

# PHP et MySQL

Site Web dynamique

- Introduction à cette formation

- Votre formateur ... Et Vous



- Le matériel et logiciels

- L'environnement de développement VisualStudio Code orienté développement Web.
    - Navigateurs Chrome et Firefox
    - Le serveur Web Apache/Tomcat, emploi de XAMPP

- L'organisation – horaires

- Formation de 4.5 jours

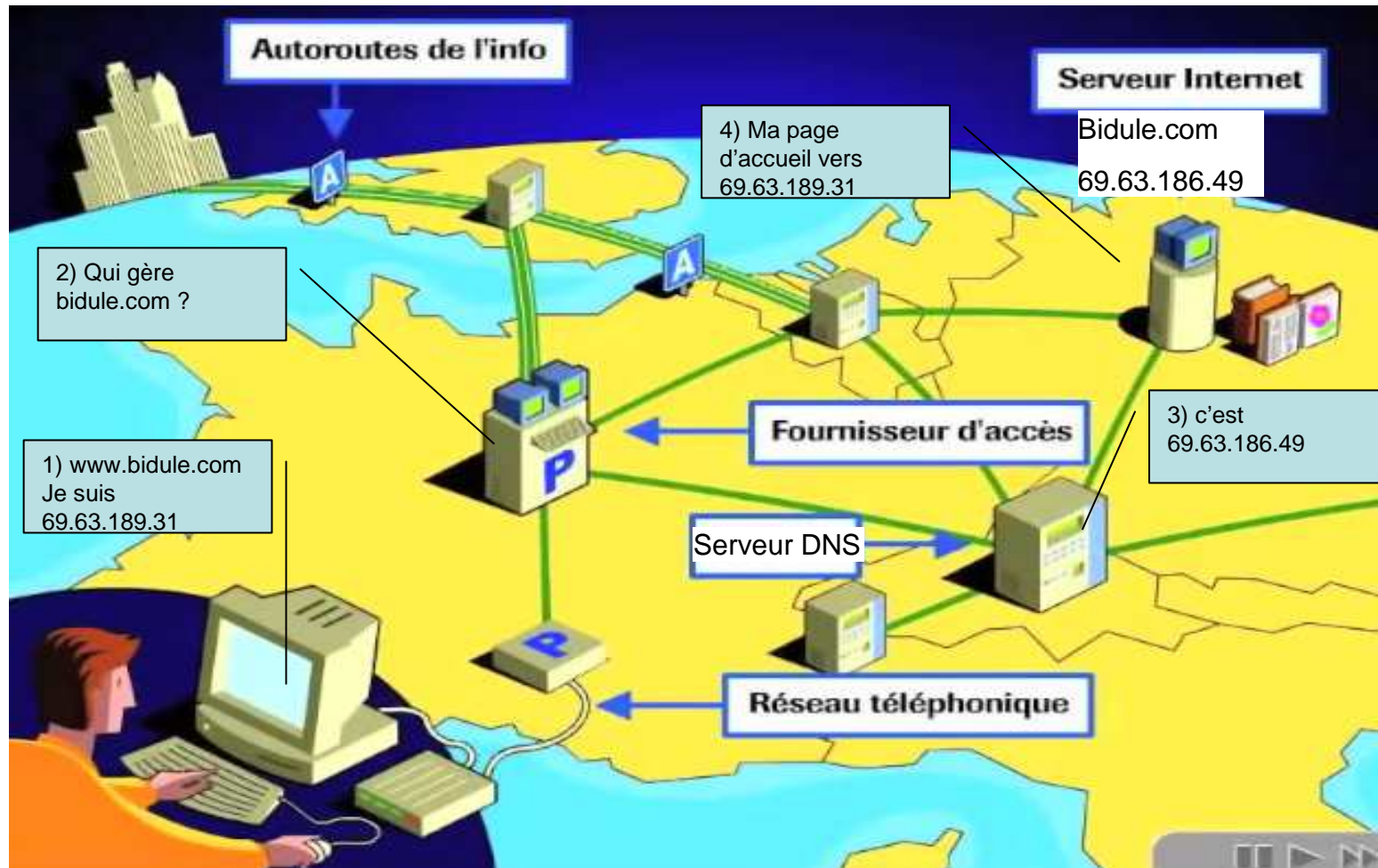
- La forme :

- Un mélange de concepts avec application directe par un exemple simple
    - Des exercices

- Les liens utiles

- <http://www.w3schools.com/>
- <https://www.futura-sciences.com/tech/videos/kezako-fonctionne-internet-5293/>
- <https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql>
- php.net/<nom fonction>

- Généralités sur le web
- Notre environnement de travail
- PHP
- MySQL



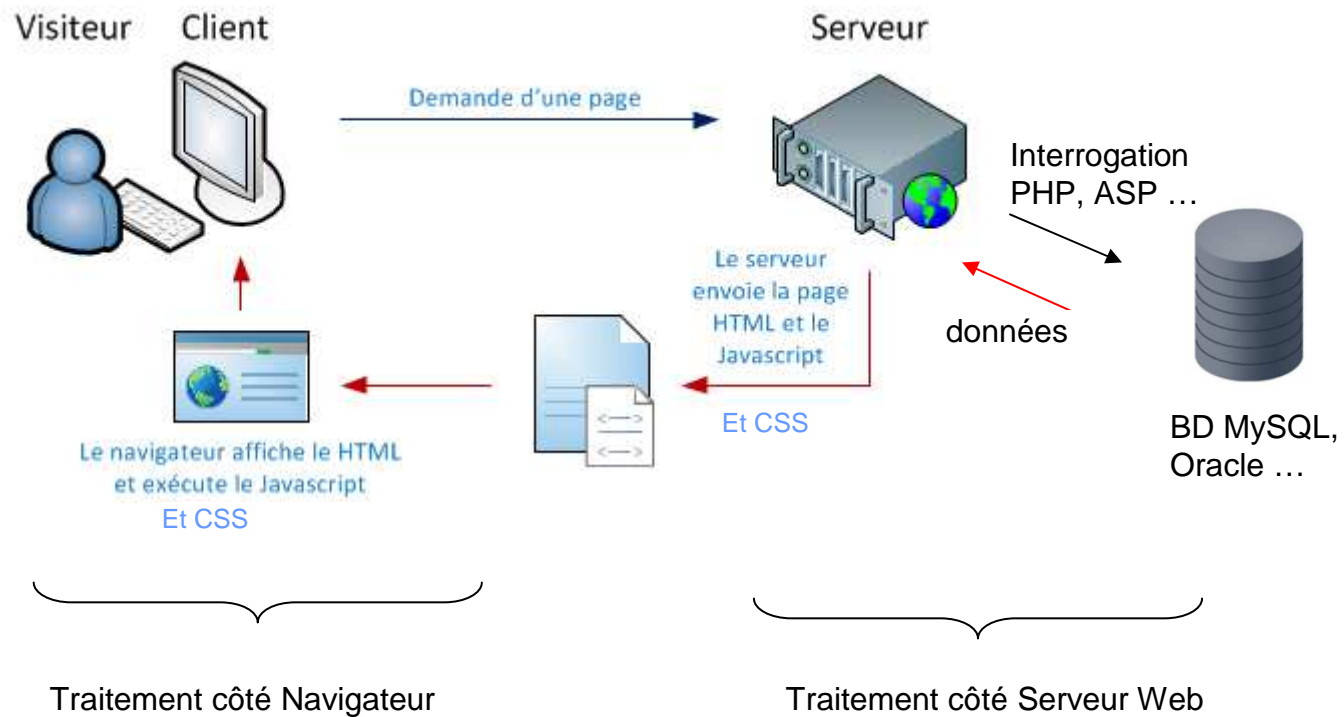
# PHP/MYSQL

- La communication client/serveur (navigateur/serveur du site Web) se fait en TCP/IP
- Protocoles utilisés
  - http, HyperText Transfert Protocol
  - https, crypté
  - ftp pour télécharger de gros fichiers
  - Imap, smtp pour les e-mails
  - ...
- Pour l'affichage d'une page web, deux acteurs principaux
  - Le serveur Web qui fournit les données
  - Le navigateur qui sait les interpréter et afficher



# PHP/MYSQL

- Traitements serveur et client



- Outils utiles :

- Côté navigateur, apprendre le HTML, CSS, Javascript : utilisation de l'éditeur de texte et simulateur, application Brackets utilise le navigateur Chrome pour visualiser les codes..



- Nous allons écrire du code côté serveur, Visual Studio Code (VSCode) va être utilisé



- Côté serveur Web, utiliser un serveur Web, interpréteur php, base MySQL : environnement XAMPP

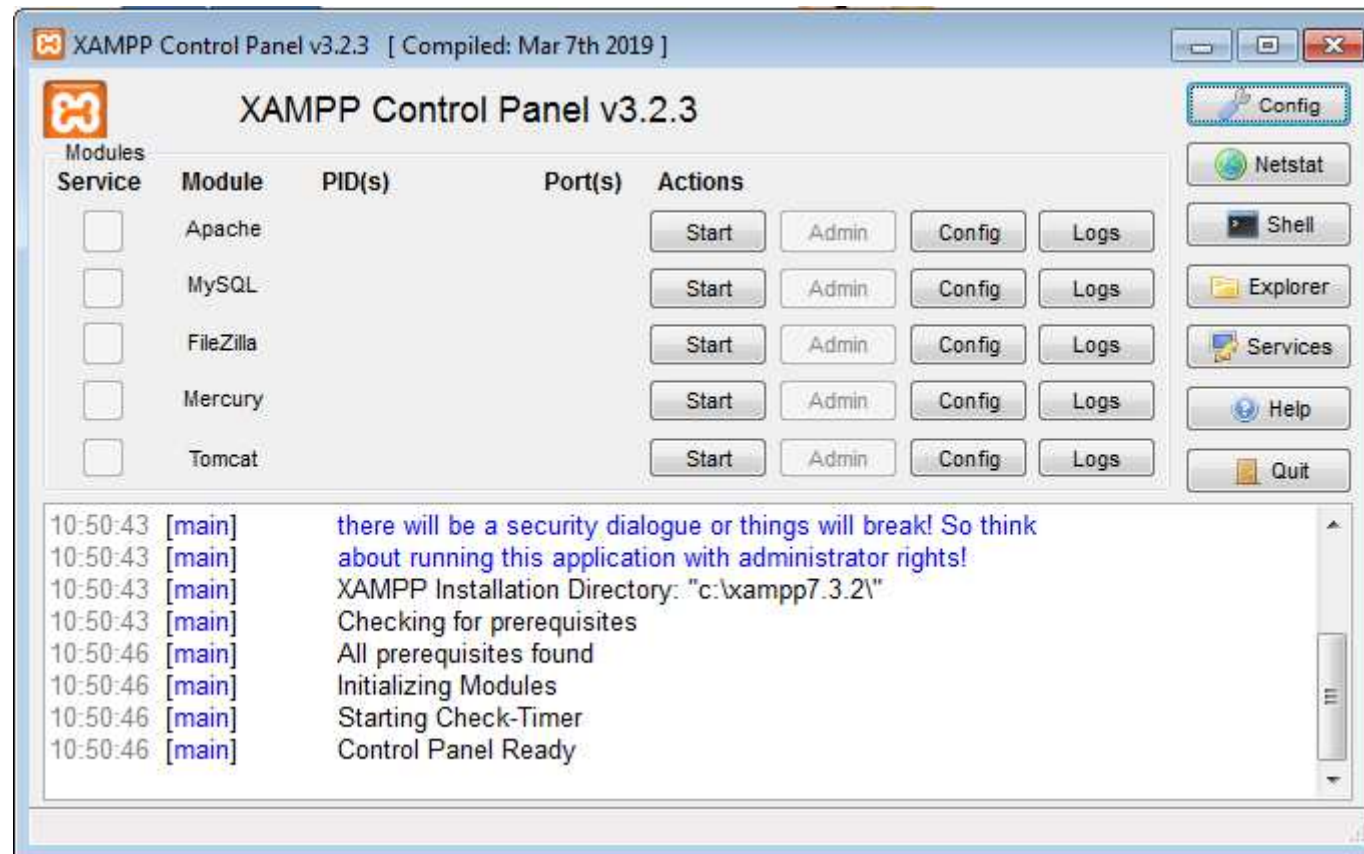


- Utiliser le document InstallXamppVSCode joint





- Lancer XAMPP



- Lancer par Start les uns après les autres
  - Apache, MySQL
- Noter où se trouve le répertoire d'installation de xampp. Xampp/htdocs est le répertoire racine du site Web
- Sous ce répertoire Xampp/htdocs créer le répertoire **test**
- Sous un navigateur Web, aller à l'adresse localhost, constater que l'on voit le répertoire test vide
- Copier le fichier **test1.html** et le regarder avec le navigateur Web
- Avec Visual Studio Code (VSCode) :
  - Fichier -> Ouvrir un dossier et choisir Xampp/htdocs
  - Editer test1.html
- Nous sommes prêts !!

- Insérer du code php dans du code html :  
    <?php  
    .... Code php  
    ?>  
    Les balises <? .. ?> et <% .. %> sont équivalentes
- Le code php peut être mis partout dans le html

```
<p>  
    Cette page contient du PHP.<br />  
    Voici quelques petits tests :  
    <?PHP echo "texte issu de PHP"; ?>  
</p>
```

- Faire une copie de test1.html en test2.html et y insérer ce code.  
Le regarder dans le navigateur. Fonctionne ?

- Tout fichier contenant du php doit avoir comme suffixe .php
- Inclusion de pages  
La problématique à résoudre est un site contenant plusieurs pages dont l'en-tête, menu, pied de pages sont identiques. Seul le corps change.



- La bonne démarche est de définir les parties communes dans des fichiers php, et de mettre des inclusions dans le fichier principal par l'instruction  
`<?php include("xxx.php"); ?>`
- Utiliser le répertoire htdocs\inclusion fourni avec le cours
  - Regarder et utiliser main\_complet.html
  - Regarder et utiliser main.php qui réalise les inclusions

- Chaque instruction se termine par ;
- Les commentaires sont // ou # ou /\* .. \*/
- Affichage pour le navigateur : echo, print, printf
- Les types de variables :  
Booléens, entiers, flottants, chaînes, tableaux, objets
- Toute variable commence par \$ . Le nom est sensible à la casse.
- Les types de variables sont implicites (pas de déclaration)  
\$aa=12.5;                      \$aa est float  
\$bb="abcd";                      \$bb est une string                      idem \$cc='abcd';
- Booléens :
  - Valeur true ou false
  - Un entier = 0 est considéré comme false
  - Une chaîne vide ou « 0 » est considérée comme false

- Guillemets et apostrophes

```
$mot="Bonjour";  
echo 'valeur : $mot';           // affiche valeur : $mot  
echo "valeur : $mot";           // affiche valeur : Bonjour  
echo "valeur : " . $mot;        // affiche valeur : Bonjour
```

#### Exercice 1



- Caractères spéciaux  
\\n nelle ligne, \\t tabulation, \\\$ dollar
- Une variable peut être déclarée (apparaître) n'importe où dans le script, mais a une portée de type
  - local
  - global
  - static
- var\_dump(\$var) montre le type et contenu d'une variable

- Une var déclarée hors d'une fonction a une portée GLOBAL et est accessible uniquement en dehors de la fonction

```
<?php
$x = 5; // global scope

function myTest() {
    // using x inside this function will generate an error
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();

echo "<p>Variable x outside function is: $x</p>";
?>
```

Variable x inside function is:

Variable x outside function is: 5

- A l'inverse, une var déclarée dans une fonction a une portée LOCAL et est accessible uniquement dans la fonction

```
<?php
function myTest() {
    $x = 5; // local scope
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();

// using x outside the function will generate an error
echo "<p>Variable x outside function is: $x</p>";
?>
```

Variable x inside function is: 5

Variable x outside function is:



- Le mot clé **global** rend une var accessible à l'intérieur d'une fonction

```
<?php
$x = 5;
$y = 10;

function myTest() {
    global $x, $y;
    $y = $x + $y;
}

myTest(); // run function
echo $y; // output the new value for variable $y
?>
```

Result:

15

- PHP stocke les var globales dans un tableau \$GLOBALS[]  
Le nom de la var sert d'index. L'accès au tableau est permis

```
<?php
$x = 5;
$y = 10;

function myTest() {
    $GLOBALS['y'] = $GLOBALS['x'] + $GLOBALS['y'];
}

myTest();
echo $y;
?>
```

Result:

15

- Le mot clé **static** rend une var rémanente (on garde sa place mémoire) dans une fonction

```
<?php
function myTest() {
    static $x = 0;
    echo $x;
    $x++;
}

myTest();
echo "<br>";
myTest();
echo "<br>";
myTest();
?>
```

Result:

0  
1  
2

Dans cet exemple la ligne **static** \$x = 0; n'est exécutée qu'une fois  
\$x reste locale à la fonction

- Déclaration par le mot array
- Peut contenir des éléments de type différents
- La taille varie au cours de son utilisation
- Ex  
\$stabcolors = array('red','green','blue');  
\$stabcolors[] = 'green'; // ajout d'un élément
- Parcours d'un tableau

```
$i = 0  
while(i<count($tab))  
    echo $tab[$i++] . "<br>";
```

```
foreach($tab as $elem)  
    echo $elem
```

- Gère la notion de tableau associatif : associer un mot clé à chaque valeur

```
<?php
$age=array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
echo "Peter is " . $age['Peter'] . " years old.";
?>
```

Peter is 35 years old.

- Boucle sur les valeurs d'un tableau associatif :

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");

foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>
```

Key=Peter, Value=35  
Key=Ben, Value=37  
Key=Joe, Value=43

- print\_r(\$tab) est un moyen rapide d'afficher un tableau (debug)
- list(\$a,\$b,\$c) = array("Dog","Cat","Horse");

Exercice 2



- Constantes

Définition : `define(nom, valeur);`

Leur portée est globale

```
<?php
define("GREETING", "Welcome to W3Schools.com!");
echo GREETING;
?>
```

- Opérateurs

Regarder le site [w3schools.com/php](http://w3schools.com/php)

- If .. else

```
<?php
$t = date("H");

if ($t < "20") {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```

- Switch

```
<?php
$favcolor = "red";

switch ($favcolor) {
    case "red":
        echo "Your favorite color is red!";
        break;
    case "blue":
        echo "Your favorite color is blue!";
        break;
    case "green":
        echo "Your favorite color is green!";
        break;
    default:
        echo "Your favorite color is neither red, blue, nor green!";
}
?>
```

- Boucles

```
<?php
$x = 1;

while($x <= 5) {
    echo "The number is: $x <br>";
    $x++;
}
?>
```

```
<?php
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: $x <br>";
}
?>
```

```
<?php
$x = 1;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>
```

```
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
    echo "$value <br>";
}
?>
```

- Les fonctions peuvent prendre des arguments dont il n'est pas besoin de spécifier le type
- Elles peuvent retourner une valeur
- Tout type peut être renvoyé : tableau, entier ...
- Les identificateurs de fonctions sont insensibles à la casse

```
<?php
function familyName($fname, $year) {
    echo "$fname Refsnes. Born in $year <br>";
}

familyName("Hege", "1975");
familyName("Stale", "1978");
familyName("Kai Jim", "1983");
?>
```

---

```
Hege Refsnes. Born in 1975
Stale Refsnes. Born in 1978
Kai Jim Refsnes. Born in 1983
```

- Il existe la notion d'argument par défaut

```
<?php
function setHeight($minheight = 50) {
    echo "The height is : $minheight <br>";
}

setHeight(350);
setHeight();
setHeight(135);
setHeight(80);
?>
```

```
The height is : 350
The height is : 50
The height is : 135
The height is : 80
```



- Valeur de retour : le mot **return** suffit

```
<?php  
function sum($x, $y) {  
    $z = $x + $y;  
    return $z;  
}
```

```
echo "5 + 10 = " . sum(5,10) . "<br>";  
echo "7 + 13 = " . sum(7,13) . "<br>";  
echo "2 + 4 = " . sum(2,4);  
?>
```

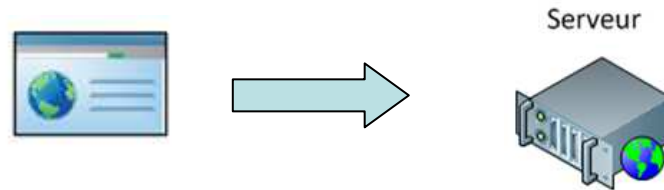
5 + 10 = 15

7 + 13 = 20

2 + 4 = 6

Exercice 3





- Afin d'obtenir des pages dynamiques et personnalisées, le navigateur doit fournir des données au serveur Web.  
2 méthodes
- Méthode 1 : Envoyer des paramètres dans l'URL  
L'URL est en effet formé par du code HTML côté navigateur  
L'URL est reçue par le serveur, du code PHP peut l'analyser
  - Un URL est de la forme  
`http://www.site.com/page.php?param1=valeur1&param2=valeur2&param3=valeur3 ....`  
  
Signifie que le navigateur a formé cette chaîne et appelle la page `page.php` avec des valeurs de paramètres.

- Comment le navigateur a-t-il fait pour former cet URL?



- Par un tag html `<a>`  
`<a href=page.php?param1=valeur1&param2=valeur2&param3=valeur3> texte`  
`</a>`
- Par l'emploi d'un formulaire `<form>` avec la `method="get"`  
Dans ce formulaire les divers contrôles ont l'attribut `name="param1"`,  
`param2` et `param3`  
`<form action=" page.php" method="get">`

- Côté serveur : la page `page.php` doit contenir du code PHP pour récupérer les valeurs

Serveur




Le tableau associatif `$_GET` contient ces valeurs :

`$_GET['param1']` contient 'valeur1'

`$_GET['param2']` contient 'valeur2'

`$_GET['param3']` contient 'valeur3'

- Cette méthode est très vulnérable car les noms et valeurs peuvent être modifiées depuis le navigateur. Il faut alors contrôler et protéger le code PHP :
  - Vérifier l'existence des paramètres  
`if (isset($_GET['param']))`
  - Si la valeur attendue est un entier, float, booléen, imposer un cast  
`$_GET['age'] = (int) $_GET['age']; // 0 si age ne contient pas un entier`
  - Si la valeur est une chaîne, s'assurer qu'il n'y pas injection de code (la faille XSS).  

  - Utiliser :  
`htmlspecialchars($_GET['param']); // remplace les balises par du texte`  
`strip_tag($_GET['param']); // supprime les balises`

#### Exercice 4



- 2<sup>ème</sup> méthode de transmission de données : la méthode POST
  - Utilisée par les formulaires html, method="post"
  - Permet d'échanger un volume important de données
  - Côté serveur, en PHP, les données sont dans le tableau associatif \$\_POST
  - Est aussi vulnérable que le GET => protéger le code



## Exercice 5



- Transfert de fichier du navigateur vers le site
  - Côté navigateur

```
<form action="echanges/transfert_fichier.php" method="post" enctype="multipart/form-data">
  <p>
    Formulaire d'envoi de fichier :<br />
    <input type="file" name="monfichier" /><br />
    <input type="submit" value="Envoyer le fichier" />
  </p>
</form>
```

- Côté serveur, le fichier php  
Sur réception côté serveur, le fichier est stocké dans un espace temporaire.  
\$\_FILES['monfichier'] fournit un tableau associatif sur le fichier  
Après vérification le fichier est copié à l'endroit voulu du serveur par  
move\_uploaded\_file();

## – Exemple de code

```
<?php
// Testons si le fichier a bien été envoyé et s'il n'y a pas d'erreur
echo "aa";
if (isset($_FILES['monfichier']) AND $_FILES['monfichier']['error'] == 0)
{
    echo "bb";
    // Testons si le fichier n'est pas trop gros
    if ($_FILES['monfichier']['size'] <= 1000000)
    {
        // Testons si l'extension est autorisée
        print_r($_FILES['monfichier']);
        $infosfichier = pathinfo($_FILES['monfichier']['name']);
        $extension_upload = $infosfichier['extension'];
        $extensions_autorisees = array('jpg', 'jpeg', 'gif', 'png', 'xml');
        if (in_array($extension_upload, $extensions_autorisees))
        {
            echo "Le fichier a la bonne extension<br/>";
            // On peut valider le fichier et le stocker définitivement
            $result = move_uploaded_file($_FILES['monfichier']['tmp_name'], 'uploads/' .
            basename($_FILES['monfichier']['name']));
            if ($result == true)
                echo "La sauvegarde a bien été effectuée !";
            else
                echo "La sauvegarde a échouée";
        }
    }
}
?>
```

## Exercice 6



- Les variables superglobales sont :
  - Créées automatiquement par PHP et sont accessibles partout
  - Ont un nom en majuscules commençant par \_ (sauf \$GLOBALS)
  - Sont des array associatifs
- print\_r(), affichage des array, permet de les découvrir
- \$\_GET, \$\_POST, \$\_FILES sont des superglobales
- Les autres sont  
\$GLOBALS \$\_SERVER \$\_REQUEST \$\_ENV \$\_COOKIE  
\$\_SESSION
- [http://www.w3schools.com/php/php\\_superglobals.asp](http://www.w3schools.com/php/php_superglobals.asp) pour le détail



Une session permet aux pages d'un site – si le site l'a permis – de partager des informations entre les pages.

Fonctionnement :

1. Un visiteur ouvre une page sur le site. Si cette page commence par `session_start()`, PHP génère un identifiant de session unique (un PHPSESSID). PHP connaît et transmet cet ID de page en page.
2. La variable superglobale `$_SESSION` est un tableau associatif qui peut être utilisé dans toutes les pages du site, à condition qu'elles commencent par `session_start()`  
`$_SESSION['nom'] = $_POST['name'];`
3. Pour terminer une session, soit `session_destroy()` est appelé sur un dialogue « deconnexion » ou un timeout termine la session

- Remarques :
  - Session\_start(); doit être placé avant <!DOCTYPE html>
  - \$\_SESSION[] stocke tout type d'information y compris des objets. C'est donc un moyen structuré d'organiser des données  
ex : droits d'accès à certaines pages, contenu d'un panier ...

- Exercice 7



- La durée de vie des données d'une session sont celles de la session
- Pour enregistrer des données au-delà de cette période, on peut les enregistrer dans des fichiers, côté client : les cookies
- Un cookie n'enregistre qu'une donnée texte à la fois. IL contient :
  - Un nom
  - Un contenu
  - Une date d'expiration
- Exemple : regarder les cookies depuis Firefox par :  
Outils->Options>Vie privée et Supprimer les cookies spécifiques

Depuis Chrome :

Paramètres ->Afficher les paramètres avancés ->Paramètres de contenu ->Cookies->Ensemble des cookies et données de site

- Ecrire un cookie :  
Utiliser la fonction PHP `setcookie()` avant tout code html de la page.

`setcookie()` nécessite 3 valeurs :

- Le nom du cookie, texte
- La valeur au format texte
- Une date d'expiration du cookie, sous forme de timestamp, nombre de seconde depuis le 01/01/1970  
`time()` fournit la date actuelle  
`time() + 365*24*3600` est une date de 1 an de plus

```
<?php setcookie('ville', 'Paris', time() + 365*24*3600 ); ?>
```

Une autre signature de `setcookie()` utilisant le mode `httpOnly`, rend le cookie plus sûr car moins accessible

```
<?php setcookie('ville', 'Paris', time() + 365*24*3600, null, null, false, true ); ?>
```

- Lire un cookie :
  - Utiliser la superglobale `$_COOKIE[]`, tableau associatif

```
echo "Votre ville est " . $_COOKIE['ville'] ;
```

- Modifier un cookie existant :  
Utiliser à nouveau `setcookie()`

- Problématique :  
Pouvoir sauvegarder des données dans un/des fichiers côté serveur Web

```
<!DOCTYPE html>
<html>
  <body>
    <h1>Fichier lectureEcritureFichier.php</h1>
    <?php
      $monfichier = fopen('compteur.txt', 'r+');

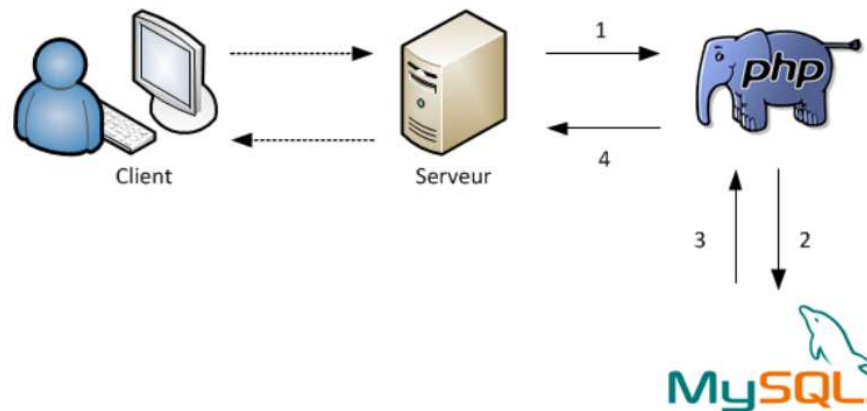
      $pages_vues = fgets($monfichier); // On lit la première ligne (nombre de pages vues)
      echo "pages_vues" . $pages_vues;
      $pages_vues += 1; // On augmente de 1 ce nombre de pages vues
      fseek($monfichier, 0); // On remet le curseur au début du fichier
      fputs($monfichier, $pages_vues); // On écrit le nouveau nombre de pages vues

      fclose($monfichier);

      echo '<p>Cette page a été vue ' . $pages_vues . ' fois !</p>';
    ?>
  </body>
</html>
```

- Les sites Web dynamiques nécessitent l'utilisation d'une base de données (BD) :
  - Par rapport à une conception avec des fichiers la BD permet d'organiser et de structurer les données du serveur Web
  - Permet aux CMS (Context Management System tels WordPress, Joomla ..) de partager des données entre les modes design et exploitation.  
Dans ce contexte tout le contenu du site, texte, photos ... se trouve dans la BD
- Les SGBD sont nombreux : MySQL, PostgreSQL, SQLite, Oracle, SQL server de Microsoft ...  
Nous utilisons ici MySQL

- Dans le contexte d'un serveur Web c'est PHP qui fait l'interface avec MySQL

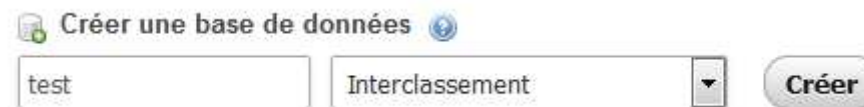


1. Le code PHP de la page demandée se déroule
2. Il contient une demande d'accès à la BD. Il élabore une requête SQL
3. MySQL exécute la requête et rend de l'information à PHP
4. PHP utilise l'information BD pour la présenter/utiliser au visiteur

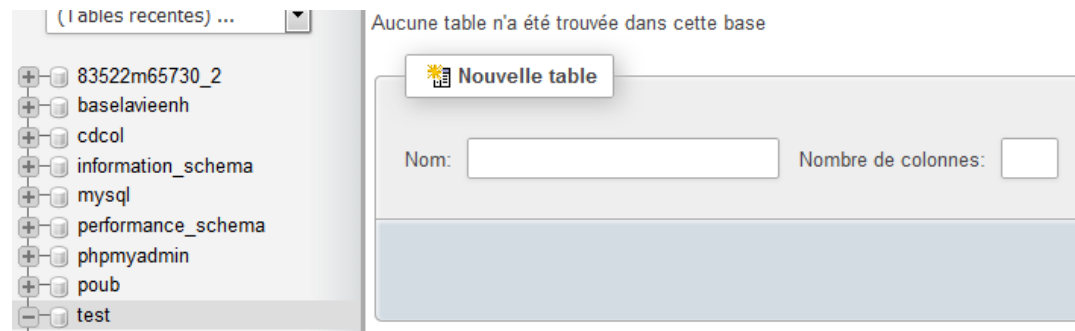


- La BD MySQL est un ensemble de tables
- Chaque table est un tableau :
  - dont les colonnes sont des *champs*
  - Dont les lignes sont des *entrées*
- Dans notre environnement de test, la BD est rangée dans des fichiers du répertoire  
XAMP\mysql\data


- Gérer la base de données : utilisation de PhpMyAdmin  
Dans un navigateur , URL localhost/phpmyadmin
- Ouvrir l'onglet base de données
- Créer la BD de nom test



- La BD est créée, sans table



- Créer la table news avec 3 colonnes

 Nouvelle table

Nom: 
 Nombre de colonnes:

Nom de la table: 
 Ajouter  colonne(s)
 Exécuter

Structure <small>?</small>									
Nom	Type <small>?</small>	Taille/Valeurs* <small>?</small>	Défaut <small>?</small>	Interclassement	Attributs	Null	Index	A	J
<input type="text" value="id"/>	INT <small>▼</small>	<input type="text"/>	Aucune <small>▼</small>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	PRIMARY <small>▼</small>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="titre"/>	VARCHAR <small>▼</small>	255	Aucune <small>▼</small>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	--- <small>▼</small>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="description"/>	TEXT <small>▼</small>	<input type="text"/>	Aucune <small>▼</small>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	--- <small>▼</small>	<input type="checkbox"/>	<input type="checkbox"/>

- Désigner « news » dans l'arbre de gauche puis onglet Insérer

- Créer 3 entrées dans la table

Colonne	Type	Fonction	Null	Valeur
id	int(11)	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>
titre	varchar(255)	<input type="text"/>	<input type="checkbox"/>	Mon 1er article
description	text	<input type="text"/>	<input type="checkbox"/>	Je crée ici ma 1ère entrée de la table news

					id	titre	description		
		Modifier		Copier		Effacer	1	Mon 1er article	Je crée ici ma 1ère entrée de la table news
		Modifier		Copier		Effacer	2	Mon 2eme article	Je crée ici ma 2ème entrée de la table news
		Modifier		Copier		Effacer	3	Mon 3eme article	Je crée ici ma 3ème entrée de la table news

- Exporter la BD  
Permet de placer dans un fichier la description SQL des tables d'une BD. Onglet Exporter le phpMyAdmin

### Exportation des tables depuis la base de données «test»

#### Méthode d'exportation :

- ☒ Rapide - n'afficher qu'un minimum d'options  
☐ Personnalisée - afficher toutes les options possibles

#### Format :

SQL ▼

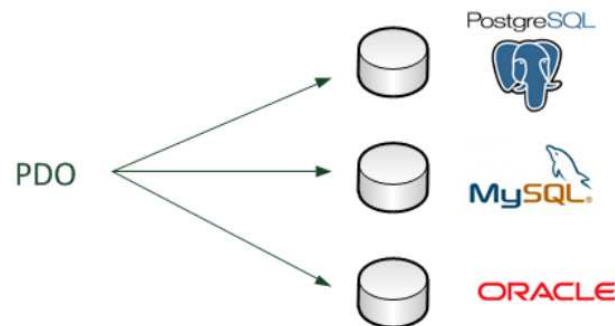
Exécuter

- Importer une BD

L'onglet Importer  Importer permet de prendre en compte un fichier exporté.

Utile pour transporter le contenu de la BD de la machine de développement vers le serveur du site Web

- PHP permet d'utiliser la BD avec plusieurs API :
  - Fonctions commençant par mysql\_  
Uniquement pour MySQL.  
Devenues obsolètes : ne pas utiliser
  - Fonctions commençant par mysqli\_  
Uniquement pour MySQL.  
Maintenues
  - Extension PDO.  
Utilisable pour plusieurs types de BD (MySQL, Oracle, PostgreSQL ...)  
L'API à utiliser.



- S'assurer que l'extension PDO est activée : fait de base à partir de PHP 5.1.0
- Se connecter en code PHP. IL faut connaître :
  - Le nom de l'hôte : localhost ou le nom que fournit l'hébergeur
  - Le nom de la base. Ex 'test'
  - Le login. 'root' par défaut, fourni par l'hébergeur
  - Le mot de passe. Vide par défaut, fourni par l'hébergeur

```
$bdd = new  
PDO('mysql:host=localhost;dbname=test;charset=utf8', 'root', '');
```



- Sur erreur de connexion, la page peut afficher des données sensibles au visiteur  
=> Protéger le code par un try catch

```
try
{
    $bdd = new PDO('mysql:host=localhost;dbname=test;charset=utf8', 'root', '');
}
catch (Exception $e)
{
    exit('Erreur : ' . $e->getMessage());
}
```

- Envoyer une requête SQL

```
$result = $bdd->query('select * from news');
```

- Lire le résultat

```
$row = $result->fetch();           // extraction d'une entrée (ligne)  
$allRow = $result->fetchAll();     // extraction de toutes les entrées
```

- Accès à chaque entrée

```
$result = $bdd->query('select * from news');  
while ($data = $result->fetch())  
{  
    echo $data['id'] . $data['titre'] . $data['description'] . "<br/>";  
}  
$result->closeCursor(); // termine le traitement
```

```
$allRows = $result->fetchAll();  
foreach ($allRows as $row)  
{  
    echo $row['id'] . $row['titre'] . $row['description'] . "<br/>";  
}  
$result->closeCursor(); // termine le traitement
```

## Exercice 8



- Écrire des **requêtes dynamiques** : une partie de la requête est définie par des variables
  - À ne pas faire :  
on souhaite rendre paramétrable la requête

```
<?php
$reponse = $bdd->query('SELECT nom FROM jeux_video WHERE possesseur=\'Patrick\'');
?>
```

par l'écriture


```
<?php
$reponse = $bdd->query('SELECT nom FROM jeux_video WHERE possesseur=\' ' .
$_GET['possesseur'] . '\');
?>
```

Ceci introduit une faille car `$_GET['possesseur']` peut contenir du code SQL malveillant (injection SQL) qui permet tout accès à la BD.



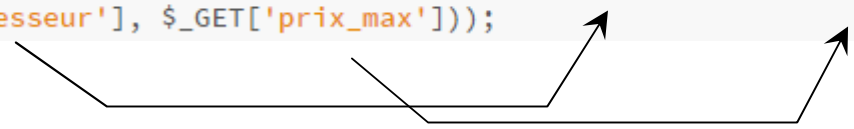
- Écrire des requêtes dynamiques (suite)  
La solution : **les requêtes préparées**, 2 étapes :
  - Préparation
  - Exécution

```
$req = $bdd->prepare('SELECT nom FROM jeux_video WHERE possesseur = ?');  
$req->execute(array($_GET['possesseur']));
```



A line connects the `$req` variable in the `prepare` statement to the `$req` variable in the `execute` statement. An arrow points from the `execute` statement to the right.

```
$req = $bdd->prepare('SELECT nom FROM jeux_video WHERE possesseur = ? AND prix <= ?');  
$req->execute(array($_GET['possesseur'], $_GET['prix_max']));
```



A line connects the `$req` variable in the `prepare` statement to the `$req` variable in the `execute` statement. An arrow points from the `execute` statement to the right.

- Écrire des requêtes dynamiques (suite)  
Utilisation de marqueurs nominatifs :marqueur

```
$req = $bdd->prepare('SELECT nom, prix FROM jeux_video WHERE possesseur = :possesseur AND  
prix <= :prixmax');  
$req->execute(array('possesseur' => $_GET['possesseur'], 'prixmax' => $_GET['prix_max']));
```

- Ajouter une entrée dans la table
  - La requête SQL d'ajout d'une entrée est  
INSERT INTO <table> (nom col, ...) VALUES ('val1', 'val2', ..)
  - La méthode PDO est  
\$bdd->exec()

```
$bdd->exec('INSERT INTO news (id, titre, description) VALUES (NULL, \'Mon 4eme article\', \'Je crée ici ma 4eme entrée de la table news\')');
```

- Utilisation d'une requête préparée

```
$req = $bdd->prepare('INSERT INTO jeux_video(nom, possesseur, console, prix, nbre_joueurs_max, commentaires) VALUES(:nom, :possesseur, :console, :prix, :nbre_joueurs_max, :commentaires)');  
$req->execute(array(  
    'nom' => $nom,  
    'possesseur' => $possesseur,  
    'console' => $console,  
    'prix' => $prix,  
    'nbre_joueurs_max' => $nbre_joueurs_max,  
    'commentaires' => $commentaires  
));
```

- Modifier des données dans la table
  - La requête SQL est  
UPDATE table SET colonne1=« val1 », colonne2=« val2 » ... WHERE condition
  - La méthode PDO est  
\$bdd->exec()

```
// modifier une entrée  
$bdd->exec('UPDATE news SET titre = "Nouveau titre" WHERE id = 1');
```

- Utilisation d'une requête préparée

```
$req = $bdd->prepare('UPDATE jeux_video SET prix = :nvprix, nbre_joueurs_max = :nv_nb_joueurs  
WHERE nom = :nom_jeu');  
$req->execute(array(  
    'nvprix' => $nvprix,  
    'nv_nb_joueurs' => $nv_nb_joueurs,  
    'nom_jeu' => $nom_jeu  
));
```

- Supprimer des entrées dans la table

- La requête SQL est  
DELETE FROM table WHERE condition

- La méthode PDO est  
\$bdd->exec()

```
$bdd->exec('DELETE FROM news WHERE id = 5');
```

- Utilisation d'une requête préparée possible aussi



- Les fonctions SQL

Il en existe de 2 types :

- Les fonctions scalaires : agissent individuellement sur chaque entrée
- Les fonctions d'agrégat : utilise un ensemble et rend une valeur

- Les fonctions scalaires

Exemple :

`SELECT UPPER(nom) FROM jeux_video` : rend les noms en majuscule, sans altérer le contenu de la table

`SELECT UPPER(nom) AS nom_maj FROM jeux_video` : idem, le résultat est mis dans un champ virtuel ou alias 'nom\_maj'

```
$reponse = $bdd->query('SELECT UPPER(nom) AS nom_maj FROM jeux_video');  
while ($donnees = $reponse->fetch())  
{  
    echo $donnees['nom_maj'] . '<br />';  
}  
$reponse->closeCursor();
```

LOWER(), LENGTH(), ROUND() ... sont d'autres exemples

- Les fonctions d'agrégat  
Ex

```
SELECT AVG(prix) AS prix_moyen FROM jeux_video
```

calcule la moyenne AVG des valeurs de la colonne prix et rend la valeur dans prix\_moyen.

- Autres fonctions d'agrégat :  
SUM(), MAX(), MIN(), COUNT(\*)

Un peu d'exercice ! : exercice 9



- Requêtes avancées

- ORDER BY

- ```
SELECT * FROM jeux_video ORDER BY prix DESC
```

- LIMIT

- ```
SELECT * FROM jeux_video LIMIT 0, 20
```

- GROUP BY

- ```
SELECT AVG(prix) AS prix_moyen, console FROM jeux_video
```

permet d'avoir un prix moyen sur l'ensemble des 'prix'

- ```
SELECT AVG(prix) AS prix_moyen, console FROM jeux_video GROUP BY console
```

- permet d'avoir un prix moyen des 'prix' pour chaque lot de 'console'

- HAVING

- Associé à GROUP BY, permet de filtrer les groupes

- ```
SELECT AVG(prix) AS prix_moyen, console FROM jeux_video GROUP BY console HAVING prix_moyen <= 10
```

- Il existe plusieurs formats de date :
  - DATE format AAAA-MM-JJ
  - TIME format HH:MM:SS
  - DATETIME format AAAA-MM-JJ HH:MM:SS
  - TIMESTAMP format AAAAMMJJHHMMSS
  - YEAR format AA ou AAAA
- Quelques requêtes SQL
  - INSERT INTO matable(pseudo, message, date) VALUES('Enzo', 'Hello message', '2010-04-02 16:32:22')
  - SELECT pseudo, message, date FROM matable WHERE date >= '2016-05-02 00:00:00' AND date <= '2016-05-18 00:00:00'
  - SELECT pseudo, message, date FROM matable WHERE date BETWEEN '2016-05-02 00:00:00' AND '2016-05-18 00:00:00'
  - Utiliser le site <https://dev.mysql.com/doc/refman/5.7/en/date-and-time-functions.html>

- Fonctions de gestion des dates
    - NOW() CURDATE() CURTIME() retournent la date courante pour les formats DATETIME, DATE et TIME  
INSERT INTO matable(pseudo, message, date) VALUES('Enzo', 'Hello message', NOW())
    - Pour extraire des parties de dates : DAY() MONTH() YEAR() HOUR() MINUTE() SECOND()  
  
SELECT pseudo, message, DAY(date) AS jour, MONTH(date) AS mois, YEAR(date) AS annee FROM matable
- Utilisation en PHP
- ```
<?php
    echo $donnees[jour] . '/' . $donnees[mois] . '/' . $donnees[annee] ;
?>
```

- Formater une date avec DATE\_FORMAT

SELECT pseudo, message, DATE\_FORMAT(date, '%d/%m/%Y %Hh%imin%ss') AS date FROM matable

Dans cet exemple les formats utilisés sont :

- %d jour
- %m mois
- %Y année
- %H heure
- %i minute
- %s seconde

date est ici une chaîne de la forme 25/12/2016 20h15min30s

‘Le bon Coin du vélo’ :

Un site permet à des propriétaires  
d'exposer des articles d'occasion de  
vélo

id	nom	tel
1	Albert95	06 11 22 33 44
2	Nemo75	01 25 32 23 45
3	Nanou	
4	Yoan22	06 11 22 11 22

Table  
proprietaires

id	nom	marque	prix	id_client
1	EX bike 700S	Asics	700	1
2	Dérailleur asics 700	Asics	80	1
3	Vélocity 400	Bobike	450	2
4	Brunch 27	Curana	350	4
5	Delta 500	Delta	650	2
6	Delta 600	Delta	700	

Table articles

- Problématique :  
Récupérer la liste des articles avec leur propriétaire
- Solution : utiliser une Jointure

- 2 types de Jointures :
  - Jointure interne : ne sélectionne que les données qui ont une correspondance dans les 2 tables  
Dans l'ex précédent le propriétaire Nanou et l'article Delta 600 se seront pas extraits  
Emploi de WHERE (ancienne méthode) et INNER JOIN
  - Jointure externe : sélectionne aussi les données qui n'ont pas de correspondance
    - Dans la 1<sup>ère</sup> table (celle de 'gauche') : LEFT JOIN
    - Dans la 2<sup>ème</sup> table (celle de 'droite') : RIGHT JOIN



id	nom	tel
1	Albert95	06 11 22 33 44
2	Nemo75	01 25 32 23 45
3	Nanou	
4	Yoan22	06 11 22 11 22

Table proprietaires

id	nom	marque	prix	id_client
1	EX bike 700S	Asics	700	1
2	Dérailleur asics 700	Asics	80	1
3	Vélocity 400	Bobike	450	2
4	Brunch 27	Curana	350	4
5	Delta 500	Delta	650	2
6	Delta 600	Delta	700	

Table articles

- Exemples

- Jointure interne

```
SELECT articles.nom, proprietaires.nom
FROM proprietaires, articles
WHERE articles.id_client = proprietaires.id
```

Avec ALIAS

```
SELECT articles.nom AS nom_art, proprietaires.nom AS nom_prop
FROM proprietaires, articles
WHERE articles.id_client = proprietaires.id
```

Avec INNER JOIN

```
SELECT articles.nom AS nom_art, proprietaires.nom AS nom_prop
FROM proprietaires
INNER JOIN articles
ON articles.id_client = proprietaires.id
```

id	nom	tel
1	Albert95	06 11 22 33 44
2	Nemo75	01 25 32 23 45
3	Nanou	
4	Yoan22	06 11 22 11 22

Table proprietaires

id	nom	marque	prix	id_client
1	EX bike 700S	Asics	700	1
2	Dérailleur asics 700	Asics	80	1
3	Vélocity 400	Bobike	450	2
4	Brunch 27	Curana	350	4
5	Delta 500	Delta	650	2
6	Delta 600	Delta	700	

Table articles

- Exemples

- Jointure externe

JOIN imposé

**LEFT JOIN** : récupérer toute la table de gauche

```
SELECT articles.nom AS nom_art, proprietaires.nom AS nom_prop
FROM proprietaires
LEFT JOIN articles
ON articles.id_client = proprietaires.id
   fournit  NULL Nanou
```

**RIGHT JOIN** : récupérer toute la table de droite

```
SELECT articles.nom AS nom_art, proprietaires.nom AS nom_prop
FROM proprietaires
RIGHT JOIN articles
ON articles.id_client = proprietaires.id
   fournit  Delta 600  NULL
```



Exercice 10

- PHP contient de nombreuses extensions
- La couche Regex, expressions régulières, permet de comparer de façon très puissante des chaînes de caractères par rapport à un modèle
  - Vérification d'un format de date
  - D'une adr mail
  - ...

- Depuis PHP 5, PHP est un langage réellement OBJET
- Définition d'une classe

```
class NomDeLaClasse {  
    // Propriétés  
  
    // Méthodes  
}  
  
$unObjet = new NomDeLaClasse();
```

- Les propriétés  
Doivent commencer par la visibilité public, private, protected

```
class NomDeLaClasse {  
    // Propriétés  
    public $nom="default";  
    private $id=1; // accessible uniquement dans la classe  
    protected $adresse; // accessible a l'interieur et classes filles
```

- Appel de propriété ou méthode
  - A l'intérieur de la classe `$this->`

```
class NomDeLaClasse {  
    // Propriétés  
    public $nom="default";  
    private $id=1;        // accessible uniquement dans la classe  
    protected $adresse;  // accessible a l'interieur et classes filles  
  
    // Méthodes  
    public function xx()  
    {  
        $val = $this->id + 1;  
        return $val;  
    }  
}
```

- Sur un objet `$obj->prop; $obj->xx();`

```
$unObjet = new NomDeLaClasse();  
echo $unObjet->nom;
```

- Constructeur : `public function __construct()`

```
public function __construct($id){
    $this->id = $id;
}
```

- Destructeur : `public function __destruct()`
- Constante de classe `const nom = valeur;`
- Utilisation dans la classe `self::nom`

```
class Ex0 {
    const DEFAULT_DATE = '2016-12-01';

    public function getDefaultDate(){
        return self::DEFAULT_DATE;
    }
}
```

Utilisation en dehors `nomClasse::nom`

```
$date = Ex0::DEFAULT_DATE;
```

- Définition d'une méthode

```
class Telephone {  
    public function quiSuisJe(){  
        echo "<br/>Je suis un telephone";  
    }  
}
```

- Définir des accesseurs (getters et setters) pour accéder et protéger les données

```
class Membre  
{  
    private $pseudo;  
    private $email;  
    private $signature;  
    private $actif;  
  
    public function getPseudo()  
    {  
        return $this->pseudo;  
    }  
  
    public function setPseudo($nouveauPseudo)  
    {  
        // Vérifier si le nouveau pseudo n'est ni vide ni trop long  
        if (!empty($nouveauPseudo) AND strlen($nouveauPseudo) < 15)  
        {  
            // Ok, on change son pseudo  
            $this->pseudo = $nouveauPseudo;  
        }  
    }  
}
```

- Attributs et méthodes statiques  
L'effet 'static' : l'information ou le comportement appartiennent à la classe et sont indépendants des objets

```
class Ex1 {  
    static private $count;  
  
    public function __construct(){  
        $this::$count ++;  
    }  
    public function __destruct(){  
        $this::$count --;  
    }  
    static public function getCount(){  
        return self::$count;  
    }  
}
```

```
$ob1Ex1 = new Ex1();  
$ob2Ex1 = new Ex1();  
echo "<br/>getCount = " . Ex1::getCount();
```

Etre vigilant sur `$this::$val`    `self::$val`



- Héritage

```
class Telephone {  
    public function quiSuisJe(){  
        echo "<br/>Je suis un telephone";  
    }  
}  
class Smartphone extends Telephone {  
    public function quiSuisJe(){  
        echo "<br/>Je suis un telephone intelligent";  
    }  
}  
class SmartphonePlus extends Telephone {  
    public function quiSuisJe(){  
        echo parent::quiSuisJe() . " plus intelligent";  
    }  
}
```

```
$phone = new Smartphone();  
$phone->quiSuisJe();  
$phone = new SmartphonePlus();  
$phone->quiSuisJe();
```

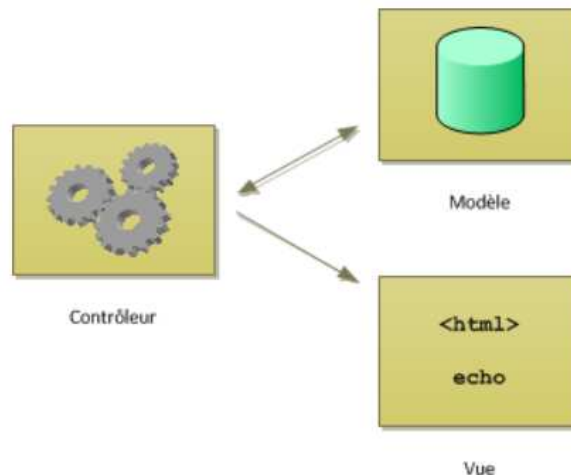
Je suis un telephone intelligent  
Je suis un telephone plus intelligent

## Exercice 11

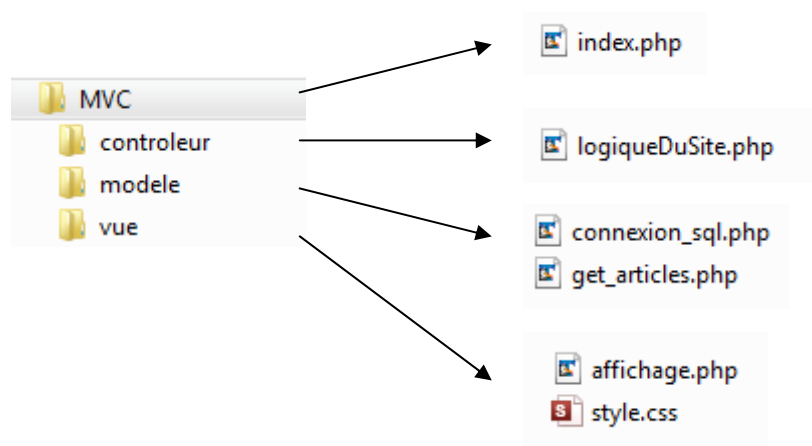


- Autres notions non abordées
  - Mot clé final pour empêcher la surcharge
  - `Include_once('Maclasse.php');` pour inclure un fichier php
  - Interfaces
  - La notion de trait
  - Namespace – espace de nom
  - `__clone()`

- L'architecture MVC permet d'organiser le source d'un site Web
  - Modèle : couche d'accès aux données du site. Requêtes SQL, accès aux fichiers ... Contient la couche métier.
  - Vue : représentation des données. Ne fait pas ou peu de calcul. Html, css
  - Contrôleur : fait le lien entre Modèle et Vue. Contient des algorithmes, prend des décisions. PHP.  
Chef d'orchestre. Le visiteur accède au site par la page Contrôleur.



- Parcourir l'exemple du répertoire MVC  
Recopier le répertoire MVC dans XAMP/htdocs
- Puis localhost/MVC dans le navigateur Web




- Votre site est prêt en local avec l'environnement XAMPP, il faut le copier vers un site Web.

Les étapes :

- Choisir un nom de domaine

[www.monsite.com](http://www.monsite.com)



Sous domaine

De nombreux sites permettent de vérifier si le nom est disponible

Ex [www.lws.fr/nom-de-domaine.php](http://www.lws.fr/nom-de-domaine.php)

- Enregistrer le domaine :
  - Utiliser un registrar : intermédiaire entre l'ICANN et vous
  - Utiliser ce service chez l'hébergeur du site : mieux

- Choisir un hébergeur

Il met a disposition un espace machine et l'espace logiciel adéquat :

- PHP
- MySQL
- Des CMS tels WordPress, Joomla ..
- Un serveur FTP ...

Le moins cher est l'hébergement mutualisé (vous partagez le serveur avec d'autres sites).

Les Stés OVH, Gandi, 1&1 sont des hébergeurs français connus.

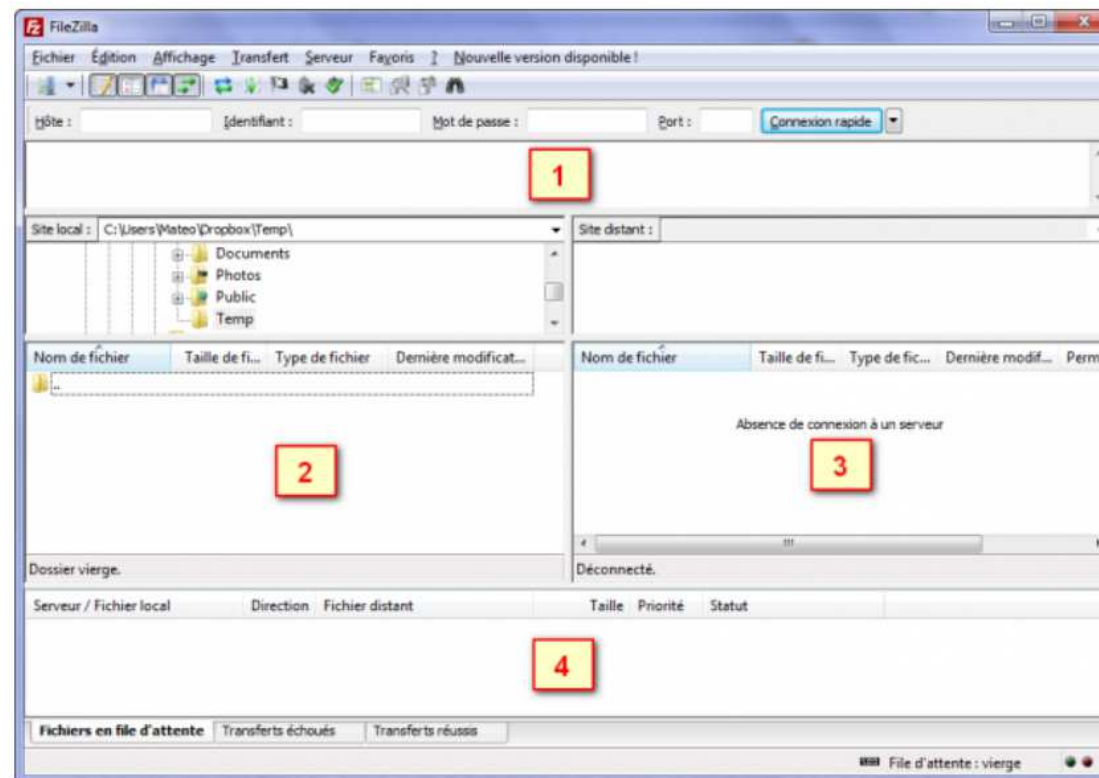
Lorsque l'hébergeur a reçu votre paiement il fournit des identifiants pour :

- Accéder aux services de l'hébergeur
- Identifiants ftp

- Transférer des fichiers

Le transfert de votre station vers le site (et l'inverse si besoin) se fait par FTP.

L'installation sur votre station d'un client FTP est nécessaire, par ex FileZilla.



- Transférer des fichiers (suite)

Copier les fichiers et répertoires depuis votre station vers le site. Généralement le répertoire racine est 'www' (équivalent de htdocs sous XAMPP).

- Le fichier index.html ou index.php sous le répertoire racine sera le point d'entrée du site à partir du nom de domaine [www.monsite.com](http://www.monsite.com) (en fait liste des fichiers visibles dans DirectoryIndex de httpd.conf de Apache)



- Transférer la base de données  
Sur le site, l'hébergeur vous communique les identifiants de la base MySQL ou vous donne les outils pour créer cette base.  
En résultent un nom et un mot de passe.
  - Sur la station de développement lancer phpMyAdmin et extraire (exporter) la base de donnée au format SQL
  - Transporter ce fichier par FTP chez l'hébergeur
  - Ensuite chez l'hébergeur lancer phpMyAdmin et importer le fichier SQL
  - Ne pas oublier de modifier la ligne php de connexion à la base de données  
`$bdd = new PDO('xxxx');`

- Il existe un mécanisme de protection d'accès à un répertoire du site
  - Dans le répertoire à protéger, créer un fichier .htaccess. Il contient

```
AuthName "Page d'administration protégée"  
AuthType Basic  
AuthUserFile "C:\<chemin absolu>\XAMP\htdocs\admin\.htpasswd"  
Require valid-user
```

- Créer le fichier .htpasswd  
Chaque ligne contient les ref utilisateur sous la forme  
nom:<mot de passe>

Dans l'env XAMPP <mot de passe> est en clair, sur site il peut être crypté. Dans ce cas la ligne

```
<?php echo crypt('mot de passe'); ?>
```

fournit la chaîne cryptée