# Cours Développement à Bases de Logiciels Libres

# Php/MySQL

Enseignant : Riadh BOUSLIMI

Dernière mise à jour le 07 juillet 2008

# TABLE DE MATIÈRES

# Leçon 1

## Introduction

**Le langage PHP** a été créé en 1994 par Rasmus Lerdorf pour les besoins personnels. A l'époque, Php signifiait **P**ersonnal **H**ome **P**age.

Les principaux atouts du langage Php sont :

- ✓ La gratuité et la disponibilité du code source ;
- ✓ Multi plate-forme : on retrouve des versions de PHP compatibles avec la majorité des plates-formes, en occurrence Windows et Lunix ;
- ✓ La simplicité d'interfaçage avec des bases de données (de nombreux SGBD sont supportés, mais le plus utilisé avec ce langage est MySQL, un SGBD gratuit disponible sur les plateformes Lunix et Windows) ;
- ✓ La disponibilité de plusieurs applications Web prêtes à l'emploi, développées à base de Php, tel que : PHPNuke, SPIP, PHPSlash, permettant de montrer facilement et gratuitement des portails Web ;
- ✓ L'intégration au sein de nombreux serveurs Web (Apache, Microsoft IIS, etc.)..

## Environnement de développement

Pour le développement d'un site Web dynamique, il faut installer en local un serveur Web, qui servira pour tester les scripts développés et un SGBD pour tester la connexion à la base de données utilisée et tester les requêtes de manipulation des données de la base.

Vous pourrez programmer en Php sans avoir besoin d'être connecté sur Internet, sinon vous serez amené à envoyer les fichiers vers le serveur de l'hébergeur.

Pour les programmeurs Windows, il existe un utilitaire très pratique (EasyPhP) qui installera Apache, Php, MySQL et aussi phpMyAdmin (une interface conviviale gratuite pour la gestion des bases de données MySQL).

## Mise en œuvre et fonctionnalité du logiciel EasyPHP

- Installez EasyPHP
- Créez un répertoire de travail tpphp sous le répertoire racine du serveur Web apache (par défaut, sous Windows, `c:\Program files\EasyPHP\www`)
- Démarrer EasyPHP
- A l'aide du menu contextuel de l'icône EasyPHP, testez les différentes fonctionnalités offertes.
- Testez le bon fonctionnement du serveur Web en accédant à l'adresse :

**http://localhost** ou **http://127.0.0.1** ou encore **http://nom_machine**

## À savoir

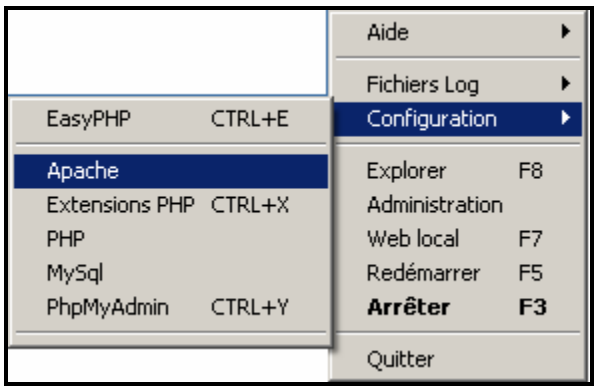



**Figure : Accès aux fonctionnalités du logiciel EasyPHP**

- L'option C**onfiguration** permet de configurer les diverses composantes d'EasyPHP ;
- L'option **Web local** permet d'accéder à la racine du surveur Web (contenu de `c:\Program files\EasyPHP\www) ;`
- Les options **arrêter** et **redémarer** permettent respectivement d'arrêter et de relancer les serveurs ;
- L'option **fichierLog** permet d'ouvrir les différents fichiers de configuration de PHP, MySQL,Apache et EasyPHP ;
- L'option **Quitter** permet d'arrêter les différents serveurs puis quitter EasyPHP.

# Syntaxe de base du langage PHP

Pour que le script soit interprété par le serveur, deux conditions sont nécessaires :

- le fichier contenant le code doit avoir l'extension .php et non .html.
- le code PHP contenu dans le code HTML doit être délimité par les balises `<?php` et `?>`.

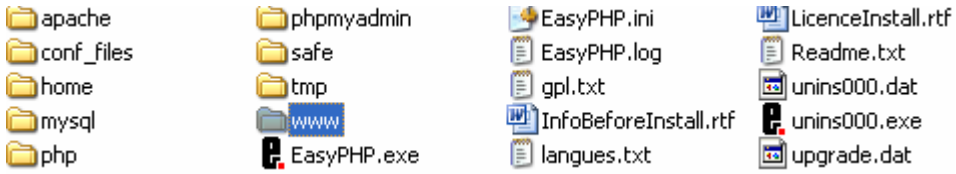
## Exemple

- On ouvre le Bloc-notes de Windows et on encode ce qui suit :

```
<html>
<head>
   <title>Exemple</title>
</head>
<body>
   <?php
       echo "Bonjour";
   ?>
</body>
</html>
```

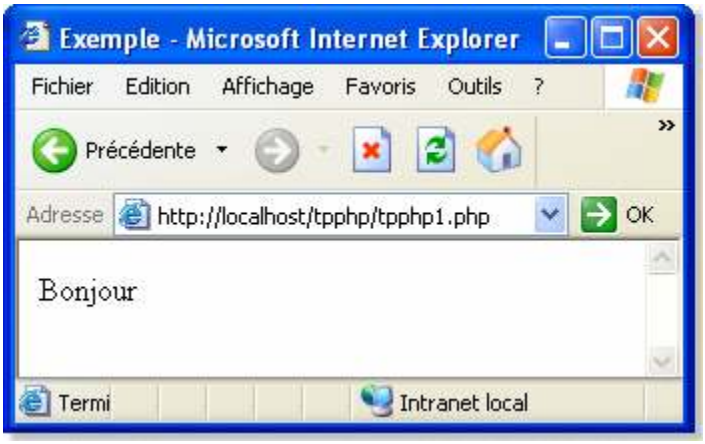
- On notera à ce stade que la fonction **`echo`** permet d'afficher une chaîne de caractères délimitée par des guillemets.

- On enregistre le fichier sous le nom "tpphp1.php" et dans le dossier www de EasyPHP (Program Files → Easyphp → www → tpphp).

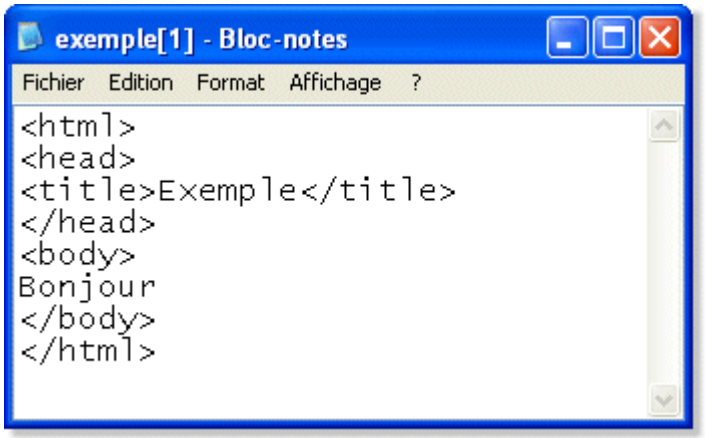



- On lance EasyPHP pour mettre en œuvre le trinôme Apache - PHP - MySQL qui activera ainsi le réseau local.

- On ouvre ensuite Microsoft Internet Explorer. Après avoir encodé l'adresse de la page soit http://localhost/tpphp/tpphp1.php ou de façon équivalente http://127.0.0.1/tpphp/tpphp1.php, on obtient ainsi dans le navigateur.




- S'il nous prend la fantaisie d'afficher la source dans le navigateur (pour Internet Explorer → Affichage → Source), vous remarquerez que toute trace de votre script en PHP a disparu.



## Les commentaires

Un commentaire sera donc noté de la façon suivante :

```
/* Voici
un commentaire ! */
```

Une autre façon d'ajouter des commentaires est le double slash (//) qui permet de mettre, sur une seule ligne, tout ce qui se situe à droite de ce symbole en commentaires.

```
// Voici un commentaire !
```

## Les constantes

Pour définir une constante, on fait appel à la fonction **`define()`**.

### Exemple

```
<html>
<head>
   <title>Les constants en PHP</title>
</head>
<body>
   <?php
        define("FACULTE","Faculté des Sciences de Jendouba");
        echo "Je suis étudiant à la ".FACULTE;
        // C'est du commentaires
   ?>
</body>
</html>
```

### Remarque

L'opérateur «.» permet la concaténation des contenus des objets.


## Les variables

Avec PHP, les noms de variables doivent répondre à certains critères :

- un nom de variable doit commencer par une lettre (majuscule ou minuscule) ou un "_" (pas par un chiffre).
- un nom de variables peut comporter des lettres, des chiffres et le caractère _ (les espaces ne sont, bien entendu, pas autorisés !).

**Quelques exemples :**

| Nom de variable correct |
|---|
| $Variable |
| $Nom_De_Variable |
| $nom_de_variable |
| $nom_de_variable_123 |
| $nom_2_variable |

| Nom de variable correct | Raison |
|---|---|
| $Nom de Variable | comporte des espaces |
| $123Nom_De_Variable | commence par un chiffre |
| $toto@mailcity.com | caractère spécial @ |
| $Nom-de-variable | signe - interdit |
| nom_de_variable | ne commence pas par $ |

Les noms de variables sont sensibles à la casse (PHP fait la différence entre majuscules et minuscules), il faut donc veiller à utiliser des noms comportant la même casse !

**Variables scalaires**

Le langage PHP propose trois types de variables scalaires:

- Entiers : nombres naturels sans décimale (sans virgule).
- Réels : nombres décimaux (on parle généralement de type double, car il s'agit de nombre décimaux à double précision).
- Chaînes de caractères : suite de caractères.

Il n'est pas nécessaire en PHP de typer les variables, c'est-à-dire de définir leur type, il suffit de leur assigner une valeur pour en définir le type :

- Entiers : nombre sans virgule.
- Réels : nombres avec une virgule (en réalité un point).
- Chaînes de caractères : ensembles de caractères entre guillemets simples ou doubles.

| Instruction | Type de la variable |
|---|---|
| $Variable = 0; | type entier |
| $Variable = 12; | type entier |
| $Variable = 0.0; | type réel |
| $Variable = 12.0; | type réel |
| $Variable = "0.0"; | type chaîne |
| $Variable = "Bonjour tout le monde"; | type chaîne |

## Exemple

```
<html>
<head>
   <title>Les constants en PHP</title>
</head>
<body>
  <?php
    $a = 1;
    $b = 3.34;
    $c = "Le monde entier";
    echo $a,"<br>",$b,"<br>",$c;
   ?>

</body>
</html>
```

**Variables tableaux**

Les tableaux stockent des données sous forme de liste. Les données contenues dans la liste sont accessibles grâce à un index (un numéro représentant l'élément de la liste).

Ainsi, pour désigner un élément de tableau, il suffit de faire suivre au nom du tableau l'indice de l'élément entre crochets :

```
$Tableau[0] = 12;
$Tableau[1] = "Bonjour";
```

Avec PHP, il n'est pas nécessaire de préciser la valeur de l'index lorsque l'on veut remplir un tableau, car il assigne la valeur 0 au premier élément (si le tableau est vide) et incrémente les indices suivants. Le code précédent est équivalent à :

```
$Tableau[] = 12;
$Tableau[] = "Bonjour";
```

**Remarque**

Il est important de noter que :

- les indices de tableau commencent à zéro.
- tous les types de variables peuvent être contenus dans un tableau.

Une autre façon de créer un tableau est de passer par l'élément du langage PHP, array().

```
<?php
$Tableau = array(12,"Bonjour");
?>
```

**Variables tableaux associatifs**

PHP permet l'utilisation de chaînes de caractères au lieu de simples entiers pour définir les indices d'un tableau, on parle alors de tableaux associatifs. Cette façon de nommer les indices peut parfois être plus facile à utiliser.

```
$Auteur["Nom"] = "BOUSLIMI";
$Auteur["Prenom"] = "Riadh";
$Auteur["Code_Postal"] = 8100;
```

Ou avec array() :

```
<?php
$Auteur =
array(Nom=>"BOUSLIMI",Prenom=>"Riadh",Code_Postal=>8100);
?>
```

**Conversion de types**

La conversion de types se fait :

- Avec la fonction **settype**, en utilisant la syntaxe :

  **Int settype(string var,string type)**

  Définit de façon explicite le type (**type**) de la variable **var**.

  Le type peut être : `integer`, `double`, `string`, `array` ou `object`.

  La fonction **settype** renvoie **TRUE** en cas de succès, **FALSE** sinon.

- En précédant les variables à convertir par des clauses (**type**)

  ***Exemple***

  ```
  $var = 15.6; // $var est un double

  $var = (int) $var; // c'est maintenant un entier (valeur 15)
  ```

```
$var = (double)$var ; //c'est de nouveau un double (valeur 15,0)

$var_ch = (string)$var ;//$var_ch est une chaine (valeur= "15")
```

**Fonctions de manipulation de variables**

Pour connaître le type d'une variable vous pouvez utiliser l'une des fonctions suivantes : **`is_long(), is_double(), is_string(), is_array() et is_object()`**.

- **`int isset(var)`** : cette fonction retourne le résultat TRUE si la variable var possède une valeur, FALSE sinon.

  ***Exemple***

  ```
  $Prenom = "Riadh" ;

  echo(isset($Prenom)) ; // TRUE
  ```

- **`boolean empty(var)`** : cette fonction retourne le résultat TRUE si la variable var est vide , FALSE sinon.

**Quelques fonctions sur les chaînes de caractères**

- **`$chaine_result = strchr($chaine,$occurrence) ;`**

Retourne une chaîne à partir de la première occurrence d'un caractère d'une chaîne spécifiée.

- **`$nombre = strcmp($chaine_1,$chaine_2) ;`**

Compare en binaire deux chaînes de caractères et retourne un nombre négatif si la première chaîne et plus petite que la seconde, un positif dans le cas contraire ou un nul en cas d'égalité.

- **`$chaine_result = substr($chaine,$position,$longueur) ;`**

Retourne un fragment de la chaîne de caractères à partir de la position spécifiée et jusqu'à une certaine longueur.

- **`$chaine_result = trim($chaine) ;`**

Supprime les espaces blancs eu début et à la fin d'une chaîne de caractères.

- **`$nombre = strlen($chaine) ;`**

Retourne la longueur de la chaîne.

- **`$tableau = explode($delimiteur,$chaine) ;`**

Scinde une chaîne de caractère en fragments à l'aide d'un délimiteur et retourne un tableau.

- **`$chaine_result = implode($delimiteur,$tableau) ;`**

Concatène tous les éléments d'un tableau dans une chaîne séparés par une chaîne de caractères délimitrice.

- **`$chaine_result = str_replace($recherche,$remplacement,$chaine);`**

Remplace toutes les occurrences de la première chaîne par la seconde dans la dernière chaîne de caractères passée en argument.

## Les opérateurs

**Les opérateurs de calcul**

Les opérateurs de calcul permettent de modifier mathématiquement la valeur d'une variable. La valeur initiale de $x est 7.

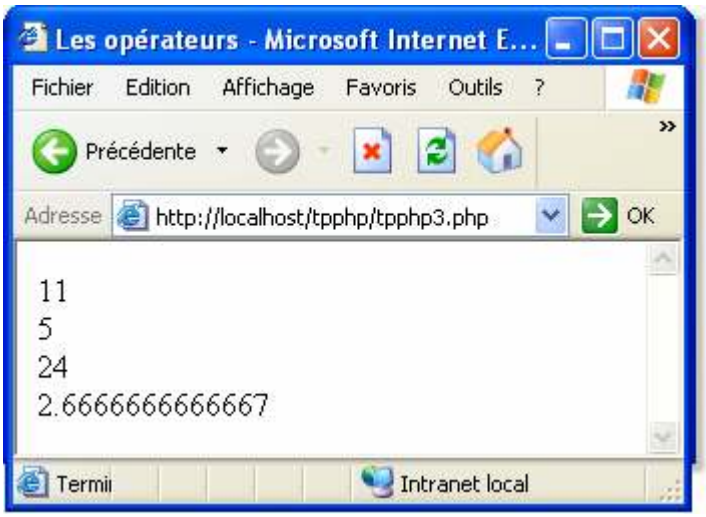
| Opérateur | Dénomination | Effet | Exemple | Résultat |
|---|---|---|---|---|
| + | Addition | Ajoute deux valeurs | $x+3 | 10 |
| - | Soustraction | Soustrait deux valeurs | $x-3 | 4 |
| * | Multiplication | Multiplie deux valeurs | $x*3 | 21 |
| / | Division | Divise deux valeurs | $x/3 | 2.3333333 |
| = | Opérateur d'affectation | Affecte une valeur à une variable | $x=3 | Met la valeur 3 à la variable $x |
| % | modulo | Reste de la division entière | $x=$x%3 | 0 |

## Exemple

```
<html>
<head>
   <title>Les opérateurs</title>
</head>
<body>

  <?php
    $a = 8;
    $b = 3;
    echo $a + $b."<br>";
    echo $a - $b."<br>";
    echo $a * $b."<br>";
    echo $a / $b."<br>";
  ?>

</body>
</html>
```




**Les opérateurs d'assignation**

Ces opérateurs permettent de simplifier des opérations telles que ajouter une valeur dans une variable et stocker le résultat dans celle-ci. Une telle opérations s'écrirait habituellement de la façon suivante par exemple: $x=$x+2

Avec les opérateurs d'assignation il est possible d'écrire cette opération sous la forme suivante: $x+=2
Ainsi, si la valeur de x était 5 avant opération, elle sera de 7 après…

Les opérateurs de ce type sont les suivants:

| Opérateur | Effet |
|---|---|
| += | addition deux valeurs et stocke le résultat dans la variable (à gauche). |
| -= | soustrait deux valeurs et stocke le résultat dans la variable. |
| *= | multiplie deux valeurs et stocke le résultat dans la variable. |
| /= | divise deux valeurs et stocke le résultat dans la variable. |
| %= | donne le reste de la division deux valeurs et stocke le résultat dans la variable. |
| \|= | Effectue un OU logique entre deux valeurs et stocke le résultat dans la variable. |
| ^= | Effectue un OU exclusif entre deux valeurs et stocke le résultat dans la variable. |
| &= | Effectue un Et logique entre deux valeurs et stocke le résultat dans la variable. |
| .= | Concatène deux chaînes et stocke le résultat dans la variable. |

**Les opérateurs d'incrémentation**

Ce type d'opérateur permet d'augmenter ou de diminuer, de façon concise, une variable d'une unité. Ces opérateurs sont très utiles pour des structures telles que les boucles qui ont besoin d'un
compteur (variable qui augmente de un en un).

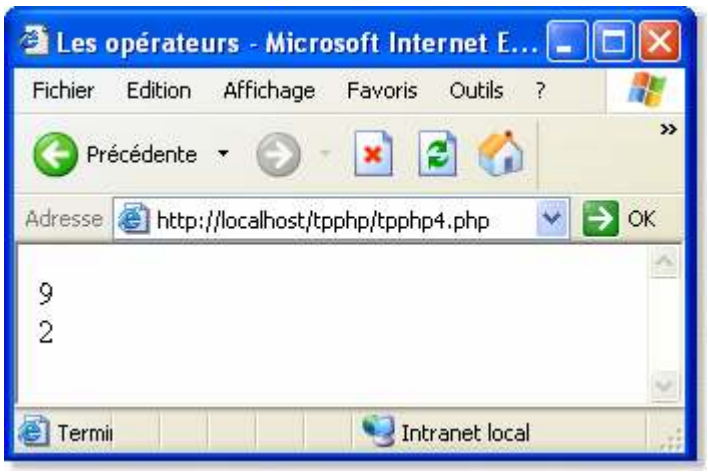
Un opérateur de type $x++ permet de remplacer des notations lourdes telles que $x=$x+1 ou bien $x+=1

| Opérateur | Dénomination | Effet | Syntaxe | Résultat |
|---|---|---|---|---|
| ++ | Incrémentation | Augmente d'une unité la variable | $x++ | 6 (avec x = 5) |
| -- | Décrémentation | Diminue d'une unité la variable | $x-- | 4 (avec x = 5) |

## Exemple

Où a=8 et b=3 :

```
<html>
<head>
   <title>Les opérateurs</title>
</head>
<body>
  <?php
    $a = 8;
    $b = 3;
    $a++;
    echo $a."<br>";
    $b--;
    echo $b."<br>";
  ?>
</body>
</html>
```



**Les opérateurs de comparaison**

Ce type d'opérateur permet de comparer la grandeur de deux données.

| Opérateur | Dénomination | Effet | Exemple | Résultat |
|---|---|---|---|---|
| **==** | opérateur d'égalité | Compare deux valeurs et vérifie leur égalité | $x==3 | Retourne 1 si $x est égal à 3, sinon 0 |
| **<** | opérateur d'infériorité stricte | Vérifie qu'une variable est strictement inférieure à une valeur | $x<3 | Retourne 1 si $x est inférieur à 3, sinon 0 |
| **<=** | opérateur d'infériorité | Vérifie qu'une variable est inférieure ou égale à une valeur | $x<=3 | Retourne 1 si $x est inférieur à 3, sinon 0 |
| **>** | opérateur de supériorité stricte | Vérifie qu'une variable est strictement supérieure à une valeur | $x>3 | Retourne 1 si $x est supérieur à 3, sinon 0 |
| **>=** | opérateur de supériorité | Vérifie qu'une variable est supérieure ou égale à une valeur | $x>=3 | Retourne 1 si $x est supérieur ou égal à 3, sinon 0 |
| **!=** | opérateur de différence | Vérifie qu'une variable est différente d'une valeur | $x!=3 | Retourne 1 si $x est différent de 3, sinon 0 |

**Ne pas confondre l'opérateur d'égalité (==) avec le signe d'affectation (=).**

**Les opérateurs logiques (booléens)**

Ce type d'opérateur permet de vérifier si plusieurs conditions sont vraies :

| Opérateur | Dénomination | Effet | Syntaxe |
|---|---|---|---|
| **\|\| ou OR** | OU logique | Vérifie qu'une des conditions est réalisée | ((condition1)\|\|(condition2)) |
| **&& ou AND** | ET logique | Vérifie que toutes les conditions sont réalisées | ((condition1)&&(condition2)) |
| **XOR** | OU exclusif | Opposé du OU logique | ((condition1)XOR(condition2)) |
| **!** | NON logique | Inverse l'état d'une variable booléenne (retourne la valeur 1 si la variable vaut 0, 0 si elle vaut 1) | |

**Autres opérateurs**

Les opérateurs suivants ne peuvent pas être classés dans une catégorie spécifique mais ils ont tout de même leur importance !

| Opérateur | Dénomination | Effet | Syntaxe |
|---|---|---|---|
| **.** | Concaténation | Joint deux chaînes bout à bout | "Bonjour"."Au revoir" |
| **$** | Référencement de variable | Permet de définir une variable | $MaVariable = 2; |
| **->** | Propriété d'un objet | Permet d'accéder aux données membres d'une classe | $MonObjet-> Propriete |

# Leçon 2

## Les structures conditionnelles

**Qu'est-ce qu'une structure conditionnelle ?**

On appelle les structures conditionnelles, les instructions qui permettent de tester si une condition est vraie ou non, c'est-à-dire si la valeur de son expression vaut 0 ou 1 (le PHP associe le mot clé true à 1 et false à 0).

Ces structures conditionnelles peuvent être associées à des structures qui se répètent suivant la réalisation de la condition. On appelle ces structures des **structures de boucle**.

**L'instruction if**

L'instruction if est la structure de test la plus basique. On la retrouve dans tous les langages de programmation. Elle permet d'exécuter une série d'instruction si une condition est réalisée. La syntaxe de cette expression est la suivante :

```
if (condition réalisée) {
  liste d'instructions
}
```

***Remarques :***

- la condition doit être mise entre des parenthèses.
- il est possible de définir plusieurs conditions à remplir avec les opérateurs ET et OU (&& et ||)
  Ainsi par exemple:
  if ((condition1)&&(condition2)) teste si les deux conditions sont vraies .
  if ((condition1)||(condition2)) exécutera les instructions si l'une ou l'autre des deux conditions est vraie.
- s'il n'y a qu'une instruction, les accolades ne sont pas indispensables.

**L'instruction `if ... else`**

L'instruction if dans sa forme basique ne permet de tester que la réalisation d'une condition. Or la plupart du temps on aimerait pouvoir choisir les instructions à exécuter en cas de non réalisation de la condition.L'expression if ... else permet d'exécuter une autre série d'instruction en cas de non-réalisation de la condition.

La syntaxe de cette expression est la suivante :

```
if (condition réalisée) {
    liste d'instructions
}
else {
    autre série d'instructions (en cas de non-réalisation).
}
```

**Exemple**

Où a=8 et b=3 :

```
<html>
<head>
   <title>Les structures conditionnelles</title>
</head>
<body>
  <?php
    $a = 8;
    $b = 3;
    if ($a < $b)
    {
      echo "a est plus petit que b";
    }
    else
    {
      echo "a n'est pas plus petit que b";
    }
   ?>
</body>
</html>
```


**L'instruction `if ... elseif ... else`**

Il est parfois nécessaire de tester plusieurs conditions de façon exclusive, c'est-à-dire que sur toutes les conditions une seule sera réalisée ...
L'expression if ... elseif ... else permet d'enchaîner une série d'instructions et évite d'avoir à imbriquer des instructions if.

La syntaxe de cette expression est la suivante:

```
if (condition réalisée) {
   liste d'instructions
}
elseif (autre condition réalisée) {
   autre série d'instructions
}
...
else (dernière condition réalisée) {
   série d'instructions
}
```

### L'instruction `switch`

L'instruction **`switch`** permet de faire plusieurs tests de valeurs sur le contenu d'une même variable. Ce branchement conditionnel simplifie beaucoup le test de plusieurs valeurs d'une variable. Cette opération aurait été compliquée (mais possible) avec des if imbriqués. Sa syntaxe est la suivante :

```
switch (Variable) {
  case Valeur1:
     Liste d'instructions
     break;
  case Valeur1:
     Liste d'instructions
     break;
  case Valeurs...:
     Liste d'instructions
     break;
  default:
     Liste d'instructions
     break;
}
```

Les parenthèses qui suivent le mot clé **`switch`** indiquent une expression dont la valeur est testée successivement par chacun des "case". Lorsque l'expression testée est égale à une des valeurs suivant un case, la liste d'instruction qui suit celui-ci est exécutée. Le mot clé break indique la sortie de la structure conditionnelle. Le mot clé **`default`** précède la liste d'instructions qui sera exécutée si l'expression n'est jamais égale à une des valeurs.

**N'oubliez pas d'insérer des instructions break entre chaque test, ce genre d'oubli est difficile à détecter car aucune erreur n'est signalée...**

### Les boucles

Les boucles sont des structures qui permettent d'exécuter plusieurs fois la même série d'instructions jusqu'à ce qu'une condition ne soit plus réalisée.

On appelle aussi ces structures des instructions répétitives ou bien des itérations. La façon la plus commune de faire une boucle, est de créer un compteur (une variable qui s'incrémente, c'est-à-dire qui augmente de 1 à chaque tour de boucle) et de faire arrêter la boucle lorsque le compteur dépasse une certaine valeur.

### La boucle for

L'instruction for permet d'exécuter plusieurs fois la même série d'instructions.

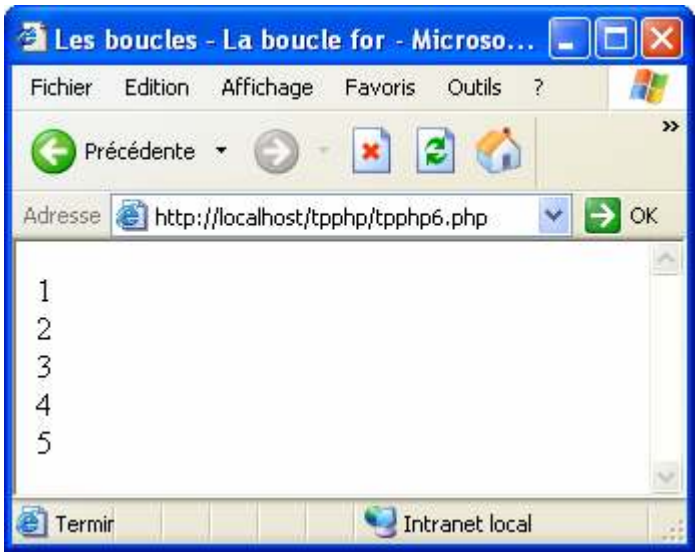
Dans sa syntaxe, il suffit de préciser le nom de la variable qui sert de compteur (et éventuellement sa valeur de départ), la condition sur la variable pour laquelle la boucle s'arrête (par exemple une condition qui teste si la valeur du compteur dépasse une limite) et enfin une instruction qui incrémente (ou décrémente) le compteur.

La syntaxe de cette expression est la suivante :

```
for (compteur; condition; modification du compteur) {
    liste d'instructions
}
```

**Exemple**

```
<html>
<head>
   <title>Les boucles – La boucle for</title>
</head>
<body>
    <?php
      for ($i=1; $i<6; $i++) {
         echo "$i<br>";
      }
    ?>
</body>
</html>
```




Cette boucle affiche 5 fois la valeur de $i, c'est-à-dire 1, 2, 3, 4, 5.
Elle commence à $i=1, vérifie que $i est bien inférieur à 6, jusqu'à atteindre la valeur $i=6, pour laquelle la condition ne sera plus réalisée. Alors la boucle s'interrompra et le programme continuera son cours.
Notons que le langage PHP autorise la déclaration de la variable de boucle dans l'instruction for elle-même !

**L'instruction while**

L'instruction while représente un autre moyen d'exécuter plusieurs fois la même série d'instructions. La syntaxe de cette expression est la suivante :

```
while (condition réalisée) {
   liste d'instructions
}
```

Cette instruction exécute la liste d'instructions aussi longtemps que (while en anglais) la condition est réalisée.

**La condition de sortie pouvant être n'importe quelle structure conditionnelle, les risques de boucle infinie (boucle dont la condition est toujours vraie) sont grands. Une boucle infinie risque de provoquer un plantage du serveur !**

**Saut inconditionnel**

Il peut être nécessaire de faire sauter à la boucle une ou plusieurs valeurs sans pour autant mettre fin à celle-ci.

La syntaxe de cette expression est "**`continue;`**" (cette instruction se place dans une boucle). On l'associe généralement à une structure conditionnelle sinon les lignes situées entre cette instruction et la fin de la boucle ne seraient jamais utilisées.

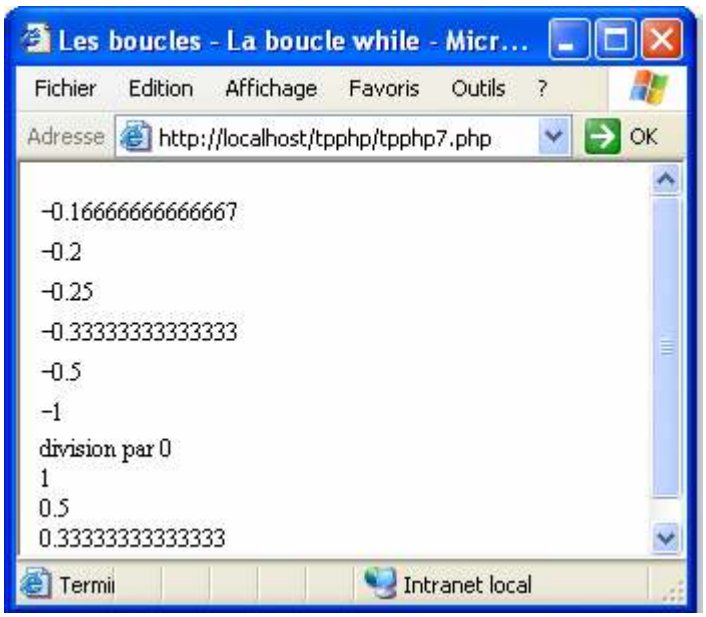
**Exemple:**

Imaginons que l'on veuille imprimer pour $x allant de 1 à 10 la valeur de 1/($x-7) ... Il est évident que pour $x=7 il y aura une erreur. Heureusement, grâce à l'instruction continue il est possible de traiter cette valeur à part puis de continuer la boucle!

```
<html>
<head>
   <title>Les boucles – La boucle while</title>
</head>
<body>

  <?php
   $x=1;
   while ($x<=10) {
      if ($x == 7) {
        echo "division par 0 <br>";
        $x++;
        continue;
      }
      $a = 1/($x-7);
      echo "$a<br>";
    $x++;
   }
  ?>

</body>
</html>
```



**Arrêt inconditionnel**

A l'inverse, on peut vouloir arrêter prématurément la boucle, pour une autre condition que celle précisée dans l'en-tête de la boucle. L'instruction break permet d'arrêter une boucle (for ou while). Il s'agit, tout comme l'instruction continue, de l'associer à une structure conditionnelle, sans laquelle la boucle ne ferait jamais plus d'un tour!

Dans l'exemple précédent, il serait possible de faire arrêter la boucle en cas d'annulation du dénominateur, pour éviter une division par zéro.

```
<html>
<head>
   <title>Les boucles – L'instruction Break</title>
</head>
<body>

  <?php
    for ($x=1; $x<=10; $x++) {
      $a = $x-7;
      if ($a == 0) {
        echo "division par 0 - arrêt !";
        break;
      }
      echo "1/$a<br>";
    }
  ?>

</body>
</html>
```


**Arrêt d'exécution du script**

PHP autorise l'utilisation de la commande exit, qui permet d'interrompre totalement l'interprétation du script, ce qui signifie que le serveur n'envoie plus d'informations au navigateur. Le script est arrêté dans son état. Cette instruction est particulièrement utile lors de l'apparition d'erreurs…

# Leçon 3

## Les entrées/sorties

### La méthode POST

On souhaite créer la page web **tpphp9.php** qui permettra à un étudiant de s'inscrire au niveau de la faculté FSJEG Jendouba. La page contient un formulaire contenant trois champs de saisie et deux boutons(figure n°1). Lorsque l'utilisateur clique sur le bouton valider les données sont envoyées avec la méthode (method=''POST'') vers l'url lui-même **tpphp9.php** et le même fichier affichera un message récapitulatif contenant les différentes informations saisies par l'utilisateur (figure n°2).

Utiliser la fonction **isset** qui permet de tester si une variable est attribuée par une valeur.

```
<html>
<head>
   <title>Formulaire d'inscription</title>
</head>
<body>
 <?php
  if (isset($_POST["ncin"]) AND isset($_POST["nom"]) AND isset($_POST["prenom"])){
    echo "NCIN :".$_POST["ncin"]." Nom:".$_POST["nom"]." Prénom:".$_POST["prenom"];
  }
  else {
 ?>
 <p>
   <form method="POST" action="tpphp9.php">
     NCIN :   <input type="text" name="ncin" size="8" maxlength="8"><br>
     Nom  :   <input type="text" name="nom" size="15" maxlength="20"><br>
     Prénom : <input type="text" name="prenom" size="15" maxlength="20"><br>
     <input type="submit" value="Valider">  
     <input type="reset" value="Annuler">
   </form>
 </p>
 <?php
  }
 ?>
</body>
</html>
```

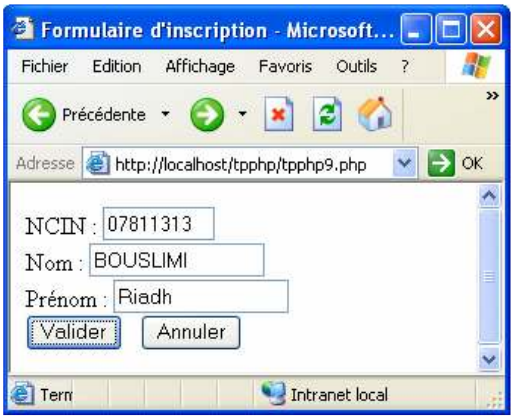



Figure n°1

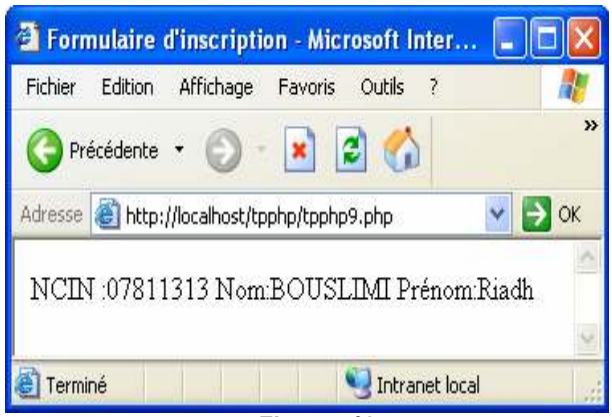



Figure n°2

## La méthode GET

On souhaite modifier le fichier **`tpphp9.php`** en changeant la méthode POST par GET. Quel sera le résultat ?

```
<html>
<head>
   <title>Formulaire d'inscription</title>
</head>
<body>
 <?php
  if (isset($_GET["ncin"]) AND isset($_GET["nom"]) AND isset($_GET["prenom"])){
    echo "NCIN :".$_GET["ncin"]." Nom:".$_GET["nom"]." Prénom:".$_GET["prenom"];
  }
  else {
 ?>
 <p>
   <form method="GET" action="tpphp9.php">
     NCIN :   <input type="text" name="ncin" size="8" maxlength="8"><br>
     Nom  :   <input type="text" name="nom" size="15" maxlength="20"><br>
     Prénom : <input type="text" name="prenom" size="15" maxlength="20"><br>
     <input type="submit" value="Valider">  
     <input type="reset" value="Annuler">
   </form>
 </p>
 <?php
  }
 ?>
</body>
</html>
```

- **Les entrées en Php** : Pour récupérer des données à partir d'une page Php, deux solutions sont possibles :

  - ✓ Récupérer des informations saisies (ou définies) à travers les objets graphiques des formulaires : on utilise l'une des deux syntaxes suivantes **$_GET["nomobjet"]** ou **$_POST["nomobjet"]**, dépendamment de la valeur de la propriété METHOD de la balise FORM (respectivement GET ou POST).

    **NB : `nomobjet` désigne la valeur de la propriété NAME de l'objet duquel on veut récupérer des données.**

  - ✓ Récupérer des informations envoyées en paramètre à un fihcier Php.

    L'envoi des données se fait à travers la syntaxe :

    **`http://adresse-url/nomfichier.php?var1=val1&var2=val2...&varN=valN`**

    La récupération (au niveau du fichier nomfichier.php) se fait grâce à la syntaxe :

    $variable1=$_GET[''var1''] …$variableN=$_GET[''varN'']

- **Les sorties en Php** : l'affichage se fait grâce à l'instruction echo qui permet d'afficher :

  - ✓ **Des chaînes de caractères**

    ***Exemple* : $x=2 ;**

    ```
    echo(''la valeur de x est : $x'');
    ```

  - ✓ **Des chaînes de caractères, des variables et des balises HTML**

    ***Exemple* : $x=2 ;**

    ```
    echo(''<B>la valeur de x est : </B>$x'');
    ```

    **NB : il est possible d'afficher des informations à travers les objets graphiques d'un formulaire.**

    ***Exemple* :**

    ```
    <INPUT TYPE=''TEXT'' NAME=''NOM'' VALUE=< ?php echo(''$nom'');?>>
    ```

# Leçon 4

## Base de données MySQL

### MySQL

MySQL est un système de base de données gratuit et rapide, fonctionnant (entre autres) sous Linux. Etant donné que la majorité des serveurs Web (dont le fameux serveur Apache) fonctionnent sous Linux, MySQL est de ce fait le système de base de données le plus utilisé avec PHP.

### Les formats des données en MySQL

- **Les données numériques :**

Les données numériques peuvent être signées (signed) ou non signées (unsigned).

| Type de champ | Description |
|---|---|
| **TINYINT** | Très petit entier. Compris entre -128 et 127 en signed et entre 0 et 256 en unsigned. |
| **SMALLINT** | Petit entier. Compris entre -32 768 et 32 0767 en signed et entre 0 et 65535 en unsigned. |
| **MEDIUMINT** | Entier moyen. Compris entre -8 388 608 et 8 388 607 en signed et entre 0 et 16 777 215 en unsigned. |
| **INT** | Entier. Compris entre 2 147 483 648 et 2 147 483 647 en signed et entre 0 et 4 294 967 295 en unsigned. |
| **BIGINT** | Grand entier. Compris entre -9 233 372 036 854 et 9 223 372 036 854 775 807 en signed et entre 0 et 18 446 744 073 709 551 615 en unsigned. |
| **FLOAT** | Nombre à virgule flottante en précision simple. L'intervalle de validité va de $-3.402823466E^{+38}$ à $-1.175494351E^{-38}$ en signed et entre 0 et de $1.175494351E^{-38}$ à $3.402823466E^{+38}$ en unsigned. . |
| **DOUBLE** | Nombre à virgule flottante en précision double. L'intervalle de validité va de $-1.7976931348623157E^{+308}$ à $-2.2250738585072014E^{-308}$ en signed et entre 0 et de $2.2250738585072014E^{-308}$ à $1.7976931348623157E^{+308}$ en unsigned. |
| **DECIMAL** | Nombre à virgule flottante signé. Les nombres sont enregistrés sous forme de chaînes de caractères. |

- **Les chaînes de caractères :**

| Type de champ | Description |
|---|---|
| **CHAR(x)** | Chaîne de caractères de longueur fixe où x (compris entre 1 et 256) est le nombre de caractères. |
| **VARCHAR(x)** | Chaîne de caractères de longueur variable où x (compris entre 1 et 256) est le nombre de caractères. |
| **TINYTEXT** | Chaîne de 256 caractères maximum. |
| **TEXT** | Chaîne de 65 585 caractères maximum. |
| **MEDIUMTEXT** | Chaîne de 16 777 215 caractères maximum. |
| **LONGTEXT** | Chaîne de 4 294 967 295 caractères maximum. |

- **Les opérateurs de MySQL**

Les opérateurs arithmétiques :

| Opérateur | Description |
|---|---|
| + | Addition |
| - | Soustraction |
| * | Multiplication |
| / | Division |

Les opérateurs de comparaison :

| Opérateur | Description |
|---|---|
| = | Egal |
| != | Inégal |
| <= | Inférieur ou égal |
| < | Inférieur |
| >= | Supérieur ou égal |
| < | Supérieur |

Les opérateurs logiques :

| Opérateur | Description |
|---|---|
| NOT ou \| | NON logique |
| OR ou \|\| | OU logique |
| AND ou && | ET logique |

## PhpMyAdmin

PhpMyAdmin qui se présente comme un site Web (en local), est un ensemble de scripts PHP permettant de gérer aisément et visuellement MySQL sans devoir passer par l'apprentissage du langage SQL.

PhpMyAdmin peut ainsi :

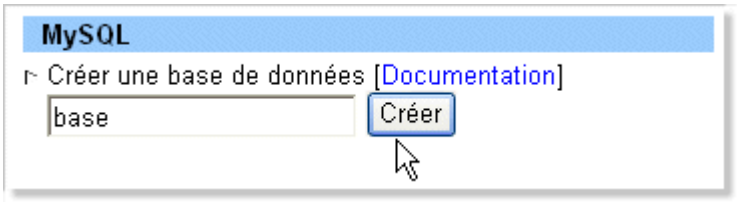
- créer et supprimer des bases de données.
- créer, modifier et supprimer des tables.
- éditer et ajouter des champs.
- insérer des données.
- gérer de multiples utilisateurs avec des permissions différentes.


## Créer une base de données

- **Créer une base de données avec PhpMyAdmin**

Pour la suite de l'apprentissage de MySQL, nous aurons besoin d'une base de données que nous allons créer par PhpMyAdmin.

Créons la base de données base sur le serveur MySQL.

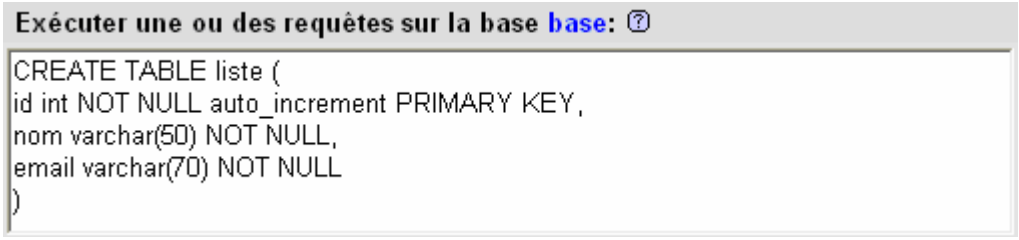



Dans cette base de données, nous allons créer la table **liste**.

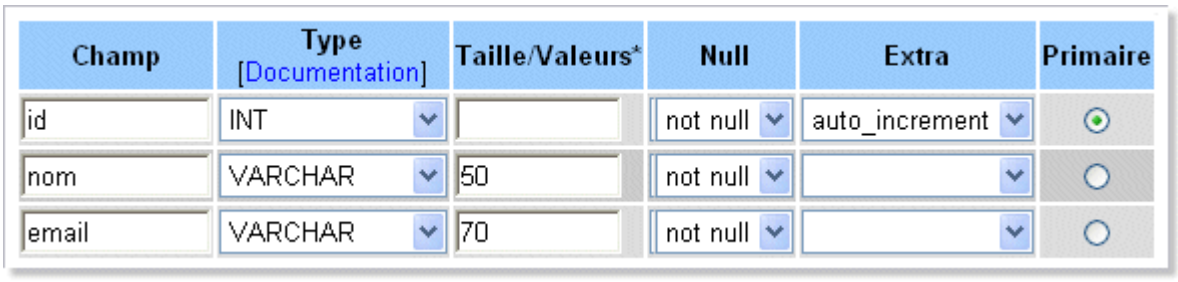
La table liste comportera 3 champs :
- un index id, un entier qui servira de clé primaire.
- un champ nom qui pourra contenir une chaîne de 50 caractères.
- un champ email qui pourra contenir une chaîne de 70 caractères.

Parmi les multiples possibilités offertes par **PhpMyAdmin**, retenons celle qui permet d'encoder les requêtes SQL.

```
CREATE TABLE liste (
id int NOT NULL auto_increment PRIMARY KEY,
nom varchar(50) NOT NULL,
email varchar(70) NOT NULL
)
```



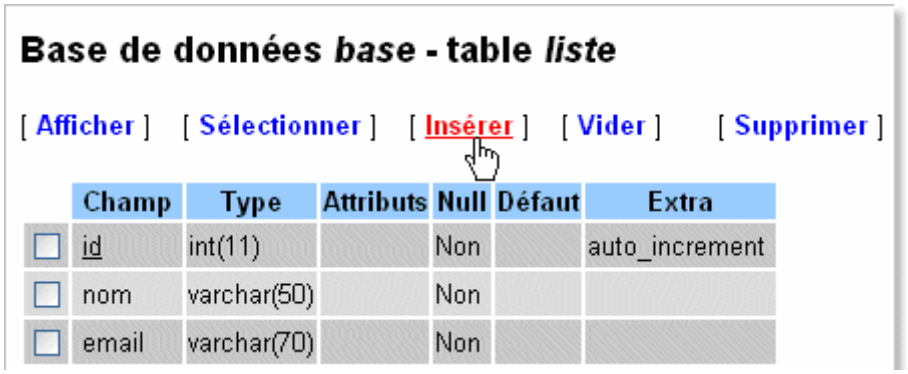

Ou tout simplement :


Insérons un enregistrement de données.

```
INSERT INTO liste VALUES ('', 'BOUSLIMI Riadh',
'riadh_inform@yahoo.fr');
```


Ou tout simplement :




# Php/MySQL

L'utilisation de MySQL avec Php s'effectue en quatre étapes :

- Connexion au serveur de données ;
- Sélection de la base de données ;
- Exécution de la requête ;
- Exploitation des résultats de la requête.

**1er étape : Connexion au serveur de données**

**int mysql_connect(string hostname,string username,string password)**

Pour se connecter, il faut définir l'adresse du serveur de données ainsi que le nom d'utilisateur et le mot de passe. La valeur par défaut de **hostname** est **''localhost''**, de **username** est **''root''** et de **password** est **'' ''**

La fonction **mysql_connect()** retourne un entier permettant de vérifier l'établissement de la connexion.

**2eme étape : Sélection de la base**

**int mysql_db(string database_name,[int link_identifier])**

Le paramètre **database_name** est obligatoire, le paramètre **link_identifier** est facultatif.

La fonction **mysql_db** retourne **true** ou **false** selon que l'opération réussit ou non.

Si on ne donne pas le paramètre **link_identifier**, la fonction utilise la dernière connexion ouverte.

**3eme étape : Exécution d'une requête SQL**

**int mysql_query(string query)**

Envoie au serveur mysql une instruction SQL à exécuter.

**4eme étape : Exploitation d'une requête SQL**

Le traitement des données en résultat est seulement pour les requêtes de sélection. A la suite d'une requête de sélection, les données sont mises en mémoire.

Pour pouvoir les exploiter, Php gère un pointeur de résultat, c'est-à-dire qu'il repère un enregistrement parmi les autres et lorsqu'on veut en lire un, c'est qui est pointé qui sera retourné et le pointeur est déplacé vers l'enregistrement suivant.

La fonction de lecture du résultat est :

**array mysql_fetch_array(int result, int result_type)**

Extrait la ligne sous forme d'un tableau associatif.

Le paramètre **result_type** est facultatif. Il peut prendre les valeurs suivantes :

- **MYSQL_NUM** : Le tableau ne contient que des indices numériques.
- **MYSQL_ASSOC** : Le tableau ne contient que des indices associatifs.
- **MYSQL_BOTH** : Le tableau contient à la fois des indices numériques et des indices associatifs. (c'est la valeur qui est par défaut).

**int mysql_num_rows(int result)** : retourne le nombre d'enregistrement qui ont été retournés par la sélection.

**NB :** L'insertion, la modification et la suppression des données dans les tables se font à travers des requêtes SQL utilisées en paramètres de la fonction **mysql_query**, à savoir respectivement des requêtes de type INSERT, UPDATE, DELETE.

Suite à une opération de mise à jour (insertion, suppression, modification), il est possible de connaître le nombre d'enregistrements affectés par la requête. Ceci est faisable grâce à la fonction suivante : int mysql_affected_rows().

# TP PhP/MYSQL

**Module : D**éveloppement à **B**ases de **L**ogiciels **L**ibres**(Php/MySQL)**

**Enseignant :** Riadh BOUSLIMI

## I. Création de la base de données

On dispose de la classe étudiant suivante :

| Etudiant |
|---|
| NCIN : VARCHAR(8)<br>NOM : VARCHAR(25)<br>PRENOM : VARCHAR(25)<br>SEXE :VARCHAR(1) |
| |

1. Lancez EasyPhp ;
2. Créez la base de données MySQL dont son nom est « **TPMySQL** » ;
3. Créez la table **Etudiant** ;
4. Insérez dans la table **Etudiant** les champs suivants :

| NCIN | NOM | PRENOM | SEXE |
|---|---|---|---|
| 07899039 | KHAMIRI | GHAYA | F |
| 08072024 | NAIMI | ATEF | H |
| 09450267 | TOUIHRI | ALAA | H |
| 07896864 | HAMDI | BOUTHEINA | F |
| ML060005 | TRAORE | GAOUSSOU | H |
| 07883548 | ECHI | HAMDI | H |
| 07881322 | SOLTANI | MAHA | F |
| 07897135 | AYEDI | MARWA | F |
| 07895883 | SAIDI | SAMEH | H |
| 07885840 | TOUIHRI | EMNA | F |
| 07886810 | OUHIIBI | NAWEL | F |

## II. Ajout d'un enregistrement

1. Créez un répertoire de travail « **tpphpmysql** » dans le chemin suivant :
   `C:\Program Files\EasyPHP1-8\www\`
2. Lancez l'éditeur de texte « **Bloc-notes** » ;
3. Créer le fichier « **AjouterEtudiant.php** » qui permet de visualiser un formulaire de saisie pour l'ajout d'un nouveau étudiant. L'apparence de ce formulaire doit ressembler à la figure n°1.
4. Lorsque l'utilisateur clique sur le bouton « **Valider** », *la page est de nouveau appelée* afin d'insérer les nouvelles données sur le nouveau étudiant dans la base de données (figure n°2). Le bouton « **Annuler** » permet d'initialiser le formulaire.
   Programmer le lien hypertexte « **Liste des étudiants** » qui permet d'accéder à la page « **lstEtudiants.php** ».
   **NB :** Avant d'ajouter un étudiant, il faut d'abord s'assurer que l'utilisateur à bien saisie les données sur l'étudiant, ensuite on test l'existence de l'étudiant dans la base de données. S'il est existant alors vous devez le signaler à l'utilisateur à travers un message (figure n°3).


**Figure n°1**

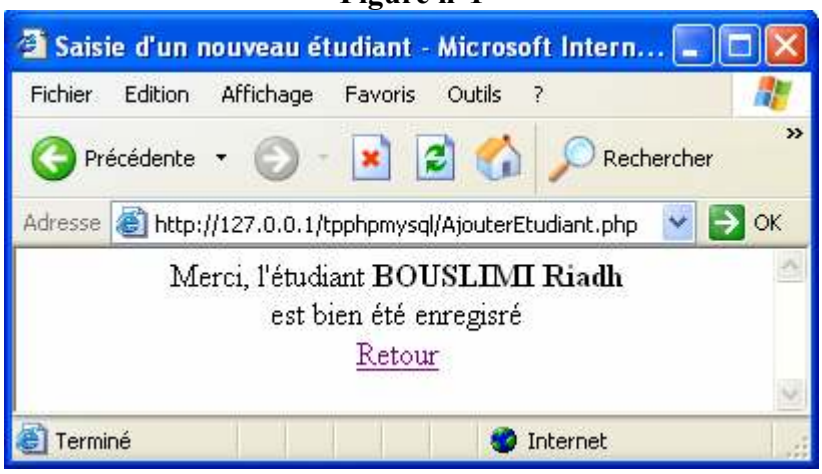



**Figure n°2**


**Figure n°3**

## III. Affichage des enregistrements

1. Lancez l'éditeur de texte « **Bloc-notes** » ;
2. Créez le fichier « **lstEtudiants.php** » qui permet d'afficher la liste des étudiants par sexe et ordonnée par nom et prénom selon l'ordre croissant. Cette liste doit être sous la forme d'un tableau de 5 colonnes.

   **NB :** La dernière colonne de ce tableau doit contenir 2 liens hypertextuels « **modifier** » et « **supprimer** », qui permettent d'appeler respectivement les fichiers « **modif_Etudiant.php** » et « **suppr_Etudiant.php** » qui effectuent les traitements correspondants à partir de la référence du **NCIN** qui doit être passée en paramètre.(figure n°4).

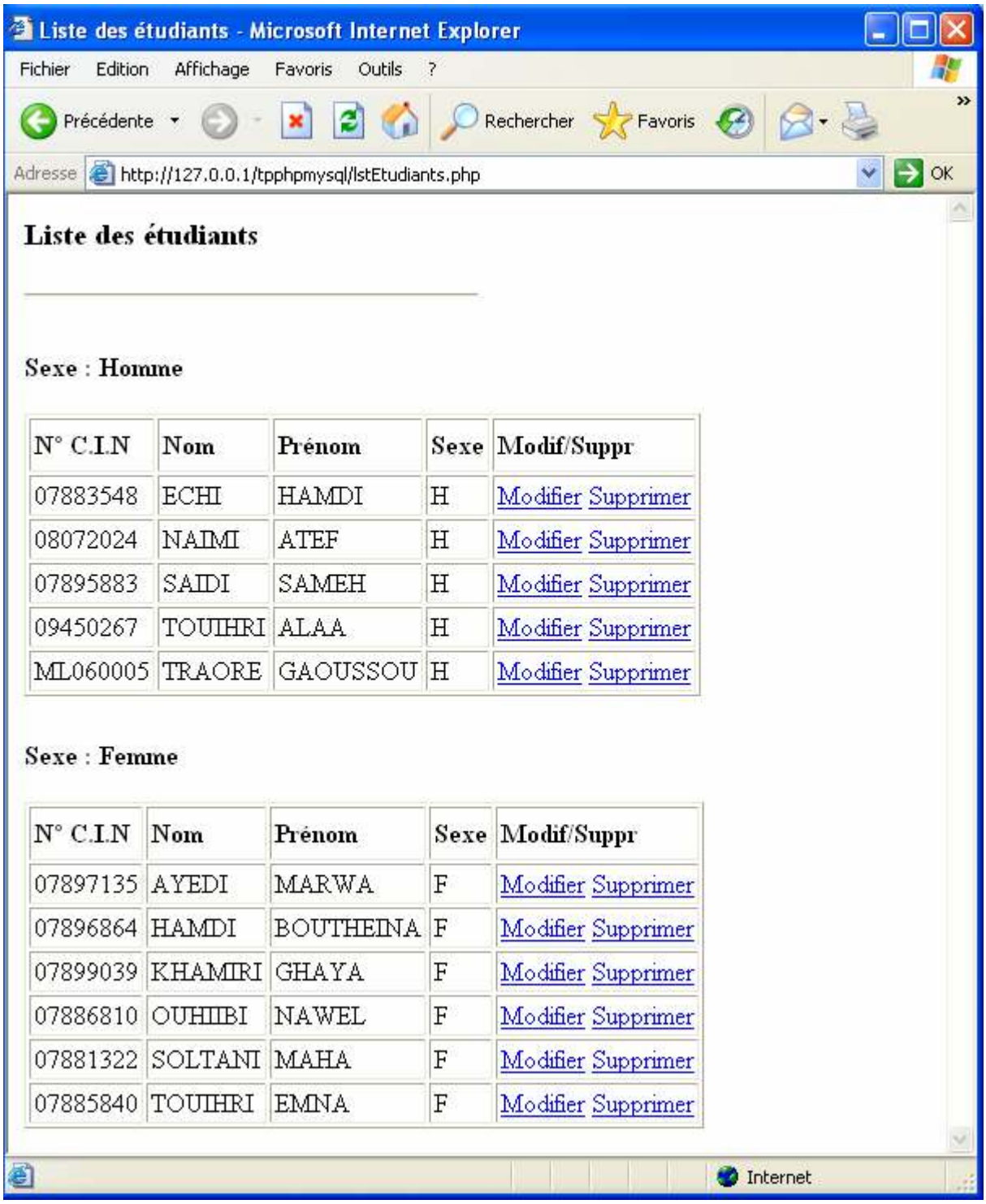



**Figure n°4**

## IV. Modification d'un enregistrement

1. Lancez l'éditeur de texte « **Bloc-notes** » ;
2. Créez le fichier « **modif_Etudiant.php** » qui permet de modifier un étudiant.

Lorsque l'utilisateur clique sur le lien hypertexte « **Modifier** » (figure n°5) dans la page « **lstEtudiants.php** » la page « **modif_Etudiant.php** » apparaîtra avec un formulaire qui contient les renseignement sur l'étudiant en question (figure n°6) .

Lorsque l'utilisateur clique sur le bouton « valider » la figure n°7 apparaîtra.

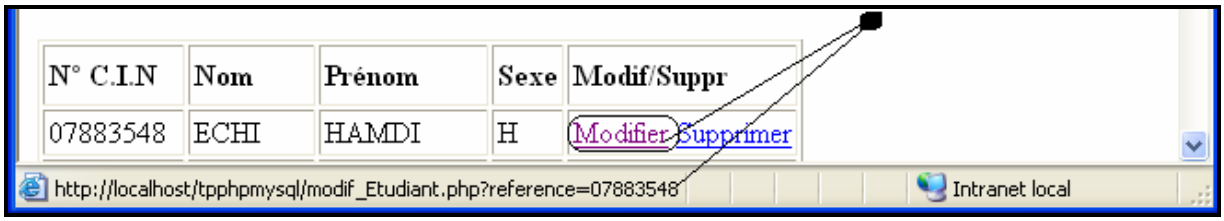



**Figure n°5**


**Figure n°6**


**Figure n°7**

## V. Suppression d'un enregistrement

1. Lancez l'éditeur de texte « **Bloc-notes** » ;
2. Créez le fichier « **suppr_Etudiant.php** » qui permet de modifier un étudiant.

Lorsque l'utilisateur clique sur le lien hypertexte « **Supprimer** » (figure n°8) dans la page « **lstEtudiants.php** » la page « **suppr_Etudiant.php** » apparaîtra avec un message pour indiquer que la suppression a été effectué avec succès (figure n°9) .

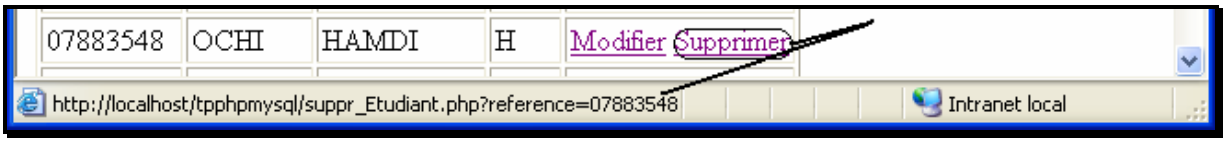



**Figure n°8**


**Figure n°9**

# CORRECTION

```
<!--
    fichier : AjouterEtudiant.php
    Auteur  : Riadh BOUSLIMI
    Module  : Développement à Bases de Logiciels Libres(Php/MySQL)
-->
<html>
<head>
     <title>Saisie d'un nouveau étudiant</title>
</head>
<body>
 <?php
   if (!empty($_POST['ncin']) AND !empty($_POST['nom'])
        AND !empty($_POST['prenom'])) {
       //connexion
       mysql_connect("localhost","root","");
       mysql_select_db("tpmysql");
       //test d'existance de l'étudiant
       $requete1="select * from etudiant where NCIN='".$_POST['ncin']."'";
       $resultat=mysql_query($requete1);
       if ($enreg=mysql_fetch_array($resultat)){
         //l'étudiant est déjà existant
       ?>
         <div align="center">
           <br>L'étudiant <b><?php echo($_POST['nom']." ".$_POST['prenom']);?></b>
               <br>est déjà existant!!!
           <br><br><a href="AjouterEtudiant.php">Retour</a>
         </div>
      <?php
      }else {
       //insertion du nouveau étudiant
       $requete2="insert into etudiant
                   values('".$_POST['ncin']."','".$_POST['nom']."',
                            '".$_POST['prenom']."','".$_POST['sexe']."')";
       mysql_query($requete2);
      ?>
       <div align="center">
        <br>Merci, l'étudiant
            <b><?php echo($_POST['nom']." ".$_POST['prenom']);?></b>
        <br>est bien été enregisré
        <br><br><a href="AjouterEtudiant.php">Retour</a>
       </div>
      <?php
       }//fin du test d'existance
 }else{?>
    <div align="center">
     <h3>Saisie d'un nouveau étudiant</h3>
     <hr size="2" width="50%">
     <!--Ici c'est le Formulaire-->
     <form name="SaisieEtudiant" method="POST" action="AjouterEtudiant.php">
       <table border="1">
        <tr>
          <td width="35%">N°.C.I.N :</td>
          <td width="65%"><input type="text" name="ncin"
                        value="" size="8" maxlength="8"></td>
        </tr>
        <tr>
          <td width="35%">Nom :</td>
          <td width="65%"><input type="text" name="nom" value=""
                        size="25" maxlength="25"></td>
        </tr>

```

```
62          <tr>
63            <td width="35%">Prénom :</td>
64            <td width="65%"><input type="text" name="prenom"
65                                  value="" size="25" maxlength="25"></td>
66          </tr>
67          <tr>
68            <td width="35%">Sexe :</td>
69            <td width="65%">
70                  <select name="sexe">
71                <option value="H">Homme</option>
72                <option value="F">Femme</option>
73              </select>
74            </td>
75          </tr>
76        </table>
77        <br>
78        <input type="submit" value="Valider">  
79        <input type="reset" value="Annuler">
80       </form>
81       <a href="lstEtudiants.php">Liste des étudiants</a>
82      </div>
83  <?php
84    }
85  ?>
86 </body>
87 </html>
```

```
<!--
    fichier : lstEtudiants.php
    Auteur  : Riadh BOUSLIMI
    Module  : Développement à Bases de Logiciels Libres(Php/MySQL)
-->
<html>
<head>
 <title>Liste des étudiants</title>
</head>
<body>
    <div align="Left">
         <h3>Liste des étudiants</h3>
             <hr width="50%" size="2">
      </div>
    <?php
    //connexion
    mysql_connect("localhost","root","");
    mysql_select_db("tpmysql");
      ?>
      <?php
    //Liste des étudiants dont le sexe est Homme
    $requete_H="select * from etudiant where SEXE='H' order by NOM,PRENOM";
    $resultat_H=mysql_query($requete_H);
    if (mysql_num_rows($resultat_H)<>0){
    ?>
       <div align="Left">
       <br>
         <h4>Sexe : <b>Homme</b></h4>
         <table border="1" cellpaccing="2" cellpadding="0">
               <tr>
                  <td><b>N° C.I.N</b></td>
                  <td><b>Nom</b></td>
                  <td><b>Prénom</b></td>
                  <td><b>Sexe</b></td>
                  <td><b>Modif/Suppr</b></td>
               </tr>
        <?php
           while ($enreg_H=mysql_fetch_array($resultat_H)){
             ?>
               <tr>
                  <td><?php echo $enreg_H["NCIN"];?></td>
                  <td><?php echo $enreg_H["NOM"];?></td>
                  <td><?php echo $enreg_H["PRENOM"];?></td>
                  <td><?php echo $enreg_H["SEXE"];?></td>
                  <td>
                     <a href="modif_Etudiant.php?reference=
                               <?php echo $enreg_H["NCIN"];?>
                              ">Modifier</a>
                     <a href="suppr_Etudiant.php?reference=
                               <?php echo $enreg_H["NCIN"];?>
                              ">Supprimer</a>
                  </td>
               </tr>
              <?php
              }
            ?>
         </table>
         </div>
    <?php
    }else{
    ?>
       <div align="Left">
         <h4>Sexe : <b>Homme</b></h4>
         <hr width="50%" size="2">
         <b> Pour le moment, il n'a aucun étudiant enregistré!!!</b>
```

```
66            <hr width="50%" size="2">
67          </div>
68       <?php
69       }
70       ?>
71       <?php
72       //Liste des étudiants dont le sexe est Femme
73       $requete_F="select * from etudiant where SEXE='F' order by NOM,PRENOM";
74       $resultat_F=mysql_query($requete_F);
75       if (mysql_num_rows($resultat_F)<>0){
76       ?>
77          <div align="Left">
78          <br>
79            <h4>Sexe : <b>Femme</b></h4>
80            <table border="1" cellpaccing="2" cellpadding="0">
81               <tr>
82                  <td><b>N° C.I.N</b></td>
83                  <td><b>Nom</b></td>
84                  <td><b>Prénom</b></td>
85                  <td><b>Sexe</b></td>
86                  <td><b>Modif/Suppr</b></td>
87               </tr>
88           <?php
89              while ($enreg_F=mysql_fetch_array($resultat_F)){
90                ?>
91               <tr>
92                  <td><?php echo $enreg_F["NCIN"];?></td>
93                  <td><?php echo $enreg_F["NOM"];?></td>
94                  <td><?php echo $enreg_F["PRENOM"];?></td>
95                  <td><?php echo $enreg_F["SEXE"];?></td>
96                  <td>
97                      <a href="modif_Etudiant.php?reference=
98                                 <?php echo $enreg_F["NCIN"];?>
99                              ">Modifier</a>
100                     <a href="suppr_Etudiant.php?reference=
101                                <?php echo $enreg_F["NCIN"];?>
102                             ">Supprimer</a>
103                 </td>
104              </tr>
105              <?php
106              }
107           ?>
108           </table>
109           </div>
110      <?php
111      }else{
112      ?>
113         <div align="Left">
114         <h4>Sexe : <b>Femme</b></h4>
115           <hr width="50%" size="2">
116         <b> Pour le moment, il n'a aucune étudiante enregistrée!!!</b>
117         <hr width="50%" size="2">
118         </div>
119      <?php
120      }
121      ?>
122 </body>
123 </html>
```

```php
<!--
    fichier : modif_Etudiant.php
    Auteur  : Riadh BOUSLIMI
    Module  : Développement à Bases de Logiciels Libres(Php/MySQL)
-->
<html>
<head>
     <title>Modification d'un étudiant</title>
</head>
<body>
 <?php
  //connexion
  mysql_connect("localhost","root","");
  mysql_select_db("tpmysql");
  if (empty($_POST['ncin']) AND empty($_POST['nom'])
       AND empty($_POST['prenom'])) {
       //Info. sur l'étudiant
       $requete1="select * from etudiant where NCIN='".$_GET['reference']."'";
       $resultat=mysql_query($requete1);
       if ($enreg=mysql_fetch_array($resultat)){
         //Remplissage des renseignements sur l'étudiant en question
       ?>
           <div align="center">
             <h3>Modifier un étudiant</h3>
             <hr size="2" width="50%">
             <form name="ModifierEtudiant" method="POST" action="modif_Etudiant.php">
               <table border="1">
                <tr>
                  <td width="35%">Nom :</td>
                  <td width="65%"><input type="text" name="nom"
                                   value="<?php echo $enreg['NOM'];?>"
                                   size="25" maxlength="25">
                  </td>
                </tr>
                <tr>
                  <td width="35%">Prénom :</td>
                  <td width="65%"><input type="text" name="prenom"
                                  value="<?php echo $enreg['PRENOM'];?>"
                                   size="25" maxlength="25">
                  </td>
               </tr>
               <tr>
                  <td width="35%">Sexe :</td>
                  <td width="65%">
                    <select name="sexe">
                       <?php if (strcmp($enreg['SEXE'],"H")==0) {?>
                           <option value="H" selected>Homme</option>
                       <?php }else{?>
                             <option value="H">Homme</option>
                       <?php }?>
                       <?php if (strcmp($enreg['SEXE'],"F")==0) {?>
                           <option value="F" selected>Femme</option>
                       <?php }else{?>
                             <option value="F">Femme</option>
                       <?php }?>
                    </select>
                   </td>
                </tr>
       </table>
         <input type="hidden" name="ncin" value="<?php echo $enreg['NCIN'];?>">
         <br>
         <input type="submit" value="Valider">  
         <input type="reset" value="Annuler">
       </form>
      <a href="lstEtudiants.php">Liste des étudiants</a>
```

```
66      </div>
67   <?php
68      } ?>
69   <?php
70    }else {
71      //modification de l'étudiant
72       $requete2="Update etudiant set NOM='".$_POST['nom'].
73                 "',PRENOM='".$_POST['prenom'].
74                 "',SEXE='".$_POST['sexe']."'
75                 where NCIN='".$_POST['ncin']."'";
76         mysql_query($requete2);
77       ?>
78         <div align="center">
79          <br>Merci, l'étudiant <b><?php echo($_POST['nom']." ".$_POST['prenom']);?></b>
80          <br>est bien été enregisré
81          <br><br><a href="AjouterEtudiant.php">Retour</a>
82         </div>
83  <?php
84   }
85  ?>
86  </body>
87  </html>
```

```php
<!--
    fichier : suppr_Etudiant.php
    Auteur  : Riadh BOUSLIMI
    Module  : Développement à Bases de Logiciels Libres(Php/MySQL)
-->
<html>
<head>
 <title>Suppression d'un étudiant</title>
</head>
<body>
 <?php
  //connexion
  mysql_connect("localhost","root","");
  mysql_select_db("tpmysql");
  //suppression de l'étudiant
  $requete="delete from etudiant where NCIN='".$_GET['reference']."'";
  mysql_query($requete);
  ?>
    <div align="center">
      <br>Merci, l'étudiant au n° C.I.N  <b><?php echo($_GET['reference']);?></b>
      <br>est bien été supprimé
      <br><br><a href="lstEtudiants.php">Retour</a>
    </div>
</body>
</html>
```

# PROGRAMME

| Leçons | Contenus | Durées |
|---|---|---|
| Leçon1 : Syntaxe de base du langage Php | ▪ Les commentaires<br>▪ Les constantes<br>▪ Les variables<br>▪ Les opérateurs | 2h |
| Leçon2 : Les structures conditionnelles | ▪ La structure **if**<br>▪ L'instruction **switch**<br>▪ La boucle **for**<br>▪ La structure **while**<br>▪ L'instruction « **continue** »<br>▪ L'instruction « **break** » | 4h |
| Leçon3 : Les entrées/Sorties | ▪ La méthode **POST**<br>▪ La méthode **GET**<br>▪ Récupération des données | 2h |
| Leçon4 : Php/MYSQL | ▪ Création d'une base de données MySQL<br>▪ Manipulation d'une base de données avec Php | 2h |
| TP : Php/MYSQL | ▪ Application pratique sur Php/MySQL | 8h |
| | | 18h |