

- **PHP**
  - Éléments de programmation
  - Notions de programmation web
- **MySQL**
  - Interface Web phpMyAdmin
- **Apache**
  - Éléments d'installation et de configuration

### ***PHP c'est quoi?***

- Définition
  - Un langage de scripts permettant la création d'applications Web
  - Indépendant de la plate-forme utilisée puisqu'il est exécuté côté serveur et non côté client.
  - La syntaxe du langage provient de celles du langage C, du Perl et de Java.
  - Ses principaux atouts sont:
    - La gratuité et la disponibilité du code source (PHP4 est distribué sous licence GNU GPL)
    - La simplicité d'écriture de scripts
    - La possibilité d'inclure le script PHP au sein d'une page HTML
    - La simplicité d'interfaçage avec des bases de données
    - L'intégration au sein de nombreux serveurs web (Apache, Microsoft IIS, ...)

### *Intégration PHP et HTML*

- Principe
  - Les scripts PHP sont généralement intégrés dans le code d'un document HTML
  - L'intégration nécessite l'utilisation de balises
    - avec le style xml : `<? ligne de code PHP ?>`
    - Avec le style php: `<?php ligne de code PHP ?>`
    - avec le style JavaScript :  
`<script language=«php»> ligne de code PHP </script>`
    - avec le style des ASP : `<% ligne de code ASP %>`

### *Intégration PHP et HTML*

- Forme d'une page PHP
  - Intégration directe

```
< ?php
//ligne de code PHP
?>
<html>
<head> <title> Mon script PHP </title>
</head>
<body>
//ligne de code HTML
< ?php
//ligne de code PHP
?>
//ligne de code HTML
....
</body> </html>
```

### *Intégration PHP et HTML*

- Forme d'une page PHP
  - Inclure un fichier PHP dans un fichier HTML : include()

Fichier Principal

```
<html>
<head>
<title> Fichier d'appel </title>
</head>
<body>
<?php
$salut = " BONJOUR" ;
include "information.inc" ;
?>
</body>
</html>
```

Fichier à inclure : information.inc

```
<?php
$chaine=$salut . « C'est PHP » ;
echo " <table border= '3'
<tr> <td width = '100%' >
<h2> $chaine</h2>
</td> </tr></table> ";
?>
```

### *Intégration PHP et HTML*

- Envoi du code HTML par PHP
  - La fonction echo : `echo Expression;`
    - `echo "Chaine de caracteres";`
    - `echo (1+2)*87;`
  - La fonction print : `print(expression);`
    - `print("Chaine de caracteres");`
    - `print ((1+2)*87);`
  - La fonction printf : `printf (chaîne formatée);`
    - `printf ("Le périmètre du cercle est %d",$Perimetre);`

### ***Syntaxe de base***

- Typologie
  - Toute instruction se termine par un point-virgule
  - Sensible à la casse
    - Sauf par rapport aux fonctions
- Les commentaires
  - `/* Voici un commentaire! */`
  - `//` un commentaire sur une ligne

### ***Les constantes***

- Les constantes
  - **Define**("nom\_constante", valeur\_constante )
    - `define ("ma_const", "Vive PHP5") ;`
    - `define ("an", 2017) ;`
  - Les constantes prédéfinies
    - `NULL`
    - `_FILE_`
    - `_LINE_`
    - `PHP_VERSION`
    - `PHP_OS`
    - `TRUE` et `FALSE`
    - `E_ERROR`



### ***Les variables***

- Principe
  - Commencent par le caractère **\$**
  - N'ont pas besoin d'être déclarées
- Fonctions de vérifications de variables
  - Doubleval(), empty(), gettype(), intval(),
  - is\_array(), is\_bool(), is\_double(), is\_float(), is\_int(), is\_integer, is\_long(), is\_object(), is\_real(), is\_numeric(), is\_string()
  - Isset(), settype(), strval(), unset()
- Affectation par valeur et par référence
  - Affectation par valeur : **\$b=\$a**
  - Affectation par (référence) variable : **\$c = &\$a**

### *Les variables*

- Visibilité des variables
  - Variable locale
    - Visible uniquement à l'intérieur d'un contexte d'utilisation
  - Variable globale
    - Visible dans tout le script
    - Utilisation de l'instruction `global` dans des contextes locaux

```
<?
$var = 100;
function test(){
    global $var;
    return $var;
}
$resultat = test();
if ($resultat)
    echo $resultat;
else
    echo "erreur ";
?>
```

### *Les variables*

- Une variable statique est une variable locale qui ne perd pas sa valeur à chaque fois que la fonction est exécutée.
  - On utilise l'attribut **static** pour déclarer une telle variable.
  - Ce type de variable est très utile pour la création de fonctions récursives.

```
<?php

function compte () {
    static $compteur = 0 ;
    $compteur++ ;
    echo "$compteur " ;
    if ($compteur < 10) compte() ;
}

compte() ;
?>
```

### *Les variables*

- Les variables dynamiques

- Permettent d'affecter un nom différent à une autre variable

```
$nom_variable = 'nom_var';
```

```
$$nom_variable = valeur; // équivaut à $nom_var = valeur;
```

- Les variables tableaux sont également capables de supporter les noms dynamiques

```
$nom_variable = array("val0", "val1", ..., "valN");
```

```
${$nom_variable[0]} = valeur; $val0 = valeur;
```

```
$nom_variable = "nom_var";
```

```
${$nom_variable}[0] = valeur;
```

```
$nom_var[0] = valeur;
```

- Les accolades servent aussi à éviter toute confusion lors du rendu d'une variable dynamique

```
echo "Nom : $nom_variable - Valeur : ${$nom_variable}";
```

```
// équivaut à echo "Nom : $nom_variable - Valeur : $nom_var";
```

### *Les variables prédéfinies*

- Le langage php est doté d'une multitude de variables d'environnement que la fonction **phpinfo()** permet d'afficher (ainsi que bien d'autres informations sur la configuration du serveur Apache). Ces variables permettent au **script** d'accéder à des informations très utiles voire quelques fois indispensables. Quelques variables :

Variable	Description
\$_SERVER["HTTP_HOST"]	Nom d'hôte de la machine du client (associée à l'adresse IP)
\$_SERVER["HTTP_REFERER"]	URL de la page qui a appelé le script PHP
\$_SERVER["HTTP_ACCEPT_LANGUAGE"]	Langue utilisée par le serveur (par défaut en-us)
\$_SERVER["HTTP_ACCEPT"]	Types MIME reconnus par le serveur (séparés par des virgules)
\$_SERVER["CONTENT_TYPE"]	Type de données contenu présent dans le corps de la requête. Il s'agit du type MIME des données
\$_SERVER["REMOTE_ADDR"]	L'adresse IP du client appelant le script CGI
\$_SERVER["PHP_SELF"]	Nom du script PHP

### *Les tableaux*

- Une variable tableau est de type **array**. Un tableau accepte des éléments de tout type. Les éléments d'un tableau peuvent être de types différents et sont séparés d'une virgule.
- Un tableau peut être initialisé avec la syntaxe **array**.
- *Exemple :*
- **\$tab\_colors = array('red', 'yellow', 'blue', 'white');**
- **\$tab = array('foobar', 2002, 20.5, \$name);**
- Mais il peut aussi être initialisé au fur et à mesure.
- *Exemples :*
- **\$prenoms[ ] = "Clément";**
- **\$prenoms[ ] = "Justin";**
- **\$prenoms[ ] = "Tanguy";**
- **\$villes[0] = "Paris";**
- **\$villes[1] = "Londres";**
- **\$villes[2] = "Lisbonne";**

### *Les tableaux*

- Parcours d'un tableau.

```
$tab = array('Hugo', 'Jean', 'Mario');
```

- *Exemple 1 :*

```
$i=0;
```

```
while($i <= count($tab)) {           // count() retourne le nombre d'éléments  
    echo $tab[$i].'\n';  
    $i++;  
}
```

- *Exemple 2 :*

```
foreach($tab as $elem) {  
    echo $elem."\n";  
}
```

La variable **\$elem** prend pour valeurs successives tous les éléments du tableau **\$tab**.

### *Les tableaux*

- Quelques fonctions:
- **count(\$tab), sizeof** : retournent le nombre d'éléments du tableau
- **in\_array(\$var,\$tab)** : dit si la valeur de **\$var** existe dans le tableau **\$tab**
- **list(\$var1,\$var2...)** : transforme une liste de variables en tableau
- **range(\$i,\$j)** : retourne un tableau contenant un intervalle de valeurs
- **shuffle(\$tab)** : mélange les éléments d'un tableau
- **sort(\$tab)** : trie alphanumérique les éléments du tableau
- **rsort(\$tab)** : trie alphanumérique inverse les éléments du tableau
- **implode(\$str,\$tab), join** : retournent une chaîne de caractères contenant les éléments du tableau **\$tab** joints par la chaîne de jointure **\$str**
- **explode(\$delim,\$str)** : retourne un tableau dont les éléments résultent du hachage de la chaîne **\$str** par le délimiteur **\$delim**
- **array\_merge(\$tab1,\$tab2,\$tab3...)** : concatène les tableaux passés en arguments
- **array\_rand(\$tab)** : retourne un élément du tableau au hasard



### ***Les tableaux associatifs***

- Un tableau associatif est appelé aussi *dictionnaire* ou *hashtable*. On associe à chacun de ses éléments une clé dont la valeur est de type chaîne de caractères.
- L'initialisation d'un tableau associatif est similaire à celle d'un tableau normal.
- *Exemple 1 :*
- **\$personne = array("Nom" => "César", "Prénom" => "Jules");**
- *Exemple 2 :*
- **\$personne["Nom"] = "César";**
- **\$personne["Prénom"] = "Jules";**
- Ici à la clé **"Nom"** est associée la valeur **"César"**.

### ***Les tableaux associatifs***

Parcours d'un tableau associatif.

```
$personne = array('Nom' => "César", "Prénom" => "Jules");
```

*Exemple 1 :*

```
foreach($personne as $elem) {  
    echo $elem;  
}
```

Ici on accède directement aux éléments du tableau sans passer par les clés.

*Exemple 2 :*

```
foreach($personne as $key => $elem) {  
    echo "$key : $elem";  
}
```

Ici on accède simultanément aux clés et aux éléments.

### *Les tableaux associatifs*

- Quelques fonctions :
- **array\_count\_values(\$tab)** : retourne un tableau contenant les valeurs du tableau **\$tab** comme clés et leurs fréquence comme valeur (utile pour évaluer les redondances)
- **array\_keys(\$tab)** : retourne un tableau contenant les clés du tableau associatif **\$tab**
- **array\_values(\$tab)** : retourne un tableau contenant les valeurs du tableau associatif **\$tab**
- **array\_search(\$val,\$tab)** : retourne la clé associée à la valeur **\$val**
- L'élément d'un tableau peut être un autre tableau.
- Les tableaux associatifs permettent de préserver une structure de données.

### ***Les tableaux associatifs***

- Quelques fonctions alternatives pour le parcours de tableaux (normaux ou associatifs) :
  - **reset(\$tab)** : place le pointeur sur le premier élément
  - **current(\$tab)** : retourne la valeur de l'élément courant
  - **next(\$tab)** : place le pointeur sur l'élément suivant
  - **prev(\$tab)** : place le pointeur sur l'élément précédant
  - **each(\$tab)** : retourne la paire clé/valeur courante et avance le pointeur

- *Exemple :*

```
$colors = array("red", "green", "blue");  
$nbr = count($colors);  
reset($colors);  
for($i=1; $i<=$nbr; $i++) {  
    echo current($colors)."<br />";  
    next($colors);  
}
```

### ***Les tableaux***

`$tableau = array_filter($variable, "fonction")`

retourne un tableau contenant les enregistrements filtrés d'un tableau à partir d'une fonction.

```
<?php
```

```
function impair($var)
```

```
{return ($var % 2 == 1);}
```

```
function pair($var)
```

```
{return ($var % 2 == 0);}
```

```
$array1 = array ("a"=>1, "b"=>2, "c"=>3, "d"=>4, "e"=>5);
```

```
$array2 = array (6, 7, 8, 9, 10, 11, 12);
```

```
echo "Impairs :\n";
```

```
print_r(array_filter($array1, "impair"));
```

```
echo "Pairs :\n";
```

```
print_r(array_filter($array2, "pair"));
```

```
?>
```

### ***Les types de données***

- Principe
  - Pas besoin d'affecter un type à une variable avant de l'utiliser
    - La même variable peut changer de type en cours de script
    - Les variables issues de l'envoi des données d'un formulaire sont du type string
- Les différents types de données
  - Les entiers : le type **Integer**
  - Les flottants : le type **Double**
  - Les tableaux : le type **array**
  - Les chaînes de caractères : le type **string**
  - Les **objets**

### *Les types de données*

- Le transtypage
  - La fonction `settype()` permet de convertir le type auquel appartient une variable

```
<? $nbre=10;
    settype($nbre, " double ");
    echo " la variable $nbre est de type " , gettype($nbre); ?>
```

- Transtypage explicite : le cast
  - (int), (integer) ; (real), (double), (float); (string); (array); (object)

```
<? $var=" 100 FRF ";
    Echo " pour commencer, le type de la variable est $var, gettype($var);
    $var =(double) $var;
    echo <br> Après le cast, le type de la variable est $var ", gettype($var);
    echo "<br> et a la valeur $var ";  ?>
```

- Détermination du type de données
  - `gettype()`, `is_long()`, `is_double()`, `is_string()`, `is_array()`,  
`is_object()`, `is_bool()`

### ***Les chaînes de caractères***

- Quelques fonctions de manipulation

`chaîne_result = addCSlashes(chaîne, liste_caractères);`  
ajoute des slashes dans une chaîne

`chaîne_result = addslashes(chaîne);`  
ajoute un slash devant tous les caractères spéciaux.

`chaîne_result = chop(chaîne);`  
supprime les espaces blancs en fin de chaîne.

`caractère = chr(nombre);`  
retourne un caractère en mode ASCII

`chaîne_result = crypt(chaîne [, chaîne_code])`  
code une chaîne avec une base de codage.

`echo expression_chaîne;`  
affiche à l'écran une ou plusieurs chaînes de caractères.

`$tableau = explode(délimiteur, chaîne);`  
scinde une chaîne en fragments à l'aide d'un délimiteur et retourne un tableau.



### Les opérateurs

- Les opérateurs
  - les opérateurs de calcul
  - les opérateurs d'assignation
  - les opérateurs d'incrémentation
  - les opérateurs de comparaison
  - les opérateurs logiques
  - les opérateurs bit-à-bit
  - les opérateurs de rotation de bit

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

Opérateur	Dénomination	Effet	Syntaxe	Résultat
&	ET bit-à-bit	Retourne 1 si les deux bits de même poids sont à 1	9 & 12 (1001 & 1100)	8 (1000)
	OU bit-à-bit	Retourne 1 si l'un ou l'autre des deux bits de même poids est à 1 (ou les deux)	9   12 (1001   1100)	13 (1101)
^	OU bit-à-bit	Retourne 1 si l'un des deux bits de même poids est à 1 (mais pas les deux)	9 ^ 12 (1001 ^ 1100)	5 (0101)
~	Complément (NON)	Retourne 1 si le bit est à 0 (et inversement)	~9 (~1001)	6 (0110)

Opérateur	Dénomination	Effet	Exemple	Résultat
+	opérateur d'addition	Ajoute deux valeurs	\$x+3	10
-	opérateur de soustraction	Soustrait deux valeurs	\$x-3	4
*	opérateur de multiplication	Multiplie deux valeurs	\$x*3	21
/	plus: opérateur de division	Divise deux valeurs	\$x/3	2.3333333
=	opérateur d'affectation	Affecte une valeur à une variable	\$x=3	Met la valeur 3 dans la variable \$x

### Les opérateurs

Opérateur	Dénomination	Effet	Syntaxe	Résultat
<<	Rotation à gauche	Décale les bits vers la gauche (multiplie par 2 à chaque décalage). Les zéros qui sortent à gauche sont perdus, tandis que des zéros sont insérés à droite	6 << 1 (110 << 1)	12 (1100)
>>	Rotation à droite avec conservation du signe	Décale les bits vers la droite (divise par 2 à chaque décalage). Les zéros qui sortent à droite sont perdus, tandis que le bit non-nul de poids plus fort est recopié à gauche	6 >> 1 (0110 >> 1)	3 (0011)
Opérateur	Dénomination	Effet	Syntaxe	Résultat
.	Concaténation	Joint deux chaînes bout à bout	"Bonjour"."Au revoir"	"BonjourAu 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	

### ***Les fonctions***

- Comme tout langage de programmation, php permet l'écriture de fonctions.
  - Les fonctions peuvent prendre des arguments dont il n'est pas besoin de spécifier le type. Elles peuvent de façon optionnelle retourner une valeur.
  - L'appel à une fonction peut ne pas respecter son prototypage (nombre de paramètres). Les identificateurs de fonctions sont insensibles à la casse.

- *Exemple :*

```
function mafonction($toto) {  
    $toto += 15;  
    echo "Salut !";  
    return ($toto+10);  
}
```

```
$nbr = MaFonction(15.1);
```

- */\* retourne 15.1+15+10=40.1, les majuscules n'ont pas d'importance \*/*

### Les fonctions

- On peut donner une valeur par défaut aux arguments lors de la déclaration de la fonction. Ainsi, si un argument est « oublié » lors de l'appel de la fonction, cette dernière lui donnera automatiquement une valeur par défaut décidée à l'avance par le programmeur.

- *Exemple :*

```
function Set_Color($color="black") {  
    global $car;  
    $car["color"] = $color;  
}
```

- Forcer le passage de paramètre par référence (équivalent à user de **global**):

- *Exemple :*

```
function change(&$var) { // force le passage systématique par référence  
    $var += 100;          // incrémentation de +100  
}  
$toto = 12;              // $toto vaut 12  
change($toto);           // passage par valeur mais la fonction la prend en référence  
echo $toto;              // $toto vaut 112
```

### Les fonctions

- Grâce à une petite astuce, il est possible de faire retourner plusieurs valeurs d'une fonction. En retournant un tableau ayant pour éléments les variables à retourner. Dans l'appel de la fonction, il faudra alors utiliser la procédure **list()** qui prend en paramètre les variables à qui ont doit affecter les valeurs retournées. On affecte à **list()** le retour de la fonction.

- Exemple :

```
function trigo($nbr) {  
    return array(sin($nbr), cos($nbr), tan($nbr)); // retour d'un tableau  
}  
$r = 12;  
list($a, $b, $c) = trigo($r);                /* affectation aux variables $a,$b et $c  
                                              des éléments du tableau retourné par la  
                                              fonction trigo */  
echo "sin($r)=$a, cos($r)=$b, tan($r)=$c";    // affichage des variables
```

- Cet exemple affichera ceci :
- sin(12)=-0,5365729180, cos(12)=0,8438539587, tan(12)=-0,6358599286**

### ***Les fonctions dynamiques***

- Il est possible de créer dynamiquement des fonctions. Pour les déclarer, on affecte à une variable chaîne de caractères le nom de la fonction à dupliquer. Puis on passe en argument à cette variable les paramètres normaux de la fonction de départ.
- *Exemple :*  

```
function divers($toto) {  
    echo $toto;  
}  
$mafonction = "divers";  
$mafonction("bonjour !"); // affiche 'bonjour !' par appel de divers()
```

### *Les fonctions dynamiques*

- Appel dynamique de fonctions
  - Exécuter une fonction dont le nom n'est pas forcément connu à l'avance par le programmeur du script
  - L'appel dynamique d'une fonction s'effectue en suivant le nom d'une variable contenant le nom de la fonction par des parenthèses

```
<?php $datejour = getdate() ; // date actuelle
//récupération des heures et minutes actuelles
$heure = $datejour[hours] ;          $minute=$datejour[minutes] ;
function bonjour(){                  global $heure;                global $minute;
echo "<b> BONJOUR A VOUS IL EST : ", $heure, " H ", $minute, "</b> <br />" ;}
function bonsoir (){
global $heure ;                      global $minute ;
echo "<b> BONSOIR A VOUS IL EST : ", $heure, " H ", $minute , "</ b> <br />" ;}
if ($heure <= 17) {$salut = "bonjour" ;}          else $salut="bonsoir" ;
//appel dynamique de la fonction
$salut() ;    ?>
```

### *Les fonctions dynamiques*

- Quelques fonctions permettant de travailler sur des fonctions utilisateur : **call\_user\_func\_array**, **call\_user\_func**, **create\_function**, **func\_get\_arg**, **func\_get\_args**, **func\_num\_args**, **function\_exists**, **get\_defined\_functions**, **register\_shutdown\_function**.
- **call\_user\_func\_array(\$str [, \$tab])** : Appelle une fonction utilisateur **\$str** avec les paramètres rassemblés dans un tableau **\$tab**.
- *Exemple :*

```
function essai($user, $pass) {           // fonction à appeler
    echo "USER: $user PASSWORD: $pass";
}
$params = array("hugo","0478");         // création du tableau de paramètres
call_user_func_array('essai', $params); // appel de la fonction
```
- Le nom de la fonction à appeler doit être dans une chaîne de caractères.



### *Les fonctions dynamiques*

- **create\_function(\$params,\$code)** : Crée une fonction anonyme (style lambda). Prend en argument la liste **\$params** des arguments de la fonction à créer ainsi que le code **\$code** de la fonction sous la forme de chaînes de caractères. Il est conseillé d'utiliser les simples quotes afin de protéger les noms de variables **'\$var'**, ou alors d'échapper ces noms de variables **\\$var**. Le nom de la fonction créée sera de la forme : *lambda\_x* où x est l'ordre de création de la fonction.
- *Exemple :*  

```
$newfunc = create_function('$a,$b','return \$a+\$b;');  
echo "Nouvelle fonction anonyme : $newfunc <br />";  
echo $newfunc(5,12)."<br />";
```
- On fabrique ici une fonction qui prend en argument 2 paramètres et renvoie leur somme. On l'appelle avec les nombres **5** et **12**. On va donc afficher le nombre **17**.
- Cet exemple est équivalent à : **function lambda\_1(\$a,\$b) { return \$a+\$b; }**

### **Formulaire HTML**

- Retourne des informations saisies par un utilisateur vers une application serveur
- La création d'un formulaire nécessite la connaissance de quelques balises HTML indispensables :
  - Structure : un formulaire commence toujours par la balise `<form>` et se termine par la balise `</form>`
  - Champ de saisie de text en ligne :  
`<input type = "text" name ="nom_du_champ" value="chaîne">`
  - Boutons d'envoi et d'effacement :  
`<input type=" submit " value = "Envoyer">`  
`<input type = "reset" name ="efface" value = "Effacer">`
  - Case à cocher et bouton radio :  
`<input type = "checkbox" name ="case1" value="valeur_case">`  
`<input type = "radio" name ="radio1" value ="valeur_radio">`

### ***Formulaire HTML***

- Liste de sélection avec options à choix unique :

```
<select name ="select" size="1">  
<option value = "un"> choix </option>  
<option value ="deux"> choix2 </option>  
</select>
```

- Liste de sélection avec options à choix multiples :

```
<select name ="select" size = "1" multiple>  
<option value = "un"> choix1 </option>  
<option value = "deux"> choix2 </option>  
</select>
```

### ***PHP et les formulaires***

- Méthodes d'envoi GET et POST
  - transmission selon une des deux méthodes d'envoi GET ou POST
  - La méthode GET place les informations d'un formulaire directement à la suite de l'adresse URL de la page appelée.
    - <http://www.site.com/cible.php?champ=valeur&champ2=valeur>
    - inconvénients :
      - rendre visibles les données dans la barre d'adresse du navigateur.
      - De plus, la longueur totale est limitée à 255 caractères, ce qui rend impossible la transmission d'un volume de données important
  - La méthode POST regroupe les informations dans l'entête d'une requête HTTP
    - » Assure une confidentialité efficace des données

### ***PHP et les formulaires***

- Récupération des paramètres en PHP
  - Si la directive `register_globals` de `php.ini` est `TRUE`, lors de la soumission, chaque élément de saisie est assimilé à une variable PHP dont le nom est constitué par la valeur de l'attribut `name` et son contenu par la valeur de l'attribut `value`  
A DECONSEILLER
  - **Les tableaux associatifs `$_GET` et `$_POST`** contiennent toutes les variables envoyées par un formulaire

### ***PHP et les formulaires***

```
<form method="GET | POST" action="page_cible.php">
<input type="text" name="Champ_saisie" value="Texte" >
<select name="Liste_Choix" size="3">
<option value="Option_1">Option_1</option>
<option value="Option_2">Option_2</option>
<option value="Option_3">Option_3</option>
</select>
<textarea name="Zone_Texte" cols="30" rows="5"> Texte par défaut </textarea>
  <input type="checkbox" name="Case_Cocher[]" value="Case_1"> Case 1<br>
  <input type="checkbox" name="Case_Cocher[]" value="Case_2"> Case 2<br>
  <input type="checkbox" name="Case_Cocher[]" value="Case_3"> Case 3<br>
  <input type="radio" name="Case_Radio" value="Case radio 1"> radio 1<br>
  <input type="radio" name="Case_Radio" value="Case radio 2"> radio 2<br>
  <input type="radio" name="Case_Radio" value="Case radio 3"> radio 3<br>
  <input type="reset" name="Annulation" value="Annuler">
  <input type="submit" name="Soumission" value="Soumettre">
</form>
```

### ***PHP et les formulaires***

```
<?php
    $resultat = $_GET["Champ_saisie"] . "<br>";
    $resultat .= $_GET["Liste_Choix"] . "<br>";
    $resultat .= $_GET["Zone_Texte"] . "<br>";
    for ($i = 0; $i < count($_GET["Case_Cocher"]); $i++)
    {
        $resultat .= $_GET["Case_Cocher"][$i] . "<br>";
    }
    $resultat .= $_GET["Case_Radio"] . "<br>";
    echo $resultat;
?>
```

page\_cible.php

### ***PHP et les formulaires***

- PHP et les formulaires
  - La plupart des éléments d'un formulaire n'acceptent qu'une seule et unique valeur, laquelle est affectée à la variable correspondante dans le script de traitement.

`$Champ_Saisie ← "Ceci est une chaîne de caractères.";`

- Pour des cases à cocher et les listes à choix multiples, plusieurs valeurs peuvent être sélectionnées entraînant l'affectation d'un tableau de valeurs aux variables correspondantes.

`$Case_Cocher[0] ← "Case radio 1";`

`$Case_Cocher[1] ← "Case radio 3";`



### ***Les cookies***

- Principe
  - Un cookie est un fichier texte créé par un script et stocké sur l'ordinateur des visiteurs d'un site
  - Les cookies permettent de conserver des renseignements utiles sur chaque utilisateur, et de les réutiliser lors de sa prochaine visite
    - Exemple : personnaliser la page d'accueil ou les autres pages du site
      - un message personnel comportant par exemple son nom, la date de sa dernière visite, ou tout autre particularité.
  - Les cookies étant stockés sur le poste client, l'identification est immédiate et ne concernent que les renseignements qui le concernent
  - Pour des raisons de sécurité, les cookies ne peuvent être lus que par des pages issues du serveur qui les a créés

### ***Les cookies***

- Principe
  - Le nombre de cookies qui peuvent être définis sur le même poste client est limité à 20 et la taille de chacun est limitée à 4ko.
  - Un navigateur peut stocker un maximum de 300 cookies
  - La date d'expiration des cookies est définie de manière explicite par le serveur web chargé de les mettre en place.
  - Les cookies disponibles sont importés par PHP sous forme de variables identifiées sous les noms utilisés par ces cookies
  - La variable globale du serveur `$_COOKIES` enregistre tous les cookies qui ont été définis

### Les cookies

- Écrire des cookies
  - L'écriture de cookies est possible grâce à la fonction `setcookie()`
    - il faut cette fonction dès le début du script avant l'envoi d'aucune autre information de la part du serveur vers le poste client.

```
Setcookie("nom_var", "valeur_var", date_expiration, "chemin",  
         "domain", "secure")
```

```
<?php  
setcookie ("PremierCookie", "Salut", time() +3600*24*7) ;  
...  
if (!$PremierCookie) {  
    echo "le cookie n'a pas été défini";  
} else {  
    echo $premierCookie, "<br>";  
}  
?>
```

### ***Les cookies***

- Écriture de cookies
  - **Chemin** : définit la destination (incluant les sous-répertoire) à laquelle le navigateur doit envoyer le cookie.
  - **Domain set** : le nom du domaine à partir duquel peuvent être lus les cookies.
    - On peut aussi utiliser la variable d'environnement `$SERVER_NAME` à la place.
  - **Secure** : un nombre qui vaut 0 si la connexion n'est pas sécurisée, sinon, il vaut 1 pour une connexion sécurisée

### Les cookies

- Lecture de cookies
  - Les cookies sont accessibles dans le tableau associatif `$_COOKIE`
    - Si celui-ci visite une des pages PHP de ce même domaine dont le chemin est inclut dans le paramètre chemin (si le chemin est / le cookie est valide pour toutes les pages de ce site).
    - Il faut d'abord vérifier l'existence des variables dont les noms et les valeurs ont été définis lors de la création du cookie.
      - Cette vérification s'effectue grâce à la fonction `isset($_COOKIE["nom_var"])` qui renvoie true si la variable `$nom_var` existe et false sinon.

```
<?php
if (isset($_COOKIE["nom_var"])){
echo "<h2> Bonjour". $_COOKIE["nom_var"]. "</h2>" ;}
else echo "pas de cookie" ;
?>
```

### ***Les cookies***

- **Écriture de cookies**
  - **Nom\_var** : nom de la variable qui va stocker l'information sur le poste client et qui sera utilisée pour récupérer cette information dans la page qui lira le cookie.
    - C'est la seule indication obligatoire pour un cookie
  - **Valeur\_var** : valeur stockée dans la variable. Par exemple une chaîne de caractères ou une variable chaîne, en provenance d'un formulaire
  - **Date\_expiration** : la date à laquelle le cookie ne sera plus lisible et sera effacé du poste client
    - on utilise en général la date du jour, définie avec la fonction `time()` à laquelle on ajoute la durée de validité désirée
    - Si l'attribut n'est pas spécifié, le cookie expire à l'issue de la session

### ***Les cookies***

- Écriture de plusieurs variables par un cookie
  - Utilisation de la fonction **compact()** pour transformer les variables en un tableau
  - Convertir le tableau en une chaîne de caractère à l'aide de la fonction **implode()**
  - Affecter la chaîne créée à l'attribut « nom\_cookie »

```
<?php
$col="#FF0000" ;
$size=24;
$font="Arial" ;
$text="Je suis le Cookie" ;
$arr=compact("col" , " size" , " font" , "text" );
$val=implode("&" , $arr);
Setcookie("la_cookie" , $val, time()+600);?>
```

### ***Les cookies***

- Lecture de plusieurs variables d'un cookie
  - Décomposé la chaîne stockée par la cookie et retrouver un tableau en utilisant la fonction `explode()`

```
<?php
echo " <b> voici le contenu de la chaîne cookie : </b><br>";
$arr=explode("&", $la_cookie);
echo "<b> ces variables ont été établies à partir de la chaîne
    cookie : </b> <br> <br>";
foreach ($arr as $k=>$elem) {
echo "$k=>$elem <br>";
}
?>
```



### ***Les cookies***

- Supprimer un cookie
  - Il suffit de renvoyer le cookie grâce à la fonction `setcookie()` en spécifiant simplement l'argument `NomDuCookie`

```
<?php  
setcookie("Visites");  
?>
```

- Une autre méthode consiste à envoyer un cookie dont la date d'expiration est passée:

```
<?php  
setcookie("Visites","",time()-1 )  
?>
```

### ***Les cookies***

- Exemples d'utilisation de cookies

//Un script permettant de savoir si un visiteur est déjà venu sur le site pendant le mois

```
setcookie("Visites","Oui",time()+2592000,"/", ".mondomaine.fr",0);
```

// Envoi d'un cookie qui restera présent 24 heures

```
SetCookie("DejaVisite","1",time()+3600*24,"/",". mondomaine.fr ",0);
```

// Envoi d'un cookie qui s'effacera le 1er janvier 2018

```
SetCookie("An2018","1",mktime(0,0,0,1,1,2018),"/",". mondomaine.fr ",0);
```

//script permettant de compter le nombre de visite de la page par le visiteur

```
<?php
```

```
$Visites++;
```

```
setcookie("Visites",$Visites,time()+2592000,"/",". mondomaine.fr ",0);?>
```

### *Les sessions*

- Les sessions sont un moyen de sauvegarder et de modifier des variables tout au cours de la visite d'un internaute sans qu'elles ne soient visibles dans l'URL et quelque soient leurs types (tableau, objet...).
- Cette méthode permet de sécuriser un site, d'espionner le visiteur, de sauvegarder son panier (e-commerce), etc.
- Les informations de sessions sont conservées en local sur le serveur tandis qu'un identifiant de session est posté sous la forme d'un cookie chez le client (ou via l'URL si le client refuse les cookies).
- Quelques fonctions :
- **session\_start()** : démarre une session
- **session\_destroy()** : détruit les données de session et ferme la session
- **session\_register("var")** : enregistre la variable **\$var** dans la session en cours, attention, ne pas mettre le signe **\$** (dollars) devant le nom de variable
- **session\_unregister("var")** : détruit la variable **\$var** de la session en cours

### Les Sessions

- Une session doit obligatoirement démarrer avant l'envoi de toute information chez le client : donc pas d'affichage et pas d'envoi de header. On peut par exemple avoir une variable globale contenant le code HTML de la page et l'afficher à la fin du script.
- Les variables de session sont accessibles comme des variables globales du script.

- *Exemple :*

```
<?
$out = "<html><body>";
session_start(); // démarrage de la session
if(! isset($client_ip)) {
    $client_ip = $REMOTE_ADDR;
    session_register('client_ip'); // enregistrement d'une variable
}
/* ... + code de la page */
$out .= "</body></html>";
echo $out;
?>
```

### ***Les sessions***

- Sauvegarder des variables de type objet dans une session est la méthode de sécurisation maximum des données : elles n'apparaîtront pas dans l'URL et ne pourront pas être forcées par un passage manuel d'arguments au script dans la barre d'adresse du navigateur.
- Les données de session étant sauvegardées sur le serveur, l'accès aux pages n'est pas ralenti même si des données volumineuses sont stockées.
- Une session est automatiquement fermée si aucune requête n'a été envoyée au serveur par le client durant un certain temps (**2 heures par défaut dans les fichiers de configuration Apache**).
- Une session est un moyen simple de suivre un internaute de page en page (sans qu'il s'en rende compte). On peut ainsi **sauvegarder son parcours**, établir son profil et établir des **statistiques** précises sur la fréquentation du site, la visibilité de certaines pages, l'efficacité du système de navigation...

### ***Les sessions***

- **Côté client**
  - Un *cookie* ou une variable contenant un numéro vous identifiant : c'est **identificateur** de session.
- **Côté serveur**
  - Un stockage contient les informations sur les **variables de session**.
  - Par défaut, Le nom du fichier correspond à la valeur de l'identificateur de session assigné.
  - En fait PHP4 utilise des fichiers pour enregistrer les variables de session mais il est possible d'utiliser une base de données ou une mémoire partagée pour le même résultat.

### *Les sessions*

```
<?php
session_start();
?>
<!DOCTYPE html>

<html>
```

```
<body>
```

```
<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>
```

```
<div>
  <?php
    echo "Favorite color is " . $_SESSION["favcolor"] . "<br>";
    echo "Favorite animal is " . $_SESSION["favanimal"] . ".";
  ?>
</div>
```

```
<div>
  <?php
    print_r($_SESSION);
  ?>

</div>
```

```
</body>
</html>
```

### ***Les sessions***

```
<?php
    session_start();
    $_SESSION['prenom'] = "Matthieu";
    $_SESSION['age'] = "inconnu";

    echo "<ul>\n";
    foreach($_SESSION as $cle => $valeur)
    {
        echo "  <li><strong>".ucfirst($cle)." : </strong><em>".$valeur."</em></li>\n";
    }
    echo "</ul>\n";
?>
```



### Les sessions

```
<?php
```

```
session_start();
$_SESSION['login'] = "";
$_SESSION['password'] = "";
```

```
if (isset($_POST['submit']))
{
```

```
    // bouton submit pressé, je traite le formulaire
    $login = (isset($_POST['login'])) ? $_POST['login'] : "";
    $pass = (isset($_POST['pass'])) ? $_POST['pass'] : "";
```

```
    if (($login == "Matthieu") && ($pass == "NewsletTux"))
    {
```

```
        $_SESSION['login'] = "Matthieu";
        $_SESSION['password'] = "NewsletTux";
        echo '<a href="accueil.php" title="Accueil de la section membre">Accueil</a>';
```

```
    }
    else
    {
```

```
        // une erreur de saisie ...?
        echo '<p style="color:#FF0000; font-weight:bold;">Erreur de connexion.</p>';
```

```
    }
```

```
}; // fin if (isset($_POST['submit']))
```

```
if (!isset($_POST['submit']))
{
```

```
    // Bouton submit non pressé j'affiche le formulaire
    echo '
```

```
<form id="conn" method="post" action="">
```

```
<p><label for="login">Login :</label><input type="text" id="login" name="login" /></p>
```

```
<p><label for="pass">Mot de Passe :</label><input type="password" id="pass" name="pass" /></p>
```

```
<p><input type="submit" id="submit" name="submit" value="Connexion" /></p>
```

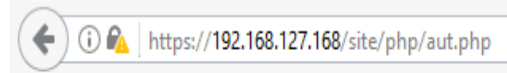
```
</form>;
```

```
}; // fin if (!isset($_POST['submit']))
```

```
?>
```

Login :

Mot de Passe :



[Accueil](#)

### ***Base de données***

- **Principe**

- PHP propose de nombreux outils permettant de travailler avec la plupart des SGBDR
  - Oracle, Sybase, Microsoft SQL Server, PostgreSQL ou encore MySQL
- Lorsqu'une base de données n'est pas directement supportée par PHP, il est possible d'utiliser un driver **ODBC** (pilote standard) pour communiquer avec les bases de données
- PHP fournit un grand choix de fonctions permettant de manipuler les bases de données.
  - Quatre fonctions sont essentielles:
    - **La fonction de connexion au serveur**
    - **La fonction de choix de la base de données**
    - **La fonction de requête**
    - **La fonction de déconnexion**

### *Les fonctions de base*

- Principe
  - Avec le SGBD MySQL, les fonctions de manipulation sont :

**`$mysqli = new mysqli($host, $user, $password, $db_name)`**

Cet appel crée un nouvel objet mysqli qui permet d'établir une connexion avec le SGBD MySQL et de sélectionner la base de données avec laquelle vous allez travailler.

En cas d'erreur lors de la connexion (mauvaise adresse, mauvais login ou mot de passe, problème de réseau...), l'attribut **`$connect_errno`** sera positionné à TRUE

### *Connexion à la base*

- En général, la première chose que vous faites avec MySQL, c'est de vous connecter au serveur et de sélectionner une base de données
  - La base de données ici est **world**

#### Notation Orientée objet:

```
<?php
$mysqli = new mysqli("localhost", "user", "password", "world");
if ($mysqli->connect_errno) {
    echo "Echec connexion à MySQL : (" . $mysqli->connect_errno . ") " . $mysqli->connect_error;
}
echo $mysqli->host_info . "\n";

?>
```

### *Envoi de la requête*

- Par exemple, une requête pour créer une table ou un select

**`$result = $mysqli->query($query)`**

Cet appel permet de faire une requête sur la base de donnée (qui a précédemment été sélectionnée lors de la création de l'objet \$mysqli). Le paramètre est une chaîne de caractère décrivant la requête.

En cas d'erreur lors de la requête (p. ex. erreur de syntaxe dans la requête SQL...), la valeur retournée sera FALSE. En cas de succès, la valeur retournée décrira le résultat de la requête (les données dans le cas d'un SELECT, et TRUE dans le cas d'un INSERT, UPDATE, DELETE ou DROP) et sera un objet de la classe mysqli\_result.

**`$result->num_rows`**

La valeur de cet attribut sera le nombre de réponses du résultat lors d'une requête SELECT.

### *Récupérer les résultats*

```
$row = $mysqli_result->fetch_assoc()
```

Cet appel permet de décomposer dans un tableau la première ligne du résultat d'une requête SELECT.

Par exemple, **\$row['field']** permettra d'accéder au champ **field** de la première ligne de résultat.

En renouvelant cet appel, c'est ensuite la valeur de la deuxième ligne qui sera retournée, et ainsi de suite.

La valeur retournée sera FALSE lorsqu'il n'y aura plus de nouvelle ligne de résultat à traiter.

### *Libérer la connexion*

#### `$mysqli_result->free()`

Cet appel permet, lorsqu'on a fini de traiter le résultat, de libérer la mémoire allouée par `$mysqli->query()` dans le cas d'une requête SELECT.

#### `$mysqli->close()`

Quand la connexion au SGBD n'est plus utile (ou avant la fin du programme), cet appel permet de rompre la connexion qui avait été établie au préalable.

### Select

```
$mysqli = new mysqli('192.20.27.12', 'root', 'HaLo1Bt', 'db_forum');

if ($mysqli->connect_errno) {
    die('<p>Connexion impossible : '.$mysqli->connect_error.'</p>');
}
$result = $mysqli->query('SELECT nom, prenom FROM users') ;

if (!$result) {
    die('<p>ERREUR Requête invalide : '.$mysqli->error.'</p>');
}
print '<table border="1">';
for ($i=0 ; $i < $result->num_rows ; $i++) {
    $row = $result->fetch_assoc() ;
    print '<tr>';
    print '<td>'.$row["nom"].'</td>';
    print '<td>'. $row["prenom"].'</td>';
    print '</tr>';
}
print '</table>';
$result->free() ;
$mysqli->close() ;
```



### *Insert*

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

### Count

```
<?php
//Open a new connection to the MySQL server
$mysqli = new mysqli('host','username','password','database_name');

//Output any connection error
if ($mysqli->connect_error) {
    die('Error : ('. $mysqli->connect_errno .') '. $mysqli->connect_error);
}

//get total number of records
$results = $mysqli->query("SELECT COUNT(*) FROM users");
$get_total_rows = $results->fetch_row(); //hold total records in variable

$mysqli->close();
?>
```

### *Prepared Statement (requêtes précompilées)*

```
$search_ID = 1;
$search_product = "PD1001";

$query = "SELECT id, product_code, product_desc, price FROM products WHERE ID=?
AND product_code=? ";
$stmt = $mysqli->prepare($query);
$stmt->bind_param('is', $search_ID, $search_product);
$stmt->execute();
$stmt->bind_result($id, $product_code, $product_desc, $price);

print '<table border="1">';
while($stmt->fetch()) {
    print '<tr>';
    print '<td>'.$id.'</td>';
    print '<td>'.$product_code.'</td>';
    print '<td>'.$product_desc.'</td>';
    print '<td>'.$price.'</td>';
    print '</tr>';
}
print '</table>';

//close connection
$stmt->close();
```

### ***Prepared Statement (requêtes précompilées)***

```
//values to be inserted in database table
```

```
$product_code = 'P1234';
```

```
$product_name = '42 inch TV';
```

```
$product_price = '600';
```

```
$query = "INSERT INTO products (product_code, product_name, price) VALUES(?, ?, ?)";
```

```
$statement = $mysqli->prepare($query);
```

```
//bind parameters for markers, where (s = string, i = integer, d = double, b = blob)
```

```
$statement->bind_param('sss', $product_code, $product_name, $product_price);
```

```
if( $statement->execute() ) {
```

```
    print 'Success! ID of last inserted record is : ' . $statement->insert_id . '<br />';
```

```
}else{
```

```
    die('Error : ('. $mysqli->errno .') '. $mysqli->error);
```

```
}
```

```
$statement->close();
```

### *Update/Delete*

**//MySqli Update Query**

```
$results = $mysqli->query("UPDATE products SET product_name='52 inch TV',  
product_code='323343' WHERE ID=24");
```

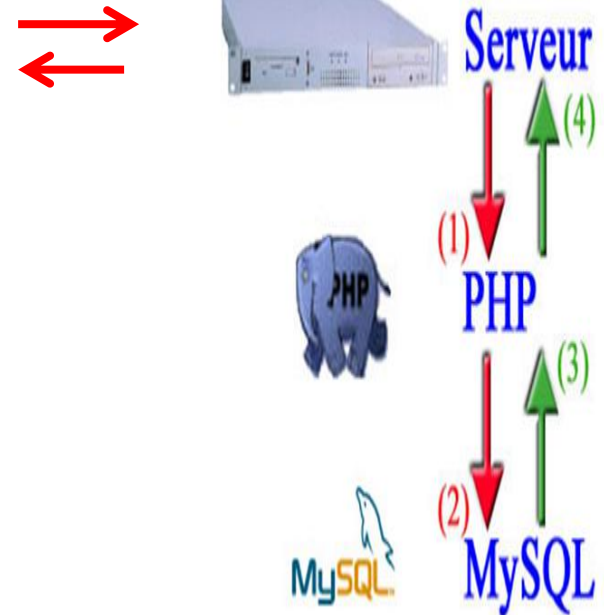
**//MySqli Delete Query**

```
//$results = $mysqli->query("DELETE FROM products WHERE ID=24");
```


```
if($results){  
    print 'Success! record updated / deleted';  
}else{  
    print 'Error : ('. $mysqli->errno .') '. $mysqli->error;  
}
```

### PhpMyAdmin

- Interface Web pour MySQL écrite en PHP
- Projet open source :  
<http://www.phpmyadmin.net>
- Fonctionnalités :
  - Affichage / création / modification de tables
  - Affichage / insertion / suppression des contenus
  - Recherches / Requêtes
  - Importation / exportation de tables / base / données
  - Optimisation de tables / base
  - Moteur MyISAM (→5.5) :
    - relationnel, transactionnel, plain text, très rapide
  - Moteur INNODB (5.5→) :
    - relationnel, transactionnel, ACID, moins rapide



### PhpMyAdmin



Bienvenue sur phpMyAdmin

Langue - Language

Français - French ▼

Connexion ⓘ

Utilisateur :

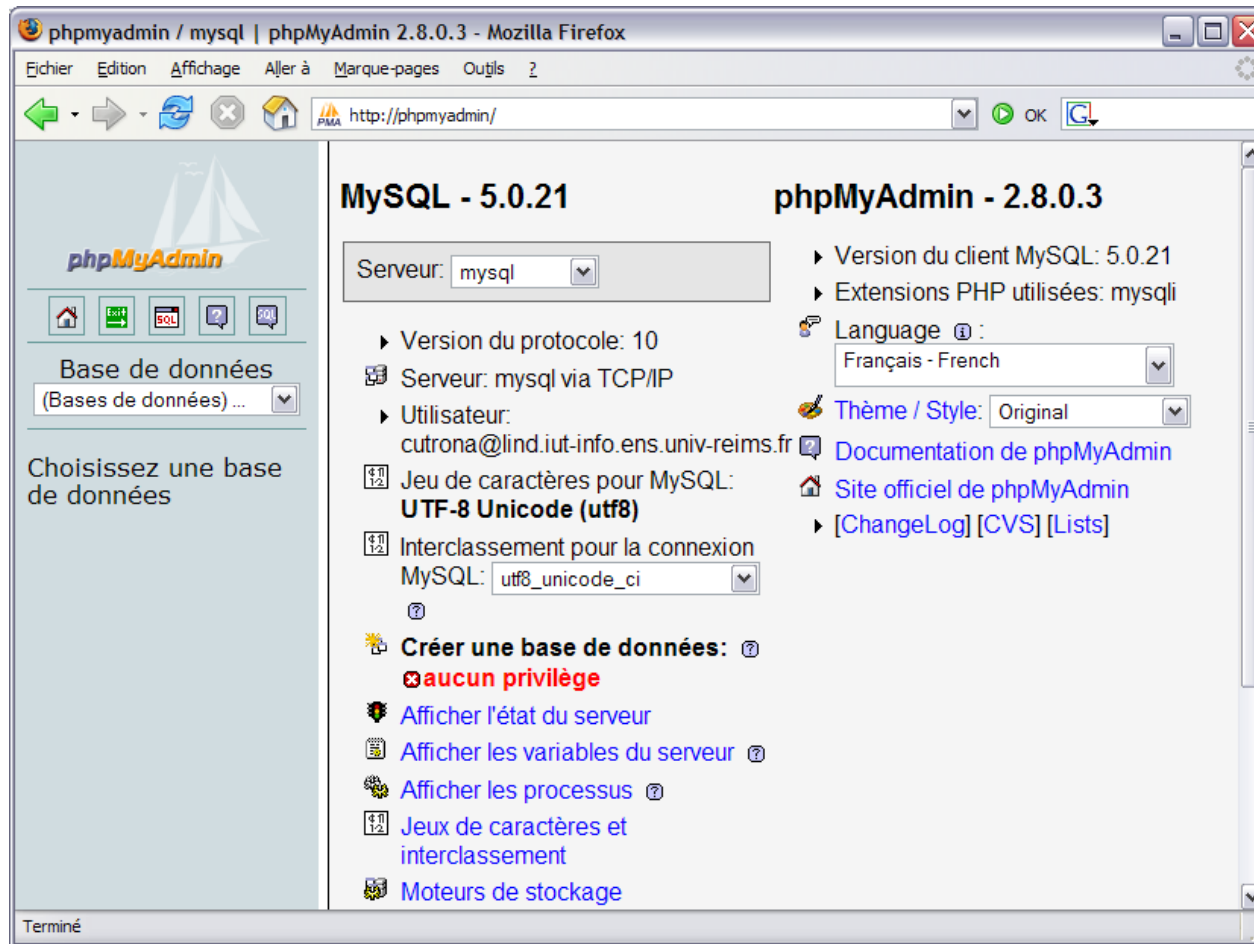
Mot de passe :

Choix du serveur: mysql ▼

Exécuter

⚠ Vous devez accepter les cookies pour poursuivre.

### PhpMyAdmin





### PhpMyAdmin

The screenshot displays the PhpMyAdmin interface for the 'mysql' database. The top navigation bar includes links for 'Bases de données', 'SQL', 'État', 'Processus', 'Exporter', 'Importer', 'Variables', and a 'plus' dropdown. The left sidebar contains a home icon, a database selection dropdown (currently showing '(Bases de données) ...'), and the text 'Choisissez une base de données'. The main content area is divided into three sections: 'Paramètres généraux', 'Paramètres d'affichage', and 'MySQL'. The 'Paramètres généraux' section shows 'Serveur actuel' set to 'mysql', a link to 'Modifier le mot de passe', and 'Interclassement pour la connexion MySQL' set to 'utf8\_general\_ci'. The 'Paramètres d'affichage' section shows 'Langue - Language' set to 'Français - French', 'Thème / Style' set to 'pmahomme', and 'Taille du texte' set to '82%'. The 'MySQL' section lists server details: 'Serveur: mysql via TCP/IP', 'Version du serveur: 5.1.56', 'Version du protocole: 10', 'Utilisateur: cutrona@lind.iut-info.ens.univ-reims.fr', and 'Jeu de caractères pour MySQL: UTF-8 Unicode (utf8)'. The 'Serveur web' section lists: 'Apache/2.2.3 (CentOS)', 'Version du client MySQL: 5.1.54', and 'Extension PHP: mysql'. The 'phpMyAdmin' section lists: 'Version: 3.4.3.1, dernière version stable : 3.4.10.1', 'Documentation', 'Wiki', 'Site officiel', 'Contribuer', 'Obtenir de l'aide', and 'Liste des changements'.

mysql

Bases de données SQL État Processus Exporter Importer Variables plus

(Bases de données) ...

Choisissez une base de données

**Paramètres généraux**

Serveur actuel: mysql

Modifier le mot de passe

Interclassement pour la connexion MySQL : utf8\_general\_ci

**Paramètres d'affichage**

Langue - Language : Français - French

Thème / Style: pmahomme

Taille du texte: 82%

Plus de paramètres

**MySQL**

- Serveur: mysql via TCP/IP
- Version du serveur: 5.1.56
- Version du protocole: 10
- Utilisateur: cutrona@lind.iut-info.ens.univ-reims.fr
- Jeu de caractères pour MySQL: UTF-8 Unicode (utf8)

**Serveur web**

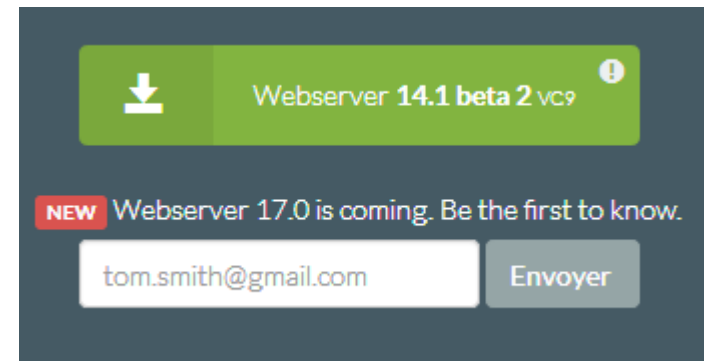
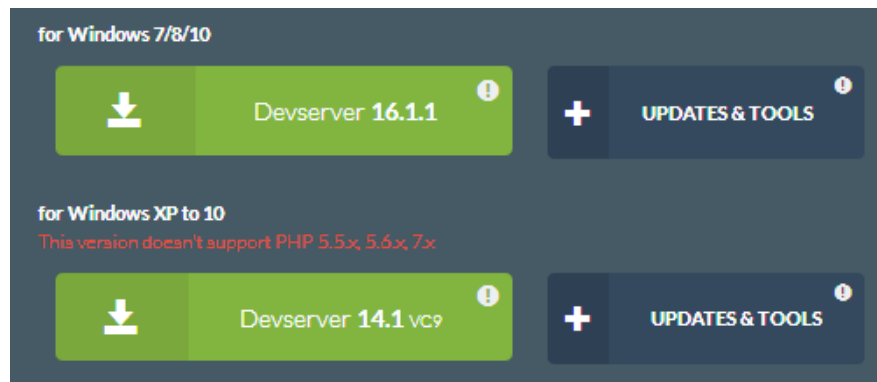
- Apache/2.2.3 (CentOS)
- Version du client MySQL: 5.1.54
- Extension PHP: mysql

**phpMyAdmin**

- Version: 3.4.3.1, dernière version stable : 3.4.10.1
- Documentation
- Wiki
- Site officiel
- Contribuer
- Obtenir de l'aide
- Liste des changements

### EasyPHP

- EasyPHP est une plateforme de développement web qui permet de travailler dans un environnement serveur composé d'un serveur web Apache, d'un serveur de base de données MySQL. De plus, l'outil se compose également d'un interpréteur de script (PHP) et d'une administration SQL PHPMyAdmin.
- Deux versions
  - Dev
  - Host



### XAMPP

- Cross-Platform (X), Apache (A), MariaDB,MySQL (M), PHP (P) and Perl (P)

#### Extensions XAMPP



#### Applications

Installez vos applications favorites par-dessus XAMPP. Bitnami fournit un outil tout-en-un gratuit pour installer Drupal, Joomla!, WordPress et beaucoup d'autres applications populaires libres, par-dessus XAMPP.



WordPress  
Blog



Joomla!  
CMS



CMS Made Simple  
CMS



Drupal  
CMS



MediaWiki  
Wiki



PrestaShop  
e-Commerce



Moodle  
eLearning



ownCloud  
Media sharing

# ZMWS

## ZazouMiniWebServer : un serveur AMP (Apache - PHP - MySQL) pour Windows

Par Jean-Christophe Becquet :: [Logiciels](#) :: [#205](#)



[ZazouMiniWebServer](#) permet de lancer un serveur AMP (Apache - PHP - MySQL) sous Windows. Il ne nécessite pas d'installation et peut même fonctionner sur un support en lecture seule. Copié sur une clef USB il permet de disposer très simplement d'un serveur portable.



Article [Discussion](#) [Lire](#) [Modifier](#) [Modifier le code](#) [t](#)

 Non connecté [Di](#)

## Zazouminiwebserver



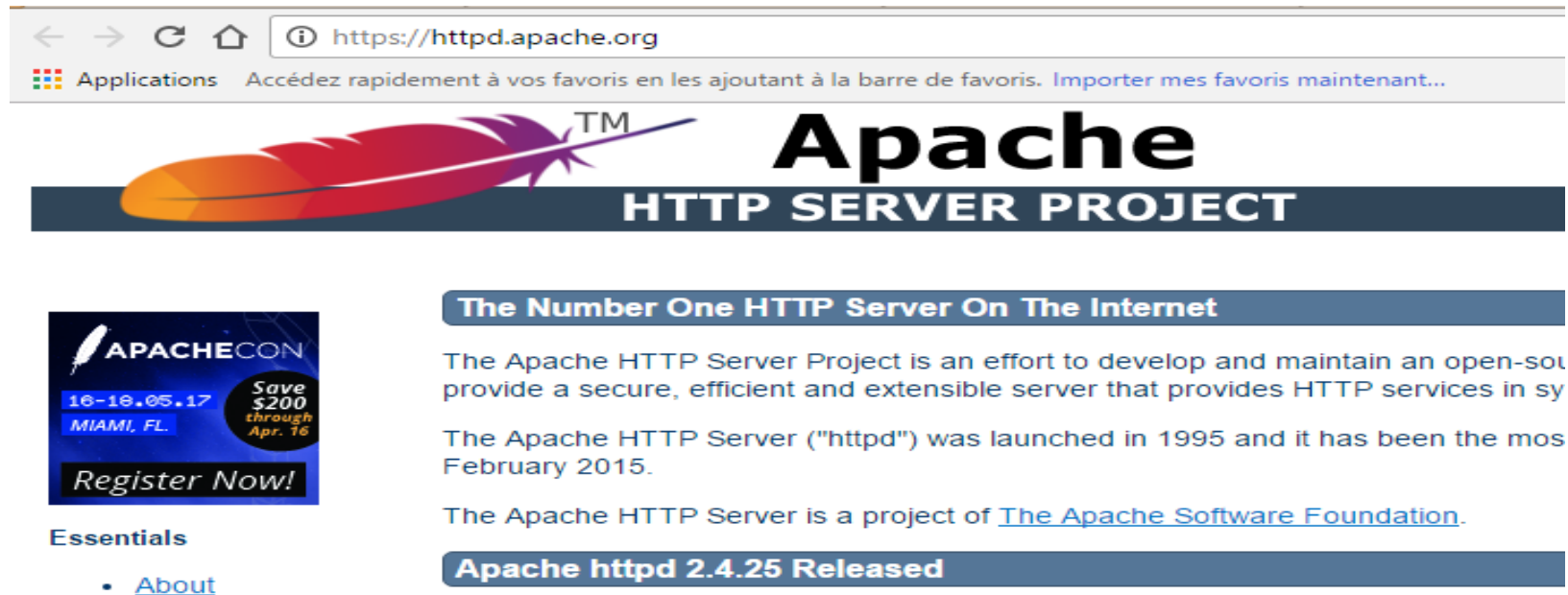
Cet article est **orphelin**. Moins de trois articles **lui sont liés**

Aidez à ajouter des liens en plaçant le code  [\[\[Zazouminiwebserver: sujet](#)

**ZazouMiniWebServeur** (en abrégé **ZMWS**) est un [serveur web](#) tournant uniquement sous l'environnement [Microsoft Windows](#). Il a la particularité d'être extrêmement léger (approximativement 500 kilooctets) et d'être fourni en tant qu'[exécutable](#) ou de [DLL](#), et peut s'installer en tant que [daemon](#). Son auteur, Xavier Garreau, a placé ce [logiciel](#) sous [licence libre GPL](#) <sup>1</sup>.


### Site web du projet


- Le projet open-source
- **Apache** est le plus populaire des serveurs HTTP. Il est produit par la « Apache Software Foundation ». C'est un logiciel libre fourni sous la licence spécifique Apache



The screenshot shows the Apache HTTP Server Project website. At the top, there's a navigation bar with a search icon and the URL <https://httpd.apache.org>. Below the navigation bar, there's a large banner featuring the Apache logo (a red and orange feather) and the text "Apache HTTP SERVER PROJECT". To the left of the banner, there's a sidebar with a section titled "Essentials" containing a link to "About". The main content area has a section titled "The Number One HTTP Server On The Internet" with a sub-header "The Apache HTTP Server Project is an effort to develop and maintain an open-source provide a secure, efficient and extensible server that provides HTTP services in sy". Below this, there's a paragraph stating "The Apache HTTP Server ('httpd') was launched in 1995 and it has been the mos February 2015." and another paragraph stating "The Apache HTTP Server is a project of [The Apache Software Foundation](#)." At the bottom, there's a section titled "Apache httpd 2.4.25 Released".

Applications Accédez rapidement à vos favoris en les ajoutant à la barre de favoris. [Importer mes favoris maintenant...](#)

 **Apache**  
HTTP SERVER PROJECT

 **APACHECON**  
16-18.05.17  
MIAMI, FL  
Save \$200 through Apr. 16  
Register Now!

Essentials

- [About](#)

**The Number One HTTP Server On The Internet**

The Apache HTTP Server Project is an effort to develop and maintain an open-source provide a secure, efficient and extensible server that provides HTTP services in sy

The Apache HTTP Server ("httpd") was launched in 1995 and it has been the mos February 2015.

The Apache HTTP Server is a project of [The Apache Software Foundation](#).

**Apache httpd 2.4.25 Released**

### ***Installation***

Installation de :

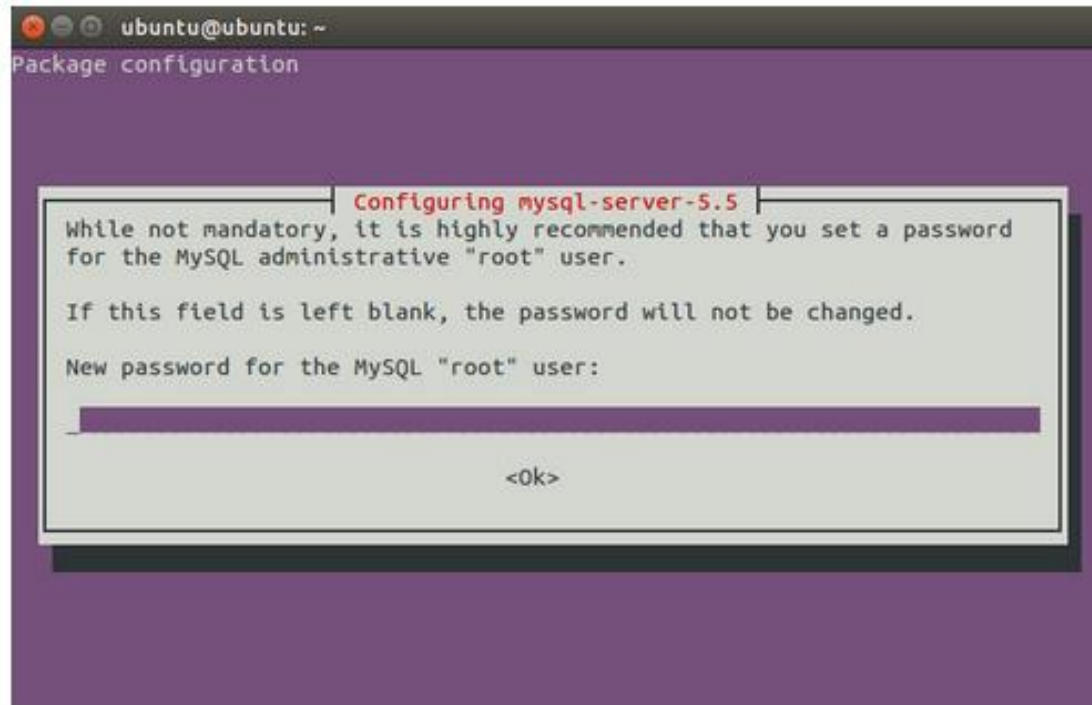
- Apache 2
  - Mysql Server 5.5
  - PHP 5.5.9
  - PhpMyAdmin
- 
- Installez les paquets nécessaires pour apache2 et php5 et mysql. Validez toutes les options par défaut.
  - Vous pouvez installer tous les paquets en une seule commande

### *Installation*

```
$ sudo apt-get update
$ sudo apt-get install php5
$ sudo apt-get install apache2
$ sudo apt-get install libapache2-mod-php5
$ sudo apt-get install mysql-server-5.5
$ sudo apt-get install php5-mysql
```

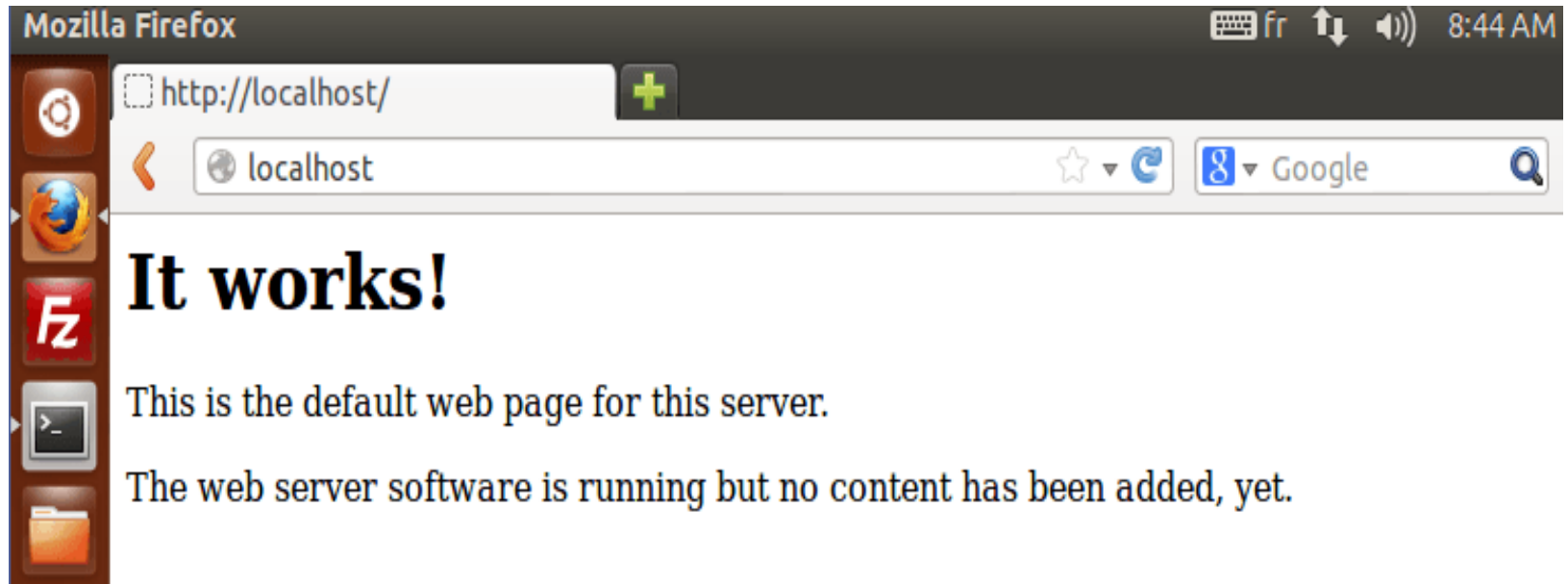
## *Installation*



- Pendant l'installation de MySQL :





### *It works*



- Pour relancer le démon apache2 (le serveur web):
  - **\$ service apache2 restart** 
  - **ou**
  - **\$ /etc/init.d/apache2 restart** 

### Vérification

- Apache 2 a très certainement été installé par défaut lors de l'installation de votre Debian. Pour le vérifier :

**dpkg -l | grep apache2**

```
root@lamp ~# dpkg -l | grep apache2
ii  apache2                2.4.10-10+deb8u8
ii  apache2-bin            2.4.10-10+deb8u8
s)
ii  apache2-data           2.4.10-10+deb8u8
ii  apache2-mpm-prefork    2.4.10-10+deb8u8
ii  apache2-utils          2.4.10-10+deb8u8
ers)
ii  libapache2-mod-perl2    2.0.9~1624218-2+deb8u1
ii  libapache2-mod-php5     5.6.30+dfsg-0+deb8u1
ache 2 module)
ii  libapache2-mod-python   3.3.1-11
root@lamp ~# █
```

### Configuration

- Tous les **fichiers de configuration** de Apache2 sont dans le dossier **/etc/apache2**.
- Rendez-vous dans le répertoire **/etc/apache2/**, et regardez les fichiers le composant :

```
root@lamp /etc/apache2# ls -l
total 80
-rw-r--r-- 1 root root 7115 Oct 24 2015 apache2.conf
drwxr-xr-x 2 root root 4096 Mar 31 20:55 conf-available
drwxr-xr-x 2 root root 4096 Apr 8 2016 conf-enabled
-rw-r--r-- 1 root root 1782 Oct 24 2015 envvars
-rw-r--r-- 1 root root 31063 Oct 24 2015 magic
drwxr-xr-x 2 root root 12288 Mar 31 21:02 mods-available
drwxr-xr-x 2 root root 4096 Mar 31 20:55 mods-enabled
-rw-r--r-- 1 root root 333 Apr 8 2016 ports.conf
drwxr-xr-x 2 root root 4096 Mar 31 20:55 sites-available
drwxr-xr-x 2 root root 4096 Apr 8 2016 sites-enabled
root@lamp /etc/apache2#
```

```
root@lamp /etc# tree apache2/ -d
apache2/
├── conf-available
├── conf-enabled
├── mods-available
├── mods-enabled
├── sites-available
└── sites-enabled

6 directories
root@lamp /etc#
```

### Configuration

- **envvars** est utilisé pour définir des variables d'environnement propres à Apache ;
- **ports.conf** contient la directive listen qui spécifie les adresses et les ports d'écoutes ;

```
root@lamp /etc/apache2# more ports.conf
# If you just change the port or add more port
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>
```

- **apache2.conf** est le fichier principal de configuration c'est à partir de lui que tous les autres fichiers sont chargés ;
- **mods-available** contient la liste des modules d'apache installés ;
- **mods-enabled** celle des modules activées ;
- **sites-available** contient la liste des **vhosts** installés ;
- **sites-enabled** celle des vhosts activées.

### *Répertoire /etc/apache2/mods-available*

On trouve tous les fichiers concernant les modules dans le répertoire /etc/apache2/mods-available/. On y trouve deux catégories de fichiers : \*.load et \*.conf

Les fichiers avec l'extension load charge effectivement les modules dynamiques :

```
cat /etc/apache2/mods-available/userdir.load
```

```
LoadModule userdir_module /usr/lib/apache2/modules/mod_userdir.so
```

Les fichiers avec l'extension conf sont les fichiers de configuration des modules :

```
cat /etc/apache2/mods-available/userdir.conf
```

```
<IfModule mod_userdir.c>
```

```
UserDir public_html
```

```
UserDir disabled root
```

```
<Directory /home/*/public_html>
```

```
AllowOverride FileInfo AuthConfig Limit
```

```
Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
```

```
</Directory>
```

```
</IfModule>
```

### *Répertoire `/etc/apache2/mods-enabled`*

- On trouve les fichiers concernant les modules activés dans le répertoire `/etc/apache2/mods-enabled/`
  - ce sont uniquement ces fichiers qui sont inclus dans le fichier de configuration principal par les directives :

**`Include /etc/apache2/mods-enabled/*.load`**

**`Include /etc/apache2/mods-enabled/*.conf`**

- Et ces fichiers sont en fait des liens qui pointent vers les fichiers de `/etc/apache2/mods-available`
- Pour activer un module (ce qui revient donc à créer le lien), il est pratique d'utiliser la commande suivante : `a2enmod mod_userdir`. Mais on peut bien évidemment créer le lien "à la main".

### *Répertoire /etc/apache2/sites-available*

- On trouve les fichiers de configuration des sites web activés dans le répertoire /etc/apache2/sites-enabled/ : ce sont uniquement ces fichiers qui sont inclus dans le fichier de configuration principal par la directive :

**Include /etc/apache2/sites-enabled/[^.#]\***

- Et ces fichiers sont en fait des liens qui pointent vers les fichiers de /etc/apache2/sites-available
- De même que pour les modules, pour activer un site, il existe une commande : **a2ensite fichier\_conf**
  - "fichier\_conf" étant un fichier de configuration présent dans le dossier **/etc/apache2/sites-available/**

### Répertoire `/etc/apache2/sites-available`

- Il est fréquent d'héberger plusieurs serveurs web sur un même poste aussi la déclaration et le paramétrage des différents serveurs est déportée dans des fichiers à placer dans `/etc/apache2/sites-available/`. Le fichier default y est déjà présent pour assurer le paramétrage du site principal par défaut dont la racine se trouve, toujours par défaut à `/var/www/`.
- Le site par défaut est déjà activé ; on retrouve donc un lien vers ce fichier dans [`/etc/apache2/sites-enabled`](#).
- Sinon, le principe est le suivant :
  - On crée un fichier de configuration (appelé `conf_site`) pour un site web spécifique dans [`/etc/apache2/sites-available/`](#).
  - On active ce fichier de configuration par la commande : **`a2ensite conf_site`** ; cette commande a pour effet de créer un lien dans `/etc/apache2/sites-enabled/` qui pointe vers [`/etc/apache2/sites-available/conf\_site`](#).



### *Les directives*

- Les directives qui suivent correspondent à des serveurs spécifiques et sont donc incluses dans les fichiers de configuration présents dans /etc/apache2/sites-available/ , notamment celui par défaut /etc/apache2/sites-available/default.
- **ServerAdmin** **webmaster@localhost**, précise quel est le compte qui reçoit les messages. Par défaut un compte spécifique administrateur de site web (à modifier pour une adresse comme root@MonDomaine.edu).
- **ServerName** **www.MonDomaine.edu**, indique le nom ou l'alias avec lequel la machine est désignée. Par exemple, l'hôte ns1.MonDomaine.edu, peut avoir le nom d'alias www.MonDomaine.edu. Ce nom doit correspondre à une adresse IP, donc être renseigné dans un serveur DNS (ou dans un premier temps mais à éviter en production dans un fichier hosts sur le client). S'il n'est pas défini, alors le serveur tentera de le résoudre à partir de sa propre adresse IP.
- **DocumentRoot** **/var/www**, indique l'emplacement par défaut des pages HTML quand une requête accède au serveur.
  - Exemple : la requête `http://www.MonDomaine.edu/index.html` pointe en fait sur le fichier local `/var/www/index.html` sauf si le répertoire est pointé par une directive tel que Alias (voir ci-après).

### *Les directives*

- **Alias /CheminVu/ /CheminRéel/** , par exemple : `"/icons/ /usr/share/apache/icons/"`, ce paramètre permet de renommer, à la manière d'un lien logique, un emplacement physique avec un nom logique.
  - Exemple: vous voulez que `www.MonDomaine.edu/test/index.html`, ne corresponde pas physiquement à un répertoire sur la racine du serveur HTTP (défini par la directive précédente `DocumentRoot`) mais à un emplacement qui serait `/usr/local/essai`. Vous pouvez mettre dans le fichier de configuration d'Apache un alias de la forme: `alias /test/ /usr/local/essai/`
- **ScriptAlias /cgi-bin/ /usr/lib/cgi-bin**, de la forme `"ScriptAlias FakeName RealName"`, indique où sont physiquement situés les scripts sur le disque, ainsi que l'alias utilisé par les développeurs pour le développement des scripts et des pages. Un développeur utilisera un lien (exemple : `/cgi-bin/NomDuScript` où `/cgi-bin/` est un alias sur `/home/httpd/cgi-bin/`), et c'est le script `/home/httpd/cgi-bin/NomDuScript` qui sera effectivement exécuté. La mise en place d'alias permet de restructurer ou déplacer un serveur sans avoir à modifier toutes les pages développées.

### *Les directives*

- **DirectoryIndex** donne le ou les noms des fichiers que le serveur doit rechercher si le navigateur passe une requête sur un répertoire. Par exemple sur une requête `http://www.MonDomaine.edu`, le serveur va rechercher dans l'ordre s'il trouve un fichier `index.html`, `index.shtml`, `index.cgi`... en fonction des paramètres de cette variable.
- On peut activer ou non (activée par défaut) l'option "**Indexes**" au niveau d'un répertoire de manière à ce que si une URL pointe sur un répertoire, et aucun fichier défini par `DirectoryIndex` n'existe dans ce dernier, alors le serveur retourne une liste du contenu du répertoire. Si l'indexation n'est pas activée (ce qui est plus sécurisé), on obtient une page d'erreur 403 ("You don't have permission to access /repertoire on this server")

### *Sécurisation des accès*

- Chaque répertoire dont le contenu doit être géré par Apache2 peut être configuré spécifiquement.
- Pour chaque répertoire "UnRépertoire", sur lequel on désire avoir une action, on utilisera la procédure suivante:

**<Directory UnRépertoire>**

**...Ici mettre les actions...**

**</Directory>**

Tout ce qui est entre les balises s'applique au répertoire "UnRépertoire".

### *Sécurisation des accès*

- Quelques options :

**<Directory UnRépertoire>**

**Options Indexes FollowSymLinks ExecCGI**

...

**</Directory>**

- Indexes : autorise l'affichage du contenu d'un répertoire (si un fichier par défaut n'y est pas trouvé).
- FollowSymLinks: le serveur est autorisé à suivre les liens symboliques dans ce répertoire.
- ExecCGI : l'exécution de scripts CGI est autorisé.

Pour désactiver les options (par exemple Indexes)

**<Directory UnRépertoire>**

**Options -Indexes FollowSymLinks ExecCGI**

...

**</Directory>**

### *Sécuriser un répertoire en autorisant/refusant l'accès*

- Exemple: On désire autoriser l'accès du répertoire "/intranet" aux machines du réseau d'adresse 192.168.1.0/24 et de nom de domaine MonDomaine.edu, et l'interdire à tous les autres.

```
<Directory /intranet>  
#Ordre de lecture des règles  
order allow,deny  
deny from all  
allow from 192.168.1 #ou encore allow from .MonDomaine.edu  
</Directory>
```

- Il importe de préciser dans quel ordre les règles de restriction vont être appliquées. Cet ordre est indiqué par le mot réservé "order", par exemple "order deny,allow" (On refuse puis on alloue l'accès à quelques adresses, c'est à dire que toutes les règles deny vont être lues d'abord, puis ce sera le tour de toutes les règles allow) ou "order allow,deny" (on accepte généralement les accès mais il sont refusés pour quelques adresses : ici, on prend en compte en premier lieu toutes les règles allow dans l'ordre trouvé, puis ensuite toutes les règles deny).

### *Sécuriser un répertoire en autorisant/refusant l'accès*

- Exemple: On désire que l'accès soit majoritairement accepté, sauf pour un ou quelques sites.

```
<directory /home/httpd/html>  
  AllowOverride none  
  Order deny,allow  
  deny from pirate.com badboy.com cochon.com  
  allow from all  
</directory>
```

### *Fichier de configuration apache2.conf*

- Le fichier de configuration principal est **apach2.conf** C'est un fichier texte qui contient des directives. Le fichier est structuré en 3 parties :
  - \* **Global Environment** : environnement global
  - \* **Main Server Configuration** : configuration du serveur « maître »
  - \* **Virtuals Hosts** : configuration des hôtes virtuelsAvec l'installation par défaut Apache ne possède pas d'hôte virtuel.
- Les directives **<IfModule nom.c>** testent la présence du module afin de s'adapter aux configurations. Les autres lignes commençant par **#** sont des commentaires



### *Les hôtes virtuels*

- Le mécanisme des sites virtuels permet à un serveur Web d'héberger de multiples sites. Les fournisseurs d'accès ont souvent recours à cette technique, car ils ne souhaitent pas gérer une machine différente pour chacun des domaines.
  - **Le terme Hôte Virtuel** se réfère à la technique qui consiste à maintenir **plus d'un serveur sur une même machine**, tout en étant différenciés par leur nom apparent.
  - Par exemple, il est souvent préférable pour des compagnies se partageant un serveur Web d'avoir leurs propres domaines accessibles par `www.orabec1.com` et `www.orabec2.com`

### Les hôtes virtuels

- Dans Apache2, les sites disponibles (on parle de virtualhosts, ou vhost) sont déclarés à l'aide de fichier dans le répertoire :
  - **/etc/apache2/sites-available/**
- Partons du principe que vous souhaitez mettre en ligne 2 sites que nous nommerons site-1 et site-2, Apache vous livre lors de son installation dans ce même **répertoire un fichier "default"** qui va nous servir de base pour nos 2 sites On va donc en faire une copie :

```
root@lamp /etc/apache2# ls -l
total 80
-rw-r--r-- 1 root root 7115 Apr 11 23:23 apache2.conf
drwxr-xr-x 2 root root 4096 Apr 15 13:09 conf-available
drwxr-xr-x 2 root root 4096 Apr 15 13:09 conf-enabled
-rw-r--r-- 1 root root 1782 Oct 24 2015 envvars
-rw-r--r-- 1 root root 31063 Oct 24 2015 magic
drwxr-xr-x 2 root root 12288 Mar 31 23:02 mods-available
drwxr-xr-x 2 root root 4096 Mar 31 22:55 mods-enabled
-rw-r--r-- 1 root root 333 Apr 8 2016 ports.conf
drwxr-xr-x 2 root root 4096 Apr 11 22:58 sites-available
drwxr-xr-x 2 root root 4096 Apr 8 2016 sites-enabled
root@lamp /etc/apache2#
```

### *Les hôtes virtuels*

Le contenu du fichier default.conf

```
ServerName localhost

<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/
</VirtualHost>

<VirtualHost *:443>
    SSLEngine on
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/
</VirtualHost>

ScriptAlias /cgi-bin/ /var/www/cgi-bin/

<Directory /var/www/>
    Options Indexes FollowSymLinks MultiViews
    Require all granted
</Directory>
```

**cp** /etc/apache2/sites-avaiaible/default.conf /etc/apache2/sites-avaiaible/site-1.conf

**cp** /etc/apache2/sites-avaiaible/default.conf /etc/apache2/sites-avaiaible/site-2.conf

### Les hôtes virtuels

- site-1.conf

```
<VirtualHost *:80>
    ServerAdmin MonMail@gmail.com
    ServerName www.site-1.fr
    ServerAlias site-1.fr
    ServerAlias *.site-1.fr
    DocumentRoot /home/www/site-1
    DirectoryIndex index.php

<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>
<Directory /home/www/site-1>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Require all granted
</Directory>
    ErrorLog /home/www/apache_log/error_site-1.log
    # Possible values include: debug, info, notice, warn, error,crit,
    # alert, emerg.
    LogLevel warn
    CustomLog /home/www/apache_log/access_site-1.log combined
</VirtualHost>
```

### *Les hôtes virtuels*

- Activation/Désactivation des sites
  - Une fois les fichiers de vhost créés, il reste à les activer pour qu'Apache2 les prenne en compte.
  - Ce mécanisme permet d'avoir tout les fichiers de vhosts que l'on veut et de n'activer que ceux nécessaires.
- Deux méthodes possible:
  - Méthode Classique: Création/suppression d'un lien symbolique dans `/etc/apache2/sites-enable` pointant vers `/etc/apache2/sites-avaible/site-1`
    - `ln -s /etc/apache2/sites-avaible/site-1.conf /etc/apache2/sites-enable/site-1.conf`
  - les Développeurs Debian, qui font bien les choses (sinon, ils ne seraient pas DD !), ont mis en place un outil très simple:
    - `a2ensite site-1.conf`
    - ...et comme ils ne font pas les choses a moitié, ils ont aussi prévu la désactivation `a2dissite site-1.conf`

### *Les hôtes virtuels*

#### ▪ **Adaptation du fichier /etc/hosts**

- Cette opération n'est nécessaire qu'à fin de tester votre installation en local, si votre serveur et votre client sont sur la même machine, ou si vous ne disposez pas de plusieurs noms de domaine.
- Éditez votre fichier /etc/hosts et modifiez le comme suit pour que votre machine fasse le rapport entre vos vhost et le localhost.

/etc/hosts

127.0.0.1	localhost
127.0.0.1	site-1
127.0.0.1	site-2

#### ▪ **Relancer le serveur apache2 et tester**

- `service apache2 restart`

#### ▪ **Testez vos sites**

- `http://site-1/` puis <http://site-2/>
- Vous devriez voir apparaître les pages d'accueil des site-1 et site-2.

### ***Les hôtes virtuels***

- Dans le cas d'une utilisation du serveur à des fins de test ou d'un développement en local, il peut-être pratique de pouvoir modifier tous les fichiers de son site sans s'empêtrer avec les problèmes de droits Root/User.
  - Pour cela on crée **un lien symbolique** depuis le répertoire du serveur vers le répertoire de son dossier personnel.
  - Par exemple on place les fichiers de site-1 dans :  
/home/utilisateur/dev/site-1, puis on crée un lien symbolique comme ceci :
    - **In -s /home/utilisateur/dev/site-1 /var/www/site-1**

### *Les hôtes virtuels*

- `<VirtualHost *:80>` et `</VirtualHost>`: Signalent le début et la fin de la section du vhost, en écoute pour toutes les interfaces/adresses IP (\*) sur le port **80**.
- `ServerAdmin webmaster@localhost` : adresse mail où envoyer les messages d'erreur. Devra donc être remplacée par la vôtre
- `ServerName www.blablabla.xxx` : Nom utilisé par le vhost, remplacez-le par le nom de votre site (ici **site-1**)
- `DocumentRoot /var/www` : Répertoire de stockage du site (sa racine). Vous devrez donc modifier cette directive pour qu'elle pointe sur votre racine (ex: `/home/www/site-1` )



### Tester votre configuration

- Document root : **/var/www/html** or **/var/www**
- Fichier de configuration principal: **/etc/httpd/conf/httpd.conf** (RHEL/CentOS/Fedora) and **/etc/apache2/apache2.conf** (Debian/Ubuntu).
- Port HTTP par défaut: **80** TCP
- Port HTTPS par défaut: **443** TCP
- Tester votre fichier de configuration (paramètres + syntaxe) : **apache2 -t**
- Fichier de log d'accès du serveur web: **/var/log/httpd/access\_log**
- Fichier de log d'erreurs du serveur web: **/var/log/httpd/error\_log**

```
root@kali:/var/log/apache2#  
root@kali:/var/log/apache2#  
root@kali:/var/log/apache2#  
root@kali:/var/log/apache2# apache2 -t  
[Tue May 02 12:40:20.671208 2017] [core:warn] [pid 5522] AH00111: Config variable ${APACHE_PID_FILE} is not defined  
[Tue May 02 12:40:20.671280 2017] [core:warn] [pid 5522] AH00111: Config variable ${APACHE_RUN_USER} is not defined  
[Tue May 02 12:40:20.671288 2017] [core:warn] [pid 5522] AH00111: Config variable ${APACHE_RUN_GROUP} is not defined  
[Tue May 02 12:40:20.671301 2017] [core:warn] [pid 5522] AH00111: Config variable ${APACHE_LOG_DIR} is not defined  
[Tue May 02 12:40:20.735709 2017] [core:warn] [pid 5522] AH00111: Config variable ${APACHE_LOG_DIR} is not defined  
[Tue May 02 12:40:20.735958 2017] [core:warn] [pid 5522] AH00111: Config variable ${APACHE_LOG_DIR} is not defined  
[Tue May 02 12:40:20.736028 2017] [core:warn] [pid 5522] AH00111: Config variable ${APACHE_LOG_DIR} is not defined  
AH00543: apache2: bad user name ${APACHE_RUN_USER}  
root@kali:/var/log/apache2#  
root@kali:/var/log/apache2#  
root@kali:/var/log/apache2# apache2 -v  
Server version: Apache/2.4.23 (Debian)  
Server built: 2016-08-12T19:44:31  
root@kali:/var/log/apache2#
```

### *Processus/service Apache2*

- Le processus doit tourner avec un utilisateur banalisé
  - Le processus est lancé par le user root
  - Ensuite, le processus passe sous l'autorité de l'utilisateur banalisé
- Création du groupe et l'utilisateur banalisé

```
User http-web  
Group http-web  
  
# useradd -d /var/www/ -g http-web -s /bin/nologin http-web
```

- Ouvrir le fichier de configuration apache2.conf

```
User http-web  
Group http-web
```

- Redémarrer le service

### Configuration SSL

- Pour encrypter la communication entre votre serveur web Apache et vos clients Web, vous devez utiliser le module **mod\_ssl**. Activer ce module:

```
bob@linux-ub: /etc/apache2
bob@linux-ub:/etc/apache2$ sudo a2enmod ssl
[sudo] password for bob:
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self
-signed certificates.
To activate the new configuration, you need to run:
    service apache2 restart
bob@linux-ub:/etc/apache2$
```

- Apache2 Enable Module : a2enmod
- Apache2 Disbale Module : a2dismod

### ***Activer le site SSL par défaut***