</p>'; echo'<form method="post" action="'.$\_SERVER['PHP\_SELF']. '">'; echo'<p><input type="text" name="prenom" />**  **<input type="submit" name="valider" value="Valider" /></p>';**  **echo'</form>';}else{ echo'<p>Bonjour !</p>'; echo'<p>Tu t\'appelles '.$\_POST["prenom"].'</p>'; echo'<p>Si tu veux changer de prénom, <a href="'.$\_SERVER['PHP\_SELF'].'">**  **clique ici </a> pour revenir à la page formulaire.php</p>';}**  **?>**  </BODY>  </HTML> |

Les données du formulaire sont soumises au serveur qui les traitera dans la page indiquée dans le paramètre action. Ici, on utilise la valeur de $\_SERVER['PHP\_SELF'] qui renvoi l’URL du script courant.

La méthode POST utilisée ici indique la façon dont PHP effectue le transfert des données : elles sont « postées » dans un tableau associé nommé $\_POST. Ce transfert est totalement transparent et l’utilisateur ne peut pas modifier les valeurs transmises. Le nom de l’élément de formulaire constitue la clé de ce tableau.

|  |  |
| --- | --- |
| **Elément du tableau** | **Variable associée** |
| <input type="text" name="prenom" /> | $\_POST["prenom"] |
| <input type="submit" name="valider" value="Valider" /> | $\_POST["valider"] |

|  |
| --- |
| Ubuntu-2011-02-26-23-31-47 |

Il existe une autre méthode pour transmettre les données de formulaire au serveur. C’est la méthode GET. Cette méthode construit une chaine de caractères constituée des noms des éléments du formulaire et des valeurs qu’ils contiennent séparés par des &, puis les associe à l’URL de la page indiquée dans le paramètre action du formulaire à l’aide du caractère «**?** ».

Ici, nous obtiendrions avec la méthode GET l’URL suivante :

http://serveur/dossier/formulaire\_get.php?prenom=Marc&valider=Valider

Fichier *formulaire\_get.php*

|  |
| --- |
| Ubuntu-2011-02-26-23-44-41 |

### Exercice

Réalisez un formulaire proposant un menu composé :

* d'une liste d'entrées,
* d'une liste de plats,
* d'une liste de desserts.

Associez un prix à chaque entrée, plat ou dessert.

|  |
| --- |
| $entree = array("Foie Gras Maison"=>14.00,  "Saumon Fumé"=>12.00,  "Terrine Maison et confiture d'oignons"=>9.00,  "Escargots de Bourgogne"=>12.00);    $plat = array("Entrecôte Poêlée"=>14.80, "Gigot d'Agneau"=>14.50,  "Confit de Canard"=>14.00, "Faux Filet Poêlé"=>13.00);    $dessert = array("Mousse au Chocolat"=>6.00, "Ile Flottante"=>6.00, "Fondant au Chocolat"=>6.50, "Crème Caramel"=>6.00, "Tarte Normande"=>6.50,  "Tarte à l'Abricot"=>6.50); |

Un internaute doit pouvoir choisir une entrée, un plat et un dessert, et valider son choix.

|  |
| --- |
| echo'<p><select name="entree">';  foreach($entree as $choix=>$prix)  { echo'<option value="'.$choix.'">'.$choix.' : '.$prix.'</option>';  }  echo'</select></p>';  ...  echo'<input type="submit" name="valider" value="Valider" /></p>'; |

Affichez alors le menu choisi, ainsi que l'addition.

|  |
| --- |
| $total=0;  echo'<p>Bonjour !</p>';  echo'<p>Rappel de votre choix :</p>';  foreach($entree as $choix=>$prix)  { if($choix==$\_POST["entree"])  {  echo $choix.' : '.$prix.'<br>';  $total+=$prix;  }  }  ...  echo'<p>Total : '.$total.'</p>'; |

Présentez ensuite les choix de menu en les alignant à l'aide d'un tableau HTML.

|  |
| --- |
| echo'<table border="1" align="center" width="50%">';  echo'<tr>';  echo'<td>';  foreach($entree as $choix=>$prix)  { if($choix==$\_POST["entree"])  {  echo $choix.'</td><td align="right">'.money\_format('%.2n',$prix).'</td>';  $total+=$prix;  }  }  echo'</td>';  echo'</tr>';  ...  echo'</td></tr></table>'; |

Remarquez l’utilisation de la fonction money\_format(). Consultez la documentation de cette fonction sur <http://php.net>

|  |
| --- |
|  |

### Proposition de solution

|  |
| --- |
| <HTML><HEAD> <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/>  <TITLE>Menu du restaurant</TITLE></HEAD><BODY><?php  $entree = array("Foie Gras Maison"=>14.00,  "Saumon Fumé"=>12.00,  "Terrine Maison et confiture d'oignons"=>9.00,  "Escargots de Bourgogne"=>12.00);    $plat = array("Entrecôte Poêlée"=>14.80, "Gigot d'Agneau"=>14.50,  "Confit de Canard"=>14.00, "Faux Filet Poêlé"=>13.00);    $dessert = array("Mousse au Chocolat"=>6.00, "Ile Flottante"=>6.00, "Fondant au Chocolat"=>6.50, "Crème Caramel"=>6.00, "Tarte Normande"=>6.50, "Tarte à l'Abricot"=>6.50);  if(!isset($\_POST["valider"])) {  echo'<p><center><h1>Faites votre menu : </center></h1></p>'; echo'<form method="POST" action="'.$\_SERVER['PHP\_SELF']. '">';  echo'<p><center><select name="entree">';  foreach($entree as $choix=>$prix)  { echo'<option value="'.$choix.'">'.$choix.' : '.$prix.'</option>';  }  echo'</select></p>';  echo'<p><select name="plat">';  foreach($plat as $choix=>$prix)  { echo'<option value="'.$choix.'">'.$choix.' : '.$prix.'</option>';  }  echo'</select></p>';  echo'<p><select name="dessert">';  foreach($dessert as $choix=>$prix)  { echo'<option value="'.$choix.'">'.$choix.' : '.$prix.'</option>';  }  echo'</select></p>';  echo'<input type="submit" name="valider" value="Valider" /></p></center>'; echo'</form>';  } else { $total=0;  echo'<p><center><h1>Bonjour !</h1></center></p>'; echo'<p><center><h2>Rappel de votre choix :</h2></center></p>';  echo'<table border="1" align="center" width="50%">';  echo'<tr><td>';  foreach($entree as $choix=>$prix)  { if($choix==$\_POST["entree"])  {  echo $choix.'</td><td align="right">'.money\_format('%.2n',$prix).'</td>';  $total+=$prix;  }  }  echo'</td></tr><tr><td>';  foreach($plat as $choix=>$prix)  { if($choix==$\_POST["plat"])  {  echo $choix.'</td><td align="right">'.money\_format('%.2n',$prix).'</td>';  $total+=$prix;  }  }  echo'</td></tr><tr><td>';  foreach($dessert as $choix=>$prix)  { if($choix==$\_POST["dessert"])  {  echo $choix.'</td><td align="right">'.money\_format('%.2n',$prix).'</td>';  $total+=$prix;  }  }  echo'</td></tr></table>';  echo'<p><h3><center>Total : '.money\_format('%.2n',$total).'</h3></center></p>'; echo'<p>Pour recommencer, <a href="'.$\_SERVER['PHP\_SELF'].'">cliquez ici</a></p>'; } ?>  </BODY></HTML> |

# Traitement des données

Lorsqu’un script PHP produit des informations, il faut les stocker quelque part sinon, elles seront irrémédiablement perdues lorsque le serveur renverra la page html produite par le script au client.

Pour stocker des données, il existe deux solutions:

* les enregistrer dans un fichier texte sur le serveur (quelque part dans l'arborescence de votre hébergement),
* les enregistrer dans une base de données relationnelle.

La sauvegarde dans un fichier présente l’avantage de pouvoir accéder rapidement aux données qu’il contient sans avoir à utiliser de logiciels tiers. Cependant, les recherches, modifications ou suppressions de données sur une partie de l'information stockée n’est pas facile. Il est encore plus difficile de croiser des informations provenant de plusieurs fichiers.

Les bases de données ont été conçues pour ça. Les informations y sont stockées sous forme de tables contenant des enregistrements. Les opérations sur les tables et les enregistrements sont réalisées à l’aide d’un langage d’interrogation de base de données comme SQL (Structured Query Language) interprété par un logiciel spécialisé qu’on appelle **SGBD (Système de Gestion de Base de Données)**. Il constitue le serveur de base de données.

Plusieurs SGBD sont disponibles sur le marché. Citons, entre autre, Oracle, Microsoft SQL Server, MySQL, SQLite…

Dans la suite de se document, nous ne traiterons que le serveur de base de données MySQL, qui est généralement, le plus souvent associé à PHP.

## Utilisation des fichiers

### Ouvrir un fichier

L’ouverture d’un fichier se fait grâce à la fonction fopen(). En voici sa syntaxe :

|  |
| --- |
| resource fopen(string $filename , string $mode) |

Elle permet la création d’une ressource sur le fichier nommé et l’ouvre selon le mode indiqué en écriture ou en lecture :

|  |  |
| --- | --- |
| **Différents modes disponibles** | |
| **r** | ouverture en lecture seulement |
| **w** | ouverture en écriture seulement (la fonction crée le fichier s'il n'existe pas) |
| **a** | ouverture en écriture seulement avec ajout du contenu à la fin du fichier (la fonction crée le fichier s'il n'existe pas) |
| **r+** | ouverture en lecture et écriture |
| **w+** | ouverture en lecture et écriture (la fonction crée le fichier s'il n'existe pas) |
| **a+** | ouverture en lecture et écriture avec ajout du contenu à la fin du fichier (la fonction crée le fichier s'il n'existe pas) |

### Lire un fichier

Fichier *lire\_fichier.php*

|  |
| --- |
| <?php  $filename = "menu.txt"; // Fichier à lire  $handle = fopen($filename, "r"); // Ouverture en lecture  while (!feof($handle)) // Tant qu’on est pas à la fin du fichier  {  $line = fgets($handle); // Lire la ligne courante  echo $line."<br>"; // Afficher la ligne puis retour à la ligne  }  fclose($handle); // Libération de la ressource  ?> |

**Remarque :**

* Pour lire le fichier tout entier, on peut utiliser la fonction fread()
* Pour un caractère du fichier, on utilise la fonction fgetc().

### Ecrire dans un fichier

Fichier *ecrire\_fichier.php*

|  |
| --- |
| <?php  $filename = 'test.txt';  $somecontent = "\r\nCeci doit être ajouté à la fin du fichier";  // Le fichier est-il accessible en écriture ?  if (is\_writable($filename))  {  // Ouverture du fichier en ajout (écriture à la fin)  if (!$handle = fopen($filename, 'a'))  {  echo "Impossible d'ouvrir le fichier ($filename)";  exit;  }  // Ecriture à la fin du fichier  if (fwrite($handle, $somecontent) === FALSE)  {  echo "Impossible d'écrire dans le fichier ($filename)";  exit;  }  echo "L'écriture de :<br>($somecontent)<br>dans le fichier ($filename) a réussi";  fclose($handle);  }  else  {  echo "Le fichier $filename n'est pas accessible en écriture.";  }  ?> |

### Exercice

Créez une page HTML (*inscription.html*) contenant un formulaire d’inscription à la formation STI2D SIN4 avec les champs suivants :

* Nom
* Prénom
* Etablissement
* Ville

Le formulaire sera soumis à un script PHP (*ajout.php*) permettant d’écrire les données recueillies dans un fichier texte appelé *stagiaires.txt (***Attention** : créez un fichier vierge et positionnez les permissions à 777).

Un second script php (*stagiaires.php*) doit permettre d’afficher le contenu du fichier *stagiaire.txt*.

|  |
| --- |
| Ubuntu-2011-03-01-11-40-31 |

### Solution minimale

* *inscription.html* : Formulaire d’inscription des stagiaires

|  |
| --- |
| <HTML>  <HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/>  <TITLE>Formation STI2D SIN4</TITLE>  </HEAD>  <BODY>  <h1><center>Formation STI2D SIN4</center></h1>  <h2>Formulaire d'inscription</center></h2>  <br><br>  <form method="post" action="ajout.php">  <p>Nom : <input type="text" name="nom" ></p>  <p>Prénom : <input type="text" name="prenom"></p>  <p>Etablissement : <input type="text" name="etablissement"></p>  <p>Ville : <input type="text" name="ville"></p>  <input type="submit" name="valider" value="Valider">  </form>  </BODY>  </HTML> |

* *ajout.php* : Ajout des données du formulaire dans le fichier *stagiaires.txt*

|  |
| --- |
| <HTML><HEAD> <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/> <TITLE>Formation STI2D SIN4</TITLE></HEAD><BODY>  <?php  $filename = 'stagiaires.txt';  /\* Le fichier doit exister et avoir 777 comme permission pour que Apache puisse écrire dedans  sinon, vous pouvez positionner les permissions du dossier www à 777 ce qui est fortement déconseillé \*/  $stagiaire = $\_POST["nom"].";".  $\_POST["prenom"].";".  $\_POST["etablissement"].";".  $\_POST["ville"]."\r\n";  // \r\n introduit un saut de page dans un fichier texte (compatible Windows et linux)  $handle = fopen($filename, 'a+');  if(fwrite($handle, $stagiaire) === FALSE)  {  echo "Impossible d'écrire dans le fichier ($filename)";  exit;  }  echo "Stagiaire enregistré";  fclose($handle);  echo "<br><br><a href='stagiaires.php'>Liste des stagiaires</a>";  echo "<br><a href='inscription.html'>Inscrire un nouveau stagiaire</a>";  ?>  </BODY></HTML> |

* *stagiaires.php* : Affiche la liste des stagiaires contenus dans le fichier *stagiaires.txt*

|  |
| --- |
| <HTML>  <HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/>  <TITLE>Formation STI2D SIN4</TITLE>  </HEAD>  <BODY>  <h1><center>Formation STI2D SIN4</h1>  <h2>Liste des stagiaires</center></center></h2>  <br><br>  <?php  $filename = 'stagiaires.txt';  $handle = fopen($filename, 'r');  while (! feof ($handle)) // Tant qu’on est pas à la fin du fichier  {  $line = fgets($handle); // Lire la ligne courante  echo $line."<br>"; // Afficher la ligne puis retour à la ligne  }  fclose($handle);  ?>  </BODY>  </HTML> |

Le séparateur de champs est un « ; ». Il existe une fonction qui permet de découper un fichier en fonction d’un caractère particulier nommé délimiteur : fgetcsv().

Le script suivant, *stagiaires\_csv.php*, affiche le contenu du fichier texte découpé par ligne et par champs séparés par des « ; » et affiche le résultat dans un tableau :

|  |
| --- |
| <HTML>  <HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/>  <TITLE>Formation STI2D SIN4</TITLE>  </HEAD><BODY>  Création de l’entête du tableau centré sur la page, bordure de 1pts et pas d’espace entre les cellules.  <h1><center>Formation STI2D SIN4</center></h1>  <h2><center>Liste des stagiaires</center></h2>  <br><br>  <?php  $filename = 'stagiaires.txt';  $handle = fopen($filename, 'r');  echo "<table align='center' border='1' cellspacing='0'>";  <tr> : une ligne  <td> : une cellule  echo "<tr><td align='center' bgcolor='#CCCCCC'>NOM</td>";echo "<td align='center' bgcolor='#CCCCCC'>PRENOM</td>";echo "<td align='center' bgcolor='#CCCCCC'>ETABLISSEMENT</td>";echo "<td align='center' bgcolor='#CCCCCC'>VILLE</td>";echo "<tr>";  //fgetcsv retourne FALSE si le fichier est vide  while (($cols = fgetcsv($handle, 1000, ";")) != FALSE)  {  echo "<tr>";  Construction du corps du tableau.  fgetcsv décompose une ligne de 1000 caractères maximum selon le délimiteur « ; » et construit le tableau $cols.  Chaque élément de $cols est affiché ans une cellule.  foreach($cols as $key => $val)  {  echo "<td>$val</td>";  }  }  echo "</tr>";  echo "</table>";  fclose($handle);  ?>  </BODY></HTML> |

Si l’affichage des données contenues dans un fichier texte n’est pas particulièrement compliqué, il est plus difficile d’y rechercher des données précises.

Par exemple, l’utilisateur recherche tous les stagiaires d’un même établissement ou d’une même ville.

Pour facilité le traitement de ce type d’opération, il est préférable d’utiliser une base de données.

## Le SGBD MySQL

### Le langage SQL

Pour apprendre à utiliser le langage SQL du serveur MySQL, nous utiliserons l’IHM phpMyAdmin. Elle se présente sous la forme d’une application web, exécutée dans un navigateur à l’URL :

http://adresse\_IP\_du\_serveur/phpmyadmin/ (ex : http://192.168.1.12/phpmyadmin)

Pour obtenir de l’aide sur phpMyAdmin, consultez la documentation officielle :

<http://www.phpmyadmin.net/localized_docs/fr/Documentation.html>

Le langage SQL permet de manipuler des données au moyen d’instructions dont les plus courantes sont les suivantes :

|  |  |
| --- | --- |
| **Instruction** | **Opération** |
| CREATE TABLE | Création d’une nouvelle table |
| ALTER TABLE | Modification de la structure d’une table (ajout / modification / suppression de champs) |
| DROP TABLE | Suppression d’une table : **DROP TABLE** Nom\_table ; |
| SELECT | Extraire des données d’une ou plusieurs tables |
| INSERT | Insérer des données dans une table |
| UPDATE | Mettre à jour les données d’une table |
| DELETE | Supprimer des données d’une table |

**Remarque** : Le langage SQL n’est pas sensible à la casse mais les instructions sont souvent écrites en majuscules pour les différencier des données.

### Les principales manipulations

#### Ouvrir phpMyAdmin

Dans votre navigateur, saisissez l’URL <http://localhost/phpmyadmin/>

|  |
| --- |
| Ubuntu-2011-03-01-22-08-05 |

L’utilisateur par défaut est *root*. Saisissez le mot de passe *root* de la distribution linux que vous utilisez.

|  |
| --- |
| Ubuntu-2011-03-01-23-30-20  Créer une nouvelle base  Bases de données disponibles |

Créez une nouvelle de donnée nommée *test*. (**Interclassement** : jeu de caractères. Laissez tel quel).

|  |
| --- |
|  |

#### Créer une table

Créez une nouvelle table nommée *stagiaires* de 5 champs.

|  |
| --- |
| Auto\_incrément |

* **id (Int – PRIMARY – Auto incrément)** :

Type entier

Index clé primaire, ce qui signifie que les valeurs de ce champ devront être uniques. Pour garantir cette unicité, nous le laisserons s’auto-incrémenté. L’index est utilisé pour mener les recherches sur la table.

* **nom (varchar(50))** : Chaine de 50 caractères.
* **prenom (varchar(50))**: Chaine de 50 caractères.
* **etablissement (varchar(100))** : Chaine de 100 caractères.
* **ville (varchar(50))**: Chaine de 50 caractères.

Cliquez sur **Sauvegarder** pour créer la table.

|  |
| --- |
| Ubuntu-2011-03-02-00-30-35 |

Notez la requête SQL qui a été soumise au serveur MySQL pour créer la table. La structure générale de la requête de création de table est la suivante :

|  |
| --- |
| **CREATE** **table** nom\_table**(**  Nom\_attribut1 **type**[not null][AUTO\_INCREMENT],  Nom\_attribut2 **type**[not null],  ...,  Nom\_attributN **type**[not null],  **primary key** (nomattribut1)**) ;** |

#### Insérez des données

|  |
| --- |
| Assistant d’insertion de données  Ecrire une requête manuellement |

Cliquez sur **Insérer**.

Inutile de préciser la valeur id, puisqu’il est auto-incrémenté. Remplissez les autre champs puis cliquez sur **Exécuter**.

|  |
| --- |
|  |

Notez la requête SQL qui a été soumise au serveur MySQL pour insérez des données dans la table. La structure générale de la requête d’insertion de données dans une table est la suivante :

|  |
| --- |
| **INSERT INTO** nom\_table (nom\_attribut1,...,nom\_attributN) **values** ('valeur1',...,'valeurN');  Ou  **INSERT INTO** nom\_table SET nom\_attribut1='valeur1', ..., nom\_attributN='valeurN'; |

Insérez les données du fichier stagiaires vu en page 15 en écrivant vous-même les requêtes SQL.

#### Sélection de données dans une table

|  |
| --- |
|  |

Notez la requête SQL qui a été soumise au serveur MySQL pour sélectionner des données dans la table. La structure générale de la requête de sélection de données dans une table est la suivante :

|  |
| --- |
| **SELECT** nom\_attribut1,...,nom\_attributN **FROM** nom\_table [**where** condition] ; |

Ecrivez la requête SQL permettant de sélectionner tous les stagiaires de la ville de Marseille.

|  |
| --- |
| **SELECT** \* **FROM** `stagiaires` **WHERE** **etablissement like "%marseille%" ;** |

|  |
| --- |
|  |

Ecrivez la requête SQL permettant de sélectionner tous les stagiaires du lycée Jean Perrin :

|  |
| --- |
| **SELECT** \* **FROM** `stagiaires` **WHERE** **etablissement like "%perrin%" ;** |

Opérateurs de conditions :

|  |  |
| --- | --- |
| **Opérateur** | Définition |
| **<** | Plus petit que |
| **<=** | Plus petit ou égal à |
| **=** | Égal à |
| **!=** ou **<>** | N'est pas égal à |
| **>=** | Plus grand ou égal à |
| **and** | ET |
| **or** | OU |
| **not** | Négation |
| **like** | Chaine de caractère comme … |
| **Like %chaine** | Chaine de caractère fini par chaine |
| **Like chaine%** | Chaine de caractère commence par chaine |
| **Like %chaine%** | Chaine de caractère contient chaine |

#### Supprimer des données d’une table

|  |
| --- |
|  |

Notez la requête SQL qui a été soumise au serveur MySQL pour supprimer des données de la table. La structure générale de la requête de suppression de données dans une table est la suivante :

|  |
| --- |
| **DELETE FROM** Nom\_table **WHERE** condition ; |

Ecrivez la requête SQL permettant de supprimer l’enregistrement d’**id=6**.

#### Mettre à jour des données d’une table

|  |
| --- |
|  |

Modifiez l’enregistrement n°3 pour mettre à jour le prénom (**André** ou **Andrée** par exemple) et l’établissement (**Lycée Jean Lurçat**) :

|  |
| --- |
|  |

Notez la requête SQL qui a été soumise au serveur MySQL pour mettre à jour des données de la table. La structure générale de la requête de mise à jour de données dans une table est la suivante :

|  |
| --- |
| **UPDATE** Nom\_table where **SET** Nom\_attribut1 = ’nouvelle valeur’, ...,  Nom\_attributN=’nouvelle valeur’ **WHERE** condition ; |

Ecrivez la requête SQL permettant de mettre à jour l’établissement (*Lycée Philippe de Girard*) et la ville (*Avignon*) pour l’enregistrement d’id=5.

#### Proposition de solutions

* Créer la table *stagiaires* :

|  |
| --- |
| CREATE TABLE stagiaires(id int AUTO\_INCREMENT,  nom varchar(50),  prenom varchar(50),  etablissement varchar(100),  ville varchar(50),  PRIMARY KEY(id)); |

**Remarque** : Les noms des champs peuvent comporter des espaces (ex : **un champ**) . Dans ce cas, il faut l’encadrer de caractères « **` »** . Par exemple : **`un champ` varchar(100)**

Ce caractère est accessible par la combinaison de touche Alt Gr + 7 suivit d’un espace pour le faire apparaitre.

Cette remarque est valable pour l’écriture les noms des champs pour tout type de requête.

* Insérer les données du fichier *stagiaires.txt* vues en page 15.

|  |
| --- |
| INSERT INTO stagiaires SET nom='Tartempion',  prenom='Maurice',  etablissement='LPO Adam de Crapone',  ville='Salon de Provence';  Ou  INSERT INTO stagiaires(nom, prenom, etablissement, ville) VALUES('Tartempion',   'Maurice',   'LPO Adam de Crapone',   'Salon de Provence') ; |

* Sélectionner tous les stagiaires de la ville de Marseille.

|  |
| --- |
| SELECT \* FROM stagiaires WHERE ville='marseille' ; |

* Supprimer l’enregistrement d’**id=6**.

|  |
| --- |
| DELETE FROM stagiaires WHERE id=6 ; |

* Mettre à jour l’établissement (*Lycée Philippe de Girard*) et la ville (*Avignon*) pour l’enregistrement d’id=5.

|  |
| --- |
| UPDATE stagiaires SET etablissement='Lycée Philippe de Girard',  ville='Avignon' WHERE id=5 ; |

## Le couple PHP/MySQL

Pour accéder au serveur de bases de données, PHP dispose de plusieurs méthodes. La plus généraliste consiste à utilisé l’extension **PDO**. Elle permet de s’affranchir du SGBD en n'utilisant pas des fonctions au nom trop explicite comme mysql\_query() ou sqlite\_query(), ce qui facilite grandement la migration d'un SGBD à l'autre, voire l'utilisation simultanée ou alternée de plusieurs SGBD avec le même code PHP.

Ainsi, ce qui valable pour *MySQL*, le sera aussi pour *SQLITE*, le SGBD de la carte **FoxBoard G20**.

PDO (*PHP Data Objects*) peut s’apparenter à une classe qui permet de créer un objet représentant la connexion à la base de données. Le constructeur de cet objet nécessite les paramètres suivants :

* Le nom d'hôte (adresse du serveur qui héberge le SGBD)
* Le nom de la base de données à ouvrir
* Le login de l’utilisateur (en général *root*)
* Le mot de passe

Les fichiers de cette partie se trouvent dans le dossier **php-mysql**.

### Ouvrir la base de données

Créez le fichier *ouvriBD.php* et saisissez le code suivant :

|  |
| --- |
| <?php  try  {  // Préparation des paramètres du constructeur  $user="root";  $password="MotDePasse"; //mot de passe de MySQL  $base="test";  $host="localhost";  $SGBD="mysql:host=$host;dbname=$base";  $options[PDO::ATTR\_ERRMODE] = PDO::ERRMODE\_EXCEPTION; //Captage des exceptions    $bdd = new PDO($SGBD, $user, $password, $options); //construction de $bdd  echo "Ouverture de la base $base : OK";  }  catch (Exception $e)  {  die('Erreur : ' . $e->getMessage());  }  ?> |

Ouvrez ce fichier dans votre navigateur. Si tout se passe bien, vous devriez obtenir ceci :

|  |
| --- |
|  |

Notez la structure algorithmique utilisée :

|  |
| --- |
| try  {  //Code à essayer  }  catch (...)  {  //Gestion des erreurs  } |

PHP exécute les instructions à l'intérieur du bloc try{ } . Si une erreur se produit, il s'arrête et saute directement au bloc catch{ }.

Les exceptions (erreurs pour simplifier) sont stockées dans un objet appelé $e et on utilise la méthode getMessage() y accéder.

La fonction die() permet d’afficher une chaine de caractères et de terminer l’exécution du script.

Modifiez le mot de passe et ré-exécuter le script :

|  |
| --- |
|  |

Cette fois, c’est la partie de code contenu dans le bloc catch{ } qui est exécuté. Le message d’erreur renvoyé par le serveur MySQL est affiché et le script se termine.

### Afficher des données

Comme nous l’avons vu précédemment, la structure générale de la requête à utiliser pour sélectionner des données à afficher est :

|  |
| --- |
| **SELECT** nom\_attribut1,...,nom\_attributN **FROM** nom\_table [**where** condition] ; |

Il nous faut maintenant introduire cette requête dans notre code PHP afin qu’il la soumette au SGBD.   
C’est la méthode query de PDO qui va s’en chargé :

|  |
| --- |
| $rep = $bdd->query('**SELECT** nom\_attribut1,...,nom\_attributN **FROM** nom\_table [**where** condition]'); |

Le résultat de la requête renvoyé par le SGBD se trouve dans l’objet $rep.

La méthode fetch (*va chercher*) permet d’accéder aux données renvoyées ligne par ligne. Elles se présenteront sous la forme d’un tableau dont les clés sont les noms des champs de la table exploitée.

|  |
| --- |
| $rep = $bdd->query('SELECT nom\_attribut1,...,nom\_attributN FROM nom\_table [where condition]');  // On affiche chaque entrée une à une  $ligne=1 ;  while ($donnees = $rep->fetch())  {  echo "ligne $ligne : $donnees[' nom\_attribut1'] |...| $donnees[' nom\_attributN']";  $ligne++ ;  } |

Une fois le traitement terminé, il faut libérer la ressource :

|  |
| --- |
| $rep->closeCursor(); |

Pour afficher toutes les informations sur les stagiaires enregistrés dans la table stagiaires, créez un fichier nommé *affichStagiaires1.php* et saisissez le code suivant :

|  |
| --- |
| <HTML><HEAD> <meta http-equiv="Content-Type" content="text/html"; charset="utf-8" />  <TITLE>Formation STI2D SIN4</TITLE></HEAD><BODY><h1><center>Formation STI2D SIN4</center></h1><h2><center>Liste des stagiaires</center></h2><br><br>  <?php  //Inclure le fichier ouvrirBD.php (retirer le message "Ouverture de la base")  include("ouvrirBD.php");  echo "<table align='center' border='1' cellspacing='0'>";echo "<tr><td align='center' bgcolor='#CCCCCC'>NOM</td>";echo "<td align='center' bgcolor='#CCCCCC'>PRENOM</td>";echo "<td align='center' bgcolor='#CCCCCC'>ETABLISSEMENT</td>";echo "<td align='center' bgcolor='#CCCCCC'>VILLE</td>";echo "</tr>";  try  {  $rep = $bdd->query('SELECT \* FROM stagiaires');  // On affiche les données dans un tableau  while ($donnees = $rep->fetch())  {  echo "<tr>"; echo "<td>".$donnees['nom']."</td>";  echo "<td>".$donnees['prenom']."</td>";  echo "<td>".$donnees['etablissement']."</td>";  echo "<td>".$donnees['ville']."</td>"; echo "</tr>";  }  $rep->closeCursor(); // Termine le traitement de la requête  }  catch (Exception $e)  {  die('Erreur : ' . $e->getMessage());  }  ?>  </BODY></HTML> |
|  |

Il est possible que les caractères accentués ne s’affichent pas correctement. En effet, le jeu de caractère par défaut utilisé par MySQL est latin1\_swedish\_ci ce qui correspond au jeu de caractères ISO 8859-1 qui contient les caractères des langues d’Europe de l’Ouest. Or, nous avons utilisé précédemment le jeu de caractère universel UTF8. Il faut donc préciser à MySQL le jeu de caractère que nous souhaitons :

|  |
| --- |
| **$bdd->exec("SET NAMES UTF8");**  $rep = $bdd->query('SELECT \* FROM stagiaires'); |

Pour le moment, pas vraiment de différence avec l’enregistrement des données dans un fichier, mais la où l’utilisation d’une base de données trouve toute son efficacité, c’est dans le traitement sélectif des données.

Par exemple, affichons maintenant uniquement les stagiaires de Marseille. Il suffit de modifier la requête soumise au SGDB :

|  |
| --- |
| $rep = $bdd->query('SELECT \* FROM stagiaires WHERE ville=\'marseille\''); |

Puis uniquement les noms et prénoms :

|  |
| --- |
| $rep = $bdd->query('SELECT nom, prenom FROM stagiaires'); |

Pour créer une liste d’émargement par exemple :

|  |
| --- |
| echo "<table align='center' border='1' cellspacing='0'>";  Création de la colonne Emargement  echo "<tr><td align='center' bgcolor='#CCCCCC'>NOM</td>";  echo "<td align='center' bgcolor='#CCCCCC'>PRENOM</td>";  **echo "<td align='center' bgcolor='#CCCCCC'>Emargement</td>";**  Sélection des champs nom et prenom, les seuls nécessaires à la création d’une liste d’émargement.  echo "</tr>";  **$bdd->exec("SET NAMES UTF8");**  $rep = $bdd->query(**'SELECT nom, prenom FROM stagiaires'**);  // On affiche les données dans un tableau  Affichage des données renvoyées par MySQL (nom et prenom) et d’une colonne vide (enfin elle contient juste un caractère espace)  while ($donnees = $rep->fetch())  {  echo "<tr>";  echo "<td>".$donnees['nom']."</td>";  echo "<td>".$donnees['prenom']."</td>";  **echo "<td>&nbsp; </td>"; //&nbsp; affiche un espace. Nécessaire pour afficher les bordures**  echo "</tr>";  }  echo "</table>"; |

Comme vous pouvez le constater, les implications sur le code sont minimes.

### Croiser des données

L’utilisation d’une base de données permet de croiser simplement des informations de différentes tables.

Nous allons ainsi croiser la table des stagiaires avec une table des villes par département pour savoir quels sont les stagiaires d’un même département.

* Importation des données : à l’aide de phpMyAdmin, cliquez sur **Importer** et sélectionner le fichier *stagiaires.sql*, puis cliquez sur **Exécuter**. Faites de même avec le fichier *ville.sql.*

|  |
| --- |
|  |

* Ecriture de la requête :

|  |
| --- |
| **SELECT** s.nom as nom, s.prenom as prenom, s.type as type,  s.etablissement as etab, v.ville as ville  **FROM** stagiaires as s, villes as v  **WHERE** s.ville=v.ville AND v.departement='Bouches du Rhone'  **ORDER BY** etab |

Exécutez cette requête dans phpMyAdmin et vérifiez son résultat.

|  |
| --- |
|  |

Le mot clé as permet de définir un alias. Les champs sélectionnés sont identifiés par la combinaison nom\_table.nom\_champ.

L’option ORDER BY permet de classer les résultats dans l’ordre croissant ou alphabétique sur le critère définit (ici, classement alphabétique des établissements).

* Modifiez le code de la page *affichStagiaires.php* :

|  |
| --- |
| <HTML>  <HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8" />  <TITLE>Formation STI2D SIN4</TITLE>  </HEAD>  <BODY>  <h1><center>Formation STI2D SIN4</center></h1>  <br><br>  <?php  **$departement="Bouches du Rhone";**  include("ouvrirBD.php");  **echo "<h2><center>Liste des stagiaires du département : $departement</center></h2>";**  echo "<table align='center' border='1' cellspacing='0'>";  echo "<tr><td align='center' bgcolor='#CCCCCC'>NOM</td>";  echo "<td align='center' bgcolor='#CCCCCC'>PRENOM</td>";  echo "<td align='center' bgcolor='#CCCCCC'>TYPE</td>";  echo "<td align='center' bgcolor='#CCCCCC'>ETABLISSEMENT</td>";  echo "<td align='center' bgcolor='#CCCCCC'>VILLE</td>";  echo "</tr>";  **$query="SELECT s.nom as nom, s.prenom as prenom,**  **s.type as type, s.etablissement as etab, v.ville as ville**  **FROM stagiaires as s, villes as v**  **WHERE s.ville=v.ville AND v.departement='$departement'**  **ORDER BY etab";**  **//echo $query; //Affichage de la requete, utile pour deboguer**  **$bdd->exec("SET NAMES UTF8");**  **$rep = $bdd->query($query);**  // On affiche les données dans un tableau  while ($donnees = $rep->fetch())  {  echo "<tr>";  **echo "<td>".$donnees['nom']."</td>";**  **echo "<td>".$donnees['prenom']."</td>";**  **echo "<td>".$donnees['type']."</td>";**  **echo "<td>".$donnees['etab']."</td>";**  **echo "<td>".$donnees['ville']."</td>";**  echo "</tr>";  }  echo "</table>";  $rep->closeCursor(); // Termine le traitement de la requête  ?>  </BODY>  </HTML> |

Testez la page dans le navigateur :

|  |
| --- |
|  |

Pour connaitre les stagiaires originaires des autres départements, il suffira de modifier la valeur de la variable $departement. Le mieux serait d’écrire une page HTML contenant un formulaire de choix de département qui serait transmis au présent script. Le contenu de la variable $departement devrait être affecté à $\_POST["departement"].

### Insérer des données

La structure générale de la requête d’insertion de données dans une table est la suivante :

|  |
| --- |
| **INSERT INTO** nom\_table (nom\_attribut1,...,nom\_attributN) **values** ('valeur1',...,'valeurN');  Ou  **INSERT INTO** nom\_table SET nom\_attribut1='valeur1', ..., nom\_attributN='valeurN'; |

Pour introduire cette requête dans notre code PHP et la soumette au SGBD, on utilise la méthode exec de PDO :

|  |
| --- |
| $nb=$bdd->exec("**INSERT INTO** nom\_table SET nom\_attribut1='valeur1',...,nom\_attributN='valeurN'"); |

Le résultat de la requête renvoyé par le SGBD se trouve dans la variable $nb. Elle contient le nombre de lignes affectées dans la table.

|  |
| --- |
| $nb=$bdd->exec("**INSERT INTO** nom\_table SET nom\_attribut1='valeur1',...,nom\_attributN='valeurN'");  if($nb > 0) echo "$nb ligne(s) ajoutée(s) à la table $table";  else echo "Erreur lors de l’insertion des données"; |

Pour insérer un nouvel enregistrement dans la table stagiaires, créez un fichier nommé *insertStagiaires.php* et saisissez le code suivant :

|  |
| --- |
| <HTML><HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8" /> <TITLE>Formation STI2D SIN4</TITLE></HEAD><BODY><h1><center>Formation STI2D SIN4</center></h1><h2><center>Insertion d’un nouveau stagiaire</center></h2><br><br>  <?php  //Inclure le fichier ouvrirBD.php (retirer le message "Ouverture de la base")  include("ouvrirBD.php");  try  {  $bdd->exec("SET NAMES UTF8"); //imposer le jeu de caractères  $req = "INSERT INTO stagiaires SET nom = '".$\_POST['nom']."',  prenom = '".$\_POST['nom']."',  type = '".$\_POST['nom']."',  etablissement = '".$\_POST['nom']."',  ville = '".$\_POST['nom']."'";  $nb = $bdd->exec($req);  if($nb > 0) echo "$nb ligne(s) ajoutée(s) à la table $table";  else echo "Erreur lors de l’insertion des données";  }  catch (Exception $e)  {  die('Erreur : ' . $e->getMessage());  }  ?>  </BODY></HTML> |

Les variables $nom, $prenom, $type, $etablissement, $ville sont créées à la soumission du formulaire de la page HTML *nouveauStagiaires.html*.

Testez le fonctionnement et assurez-vous que les nouvelles données aient bien été enregistrées dans la base de données (utilisez phpMyAdmin bien sur !).

Si vous tentez d’entrer une chaine de caractères contenant une apostrophe, il est fort probable que MySQL retourne une erreur. Par exemple, $\_POST['nom'] contient la chaine « *l’Hermet* » donc dans la requête on aura : **... nom='l'hermet' ...**

MySQL comprend que **nom** contient **'l'** et ne sait pas interpréter la suite **hermet' ...**.

PHP dispose d’une fonction qui permet « d’habiller » une chaine de caractères en y ajoutant le caractère \ devant l’apostrophe afin qu’ils soit échappé lors de la construction de la requête : addslashes($chaine)

La fonction stripslashes($chaine) permet de « déshabiller » la chaine de ces \.

### Mettre à jour des données

La structure générale de la requête de mise à jour de données dans une table est la suivante :

|  |
| --- |
| **UPDATE** Nom\_table **SET** Nom\_attribut1 = ’nouvelle valeur’, ...,   Nom\_attributN = ’nouvelle valeur’ **WHERE** condition ; |

Pour introduire cette requête dans notre code PHP et la soumette au SGBD, on utilise la méthode exec de PDO :

|  |
| --- |
| $nb = $bdd->exec("**UPDATE** Nom\_table **SET** Nom\_attribut1 = 'nouvelle valeur', ...,   Nom\_attributN = 'nouvelle valeur' **WHERE** condition"); |

Le résultat de la requête renvoyé par le SGBD se trouve dans la variable $nb. Elle contient le nombre de lignes affectées dans la table.

|  |
| --- |
| $nb = $bdd->exec("**UPDATE** Nom\_table **SET** Nom\_attribut1 = 'nouvelle valeur', ...,   Nom\_attributN = 'nouvelle valeur' **WHERE** condition");  if($nb > 0) echo "$nb ligne(s) modifiée(s) dans la table $table";  else echo "Erreur lors de la modification des données"; |

Pour insérer un nouvel enregistrement dans la table stagiaires, créez un fichier nommé *updateStagiaires.php* et saisissez le code suivant :

|  |
| --- |
| <HTML><HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8" /> <TITLE>Formation STI2D SIN4</TITLE></HEAD><BODY><h1><center>Formation STI2D SIN4</center></h1><h2><center>Modification d’un stagiaire</center></h2><br><br>  <?php  //Inclure le fichier ouvrirBD.php (retirer le message "Ouverture de la base")  include("ouvrirBD.php");  try  {  $bdd->exec("SET NAMES UTF8"); //imposer le jeu de caractères  $req = "UPDATE stagiaires SET nom = '".$\_POST['nom']."',  prenom = '".$\_POST['nom']."',  type = '".$\_POST['nom']."',  etablissement = '".$\_POST['nom']."',  ville = '".$\_POST['nom']."'  WHERE id=".$\_POST['id'];  $nb = $bdd->exec($req);  if($nb > 0) echo "$nb ligne(s) modifiée(s) dans la table $table";  else echo "Erreur lors de la modification des données";  }  catch (Exception $e)  {  die('Erreur : ' . $e->getMessage());  }  ?>  </BODY></HTML> |

Les variables $nom, $prenom, $type, $etablissement, $ville et $id sont créées à la soumission du formulaire de la page *modifierStagiaires.php*.

Testez le fonctionnement et assurez-vous que les nouvelles données aient bien été modifiées dans la base de données (utilisez phpMyAdmin bien sur !).

### Supprimer des données

La structure générale de la requête de suppression de données dans une table est la suivante :

|  |
| --- |
| **DELETE FROM** Nom\_table **WHERE** condition ; |

Pour introduire cette requête dans notre code PHP et la soumette au SGBD, on utilise la méthode exec de PDO :

|  |
| --- |
| $nb = $bdd->exec("**DELETE FROM** Nom\_table **WHERE** condition "); |

Le résultat de la requête renvoyé par le SGBD se trouve dans la variable $nb. Elle contient le nombre de lignes affectées dans la table.

|  |
| --- |
| $nb = $bdd->exec("**DELETE FROM** Nom\_table **WHERE** condition ");  if($nb > 0) echo "$nb ligne(s) supprimée(s) dans la table $table";  else echo "Erreur lors de la suppression des données"; |

Pour supprimer un enregistrement dans la table stagiaires, créez un fichier nommé *deleteStagiaires.php* et saisissez le code suivant :

|  |
| --- |
| <HTML><HEAD>  <meta http-equiv="Content-Type" content="text/html"; charset="utf-8" /> <TITLE>Formation STI2D SIN4</TITLE></HEAD><BODY><h1><center>Formation STI2D SIN4</center></h1><h2><center>Suppression d’un stagiaire</center></h2><br><br>  <?php  //Inclure le fichier ouvrirBD.php (retirer le message "Ouverture de la base")  include("ouvrirBD.php");  try  {  $bdd->exec("SET NAMES UTF8"); //imposer le jeu de caractères  $req = "DELETE FROM stagiaires WHERE num=".$\_GET[num];  $nb = $bdd->exec($req);  if($nb > 0) echo "$nb ligne(s) supprimée(s) dans la table $table";  else echo "Erreur lors de la suppression des données";  }  catch (Exception $e)  {  die('Erreur : ' . $e->getMessage());  }  ?>  </BODY></HTML> |

La variable $num est créée à la soumission du formulaire de la page *supprimerStagiaire.php*.

Testez le fonctionnement et assurez-vous que les nouvelles données aient bien été supprimées dans la base de données (utilisez phpMyAdmin bien sur !).

### Aller plus loin

Avant de supprimer des données, il vaut mieux s’assurer de ne pas s’être trompé dans la sélection.

* Modifiez le script *supprimerStagiaires.php* pour qu’il demande à l’utilisateur de confirmer le choix de la suppression (exécution du script *deleteStagiaires.php*) ou d’annuler (retour à la page *supprimerStagiaire.html*).

Pour mener à bien cette opération, il faut se souvenir que les variables PHP n’ont qu’une portée d’une page. Les données transmises par le formulaire de la page *supprimerStagiaires.php* doivent être retransmises au script *deleteStagiaire.php*.

Il faudra donc créer un formulaire dont l’action est *deleteStagiaires.php*, contenant un bouton OUI pour la soumission et des champs cachés <input type=hidden> pour retransmettre au moins le champ **Id** du stagiaire à supprimer.

Vous pouvez vous inspirer du script *modifierStagiaires.php*.

L’utilisateur peut avoir à supprimer plusieurs stagiaires simultanément.

* Modifiez les scripts *supprimer.html*, *deleteStagiaires.php* et *confirmDeleteStagiaire.php* pour répondre à cette exigence.

Le formulaire de choix devra contenir des cases à cocher.

On pourra utiliser une structure algorithmique du type foreach( ){…} pour le traitement.