

Programmation Web Dynamique

Pr. Rachid DAKIR

Filières : SMI & IGE

FP OUARZAZATE

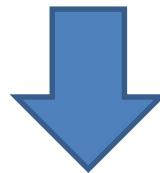
UNIVERSITE IBN ZOHR

Année Universitaire : 2019-2020

PARTIE : I

Prérequis

- Création de sites web.
- Programmation Web statique : HTML, XHTML et CSS
- Programmation Web dynamique coté client : JavaScript (intégration de script dans une page HTML, utilisation de contrôles ActiveX et d'applets Java).



Technologie Web

Plan

Objectifs :

- Les Techniques de Scripting côté Serveur : Programmation PHP ;
- La configuration d'une plateforme de développement Web (serveur Web Apache, base de données MySQL et moteur de script PHP) ;
- La Gestion des sessions utilisateurs et techniques d'authentification ;
- Application : construction d'un site Web dynamique avec base de données.

Plan

Cours en 3 Parties

Chapitre I : Programmation PHP

- ❑ Notions et Vocabulaires de base
- ❑ Les Techniques de Scripting côté Serveur : Programmation PHP.
- ❑ Configuration d'une plateforme de développement Web (serveur Web Apache, base de données MySQL et moteur de script PHP).

Chapitre II : Gestion d'une base de données MySQL

- ❑ Connexion à MySQL.
- ❑ Sélection de la base de données. – Requête sur la base de données. – Exploitation des résultats de la requête.
- ❑ Fermeture de la connexion à MySQL.

❑ Chapitre III : Gestion des sessions utilisateurs et techniques d'authentification ;

- ❑ Application : construction d'un site Web dynamique avec base de données.

Notions et Vocabulaires

Web ?

Système d'information réparti en pages (documents)Web basé la notion **d'hypertexte** et **d'hyperlien** fonctionnant sur Internet permettant de consulter des pages web avec un navigateur.

- Protocole de communications **HTTP,HTTPS**
- Adresse pour identifier des pages (ou site) Web : **URL**
- Les langages pour créer les pages web : **HTML, CSS et Java script**
- Les navigateurs permettent de visualiser les pages Web : **Chrome, firefox, Internet Explorer, Opéra**

Notions et Vocabulaires

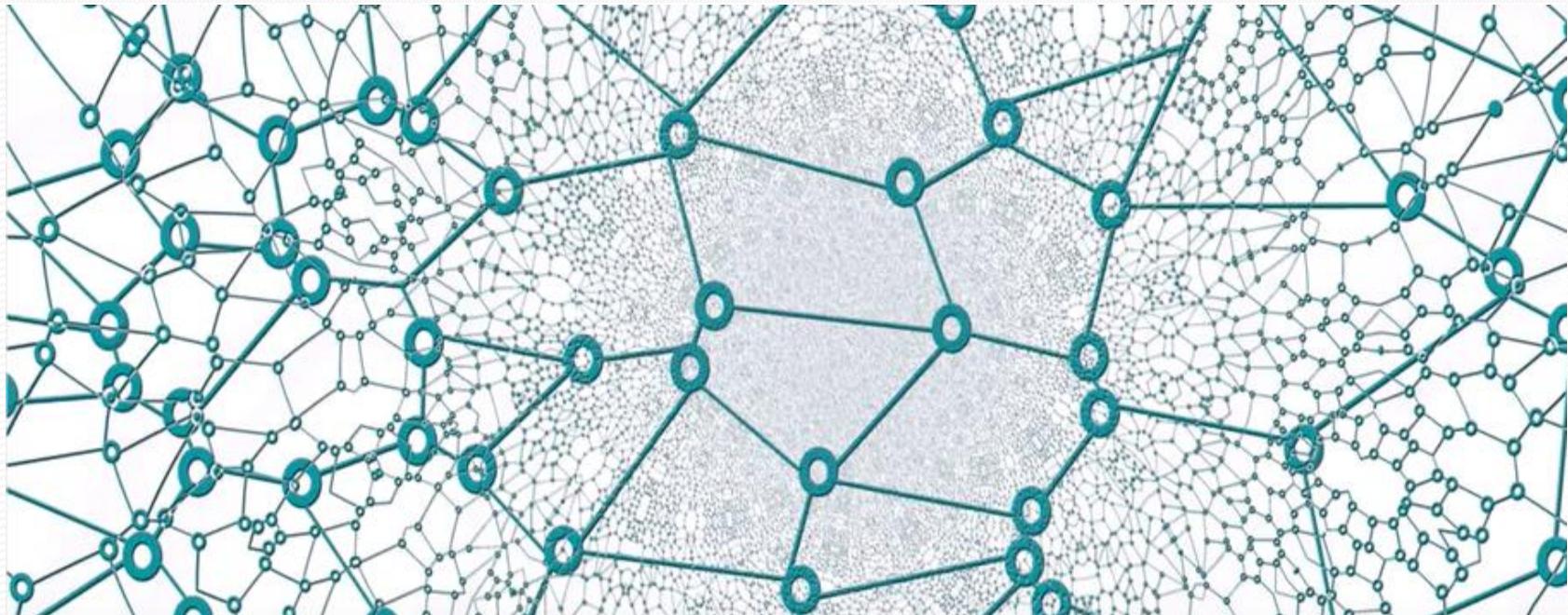
Web ?

- **Hypertexte** : Ensemble de documents contenant des unités d'information liées entre eux par des hyperliens. Ce système permet à l'utilisateur d'aller directement à l'unité qui l'intéresse, à son gré, d'une façon non linéaire
- **Hyperlien** : Référence dans un système hypertexte permettant de passer automatiquement d'un document consulté à un autre document.
- **Navigateur web** : logiciel client HTTP conçu pour accéder aux ressources du web. Sa fonction de base est de permettre la consultation des documents HTML disponibles sur les serveurs HTTP.
- **Page web** : Document destiné à être consulté avec un navigateur web. Une page web est toujours constituée d'une ressource centrale (généralement un document HTML) et d'éventuelles ressources liées automatiquement accessibles (par exemple, des images).

Notions et Vocabulaires

Internet ?

- Un réseau de réseaux (Nombre réseaux connectés entre eux)
- Réseau informatique mondial accessible au public.



Notions et Vocabulaires

Internet ?

- **Protocoles de communication TCP/IP :**

Plusieurs applications qui permettent de partages des informations entre les ordinateurs sur le réseau Internet comme :

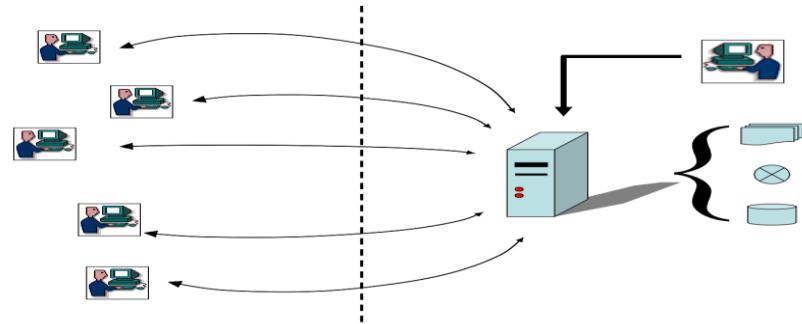
- Courrier électronique (SMTP, POP3 ou IMAP)
- Transfert de fichiers (FTP...)
- Messages instantanés (XMPP/JAPPER=
- World Wide Web (HTTP)

Etc....*

Notions et Vocabulaires

Communication Web ?

- **Architecture Client – Serveur**



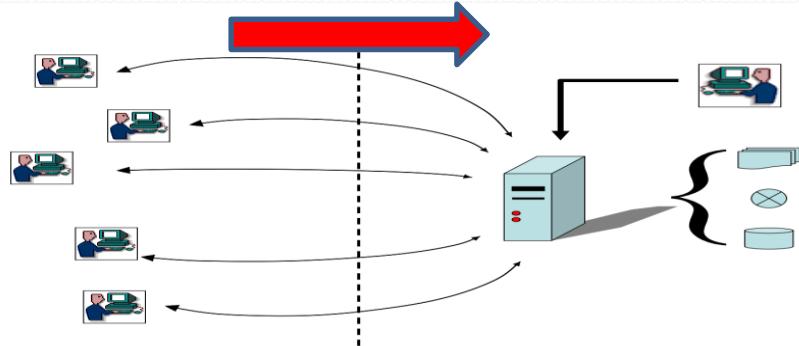
- **Serveur** : Ordinateur sur lequel se trouve une ressource. Joue en premier lieu un rôle de distributeur de fichiers. Autour de ce rôle de serveur de fichiers, il peut rendre diverses sortes de services.
- **Client** : qui demande une ressource via une interface utilisateur chargée de la présentation de la ressource
- **Protocoles de communication TCP/IP** : Ensemble des règles de communication sur internet et se base sur la notion adressage IP, c'est-à-dire le fait de fournir une adresse IP à chaque machine du réseau afin de pouvoir acheminer des paquets de données.

Notions et Vocabulaires

Communication Web ?

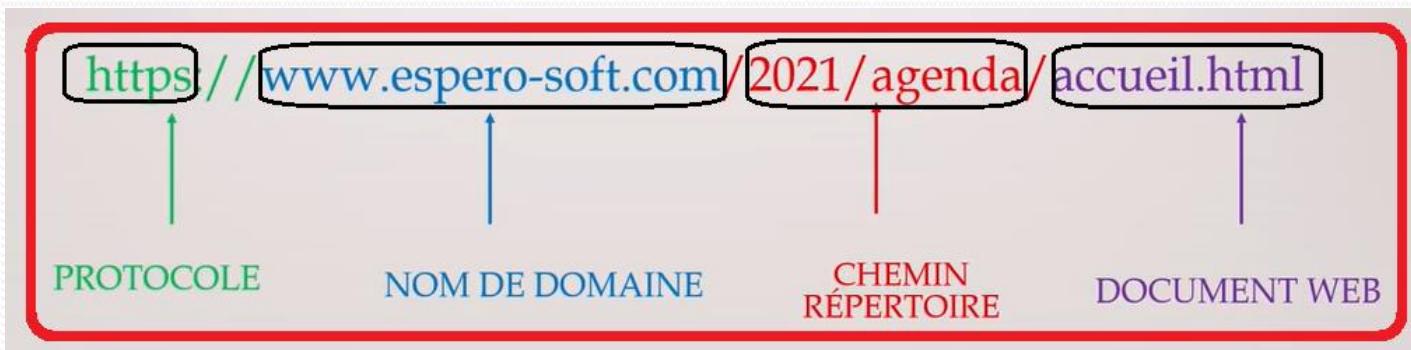
C'est quoi une requête ?

Une **requête HTTP** est une demande effectuée par le navigateur WEB (ex: Internet Explorer, Firefox, Mozilla,...) au **serveur HTTP** lorsqu'il souhaite **télécharger une page WEB**.



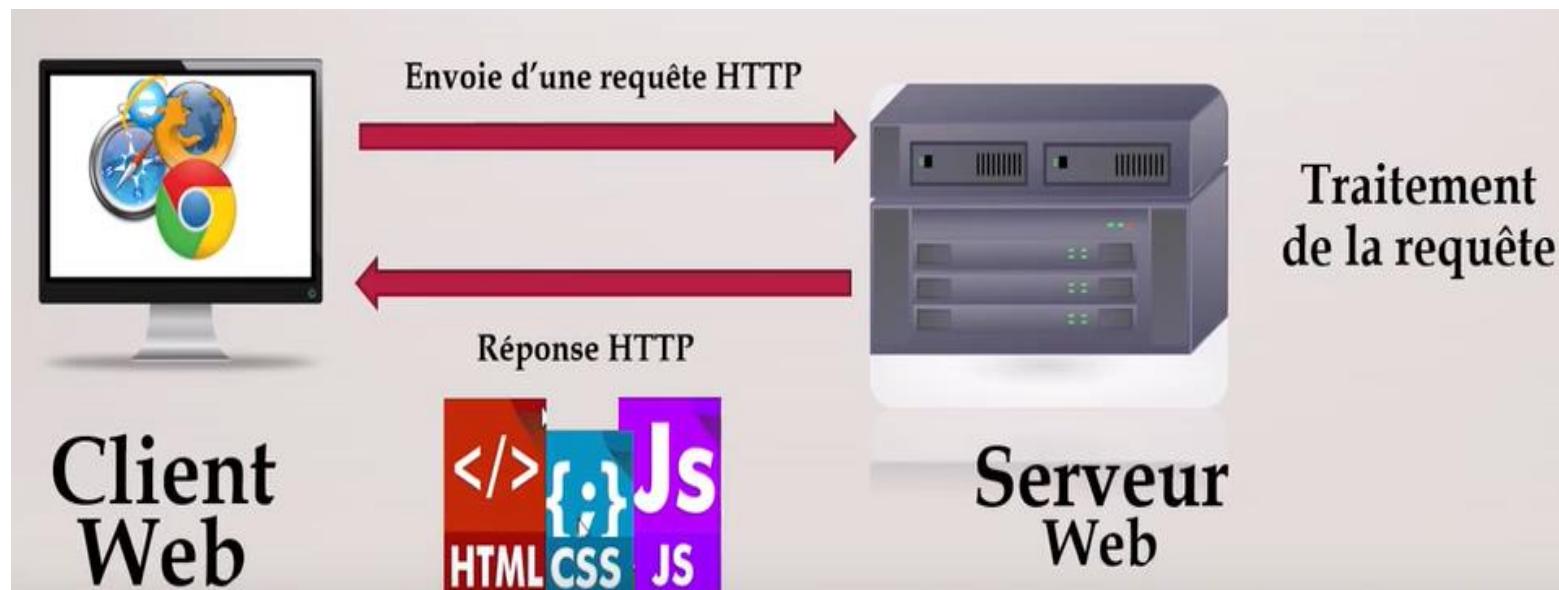
Se présente sous forme d'une adresse URL.

- Une URL complète est également composé au minimum de trois parties



Communication Web ?

- Architecture Client – Serveur



Notions et Vocabulaires

Communication Web ?

Langage Web Client

- ❑ **HTML** : Langage de balisage utilisé pour structurer et donner un sens au contenu web. Par exemple : définir des paragraphes, entêtes et tables de données ou encore intégrer des images ou des vidéos dans une page.
- ❑ **CSS** : Langage de règles de style utilisé pour appliquer un style au contenu HTML. Par exemple : en modifiant la couleur d'arrière-plan ou les polices, ou en disposant le contenu en plusieurs colonnes.
- ❑ **JavaScript** : Langage de programmation qui permet par exemple de créer du contenu mis à jour de façon dynamique, de contrôler le contenu multimédia, afficher du contenu mis à jour à des temps déterminés, des cartes interactives, des animations 2D/3D, des menus vidéo défilants, etc...

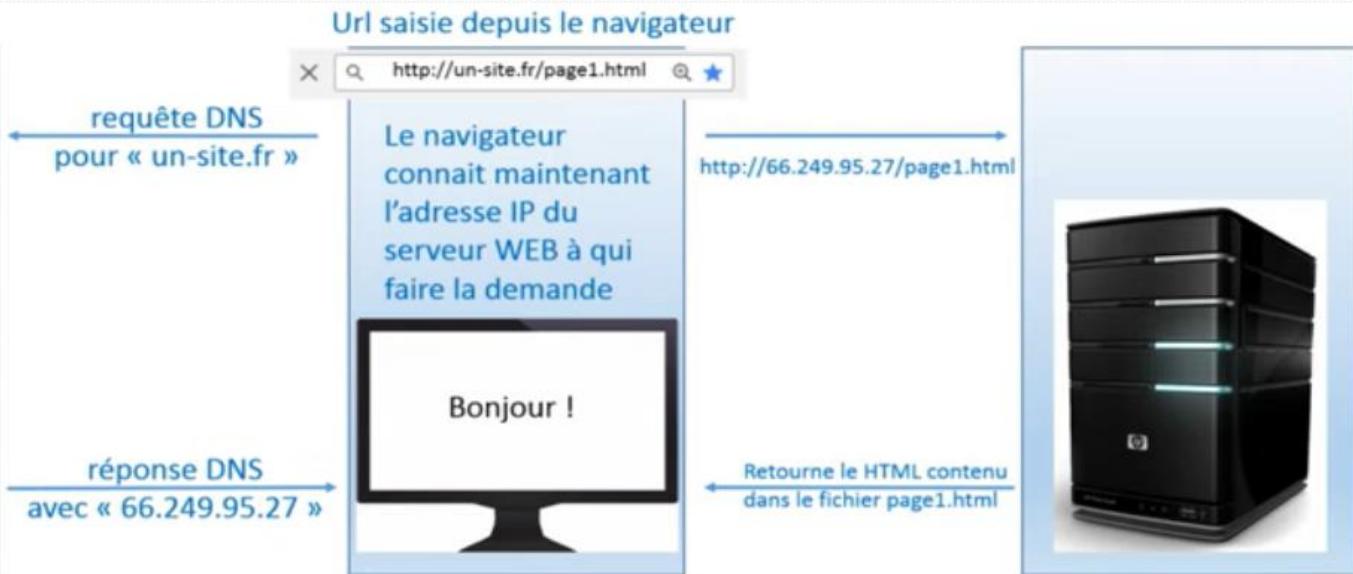


Principe du fonctionnement Web

Demande d'une page web ne contenant que du html

**** **Serveur Web installé sans autre service** ****

Statique



Serveur DNS (port 53)
(celui du FAI, ou à défaut, un autre situé quelque part dans le monde)

Principe du fonctionnement Web

Demande d'une page web contenant du code PHP
**** **Serveur Web installé sans autre service** ****

Url saisie depuis le navigateur

X http://un-site.fr/page1.php



Poste client
utilisant un navigateur

http://un-site.fr/page1.php

le serveur envoie le contenu du fichier,
à savoir du **HTML**



Serveur WEB
(Apache, Nginx, IIS,...)

Principe du fonctionnement Web

Demande d'une page web ne contenant que du code PHP
**** Serveur Web installé avec un interpréteur PHP****

Dynamique



Principe du fonctionnement Web

Demande d'une page web ne contenant que du code PHP avec une requête SQL

**** Serveur Web installé avec un interpréteur PHP et SGBD ****

Url saisie depuis le navigateur

X http://un-site.fr/page1.php

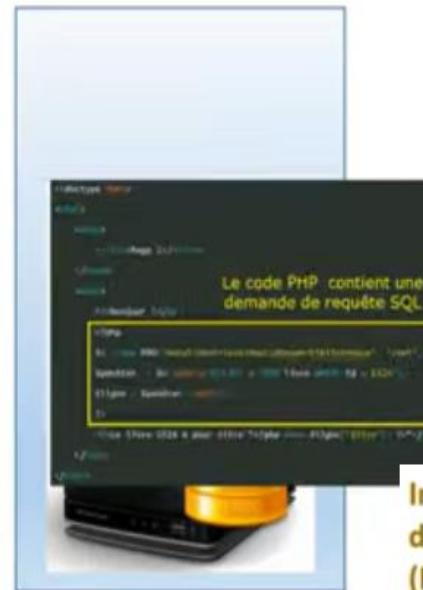
http://un-site.fr/page1.php

Le navigateur ne reçoit que du
HTML. De toutes façons, il ne
comprend rien d'autre (ou presque...)



Poste client
utilisant un navigateur

le serveur comprend
qu'on lui demande de
servir le fichier
« page1.php »



Serveur WEB
(Apache, Nginx, IIS,...)

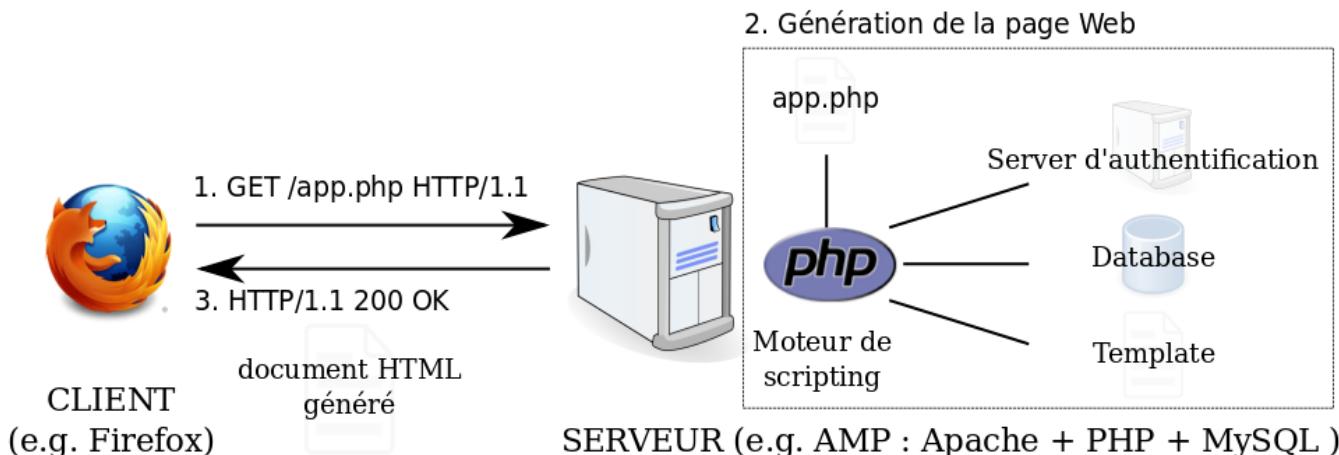
Installation d'un serveur
de base de données
**(Mysql, Postgres, Oracle,
SQLServer, ...).** Possible
sur la même machine ou
sur une autre

Principe du fonctionnement Web

Dynamique

Les pages web écrites en langage HTML sont **statique** : Dans un site web statique la forme et le contenu de la page sont définis par l'auteur du site

- ✓ Un **site web dynamique** est un site web dont les pages peuvent être générées *dynamiquement*, soit à la demande du client et le contenu peut être obtenu en combinant l'utilisation d'un **langage de scripts** ou de **programmation** et une **base de données**



Avec le web dynamique, nous allons avoir des pages qui pour une même adresse peuvent prendre des formes différentes selon les actions de l'utilisateur.

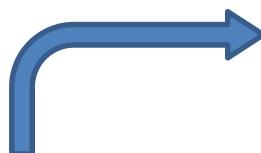
Principe du fonctionnement Web

Langage Web Serveur

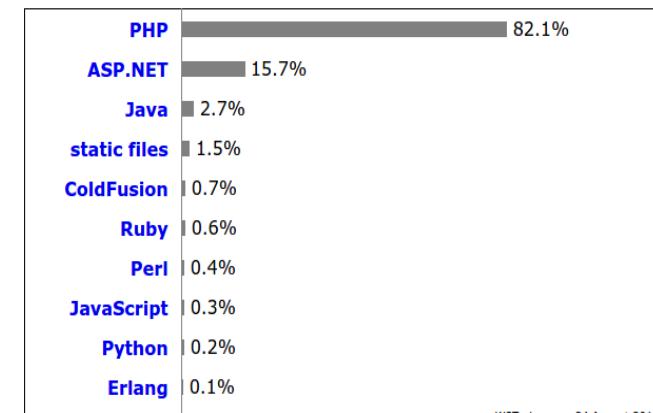
Web Dynamique

Programmes qui vont générer automatiquement du code HTML et constitués de morceaux de script, inclus dans du code HTML. Lorsque le fichier écrit en langage web serveur est interprété, les morceaux de scripts affichent les parties variables de la page. Le code HTML reste inchangé : il représente les parties fixes.

- PHP** : Langage populaire, s'intègre très facilement avec Apache (libre)
- ASP et ASP.NET** : langage destiné à être utilisé avec IIS (Microsoft, commercial)
- ColdFusion** (Adobe, commercial)
- JSP (Java Server Pages)** : langage qui permet de mêler instructions Java et code HTML
- Servlets Java** : Véritables programmes Java, plutôt pour les applications complexes côté serveur



Il existe d'autres langages qui permettent de construire des applications web dynamiques



Principe du fonctionnement Web

Accès aux bases de données

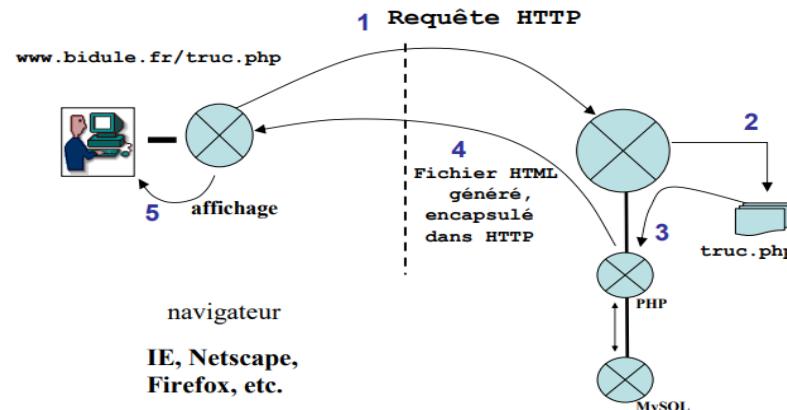
Web Dynamique

- ↳ Assurer la cohérence, le stockage et l'interrogation des données dans une base de données.
- Système propriétaire : Oracle Database, Microsoft SQL Server, DB2, MaxDB, 4D, dBase, Informix, Sybase
- Système libre : MySQL, PostgreSQL, MariaDB, Firebird, Ingres, HSQLDB, Derby, Apache Derby
- Orienté objet : ZODB, db4o
- Embarqué : SQLite, Berkeley DB
- NoSQL : Cassandra, Redis, MongoDB, SimpleDB, BigTable, CouchDB, Couchbase, HBase, LevelDB,
- Autre système : Access, OpenOffice.org Base, FileMaker, HyperFileSQL, Paradox, Neo4j, Riak, Voldemort

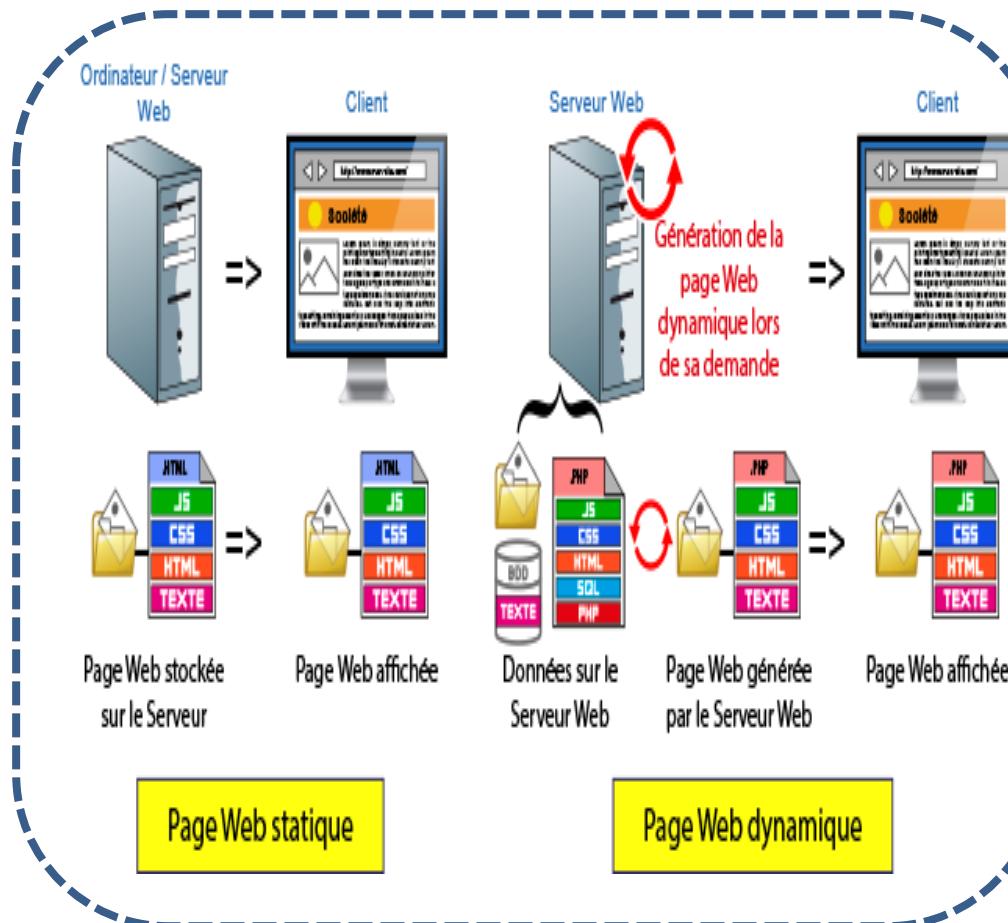
Parmi les SGBD les plus connus:



Oracle, MySQL, PostGreSQL et SQL Server



Web Statique && Web Dynamique?



	Pages Web statiques	Pages Web dynamiques
Présentation	Les pages Web statiques resteront les mêmes jusqu'à ce que quelqu'un le modifie manuellement, à moins que quelqu'un le modifie.	Les pages Web dynamiques sont comportementales et ont la capacité de produire un contenu distinct pour différents visiteurs.
Base de données	N'utilise pas de bases de données	Une base de données est utilisée.
Temps de chargement de la page	Demande moins de temps	Demande plus de temps
Changement d'information	Se produit rarement	Fréquemment
Complexité	Simple à concevoir.	Compliqué à construire.

Editeurs de programmation Web ?

Il existe plusieurs éditeurs de code de programmation Web comme:

➤ Sublime Text

➤ Notepad++

➤ Brackets

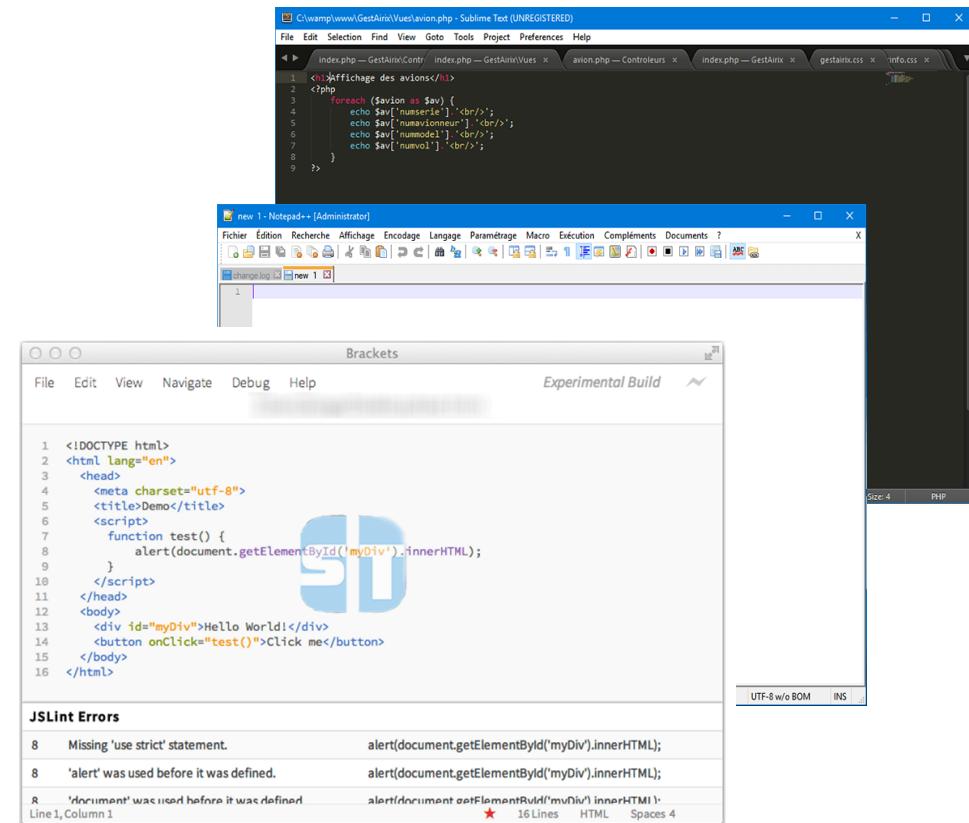
➤ Ultra Edit

➤ jEdit

➤ Komodo

➤ TextWrangler

➤ Dreamweaver



Principe du fonctionnement Web

Comment vais-je faire pour créer un site web s'il faut un serveur Web

✓ S. Web (OS + HTTP)

✓ Interpréteur PHP

✓ SGDB



Differentes solutions :

Vraie Machine

- ❖ Système Linux
- ❖ Service Apache
- ❖ Interprète PHP
- ❖ SGBD



Machine virtuelle

- ❖ Système Linux
- ❖ Service Apache
- ❖ Interprète PHP
- ❖ SGBD



Ex : WAMP-Server



Espace web

- ❖ Exemple : free.fr



Nécessite
des compétences spécifiques

Nécessite
Aucune compétence particulière

Programmation Web Dynamique

Pr. Rachid DAKIR

Filières : SMI & IGE

FP Ouarzazate

UNIVERSITE IBN ZOHR

Année Universitaire : 2019-2020

PARTIE : II

PHP - Les fonctionnalités du langage

- PHP est un langage de scripts libre principalement utilisé pour être exécuté par un serveur Web.



C'est un langage que seuls les serveurs comprennent et qui permet de rendre votre site dynamique.

- Langage Interprété
- Syntaxe plus proche de C et Perl
- Spécialisé dans la génération des fichiers et des documents
- Les fichiers sont enregistrés avec l'extension .php et permettent d'intégrer :
 - Code html
 - Code php entre les balises <?php et ?>
- Le rôle de PHP est justement de générer du code HTML.

- code HTML alterné au code PHP
- code PHP qui génère le code HTML

Utilisation :

- Grand succès
- Utilisation par de très grands sites avec beaucoup de code libre disponible.
- Des dizaines de millions de sites Web l'utilisent à travers le monde comme : Le Monde, Facebook ou Yahoo

PHP - Les fonctionnalités du langage

Imbrication de PHP dans le HTML

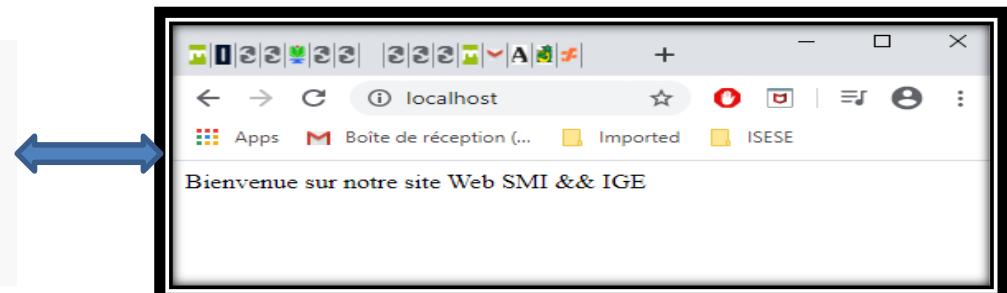
A

```
<html> <head>
<title> Bienvenue sur notre site
Web SMI && IGE</title>
</head>
<body>
<?php echo " Bienvenue sur notre
site Web SMI "; ?>
</body> </html>
```

B

```
<?php echo "<html>";
echo "<html> <head> <title> Mon premier
php
</title> </head> <body>";
echo "Bienvenue sur notre site Web IGE ";
echo "</body></html>";
?>
```

```
<html> <head>
<title> Mon premier php </title>
</head>
<body> Bonjour </body>
</html>
```



Syntaxe :

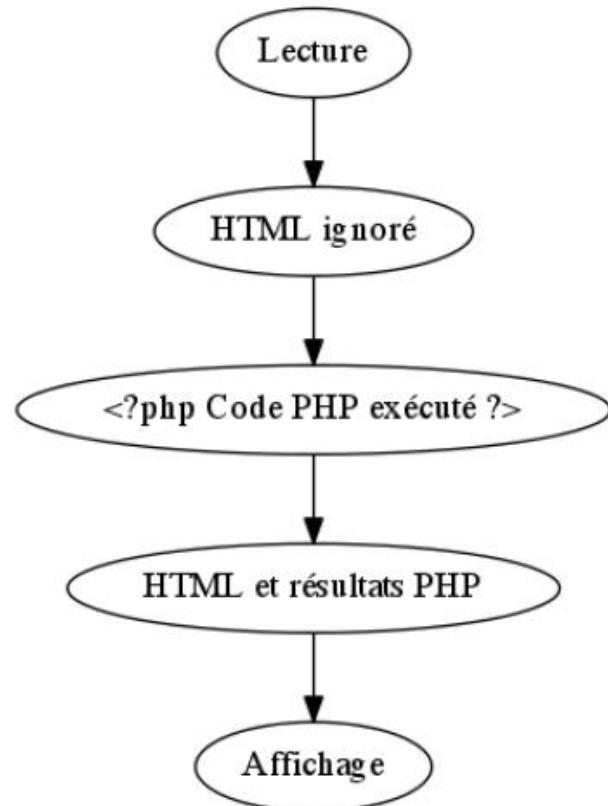
Une syntaxe PHP se termine TOUJOURS par un point-virgule, si vous l'oubliez vous verrez apparaître une PARSE ERROR lors de l'exécution de votre fichier.

PHP - Les fonctionnalités du langage

Interpréteur PHP

lit un fichier source .php puis génère un flux de sortie avec les règles suivantes :

- ❖ Toute ligne située à l'extérieur d'un bloc PHP (entre **<?php et ?>**)
est recopiée inchangée dans le flux de sortie
- ❖ Le code PHP est interprété et génère éventuellement des résultats intégrés eux aussi au flux de sortie
- ❖ Les erreurs éventuelles donnent lieu à des messages d'erreurs qu'on retrouve également dans le flux de sortie (selon la configuration du serveur)
- ❖ Une page html pure sauvegardée avec l'extension **.php** sera donc non modifiée et renvoyée telle quelle . . .



PHP - Les fonctionnalités du langage

Commentaires, Variables, chaînes et concaténation

A. Commentaires, variables

- Les commentaires peuvent se présenter sous deux formes :

```
<?php  
//ceci est un commentaire  
/* ceci est un commentaire */  
?>
```

- Déclaration d'une variable

```
<?php $variable = "une variable en PHP"; ?>
```

PHP - Les fonctionnalités du langage

Commentaires, Variables, chaînes et concaténation

A. variables

- Existence de variables, la fonction isset() :

```
<?php  
$a = "une variable en PHP";  
if(isset($a)) echo "la variable a existe";  
unset($a);  
echo "la variable a été supprimée ...";  
?>
```

- Test de variables, la fonction empty() :

```
<?php  
$a = "une variable en PHP";  
if (!empty($a)) echo " La variable existe et elle n'est pas vide !";  
?>
```

PHP - Les fonctionnalités du langage

Commentaires, Variables, chaînes et concaténation

A. variables

- Test de variables en PHP 7 avec l'opérateur coalescent

```
<?php
$b = 143;
echo $a ?? 3; // affiche 3
// Récupère la valeur de $a et retourne 3 s'il n'existe pas.
echo $a ?? $b ?? 7; // affiche 143
?>
```

The diagram shows a blue arrow pointing from the code line 'echo \$a ?? 3;' to the expression '\$a ?? 3 = isset(\$a)? 3' enclosed in a purple box.

- Ce qui permet de limiter le recours à isset dans de nombreuses situations comme :

```
<?php
// Récupère la valeur de $_GET['email'] et retourne 'nobody' si elle n'existe pas.
$mail = $_GET['email'] ?? 'nobody@null';
// Équivalent à:
$mail = isset($_GET['email']) ? $_GET['email'] : nobody@null';
// Coalescing ?? peut être chainé :
// On renvoie la première valeur définie parmi // $_GET['email'], $_POST['email'], et
// 'nobody@null.com'.
$mail = $_GET['email'] ?? $_POST['email'] ?? 'nobody@null';
echo "$mail\n";
```

PHP - Les fonctionnalités du langage

Commentaires, Variables, chaînes et concaténation

A. variables

➤ Portée des variables :

- Par défaut, toutes les variables sont locales
- Leur portée se réduit à la fonction ou au bloc de leur déclaration
- Pour déclarer une variable globale, on peut utiliser le tableau `$_GLOBALS[]`

```
<?php $_GLOBALS['MaVar']="Bonjour"; ?>
```

□ Constantes

➤ Déclaration

```
<?php  
define("USER", "admin");  
echo USER; // Notez l'absence de $ ici  
?>
```

PHP - Les fonctionnalités du langage

Commentaires, Variables, chaînes et concaténation

A. variables

➤ Typage en PHP

□ **gettype() renvoie le type de la variable comme :**

Integer, double, string, array, object, Class,...etc

□ **settype() change le type d'un élément**

```
<?php  
$a=3.5;  
settype($a,"integer");  
echo "le contenu de la variable a est ".$a;  
?>
```

□ **Fonctions de test**

`is_int()`, `is_long()`, `is_double()`, `is_array()`, `is_object()` et `is_string`

PHP - Les fonctionnalités du langage

Commentaires, Variables, chaînes et concaténation

B. Chaines, concaténation

Guillemets ou Cotes pour les chaines

L'affichage des variables combinées à des chaînes de caractères peut se faire de plusieurs manières en utilisant les cotes simples (' ') ou les doubles cotes ("").

```
<?php  
$var="Hello PHP";  
$machaine="le contenu de \$var est $var<br>";  
echo $machaine;  
//ou avec des '' :  
$mystring='le contenu de $var est '.$var;  
echo $mystring;  
?>
```

1
<?php
\$nom = 'visiteur';
echo 'bonjour '.\$nom;
?>

2
<?php
\$nom = 'visiteur';
echo "bonjour \$nom";
?>

3
<?php
\$nom = 'visiteur';
echo 'bonjour \$nom';
?>

bonjour visiteur
bonjour visiteur
bonjour \$nom

N.B : La concaténation se fait à l'aide de « . »

PHP - Les fonctionnalités du langage

Commentaires, Variables, chaînes et concaténation

B. Chaînes, concaténation

- Longueur d'une chaine :

```
<?php int lg=strlen($chaine); ?>
```

- Accéder au caractère i d'une chaine :

```
<?php echo $chaine[i]; ?>
```

- Mettre en majuscules/minuscules

- ❑ **strtoupper()** : pour obtenir des majuscules
 - ❑ **strtolower()** : pour mettre en minuscules
 - ❑ **ucfirst()** : pour mettre en majuscule la première lettre d'une chaine
 - ❑ **ucwords()** : pour mettre en majuscule la première lettre de chaque mot dans une chaine

PHP - Les fonctionnalités du langage

Commentaires, Variables, chaînes et concaténation

B. Chaines, concaténation

➤ Recherche de sous-chaines ou de motifs dans une chaîne :

- stristr()
- ereg() ou eregi()

```
<?php
$AGENT=$_SERVER['HTTP_USER_AGENT'];
echo $AGENT;
echo("\n<P>");
if (stristr($AGENT,"MSIE"))
echo "Vous semblez utiliser Internet Explorer
!</b>";
elseif (ereg("Firefox",$AGENT))
echo "Vous semblez utiliser Firefox !</b>";
elseif (ereg("chrome",$AGENT))
echo "Vous semblez utiliser Chrome !</b>";
```

PHP - Les fonctionnalités du langage

Les tableaux en PHP

A. Introduction

Il existe deux type de tableaux de variables sous PHP :

- Les tableaux à index numériques (tableaux numérotés) dans lesquels l'accès à la valeur de la variable passe par un index numérique

Exemple : \$tableau[0], \$tableau[1], \$tableau[2],.....

- Les tableaux à index associatifs (ou tableaux associatifs) dans lesquels l'accès à la valeur de la variable passe par un index nominatif

Exemple : \$tableau[nom], \$tableau[prénom], \$tableau[adresse],

PHP - Les fonctionnalités du langage

A- Syntaxe (Tableaux associatifs)

```
<?php  
$tableau[variable2] donnera valeur2  
//Tableau à index numéroté  
$tableau = array (valeur0,valeur1,valeur2, ...);  
?>
```

//Accès à chacune des valeurs :
\$tableau[0] donnera valeur0;
\$tableau[1] donnera valeur1;
.....

\$prenoms = **array** ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît')

A'- Syntaxe (Tableaux numérotés)

```
<?php  
$tableau = array (variable1 => valeur1, variable2 => valeur2, ...);  
//Accès à chacune des valeurs  
$tableau[variable1] donnera valeur1  
$tableau[variable2] donnera valeur2  
?>
```

//Accès à chacune des valeurs
\$tableau[variable1] donnera valeur1
\$tableau[variable2] donnera valeur2

\$jours=**array**("Lu"=>"Lundi","Ma"=>"Mardi","Me"=>"Mercredi","Je"=>"Jeudi","Ve"=>"Vendredi", "Sa"=>"Samedi","Di"=>"Dimanche");

PHP - Les fonctionnalités du langage

B- Parcours des tableaux

La boucle for

```
<?php
// On crée notre array $prenoms
$prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');
// Puis on fait une boucle pour tout afficher :
for ($numero = 0; $numero < 5; $numero++)
{ // affichera $prenoms[0], $prenoms[1]
    echo $prenoms[$numero] . '<br />'.
}
?>
```

PHP - Les fonctionnalités du langage

B- Parcours des tableaux

foreach

PHP intègre une structure de langage qui permet de parcourir un à un les élément d'un tableau : **foreach()**.

```
//foreach  
$tableau = array (valeur0,valeur1,valeur2, ...);  
foreach ( $tableau as $valeur )  
{ //Appeler ici la valeur courante par $valeur }
```

→ <?php
\$jours=array("Lu"=>"Lundi","Ma"=>"Mardi","Me"=>"Mercredi","Je"=>"Jeudi",
"Ve"=>"Vendredi",
"Sa"=>"Samedi","Di"=>"Dimanche");
foreach(\$jours as \$key=>\$val)
echo \$key." ".\$val."
\n";?>

Lu Lundi
Ma Mardi
Me Mercredi
Je Jeudi
Ve Vendredi
Sa Samedi
Di Dimanche

→ <?php
\$prenoms = ['François', 'Michel', 'Nicole', 'Véronique', 'Benoît'];
foreach(\$prenoms as \$element)
{ echo \$element . '
'; }
?>

François, Michel, Nicole, Florian, Benoît,

PHP - Les fonctionnalités du langage

C- Affichage avec print_r() :

```
<?php  
print_r($jours);  
?>
```

Résultat brut html :

```
Array  
(  
[Lu] => Lundi  
[Ma] => Mardi  
[Me] => Mercredi  
[Je] => Jeudi  
[Ve] => Vendredi  
[Sa] => Samedi  
[Di] => Dimanche  
)
```

D- array_walk :

Donne même résultat qu'avec la boucle foreach

```
<?php array_walk($jours,'aff_tab'); ?>
```

En ayant défini au préalable :

```
<?php  
function aff_tab($val, $key){  
echo "$key-$val<br/>\n";}  
?>
```

PHP - Les fonctionnalités du langage

E- Recherche dans un tableau

in_array() : permet de déterminer si une valeur existe dans le tableau. Elle retourne donc une valeur booléenne (True ou False).

```
$resultat=in_array ( nom_de_la_valeur, tableau) ;
```

```
<?php  
$tableau2 = array ('prenom' =>'Tarak','nom' =>'Joulak','ville' =>'Tunis') ;  
if (in_array("Tarak", $tableau2))  
{ echo 'La valeur "Tarak" existe bien dans le tableau'; }  
else { echo 'La valeur "Tarak" ne se trouve pas dans le tableau'; }  
?>
```

PHP - Les fonctionnalités du langage

E- Recherche dans un tableau

array_key_exists() : permet de vérifier si dans un tableau associatif une variable associative (clef) existe ou non. Elle retourne donc une valeur booléenne (True ou False).

→ \$resultat= **array_key_exists('nom_de_la variable', tableau_associatif)** ;

```
<?php
$tableau2 = array ('prenom' =>'Tarak','nom' =>'Joulak','ville' =>'Tunis') ;
if (array_key_exists("nom", $tableau2))
{ echo 'La variable "Nom" se trouve dans le tableau et a pour valeur: '.$tableau2['nom'];}
else
{ echo 'La variable "Nom" ne se trouve pas dans le tableau'; }
?>
```

PHP - Les fonctionnalités du langage

E- Recherche dans un tableau

array_search : fonctionne comme **in_array** : il travaille sur les valeurs d'un tableau.

Si il a trouvé la valeur, **array_search** renvoie la clé correspondante (c'est-à-dire le numéro si c'est un tableau numéroté, ou le nom de la variable si c'est un tableau associatif). Si il n'a pas trouvé la valeur, **array_search** renvoie false (comme **in_array**).

→ \$resultat=array_search (nom_de_la_valeur, tableau) ;

```
<?php
$Tableau2 = array ('prenom' =>'Tarak','nom' =>'Joulak','ville' =>'Tunis') ;
if ($position = array_search("Tunis", $Tableau2))
{ echo '"Tunis" se trouve en position ' . $position . '<br />';
} else
{ echo '"Tunis" ne se trouve pas dans le tableau.';
?>
```

PHP - Les fonctionnalités du langage

Variables serveur

PHP propose l'accès à toute une série de variables comme les en-têtes, dossiers et chemins des scripts, sans que vous ayez à les créer, on les appelle les variables serveur. Ces variables sont créées par le serveur web.

Le tableau `$_SERVER` permet d'accéder à ces variables. Si la directive `register_globals` de PHP.INI est active, alors ces variables seront aussi rendues directement accessible dans le contexte d'exécution global c'est à dire séparément du tableau `$_SERVER`.

Description de certaines variables :

- `$_SERVER['SERVER_NAME']` : Le nom du serveur hôte qui exécute le script
- `$_SERVER['PHP_SELF']` : Le nom du fichier du script en cours d'exécution
- `$_SERVER['DOCUMENT_ROOT']` : Racine du serveur
- `$_SERVER['REMOTE_ADDR']` : Adresse IP du client
- `$_SERVER['QUERY_STRING']` : Liste des paramètres passés au script
- `$_SERVER['REQUEST_METHOD']` : Méthode d'appel du script

N.B : La fonction PHP permettant d'éditer l'intégralité des variables prédéfinies disponibles est `phpinfo()`

PHP - Les fonctionnalités du langage

Variables, chaînes et concaténation

Remarques !!!!

Lors de l'usage des ' pour l'affichage d'une chaîne de caractère contenant des apostrophes, vous devez impérativement faire précéder ces apostrophes d'antislash. De la sorte l'apostrophe ne sera pas confondue avec le caractère de fin de la chaîne à afficher. Dans le cas contraire un message d'erreur sera affiché.

Il en va de même lors de l'usage de " pour l'affichage d'une chaîne de caractère contenant au préalable des doubles cotes.

→
`<?php
echo 'vous n\'êtes pas inscrit';
?>`

`<? php
echo"lien.php";
?>`

PHP - Les fonctionnalités du langage

Conditions et boucles.

A. Conditions

La syntaxe de base d'une instruction conditionnelle est la suivante :

```
<?php  
if ($var == 'condition')  
{  
// 'condition vérifiée'  
}  
else  
{  
// 'condition non vérifiée'  
}  
?>;
```

== : strictement égal
!= : différent
> : plus grand que
< : inférieur à
>= : supérieur à
<= : inférieur à
&& ou AND : et
|| : ou
OR : ou
TRUE : 1 ou oui
FALSE : 0 ou non

Les opérateurs de contrôle

PHP - Les fonctionnalités du langage

A'. Conditions

```
<?php  
If ($var == 'condition1')  
{  
// 'condition1 vérifiée';  
}  
elseif ($var == 'condition2')  
{  
// 'condition2 vérifiée';  
}  
... elseif ($var == 'conditionN')  
{  
// 'conditionN vérifiée';  
} ... else  
{  
echo 'Aucune condition n'est vérifiée';  
}  
?>
```

```
<? php  
$montant='100' ;  
if ($montant >=0 && $montant<1000)  
{  
echo ' Votre montant '. $montant. ' est compris  
entre 0 et 1000';  
}  
elseif ($montant >=1000 && $montant<5000)  
{  
echo ' Votre montant '. $montant. ' est compris  
entre 1000 et 5000';  
}  
else  
{  
echo ' Votre montant '.$montant. ' est supérieur  
à 5000';  
}  
?>
```

Exemple

PHP - Les fonctionnalités du langage

A''. Conditions

```
<?php
switch ($variable)
{
    case condition1:
        //Traitement de la condition 1
        break;
    case condition2:
        //Traitement de la condition 2
        break;
    ....
    case conditionN:
        //Traitement de la condition N
        break;
    default:
        //Traitement par défaut
}
?>
```

```
<?php
$variable = 3;
switch ($variable) {
    case 1:
        echo 'La variable a pour valeur 1'; break;
    case 2:
        echo 'La variable a pour valeur 2'; break;
    case 3:
        echo 'La variable a pour valeur 3'; break;
    case 4:
        echo 'La variable a pour valeur 4'; break;
    case 5:
        echo 'La variable a pour valeur 5'; break;
    default:
        echo 'La variable n'appartient pas à l'intervalle [1,5]'; }
?>
```

Exemple

PHP - Les fonctionnalités du langage

- L'instruction ***break***: permet d'arrêter une boucle **for**, **foreach** ou **while** avant son terme normal
- L'instruction ***continue***: n'arrête pas la boucle en cours, mais les instructions situées après **Continue** ne seront pas exécutées.

PHP - Les fonctionnalités du langage

B. Boucles : Itération avec for

```
<?php  
for($i=0; $i != condition ; $i++)  
{  
//Traitements réalisés  
}  
?>
```

```
<?php  
$nbre_maximum = 6;  
//-----DEBUT BOUCLE-----  
for($i=0; $i < $nbre_maximum ; $i++)  
{  
echo $i.' est inférieur à '.  
$nbre_maximum.'  
';  
}  
//-----FIN BOUCLE-----  
echo $i.' est égal à '. $nbre_maximum;  
?>
```

Exemple

PHP - Les fonctionnalités du langage

B. Boucles : Itération avec WHILE

```
<?
While (condition)
{
//Traitements
}
?>
```

```
<?php
$nbre_maximum = 6;
$i = 0; //initialisation de l'indice d'incrémentation
while ($i < $nbre_maximum) //condition
{
echo $i.' est inférieur à '. $nbre_maximum.'  
';
$i++; // $i++ est équivalent à ($i+1)
}
echo $i.' est égal à '. $nbre_maximum;
?>
```

Exemple

PHP - Les fonctionnalités du langage

Fonctions

PHP propose une palette approximative de 2000 fonctions prédéfinies. Toutefois il vous est possible de créer vos propres bibliothèques de fonctions pour vos usages spécifiques, une meilleure lisibilité du code et une réutilisation.

Les fonctions peuvent se distinguer en deux sous groupes :

- Les fonctions qui effectuent un traitement (affichage par exemple)
- Les fonctions qui effectuent un traitement et retournent un résultat

PHP - Les fonctionnalités du langage

Fonctions

```
function nom_de_la_fonction ($paramètres)
{ //traitement sur les paramètres effectué
return ($resultat) ;
}
```

L'appel de la fonction se fait de la manière suivante :



```
$variable = nom_de_la_fonction ($paramètres) ;
```

```
<?php
function afficher_nom_prenom ($nom,$prenom)
{
echo 'Bonjour '.$nom ' '.$prenom ;
}
afficher_nom_prenom ('Tarak','Joulak') ;
echo ' <br>' ;
$nom1='Mourad' ;
$prenom1='Zouari' ;
afficher_nom_prenom ($nom1,$prenom1) ;
?>
```

PHP - Les fonctionnalités du langage

Fonctions

```
<?php
// fonction avec 2 paramètres retourne la somme des deux
paramètres
function Somme($a, $b) {
return $a+$b;
}
$res = Somme(10, 11); echo "$res= ", $res;
//fonction sans paramètre qui affiche "Ceci est un exemple"
function Afficher_message(){
echo "ceci est un message <br />";
}
Afficher_message();
?>
```

PHP - Les fonctionnalités du langage

Fonctions

Valeurs de retour

Il est possible de retourner plusieurs valeurs d'une fonction sous forme d'un tableau. Dans l'appel de cette fonction, il faudra affecter le tableau retourner à la Procédure list() qui prend en paramètre la taille de ce tableau. On affecte à list() le retour de la fonction

```
<?php
function opération($arg1,$arg2){
return array ($arg1+$arg2, $arg1-$arg2, $arg1*$arg2 ) ;
}
$a=5 ; $b=3 ;
list($a1,$a2,$a3)= opération($a,$b) ;
echo " somme : $a1 <br />" ;
echo " soustraction : $a2 <br />" ;
echo " produit : $a3 <br />" ;
?>
```

PHP - Les fonctionnalités du langage

Fonctions

Visibilité de la fonction

Une fonction est utilisable uniquement dans le script où elle est définie. Pour l'employer dans plusieurs scripts, il faut, soit recopier sa définition dans les différents scripts, soit la définir dans un fichier inclus partout où la fonction est nécessaire.

Exemple :

Fichier fonctions.inc contenant des définitions de fonctions :

```
<?php  
function somme($arg1,$arg2){  
    return $arg1+$arg2;  
}  
?>
```

Script utilisant les fonctions définies dans fonctions.inc :

```
<?php  
Include 'fonctions.inc' ; //inclusion du fichier fonctions.inc  
echo somme(3,3) ; //utilisations de la fonction somme  
?>
```

PHP - Les fonctionnalités du langage

Fonctions

Dans le cas suivant un traitement est effectué à l'intérieur de la fonction puis retourné par cette dernière. De ce fait la fonction doit donc être affectée à une variable.

```
<?php
function additionner ($variable1,$variable2)
{
    $total = $variable1 + $variable2;
    return ($total) ;
}
$resultat= additionner (1,2) ;
echo $resultat.' <br>' ;
$var1=6 ;
$var2=7 ;
$resultat= additionner ($var1,$var2) ;
echo $resultat.' <br>' ;
?>
```

PHP - Les fonctionnalités du langage

Librairie de fonctions

Idéalement toutes les fonctions créées devraient être regroupées dans un même fichier créant ainsi une bibliothèque de fonctions. Ce fichier sera appelé à l'intérieur des autres fichiers par le biais de la fonction **include**.

```
<?php
//Ce fichier contiendra l'ensemble des fonctions que
vous développerez
function additionner ($variable1,$variable2)
{
    $total = $variable1 + $variable2;
    return ($total);
}
function afficher_nom_prenom ($nom,$prenom)
{
    echo 'Bonjour '.$nom. ' '.$prenom ;
}
?>
```

Le résultat obtenu sera :
10
Bonjour SMI IGE



```
<?php
Include ("fonction.inc.php") ;
$resultat= additionner (4,6) ;
echo $resultat.' <br>' ;
afficher_nom_prenom ('SMI ','IGE') ;
?>
```

PHP - Les fonctionnalités du langage

Mathématiques (I)

Il existe une miriade de fonctions mathématiques.

abs(\$x) : valeur absolue

ceil(\$x) : arrondi supérieur

floor(\$x) : arrondi inférieur

pow(\$x,\$y) : x exposant y

round(\$x,\$i) : arrondi de x à la ième décimale

max(\$a, \$b, \$c ...) : retourne l'argument de valeur maximum

pi() : retourne la valeur de Pi

Et aussi : **cos, sin, tan, exp, log, min, pi, sqrt...**

PHP - Les fonctionnalités du langage

Mathématiques (I)

```
<?php
$a=5;
$b=8;
$c=6.5;
$x=$a-$b;
$y=3;
//valeur absolue
echo abs($x)."  
";
// arrondi supérieur
echo ceil($c)."  
" ;
// arrondi inférieur
echo floor($c)."  
" ;
// x exposant y
echo pow($x,$y)."  
";
//arrondi de x à la ième décimale
//retourne l'argument de valeur maximum
echo max($a, $b)."  
" ;
// retourne la valeur de Pi
echo pi().";
//Et aussi : cos, sin, tan, exp, log, min, pi, sqrt...
?>
```

Exemple

PHP - Les fonctionnalités du langage

Mathématiques (II)

Nombres aléatoires :

- **rand([\$x],[\\$y])** : valeur entière aléatoire entre 0 et RAND_MAX si x et y ne sont pas définis, entre x et RAND_MAX si seul x est défini, entre x et y si ces deux paramètres sont définis.
- **srand()** : initialisation du générateur aléatoire
- **getrandmax()** : retourne la valeur du plus grand entier pouvant être généré
- L'algorithme utilisé par la fonction **rand()** - issu de vieilles bibliothèques libcs - est particulièrement lent et possède un comportement pouvant se révéler prévisible. La fonction **mt_rand()** équivalente à **rand()** est plus rapide et plus sûre puisque l'algorithme utilisé se base sur la cryptographie.
- En cas d'utilisation de la fonction **mt_rand()**, ne pas oublier d'utiliser les fonctions de la même famille : **mt_rand([\$x[\$y]])**, **mt_srand()** et **mt_getrandmax()**.

PHP - Les fonctionnalités du langage

Mathématiques (I)

```
<?php
$a=5;
$b=8;
$c=6.5;
$x=$a-$b;
$y=8;
//Et aussi : cos, sin, tan, exp, log, min, pi, sqrt...
//défini, entre x et y si ces deux paramètres sont définis.
/* rand : Cette fonction ne génère pas de valeurs sécurisées d'un point de vue cryptographique. */
echo rand($x,$y)."<br>";
// retourne la valeur du plus grand entier pouvant être générél
echo getrandmax()."<br>";
/*L'algorithme utilisé par la fonction rand() - issu de vieilles bibliothèques libcs - est particulièrement lent et possède un comportement pouvant se révéler prévisible. La fonction mt_rand() équivalente à rand() est plus rapide et plus sûre puisque l'algorithme utilisé se base sur la cryptographie.
En cas d'utilisation de la fonction mt_rand(), ne pas oublier d'utiliser les fonctions de la même famille :
mt_rand([$x[$y]]), mt_srand() et mt_getrandmax().
*/
?>
```

Exemple

PHP - Les fonctionnalités du langage

Dates et calendriers

- **Les fonctions de jours, dates et heures** sont incontournables sur Internet et sont indispensables pour la conversion en français des dates fournies par la base de données MySQL qui les code au format anglophone (YYYY-DD-MM hh:mm:ss).
- Quelques fonctions:
- **time()**: retourne le timestamp UNIX de l'heure locale (utilisée pour calculer des durées et déterminer des dates future ou passées).
- **Date()**: retourne une chaîne de caractères contenant la date et/ou l'heure locale au format spécifié.
- **getdate()** , **checkdate(\$month, \$day, \$year)**, **mktime(\$hour, \$minute, \$second, \$month, \$day,\$year)**

PHP - Les fonctionnalités du langage

Dates et calendriers

time()

- Retourne le timestamp de l'instant présent
- Cette valeur n'est affichée
- Sert pour le calcul du temps
- Sert à stocker une date à un seul nombre

```
<?php  
echo "A cet instant précis le timestamp est : ", time(), "<br />";  
echo "Dans 23 jours le timestamp sera : ", time()+23*24*3600, " <br />";  
echo "Il y a 12 jours le timestamp était : ", time()-12*24*3600,"<br />";  
echo "Nombre d'heures depuis le 1/1/1970 = ",round(time()/ 3600),"<br />";  
echo "Nombre de jours depuis le 1/1/1970 = ",round(time()/3600/ 24),"<br />";  
?>
```



A cet instant précis le timestamp est : 1362425510
Dans 23 jours le timestamp sera : 1364412710
Il y a 12 jours le timestamp était : 1361388710
Nombre d'heures depuis le 1/1/1970 = 378452
Nombre de jours depuis le 1/1/1970 = 15769

PHP - Les fonctionnalités du langage

Dates et calendriers

Définir une date mktime()

- Syntaxe:

int mktime(int heure, int minute, int seconde, int mois, int jour, int année,
int été)

La dernier paramètre prend 1 pour l'heure d'hiver et 0 pour l'été.

Exemple d'application

```
<?php
//la fonction mktime()
$timepasse= mktime(12,5,30,5,30,1969);
$timeaujour = time();
$duree = $timeaujour-$timepasse;
echo "Entre le 30/05/1969 à 12h05m30s et maintenant il s'est
écoulé",$duree, " secondes <br/>";
echo "Soit ",round($duree/3600), " heures <br/>";
echo "Ou encore ",round($duree/3600/24)," jours <br/>";
```

PHP - Les fonctionnalités du langage

Dates et calendriers

```
<?php  
  
$timefutur = mktime(12,5,30,12,25,2100);  
$noel = $timefutur-$timeaujour;  
echo "Plus que ",$noel, "secondes entre maintenant et Noël, soit ",  
    round($noel/3600/24)," jours, Patience! <br />";  
//la fonction gmmktime()  
$timepassegmt = gmmktime(12,5,30,5,30,1969);  
echo "Timestamp serveur pour le 30/5/1969= ",$timepasse,"<br />"; echo "Timestamp  
GMT pour le 30/5/1969= ",$timepassegmt,"<br />";  
echo "Décalage horaire = ",$timepasse-$timepassegmt," secondes<br />";  
?>
```



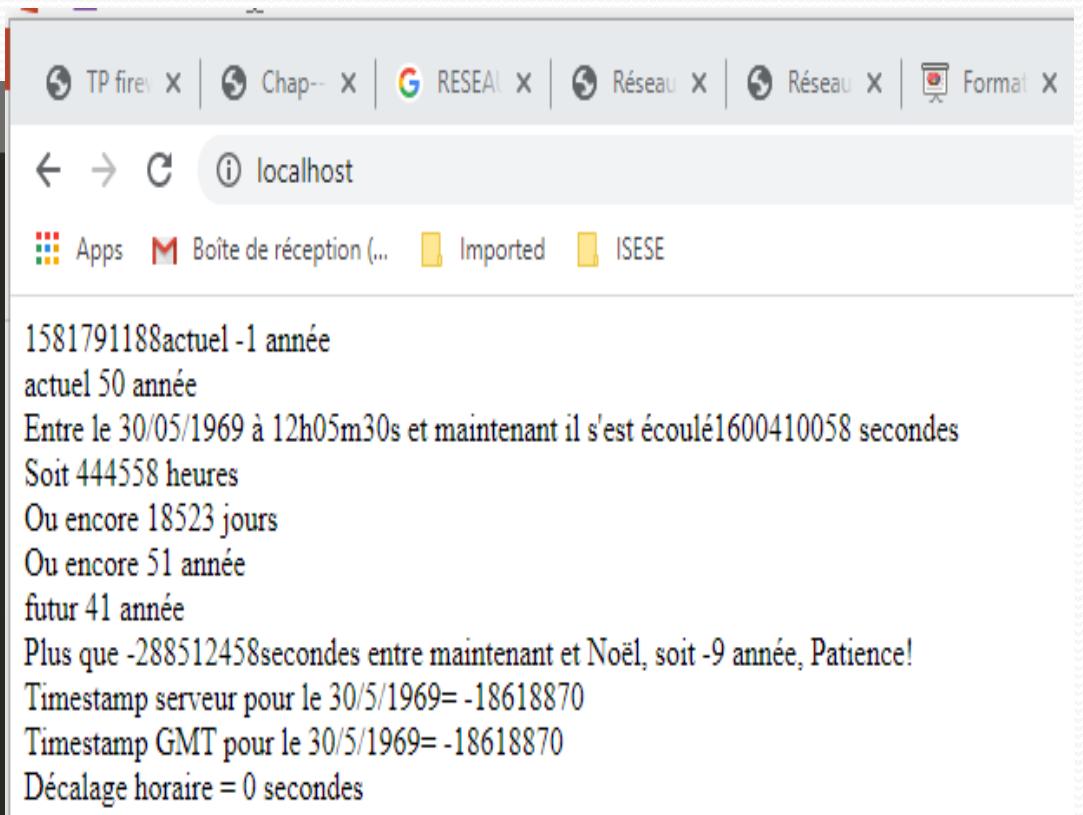
127.0.0.1/fonctionmktime.php

Entre le 30/05/1969 à 12h05m30s et maintenant il s'est écoulé 1381049197 secondes
Soit 383625 heures
Ou encore 15984 jours
Plus que -132223597 secondes entre maintenant et Noël, soit -1530 jours, Patience!
Timestamp serveur pour le 30/5/1969= -18622470
Timestamp GMT pour le 30/5/1969= -18618870
Décalage horaire = -3600 secondes

PHP - Les fonctionnalités du langage

Dates et calendriers

```
index.php      x  bonjour.php      .  reponse.php
1 <?php
2 echo time();
3 //la fonction mktime()
4 $timepasse= mktime(12,5,30,5,30,1969);
5 echo "actuel ",round($timepasse/3600/24/365)," année <br />";
6 $timeaujour = time();
7 echo "actuel ",round($timeaujour/3600/24/365)," année <br />";
8 $duree = $timeaujour-$timepasse;
9 echo "Entre le 30/05/1969 à 12h05m30s et maintenant il s'est
10 écoulé",$duree, " secondes <br />";
11 echo "Soit ",round($duree/3600), " heures <br />";
12 echo "Ou encore ",round($duree/3600/24)," jours <br />";
13 echo "Ou encore ",round($duree/3600/24/365)," année <br />";
14 ?>
15 <?php
16 $timefutur = mktime(12,5,30,12,25,2010);
17 echo "futur ",round($timefutur/3600/24/365)," année <br />";
18 $noel = $timefutur-$timeaujour;
19 echo "Plus que ",$noel, "secondes entre maintenant et Noël, soit ",
20 round($noel/3600/24/365)," année, Patience! <br />";
21 //la fonction gmmktime()
22 $timepassegmt = gmmktime(12,5,30,5,30,1969);
23 echo "Timestamp serveur pour le 30/5/1969= ",$timepasse,"<br />"; echo "Timestamp GMT pour le 30/5/1969= ",$
24 timepassegmt,"<br />";
25 echo "Décalage horaire = ",$timepasse-$timepassegmt," secondes<br />";
26 ?>
```



PHP - Les fonctionnalités du langage

Dates et calendriers

Date()

- Syntaxe:

```
string date(string format_de_date,[int timestamp])
```

- Exemple:

```
<?php  
echo "Aujourd'hui ",date("l, d F Y \i\l \e\s\\t H:i:s");  
?>
```

127.0.0.1/afficheDate.php

```
<?php  
$date_du_jour=date("d.m.y");  
$heure=date("H:i:s");  
echo "Bonjour.<BR>Nous sommes le : <B>$date_du_jour</B><BR>"  
echo "Et voici l'heure : <B>$heure</B><BR>";  
?>
```

Aujourd'hui Monday, 04 March 2013 il st 21:01:22

localhost/bonjour.php

Apps Boîte de réception (...) Imported ISESE

Bonjour.
Nous sommes le : 15.02.20
Et voici l'heure : 18:36:22

PHP - Les fonctionnalités du langage

Dates et calendriers

Vérifier une date

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Validation de date</title>
</head>
<body>
<form method="post" action=<?= $_SERVER["PHP_SELF"] ?>>
<fieldset>
<legend>Entrez votre date de naissance sous la formeJJ/MM/AAAA</legend>
<input type="text" name="date" />
<input type="submit" value="Envoi"/>
</fieldset>
</form>
<?php
//checkdate
```

PHP - Les fonctionnalités du langage

Dates et calendriers

```
if(isset($_POST["date"]))
{
$date=$_POST["date"];
$tabdate=explode("/",$date);
if(!checkdate($tabdate[1],$tabdate[0],$tabdate[2])) {echo "<hr />
➥La date $date n'est pas valide. Recommencez! <hr />";}
else {echo "<h3> La date $date est valide. Merci!</h3>";}
}
?>
</body>
</html>
```



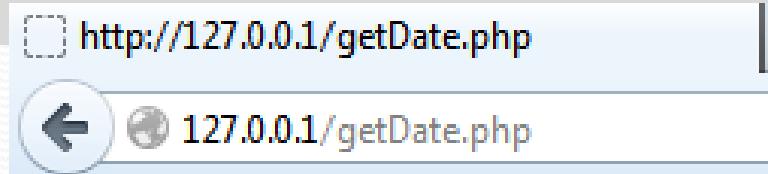
PHP - Les fonctionnalités du langage

Dates et calendriers

Fonction getdate()

- Retourne un tableau d'informations sur la date.
- Syntaxe: `array getdate([inttimestamp])`
- Exemple:

```
<?php  
$jour = getdate();  
echo "Aujourd'hui {$jour["weekday"]} {$jour["mday"]} {$jour["month"]}  
{$jour["year"]};  
?>
```



Aujourd'hui Monday 4 March 2013

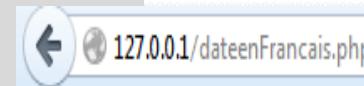
PHP - Les fonctionnalités du langage

Dates et calendriers

- Créer deux tableaux indexés des et jours mois en français.

Date en français

```
<?php
//Date en français
$jour = getdate();
echo "Anglais: Aujourd'hui {$jour["weekday"]}
{$jour["mday"]}{$jour["month"]}
{$jour["year"]} <br />";
$semaine = array(" dimanche "," lundi "," mardi "," mercredi "," jeudi ",
    " vendredi "," samedi ");
$mois =array(1=>" janvier "," février "," mars "," avril "," mai "," juin ",
" juillet ",      " août "," septembre "," octobre "," novembre "," décembre ");
//Avec getdate()
echo "Français: Avec getdate() : Aujourd'hui ",$semaine[$jour['wday']] ,
$jour['mday'],
    $mois[$jour['mon']], $jour['year'], "<br />";
//Avec date()
echo " Français: Avec date() : Aujourd'hui ", $semaine[date('w')] ,
",date('j'),"
    ", $mois[date('n')], date('Y'),"<br />";
?>
```



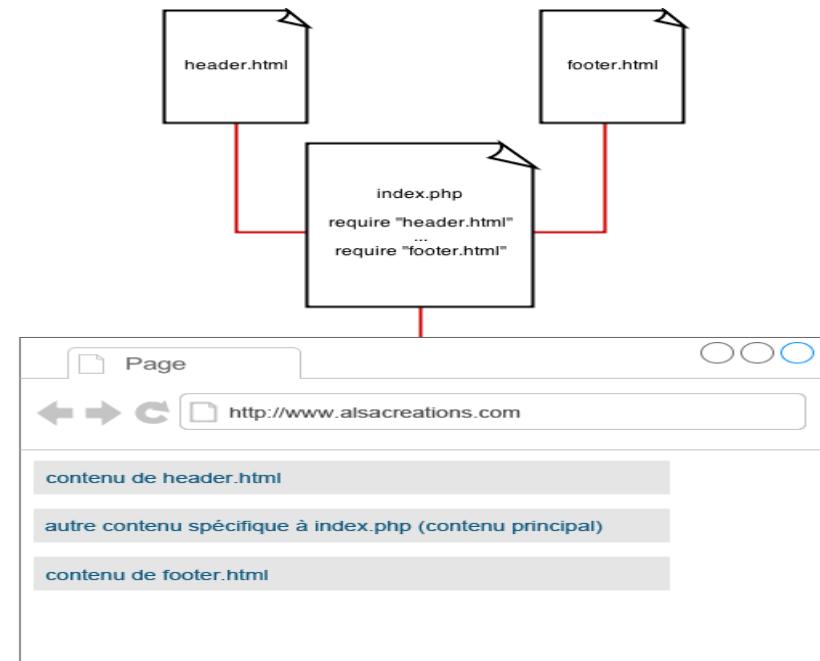
PHP - Les fonctionnalités du langage

Inclusion de fichiers externes

Il est possible d'incorporer le code PHP ou HTML écrit dans d'autres fichiers
Plusieurs possibilités

- `include("nom_du_fichier.ext");`
- `include_once("nom_du_fichier.ext");`

- `require("nom_du_fichier.ext");`
- `require_once("nom_du_fichier.ext");`



PHP - Les inclusions

La structure **include** permet d'appeler un fichier dans la page où elle a été déclarée. On la prend à tort pour une fonction, mais en réalité il s'agit d'un structure et par conséquent, les parenthèses ne sont pas obligatoire.

La syntaxe de la structure include

```
<?php  
    include "fichier_à_inclure";  
    // ou bien  
    include("fichier_à_inclure");  
?>
```

La structure **require** fonctionne pratiquement comme **include**. La principale différence entre les deux structures c'est qu'à l'inverse de **include** qui se contente d'afficher une notification si le fichier appelé n'est pas accessible, **require** quant-à-elle arrête nettement l'exécution du programme.

La syntaxe de la structure require

```
<?php  
    require "fichier_à_inclure";  
    // ou bien  
    require("fichier_à_inclure");  
?>
```

PHP - Les inclusions

La structure **include** permet d'appeler un fichier dans la page où elle a été déclarée. On la prend à tort pour une fonction, mais en réalité il s'agit d'un structure et par conséquent, les parenthèses ne sont pas obligatoire.

La syntaxe de la structure include

```
<?php  
    include "fichier_à_inclure";  
    // ou bien  
    include("fichier_à_inclure");  
?>
```

La structure **require** fonctionne pratiquement comme **include**. La principale différence entre les deux structures c'est qu'à l'inverse de **include** qui se contente d'afficher une notification si le fichier appelé n'est pas accessible, **require** quant-à-elle arrête nettement l'exécution du programme.

La syntaxe de la structure require

```
<?php  
    require "fichier_à_inclure";  
    // ou bien  
    require("fichier_à_inclure");  
?>
```

PHP - Les fonctionnalités du langage

Inclusion de fichiers externes

A Include && Include once

❑ `include("nom_du_fichier.ext");`

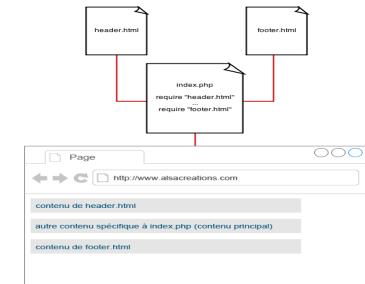
- Semblable aux include du C/C++
- Réalise une inclusion physique du fichier demandé

```
<?php include("connect.php"); ?>
```

❑ `include_once("nom_du_fichier.ext");`

- identique au include
- protège contre d'éventuelles inclusions multiples qui pourraient mener à des erreurs (redéclarations, etc.)

```
<?php include_once("connect.php"); ?>
```



PHP - Les fonctionnalités du langage

Inclusion de fichiers externes

B Require && Require_once

Fonctionnent comme le include et le include_once respectivement mais le programme s'arrête si le fichier inclus n'existe pas

require("nom_du_fichier.ext");

`<?php require("malib.php"); ?>`

require_once("nom_du_fichier.ext");

`<?php require_once("connect.php"); ?>`

include_once et require_once

La structure **include_once** fonctionne exactement comme la structure **include**. Cependant **include_once** n'inclue le même fichier qu'une seule fois. En effet, au moment de son exécution, le compilateur vérifie si le fichier n'est pas déjà inclus. Si c'est le cas, il sera ignoré sinon il sera inclus

Avertissement

L'utilisation systématique de la version avec **once** (**include_once ou require_once**) n'est pas recommandée car elle peut causer des ralentissements à l'exécution du programme.

PHP - Les arrêts prématurés

Les arrêts prématurés

- Erreur fatale:** il s'agit d'une erreur stricte qui empêche le programme de se poursuivre. Néanmoins, ce genre d'erreur pousse le compilateur à ne rien exécuter du tout car le langage PHP est compilé.
- Arrêt prématué:** il s'agit d'un arrêt programmé par le développeur. Dans ce cas, même si le programme ne s'exécute pas en entier, son arrêt est considéré comme normal car il a été prévu

Arrêt prématué avec exit() et die()

Les fonctions **exit()** et **die()** sont similaires (des alias). Elles arrêtent le programme à l'endroit où elles sont déclarées en affichant le message passé en argument (en tant que chaîne de caractères).

```
<?php  
    for($i=1;$i<=10;$i++) {  
        if($i>5)  
            die("Fin");  
        echo "Ligne $i <br />";  
    }  
?>
```

Ligne 1
Ligne 2
Ligne 3
Ligne 4
Ligne 5
Fin

Programmation Web Dynamique

Pr. Rachid DAKIR

Filières : SMI & IGE

FP Ouarzazate

UNIVERSITE IBN ZOHR

Année Universitaire : 2019-2020

PARTIE : III

PHP - TRANSMISSION DES PARAMETRES

Le langage PHP a été conçu pour que vous puissiez transmettre des informations de page en page, au fil de la navigation d'un visiteur sur votre site. C'est notamment ce qui vous permet de retenir son pseudonyme tout au long de sa visite, mais aussi de récupérer et traiter les informations qu'il rentre sur votre site, notamment dans des formulaires.



Transmission et Récupération des données en PHP

PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

Les pages Web se transmettent des données entre elles.



PHP est capable de transmettre et de récupérer
les données saisies



Comment ça marche ?

GET

POST

PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

Dans le protocole HTTP, il existe deux méthodes pour envoyer une requête :

Lorsque les valeurs saisies dans un formulaire sont de petites tailles, ce qui est le cas la plupart du temps, on a donc le choix :

- soit on stocke les variables dans l'adresse de la page, comme nous venons de le voir au chapitre précédent, et dans ce cas il n'y a pas besoin d'un corps de requête : c'est la méthode GET ;
- soit on les stocke dans le corps de la requête, et on choisit pour cela la méthode POST.

PHP - TRANSMISSION DES PARAMETRES

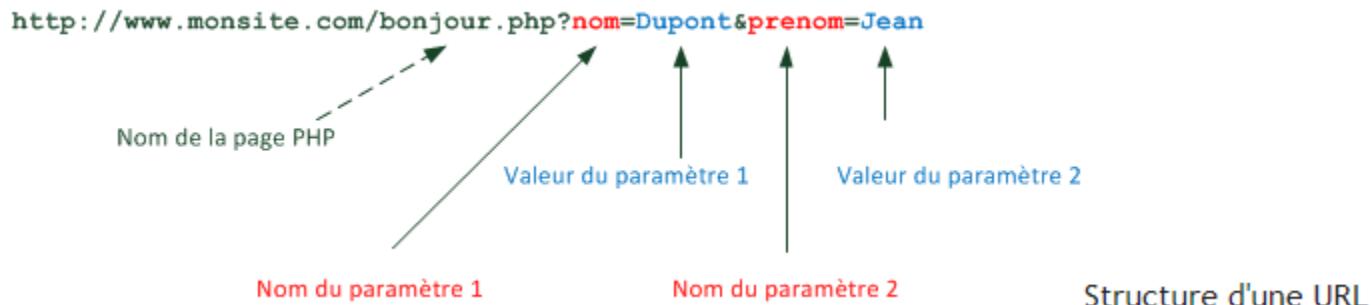
Transmission des données via URL

Former une URL pour envoyer des paramètres

Savez-vous ce qu'est une URL ?

Cela signifie Uniform Resource Locator, et cela sert à représenter une adresse sur le web. Toutes les adresses que vous voyez en haut de votre navigateur, comme <http://www.siteduzero.com>, sont des URL.

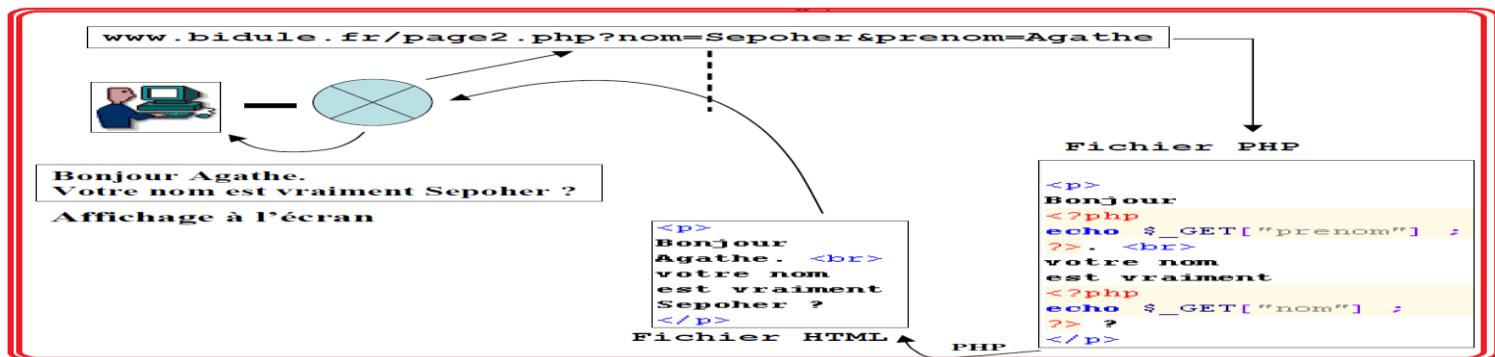
URL sont assez longues et comportent parfois des caractères un peu curieux. Par exemple, après avoir fait une recherche sur Google, la barre d'adresse contient une URL longue qui ressemble à ceci :



PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

la façon dont nous allons récupérer et traiter ces informations dans un script PHP sur le serveur.



- comment le client envoie les valeurs du formulaire vers le serveur
- comment le programme PHP récupère ces valeurs dans des variables pour générer sur le serveur une page dynamique qui tienne compte de ces valeurs

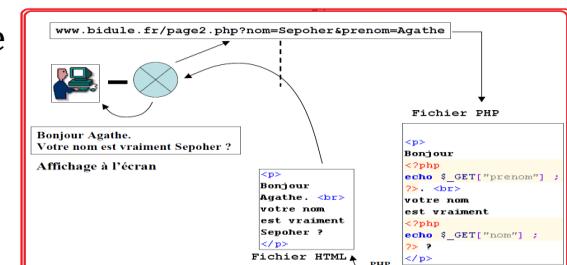
PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

Dans le protocole HTTP, il existe deux méthodes pour envoyer une requête :

➤ GET :

- La méthode GET envoie les données sous forme d'une suite de couples nom/valeur ajoutés à l'URL de la page appelée.
- La partie d'une URL précédée par le caractère point d'interrogation (?) est appelée chaîne de requête. Si la chaîne de requête contient plusieurs éléments, alors chaque élément/valeur doit être séparé par le caractère &
- Par ailleurs, elle ne peut pas dépasser 255 caractères. Les données transmises au serveur par la méthode GET sont visibles par les utilisateurs directement dans la barre d'adresse du navigateur et dans ce cas il n'y a pas besoin d'un corps de requête



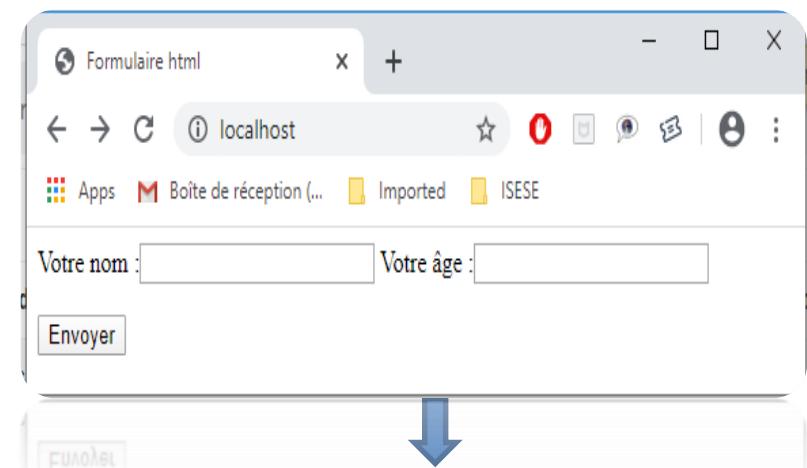
PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ GET :

```
<html><head>
<meta charset="utf-8" />
<title>Formulaire html</title>
</head>
<body>
<form action="reponse.php" method="GET">
Votre nom :<input type="text" name="nom">
Votre âge :<input type="text" name="age">
<p>
<input type=submit value="Envoyer">
</form>
</body></html>
```

```
<html><head>
<title>Test Formulaire PHP</title>
</head>
<body>
<h1>Bonjour, <?php echo $_GET['nom'] ?></h1>
<h2>Vous semblez avoir <?php echo $_GET['age'] ?></h2>
</body></html>
```



PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ GET :

```
<html><head>
<meta charset="utf-8" />
<title>Formulaire html</title>
</head>
<body>
<form action="reponse.php" method="GET">
Votre nom :<input type="text" name="nom">
Votre âge :<input type="text" name="age">
<p>
<input type=submit value="Envoyer">
</form>
</body></html>
```

L'attribut **action** de l'objet formulaire **form** donne le nom de la page qui traite, sur le serveur, les informations postées par le formulaire. Les informations traitées sont définies par les objets **input**, **textarea** et **select** définis à l'intérieur du formulaire.

PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ GET :

```
<html>
<head>
<title> Ma première page </title>
</head>
<body>
<p> Bonjour ! </p>
</body>
</html>
```

page2.php

Les informations postées par ce formulaire doivent donc être traités par la page **page2.php**.

En effet, si on remplis les champs **nom** et **prenom** avec les valeurs « MOHAMED » et « ALI », et lorsqu'on clique sur le bouton **submit**, voici ce qui s'affiche dans ma barre d'adresse :

page2.php?nom=MOHAMED&prenom=ALI

PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

<[a href="bonjour.php?nom=Dupont&prenom=Jean">Dis-moi bonjour !](bonjour.php?nom=Dupont&prenom=Jean)

Ce lien appelle la page bonjour.php et lui envoie deux paramètres :

- nom : RACHID
- prenom : DAKIR

Vous avez sûrement deviné ce qu'on essaie de faire ici : on appelle une page bonjour.php qui va dire « Bonjour » à la personne dont le nom et le prénom ont été envoyés en paramètres.

!!Comment faire dans la page bonjour.php pour récupérer ces informations ? !!

PHP - TRANSMISSION DES PARAMETRES

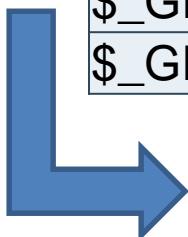
Transmission et Récupération des données en PHP

Récupérer les paramètres en PHP

En script PHP, on peut donc récupérer ces informations, les traiter et les afficher.

\$_GET :

Nom	Valeur
<code>\$_GET['nom']</code>	RACHID
<code>\$_GET['prenom']</code>	DAKIR



Comment ?

PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

Récupérer les paramètres en PHP

<p>Bonjour <?php echo \$_GET['prenom'] . ' ' . \$_GET['nom']; ?> !</p>

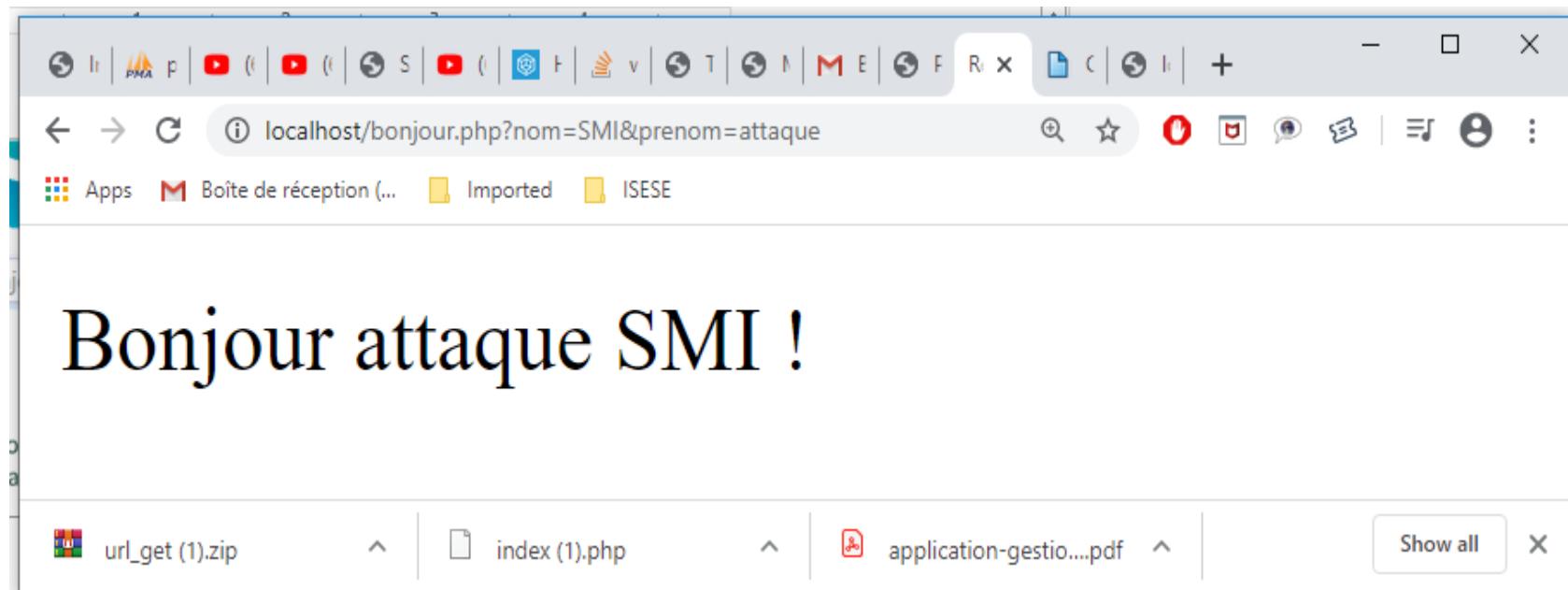
```
<html <head>
    <title>Envoi de paramètres dans l'URL</title>
</head>
<body>
    <p><a href="bonjour.php?nom=SMI&prenom=attaque">Dis-moi bonjour Mr :
    !</a></p>
</body></html>
```

```
<html> <head>
    <title>Réception de paramètres dans l'URL</title>
</head>
<body>
    <p>Bonjour <?php echo $_GET['prenom'] . ' ' . $_GET['nom']; ?> !</p>
</body></html>
```

PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

Récupérer les paramètres en PHP

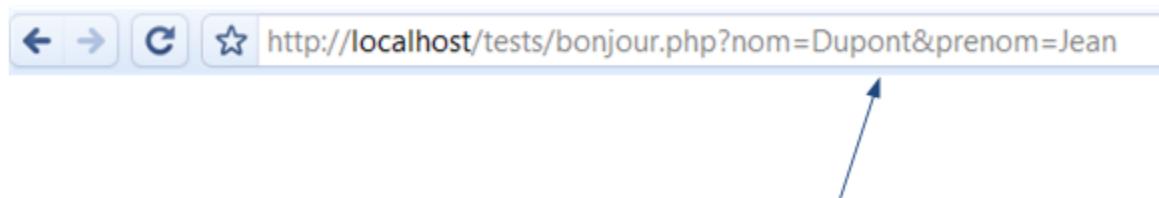


PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

Récupérer les paramètres en PHP

Ne faites jamais confiance aux données reçues !



N'importe qui peut modifier les paramètres
dans la barre d'adresse de son navigateur !

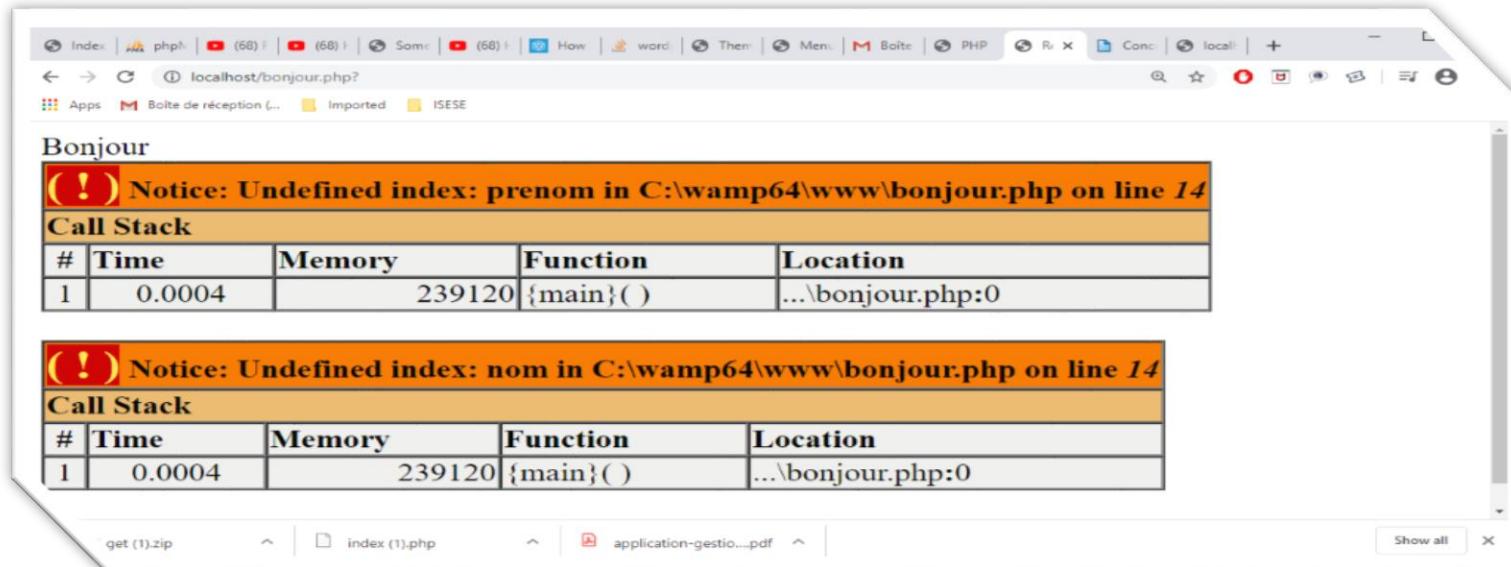
PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

Mécanisme GET

Récupérer les paramètres en PHP

Suppression des paramètres transmises entre client et serveur



Solution ??

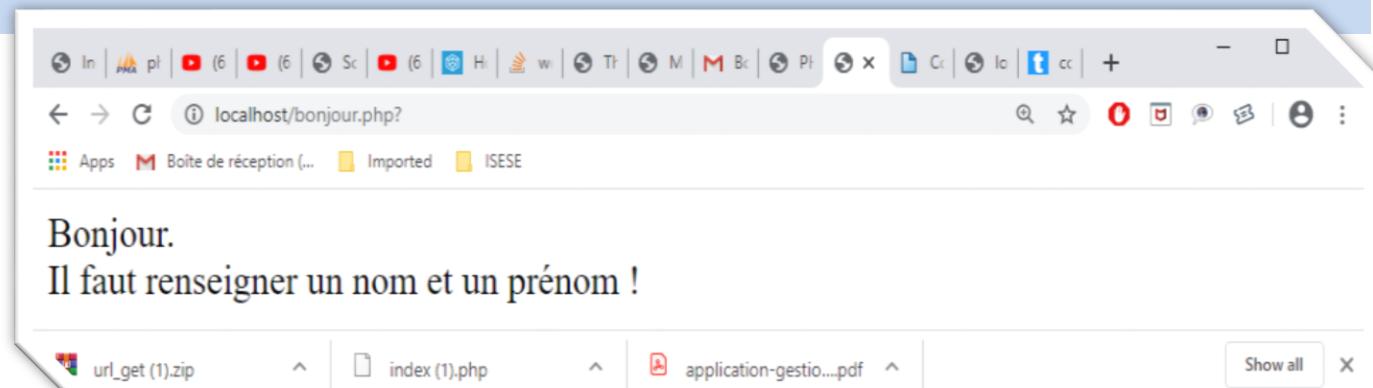
PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

Récupérer les paramètres en PHP

Solution

```
<?php
// On a le nom et le prénom
if (isset($_GET['prenom']) AND isset($_GET['nom'])) {
    echo 'Bonjour ' . $_GET['prenom'] . ' ' . $_GET['nom'] . ' !';
// Il manque des paramètres, on avertit le visiteur
else {    echo 'Il faut renseigner un nom et un prénom !'; }
?>
```

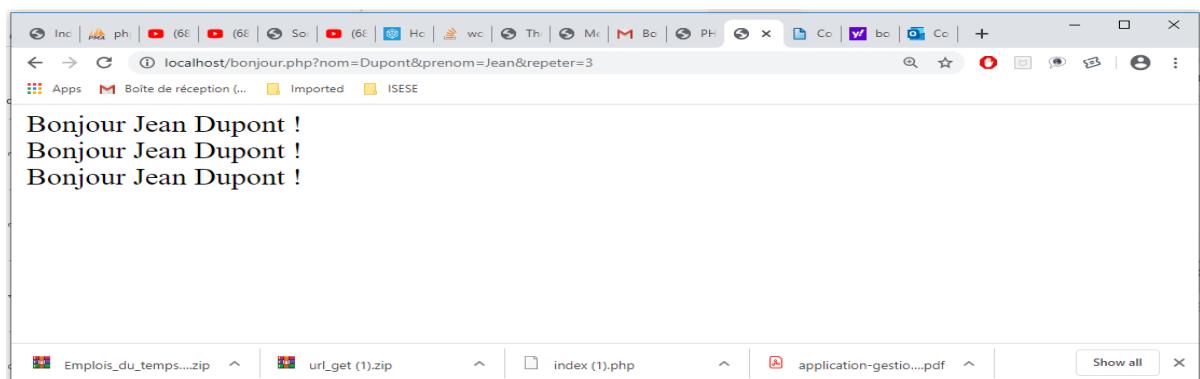


PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

```
<?php
if (isset($_GET['prenom']) AND isset($_GET['nom']) AND isset($_GET['repeter']))
{   for ($i = 0 ; $i < $_GET['repeter'] ; $i++)
    {         echo 'Bonjour ' . $_GET['prenom'] . ' ' . $_GET['nom'] . ' !<br />';
    }
} else
{ echo 'Il faut renseigner un nom, un prénom et un nombre de répétitions !';
}
?>
```

<http://localhost/bonjour.php?nom=Dupont&prenom=Jean&repeter=3>



PHP - TRANSMISSION DES PARAMETRES

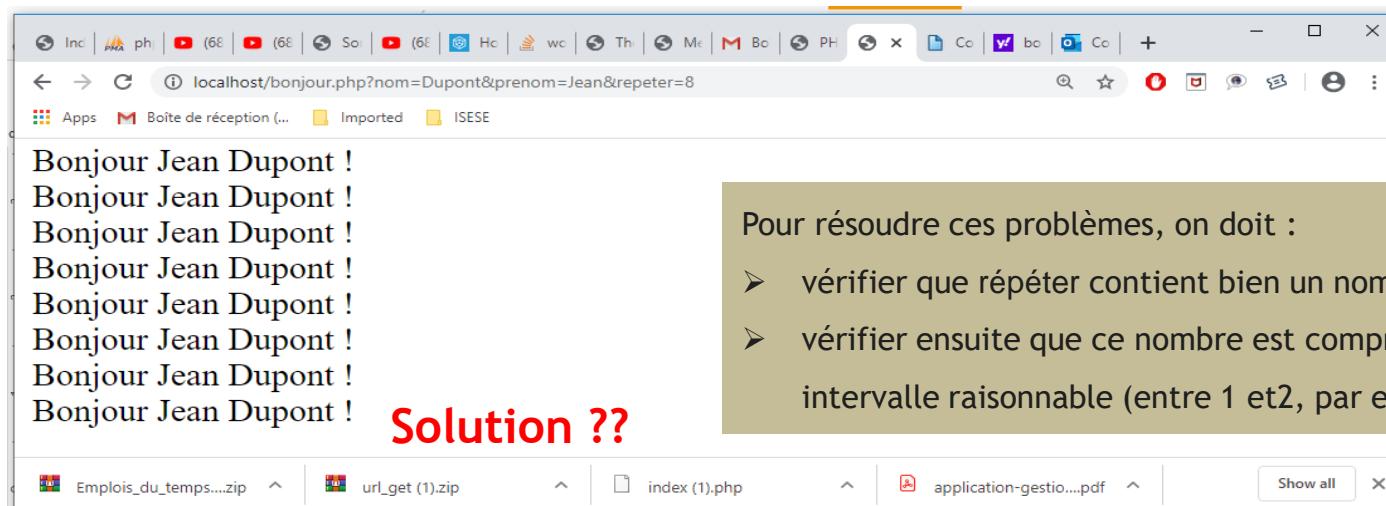
Transmission et Récupération des données en PHP

Mécanisme GET

Récupérer les paramètres en PHP

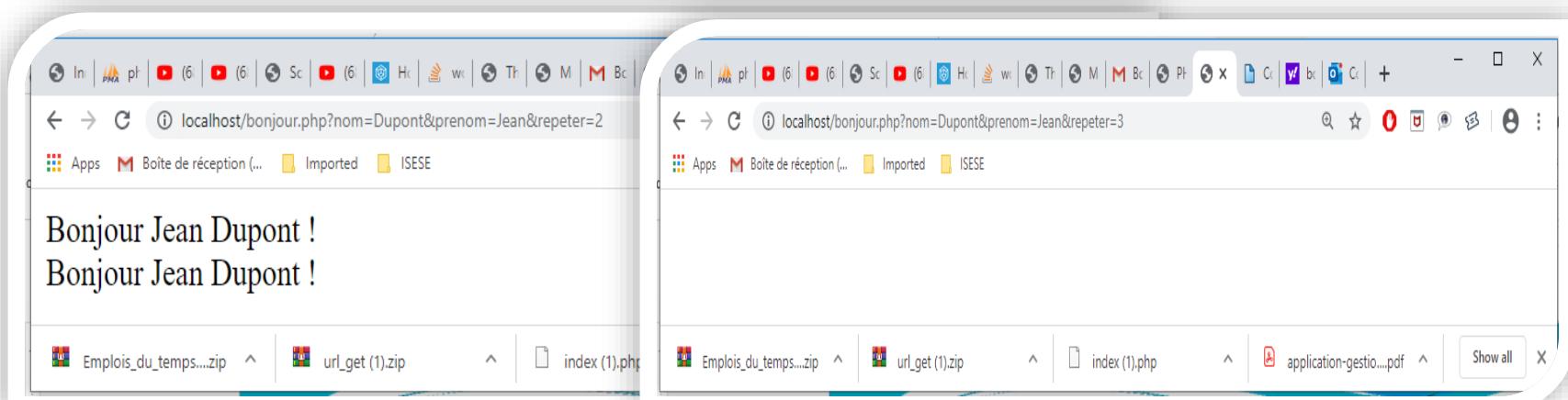
problème

<http://localhost/tests/bonjour.php?nom=Dupont&prenom=Jean&repeter=8>



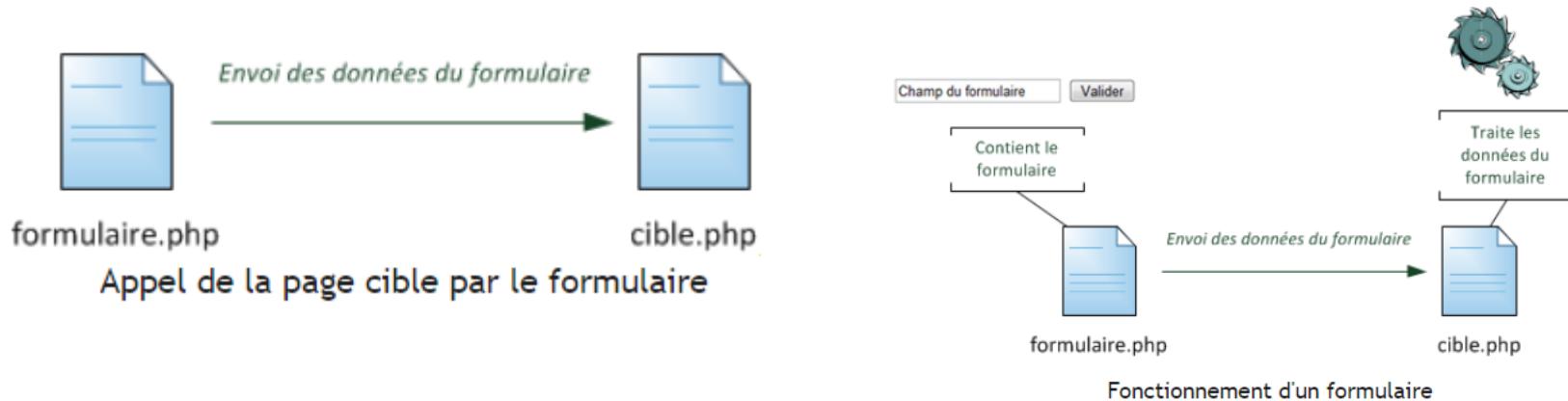
PHP - TRANSMISSION DES PARAMETRES

```
<?php
if (isset($_GET['prenom']) AND isset($_GET['nom']) AND isset($_GET['repeter']))
{
    // 1 : On force la conversion en nombre entier
    $_GET['repeter'] = (int) $_GET['repeter'];
    // 2 : Le nombre doit être compris entre 1 et 100
    if ($_GET['repeter'] >= 1 AND $_GET['repeter'] <= 2)
    {
        for ($i = 0 ; $i < $_GET['repeter'] ; $i++)
        {
            echo 'Bonjour '. $_GET['prenom'] . '' . $_GET['nom'] . ' !<br />'; }}}
else
{ echo 'Il faut renseigner un nom, un prénom et un nombre de répétitions !';?>
```



PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP



```
<form method="POST" action="cible.php">
<p> On insérera ici les éléments de notre formulaire.</p>
</form>
```

```
<form method="GET" action="cible.php">
<p> On insérera ici les éléments de notre formulaire.</p>
</form>
```

La cible

L'attribut `action` sert à définir la page appelée par le formulaire. C'est cette page qui recevra les données du formulaire et qui sera chargée de les traiter.

PHP - TRANSMISSION DES PARAMETRES

Transmission et Récupération des données en PHP

Transmettre des données avec des formulaires

Les formulaires constituent le principal moyen pour vos visiteurs d'entrer des informations sur votre site . Les formulaires permettent de créer une interactivité.

Par exemple :

- Forum
- Livre d'or
- Mini-chat

Dans un formulaire, on peut insérer beaucoup d'éléments différents : zones de texte, boutons, cases à cocher, etc.

Au lieu de recevoir un array `$_GET`, vous allez recevoir un array `$_POST` contenant les données du formulaire !

PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

Dans le protocole HTTP, il existe deux méthodes pour envoyer une requête :

- **POST** : C'est une requête qui met des données dans le corps de la requête. Le passage de paramètre se fait de manière invisible dans l'URL(les données transmises par un formulaire restent confidentielles et n'apparaissent pas dans l'URL). Cette méthode est donc la plus utilisée. Toutes les données contenues dans un formulaire seront envoyées à l'autre page PHP via la méthode POST et reçues dans tableau superglobal **`$_POST`**.

Selon que la méthode d'envoi a été du GET ou du POST la récupération du contenu des variables est faite selon une syntaxe différente :

- Dans le cas d'un envoi des paramètres en POST
`<?php $variable1=$_GET['nom_du_champ'] ; ?php>`
- Dans le cas d'un envoi des paramètres en GET
`<?php $variable1=$_POST['nom_du_champ'] ; ?php>`

Récupération des variables **au sein des superglobales**

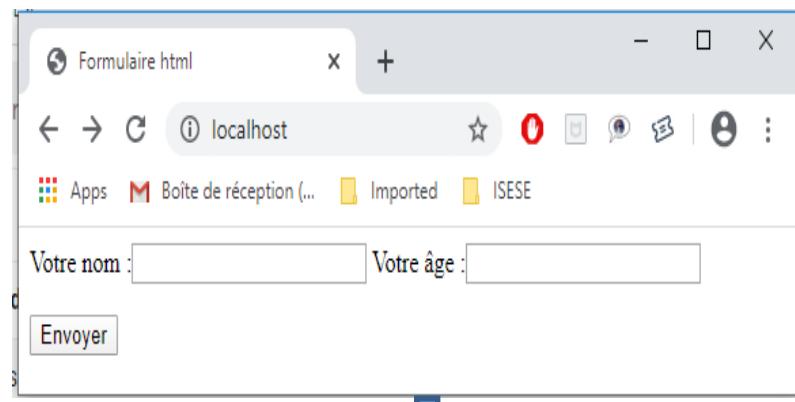
PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ POST :

```
<html><head>
<meta charset="utf-8" />
<title>Formulaire html</title>
</head>
<body>
<form action="reponse.php" method="POST">
Votre nom :<input type="text" name="nom">
Votre âge :<input type="text" name="age">
<p>
<input type=submit value="Envoyer">
</form>
</body></html>
```

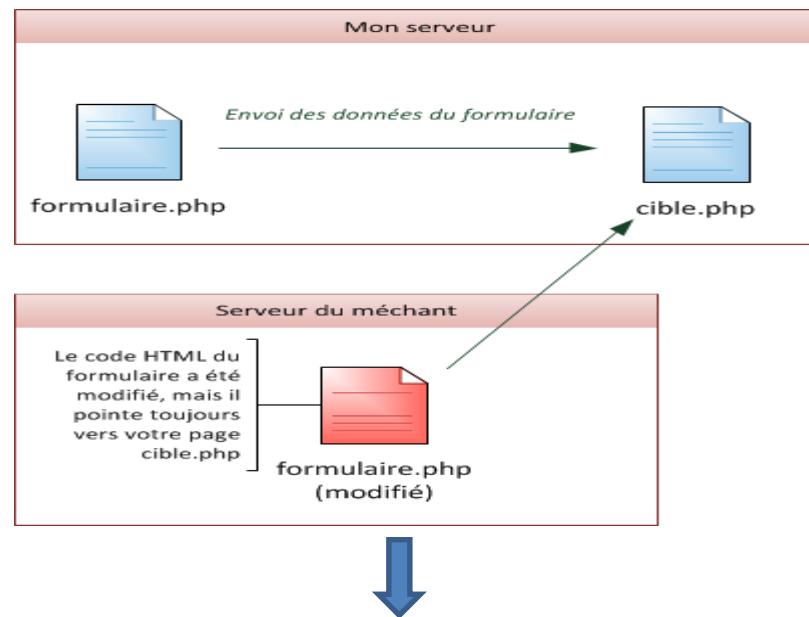
```
<html><head>
<title>Test Formulaire PHP</title>
</head>
<body>
<h1>Bonjour, <?php echo $_POST['nom'] ?></h1>
<h2>Vous semblez avoir <?php echo $_POST['age'] ?></h2>
</body></html>
```



PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ POST :

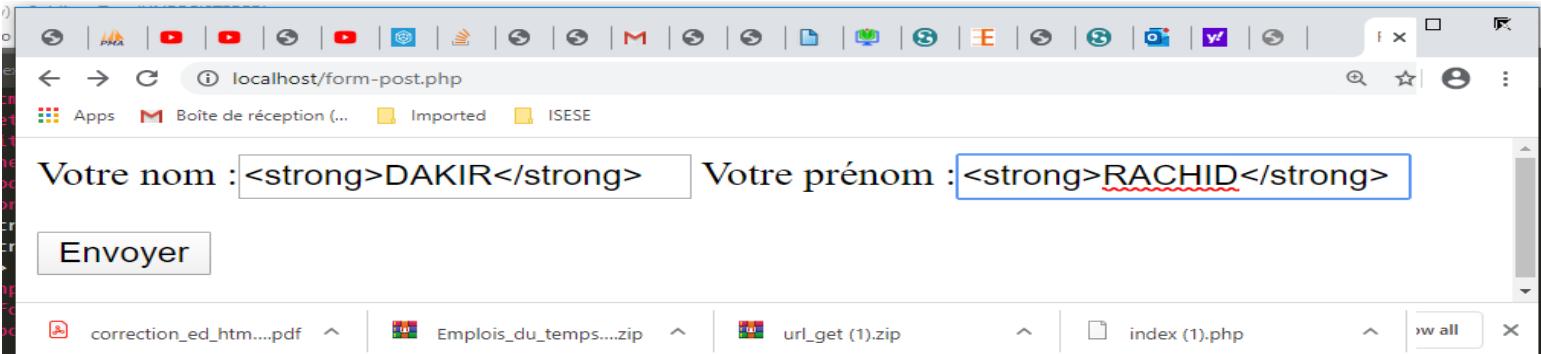


!! Qu'est-ce qui empêche quelqu'un de créer une copie légèrement modifiée de la formulaire et de la stocker sur son serveur !!

PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ POST :



Votre nom : **DAKIR** Votre prénom : **RACHID**

Envoyer

<p>Je sais comment tu t'appelle. Tu t'appelles **Badaboum** !</p>

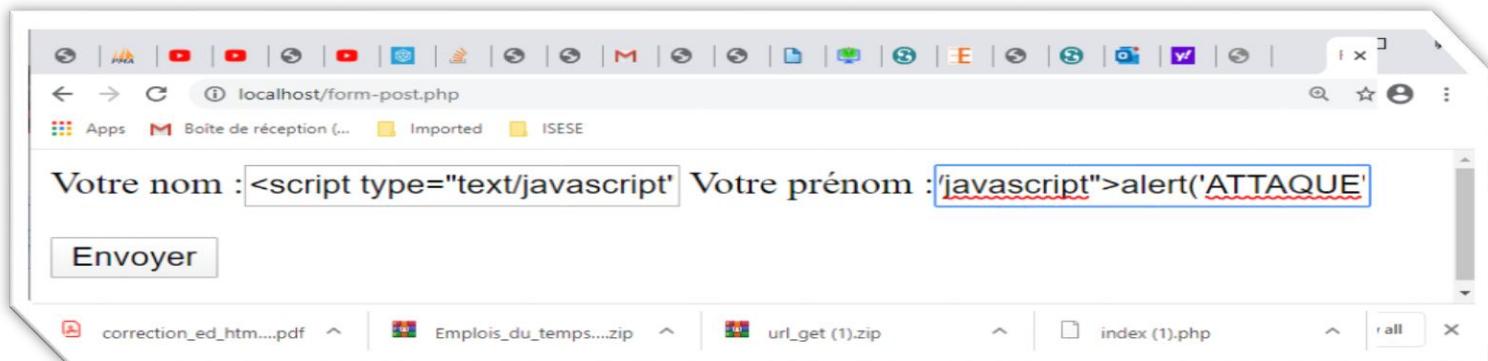


Je sais comment tu t'appelles. . Tu t'appelles **rachid dakir** !

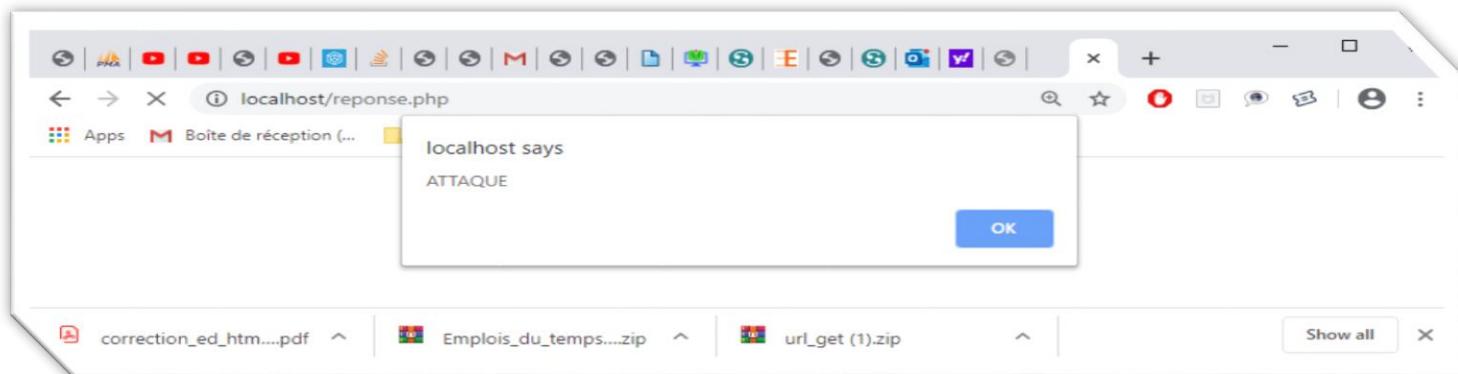
PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ POST :



<p>Je sais comment tu t'appelle. Tu t'appelles **ATTACQUE !</p>**



PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ POST :

Solution

Mal : le code HTML a été inséré tel quel dans la page, n'importe quel visiteur peut placer du HTML

Pour échapper le code HTML, il suffit d'utiliser la fonction **htmlspecialchars** qui va transformer les chevrons des balises HTML <> en < et > respectivement. Cela provoquera l'affichage de la balise plutôt que son exécution

Je sais comment tu t'appelles, Tu t'appelles Badaboum !

Je sais comment tu t'appelles, Tu t'appelles Badaboum !



Bien : le code HTML a été « échappé » et il n'est pas interprété par le navigateur. On voit le HTML mais celui-ci est inoffensif, il ne s'exécute pas.

```
<p>Je sais comment tu t'appelles.  
Tu t'appelles <?php echo htmlspecialchars($_POST['prenom']); ?> !</p>  
  
<p>Je sais comment tu t'appelles.  
Tu t'appelles &lt;strong&gt;Badaboum&lt;/strong&gt; !</p>
```

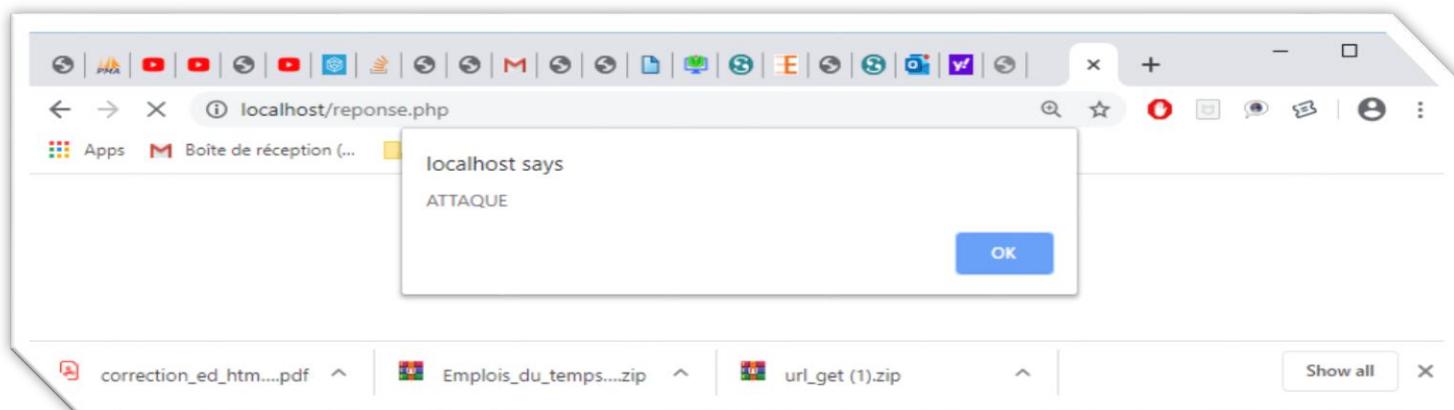
PHP - TRANSMISSION DES PARAMETRES

Passage et transmission de variables

➤ POST :



<p>Je sais comment tu t'appelle. Tu t'appelles **ATTAQUE !</p>**



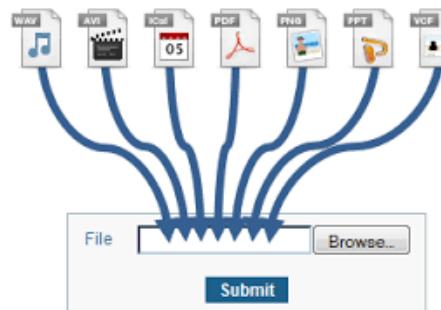
PHP - Chargement des fichiers

Le chargement de fichiers (ou upload) consiste à envoyer des fichiers au serveur. L'opération inverse s'appelle téléchargement (ou download).

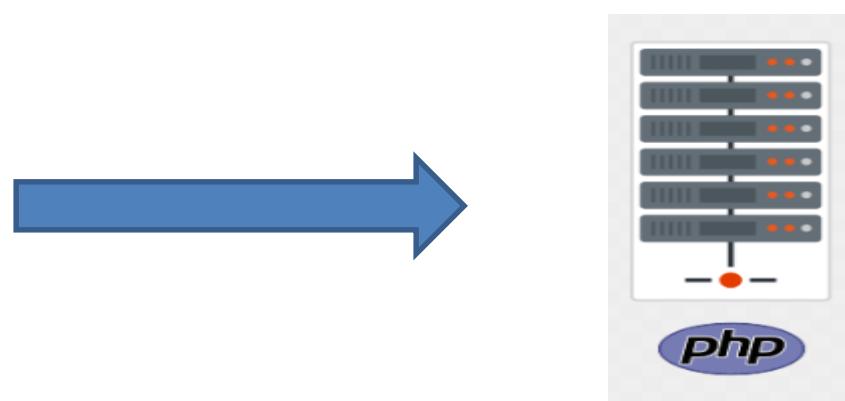
Le chargement de fichier peut être traité sur deux parties :

- ❑ la partie HTML où on va s'intéresser au formulaire qui accueillera le champ approprié
- ❑ la partie PHP où on va voir comment manipuler le fichier posté au serveur.

Partie HTML: le formulaire



Partie PHP: la variable \$_FILES

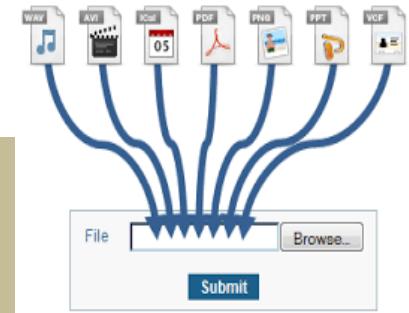


PHP - Chargement des fichiers

Partie HTML: le formulaire

Syntaxe :

```
<form method="post" action="" enctype="multipart/form-data">
    <input type="hidden" name="MAX_FILE_SIZE" value="2000000" />
    <input type="file" name="file-upload" /><br />
    <input type="submit" name="valider" value="Uploader" />
</form>
```



Attributs :

enctype : Type d'encodage:**multipart/form-data** qui signifie que des fichiers (binaires) vont être inclus aux données postées.

file : Le champs qui permet naturellement de joindre des fichiers à un document HTML

hidden : champ caché qui a comme nom **MAX_FILE_SIZE** (optionnel): représente la taille maximale du fichier à uploader en octets.

N.B : Il existe une directive dans le fichier **php.ini** qui permet de spécifier la taille maximale des fichiers que l'on peut uploader sur le serveur. Elle ressemble à ceci:

upload_max_filesize = 2M

PHP - Chargement des fichiers

Parie PHP: la variable `$_FILES`

Syntaxe :

`$_FILES[A][B]` : permet de faire des traitement important une fois le chargement du fichier effectué sur le serveur, comme par exemple, vérifier le type du fichier, sa taille.

Attributs :

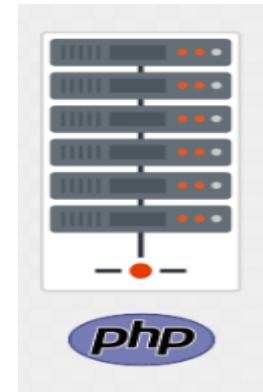
`$_FILES` : Il contient donc deux crochets (A,B) :

A :**name** du champ de type **file**

B : l'un de 5 valeurs suivantes:

- **name**: Nom d'origine du fichier chargé.
- **tmp_name**: désigne le nom temporaire donné par serveur au fichier chargé.
- **type**: désigne le type et le MIME du fichier chargé (par exemple images/jpeg).
- **size**: désigne la taille en octets du fichier chargé.
- **error**: désigne le code d'erreur dans le cas où les choses ne se sont pas déroulées comme prévu.

- ❑ `$_FILES["monfichier"]["name"]` : désigne le nom du fichier chargé depuis le formulaire précédent.
- ❑ `move_uploaded_file($tmp,$upload)` : permet de copier le fichier temporaire \$tmp (qui correspond à la clé **tmp_name**) dans l'emplacement \$upload choisi.



PHP - Chargement des fichiers

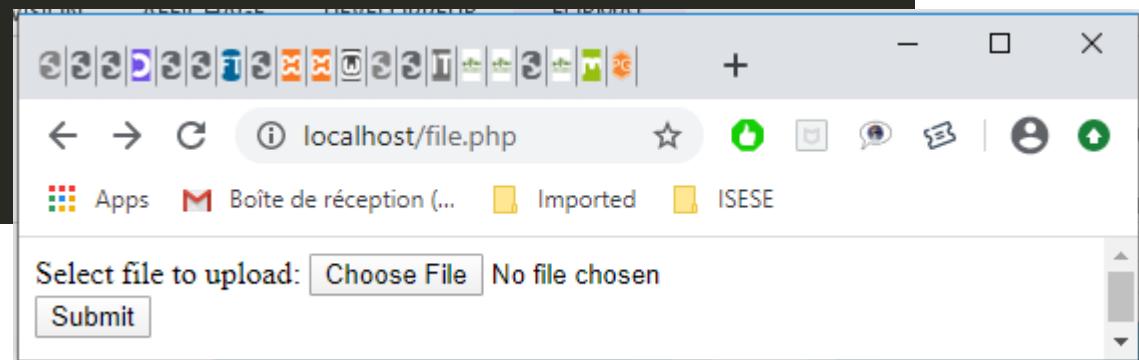
Partie HTML: le formulaire

Exemple :

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

<form action="upload.php" method="post" enctype="multipart/form-data">
    Select file to upload:
    <input type="file" name="file"><br>
    <input type="submit" name="submit">
</form>

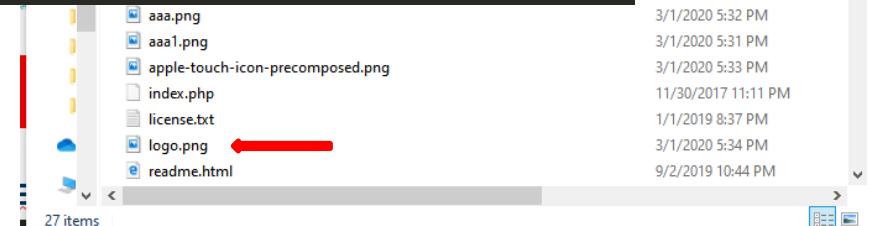
</body>
</html>
```



PHP - Chargement des fichiers

Exemple :

```
<?php
if(isset($_POST["submit"])) { $fileName = $_FILES['file']['name'];
    $fileTmpName = $_FILES['file']['tmp_name'];
    $fileSize = $_FILES['file']['size'];
    $fileError = $_FILES['file']['error'];
    $fileType = $_FILES['file']['type'];
    $fileExt = explode('.',$fileName);
    $fileActualExt = strtolower(end($fileExt));
    $allowed = array('jpg','jpeg','png','pdf');
    if(in_array($fileActualExt,$allowed)) {
        if ($fileError==0) {
            $fileNameNew = $fileName;
            $fileDestination = 'test/'.$fileNameNew;
            move_uploaded_file($fileTmpName, $fileDestination);
            echo "<img src='test/".$_FILES['file']['name']."'>";
        } else {
            echo "you cannot upload files of this type!";
        }
    } else {
        echo "you cannot upload files of this type!";
    }
}
?>
```



PHP - Envoi Email

Fonction mail :

Il vous est certainement déjà arrivé de vous inscrire sur un site , et juste après que vous ayez fini votre inscription, vous recevez un mail de confirmation d'inscription et d'informer le webmaster qu'il ya une nouvelle inscription .

En PHP il est possible d'envoyer un mail via un script, c'est grâce à la fonction **mail()**.

Syntaxe :

```
mail ($dest , $objet , $message , $entetes);
```



PHP - Envoi Email

Fonction mail :

Paramètres (Attributs):

- **\$dest:** spécifie l'adresse du destinataire.
- **\$objet :** décrit l'objet du message.
- **\$message:** désigne le message à envoyer au(x) destinataire(s).

PHP - Envoi Email

Fonction mail :

Paramètres (Attributs):

- **Paramètre \$entetes:**

Le paramètre **\$entetes** est facultatif mais très utile. Il spécifie les informations supplémentaires qui seront envoyées avec le mail. Un entête respecte la forme **propriété:valeur**. Si plusieurs entêtes sont combinés alors il faut les séparer par \n.

Voici la liste des entêtes les plus utilisés:

- **From:** désigne l'adresse de l'expéditeur.
- **Cc:** (pour Carbon Copy) désigne l'adresse qui recevra une copie du mail.
- **Bcc:** (pour Blind Carbon Copy) désigne l'adresse qui recevra une copie cachée du mail.
- **Reply-To:** spécifie l'adresse de réponse.

PHP - Envoi Email

Fonction mail :

Exemple :

```
<?php

$dest="contact@chiny.me";
$objet="Rendez-vous";
$message="Bonjour\nPrière de se retrouver sur Skype à <b>18h</b>
aujourd'hui.\n Merci et bonne journée";
$entetes="From: sc@example.com\n";
$entetes.="Cc: chiny@example.com\n";
$entetes.="Content-Type: text/html; charset=iso-8859-1";
if(mail($dest,$objet,$message,$entetes)){ echo "Mail envoyé avec
succès.";}
else
    echo "Un problème est survenu."; exit;
?>
```

PHP - Envoi Email

Fonction mail :

Exemple :

```
<?php

    $dest="contact@chiny.me";
    $objet="Rendez-vous";
    $message="Bonjour\nPrière de se retrouver sur Skype à <b>18h</b>
aujourd'hui.\n Merci et bonne journée";
    $entetes="From: sc@example.com\n";
    $entetes.="Cc: chiny@example.com\n";
    $entetes.="Content-Type: text/html; charset=iso-8859-1";
    if(mail($dest,$objet,$message,$entetes)){ echo "Mail envoyé avec
succès.";}
    else
        echo "Un problème est survenu."; exit;
?>
```

Programmation Web Dynamique

Pr. Rachid DAKIR

Filières : SMI & IGE

FP Ouarzazate

UNIVERSITE IBN ZOHR

Année Universitaire : 2019-2020

PARTIE : IV

PHP - BDD

Plusieurs applications Web nécessitent le stockage des données pour de longues périodes
L'information peut être stockée dans des fichiers :

- Ne nécessite aucun logiciel supplémentaire
- Facile à utiliser
- Ne nécessite pas beaucoup d'espace disque
 - Redondance des données et incohérence
 - Difficulté à accéder aux données
 - Isolation des données: plusieurs fichiers, plusieurs formats
 - Problèmes d'intégrité
 - Problèmes d'atomicité
 - Problèmes de concurrence
 - Problèmes de sécurité

Inconvénients des fichiers

PHP - BDD

□ Base de Données (BD):

Gros ensemble persistant de données structurées et cohérentes exploitable simultanément

Exemples d'utilisation de BD :

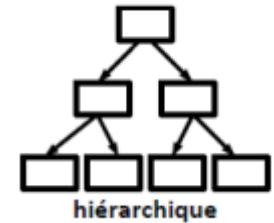
- Gestion des personnels, étudiants, cours, inscriptions, ... De l'université
- Système de réservation de places d'avion chez Royal Air Maroc, de places de train à la ONCF
- Gestion des comptes clients de la banque
- Gestion des commandes chez Amazon.com
- Gestion d'une bibliothèque

PHP - BDD

Différents types de bases de données

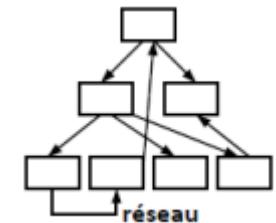
Base hiérarchique

- Contenu organisé dans une structure arborescente.



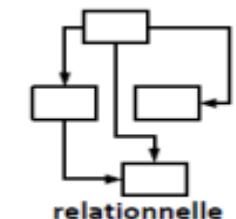
Base en réseau

- Hiérarchique mais avec des relations transverses.



Base relationnelle

- Informations organisées dans des matrices appelées relations ou tables.



Base XML

- S'appuie sur le modèle fourni par XML.

Base objet

- Informations groupées sous forme de collections d'objets.
- Chaque donnée est active et possède ses propres méthodes d'interrogation et d'affectation.

PHP - BDD

□ Utilisateurs (SGBD):

Plusieurs types d'utilisateur:

② **Utilisateurs naïfs** : accèdent à la BD à travers une interface (ex. sur le web)

Utilisateurs sophistiqués : accèdent à la BD en écrivant des requêtes en Langage de manipulation de données (LMD)

Programmeurs d'application : accèdent à la BD en écrivant des programmes d'application

Administrateur BD ("DBA") : donne les droits d'accès aux utilisateurs, définit la structure de stockage et l'organisation physique de la BD, surveille les performances du système, etc

PHP - BDD

BD relationnelle : Terminologie

Une **BD relationnelle** est un **ensemble de données organisées** sous forme de **tables** (appelée aussi **relation**) qui décrivent les types des données.

- En utilisant un **identifiant commun** (clé) entre les tables il est possible de les « **relier** » entre elles.
- Une relation est une table comportant des **colonnes** (dits aussi **attributs**) dont le nom et le type caractérise le contenu qui sera inséré dans la table.
- Les tables contiennent à leur tour des lignes (appelées aussi des **enregistrements** ou **tuples**) qui sont les vraies données.



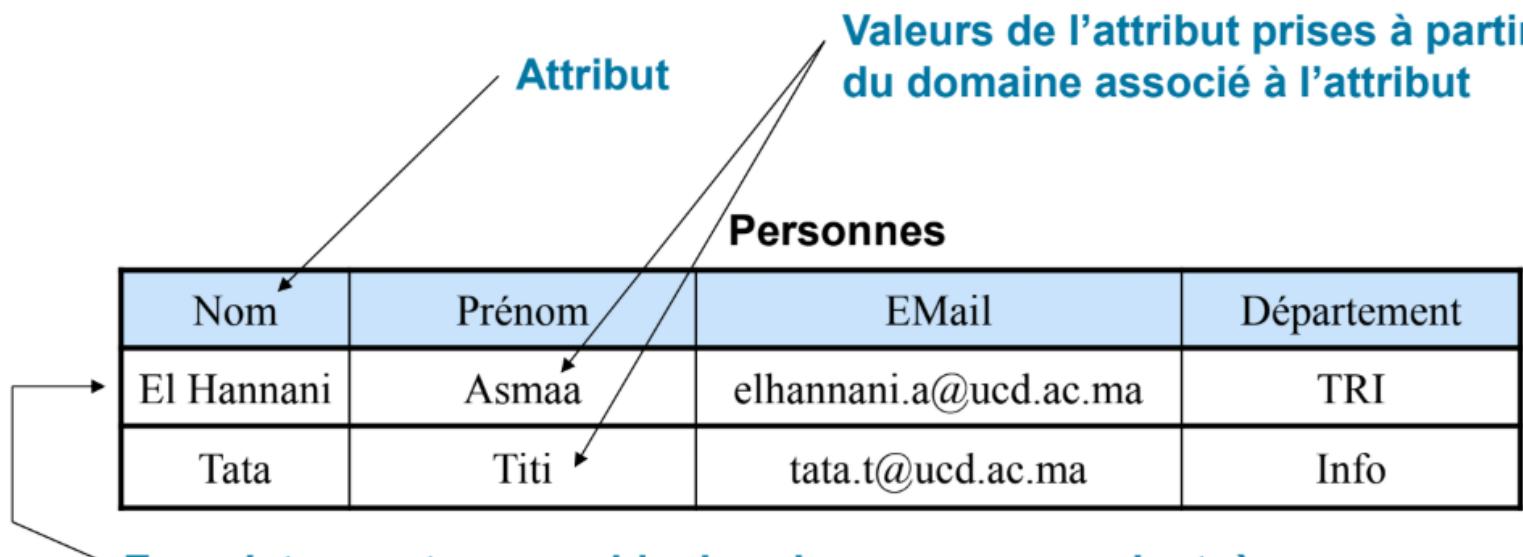
Exemple : Base de données 'Monde'

Table 1: 'table_continents'
Colonnes de la Table 1: 'continent', 'description'
Table 2: 'liste_pays'
Colonnes de la Table 2: 'continent', 'pays', 'carte'
Table 3: 'table_villes'
Colonnes de la Table 3: 'pays', 'ville', 'nb_habitants'

PHP - BDD

Exemple

- Imaginons que l'on veuille stocker dans notre base de données notre carnet d'adresses. On va donc créer la **relation** (table) **Personne** qui aura pour **attributs** : nom, prénom, email, département.



Enregistrement : ensemble de valeurs correspondant à une occurrence particulière de l'objet représentée par la table

PHP - BDD

relation

- ❑ Chaque relation ou table a un nom unique dans la BD
- ❑ Une relation représente soit :
 - un type d'objet d'affaires spécifique sur lequel l'entreprise désire conserver de l'information
ex: Coureur, Course
 - ou une association logique entre deux objets
ex: Résultat – représente l'association logique entre les coureurs et les courses auxquelles ils ont participé
- ❑ Synonymes
 - relation, table, entité

PHP - BDD

Enregistrement

- ❑ Chaque ligne correspond à un enregistrement.
- ❑ Chaque enregistrement est unique dans une relation.
- ❑ Un enregistrement est formé par l'ensemble des valeurs des attributs décrivant une occurrence particulière de l'objet d'affaire représenté par la relation.
- ❑ Chaque enregistrement contient exactement une valeur pour chaque attribut.
- ❑ L'ordre des enregistrements n'a pas d'importance.
- ❑ Le nombre des enregistrements est variable dans le temps.
- ❑ Synonymes
 - Enregistrement, tuple

PHP - BDD

Champ

- ❑ Chaque colonne d'une relation correspond à un attribut et à ses valeurs.
- ❑ L'ordre des colonnes dans une relation n'a pas d'importance.
- ❑ Le nom d'un attribut est unique dans une relation.
- ❑ Le nom d'attribut décrit un aspect de l'objet d'affaires.
 - Ex: « fonction » dans la relation « employé »
- ❑ Un attribut prend des valeurs à partir d'un domaine donné.
- ❑ Chaque valeur d'un attribut est atomique.
- ❑ Synonymes
 - attribut, colonne, champ

PHP - BDD

La plupart de SGBDR sont accessibles via le Langage structuré de requêtes (SQL):

Langage de Définition de Données (LDD)

- Créer, modifier, et supprimer des tables

Langage de Manipulation de Données (LMD)

- Insérer, modifier, supprimer des enregistrements
- Extraire des données (requêtes)

Langage de Contrôle de Données (LCD)

- Créer, modifier, supprimer des droits d'accès

PHP - BDD

MySQL

❑ Avantages :

- très courant chez les hébergeurs, très bonne intégration Apache/PHP,
- OpenSource,
- version cluster (plusieurs serveurs reliés entre eux),
- facilité de prise en main.

❑ Inconvénients :

- faible richesse fonctionnelle (contrôle d'intégrité),
- peu robuste avec des gros volumes,
- C'est le SGBD qu'on va utiliser dans le cadre de ce cours

PHP - BDD

Nous utiliserons phpMyAdmin

- ❑ phpMyAdmin est développé en PHP et offre une interface intuitive pour l'administration des bases de données du serveur.
- ❑ Il est téléchargeable ici : <http://phpmyadmin.sourceforge.net>
- ❑ Cet outil permet de :
 - ❑ créer de nouvelles bases
 - ❑ créer/modifier/supprimer des tables
 - ❑ afficher/ajouter/modifier/supprimer des tuples dans des tables
 - ❑ effectuer des sauvegarde de la structure et/ou des données
 - ❑ effectuer n'importe quelle requête
 - ❑ gérer les privilèges des utilisateurs

PHP - BDD

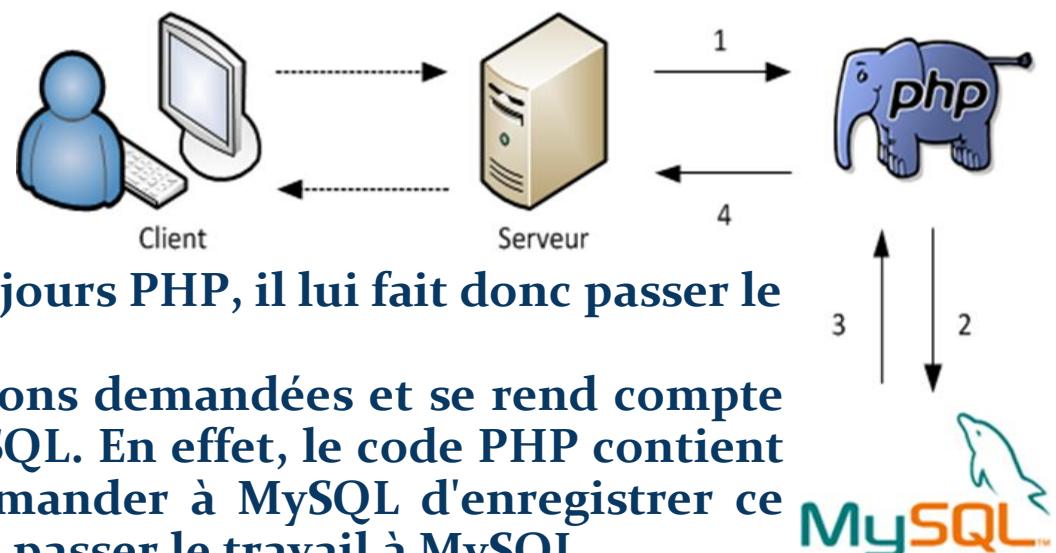
Le Structured Query Language est un langage standardisé qui permet d'effectuer des opérations sur des bases de données.

Il se compose de:

- Langage de Définition de Données (LDD) permettant de gérer les structures de la base
- Langage de Manipulation de Données (LMD) pour interagir avec les données.
- Langage de Contrôle de Données (LCD) pour Créer, modifier, supprimer des droits d'accès

PHP - BDD

PHP fait la jonction avec MySQL



1. Le serveur utilise toujours PHP, il lui fait donc passer le message.
2. PHP effectue les actions demandées et se rend compte qu'il a besoin de MySQL. En effet, le code PHP contient à un endroit "Va demander à MySQL d'enregistrer ce message". Il fait donc passer le travail à MySQL.
3. MySQL fait le travail que PHP lui avait soumis et lui répond "OK, c'est bon !"
4. PHP renvoie au serveur que MySQL a bien fait ce qui lui était demandé.

PHP - BDD

Pour pouvoir travailler avec la base de données en PHP, il faut d'abord s'y connecter.

Comment se connecte-t-on à la base de données en PHP ?

- PHP propose plusieurs moyens de se connecter à une base de données MySQL :
 - ✓ L'**extension mysql_** : ce sont des fonctions qui permettent d'accéder à une base de données MySQL et donc de communiquer avec MySQL. Leur nom commence toujours par mysql_. Toutefois, ces fonctions sont vieilles et on recommande de ne plus les utiliser aujourd'hui.
 - ✓ L'**extension mysqli_** : ce sont des fonctions améliorées d'accès à MySQL. Elles proposent plus de fonctionnalités et sont plus à jour.
 - ✓ L'**extension PDO** : c'est un outil complet qui permet d'accéder à n'importe quel type de base de données. On peut donc l'utiliser pour se connecter aussi bien à MySQL que PostgreSQL ou Oracle.

PHP - BDD

Interfaçage avec une base de données

Principe

Avec le SGBD MySQL, les fonctions de manipulation sont :

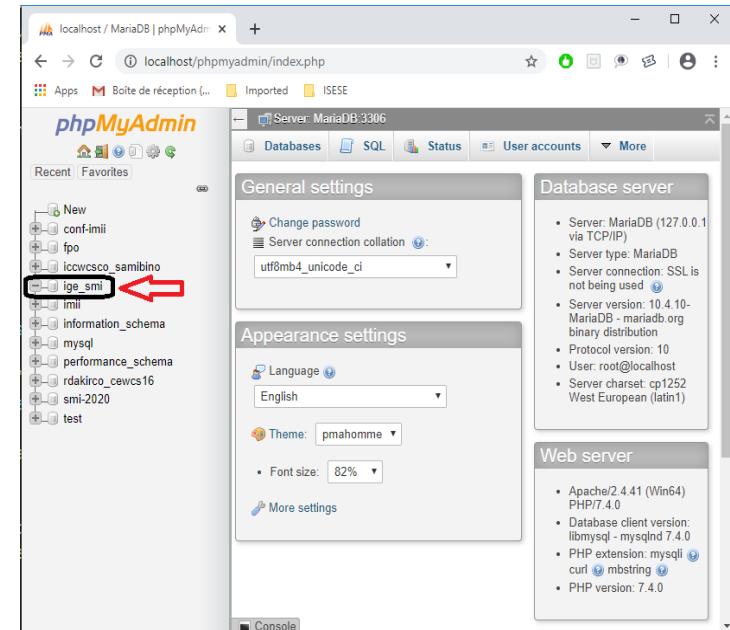
- mysql_connect
- mysql_select_db
- mysql_query
- mysql_close

PHP - BDD

A- Connexion à un serveur MYSQL en PHP

- La première opération à réaliser pour accéder à MYSQL via un script PHP correspond à la connection à un serveur de base de données MYSQL.
- Le PHP nous fournit donc deux API pour nous connecter à MySQL et manipuler nos bases de données (sans passer par phpMyAdmin).
 - ❑ L'extension MySQLi ;
 - ❑ L'extension PDO (PHP Data Objects).

Quelle API préférer : MySQLi ou PDO ?



PHP - BDD

A- Connexion à un serveur MYSQL

Quelle API préférer : MySQLi ou PDO ?

Il existe notamment une différence notable entre ces deux API :

- ❑ Eextension MySQLi fonctionne 'avec les bases de données MySQL
- ❑ PDO fonctionne avec 12 systèmes de bases de données différents.
- ❑ Pour cette raison, nous préférerons généralement le PDO car si vous devez un jour utiliser un autre système de bases de données, le changement sera beaucoup plus simple que si vous avez tout codé en MySQLi auquel cas vous devrez réécrire le code dans son ensemble.
- ❑ MySQLi et PDO sont tous les deux orienté objet (bien que MySQLi propose également une API en procédural),

PHP - BDD

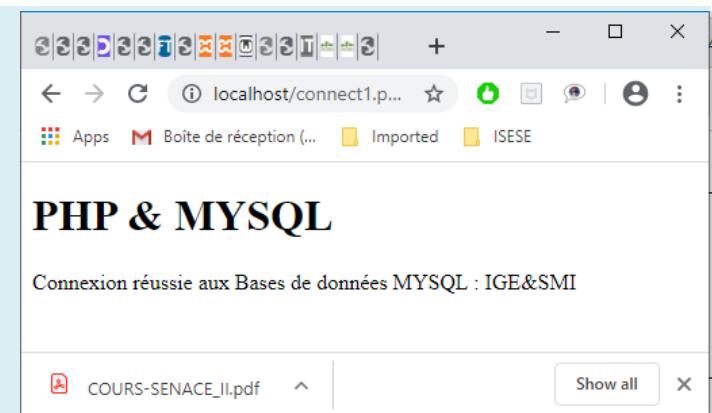
A- Connexion à un serveur MYSQL

- ❑ L'extension PDO (PHP Data Objects).
- Pour se connecter à MySQL. Nous allons avoir besoin de 4 renseignements :
 - ❖ Le nom de l'hôte
 - ❖ La base
 - ❖ Le login
 - ❖ Le mot de passe

PHP - BDD

Procédurable : Mysqli_connect

```
<html> <head>
    <title>SMI & IGE</title>
</head>
<body>
    <h1>PHP & MYSQL</h1>
    <?php
        $servername = 'localhost';
        $username = 'root';
        $password = "";
        //On établit la connexion
        $conn = mysqli_connect($servername, $username, $password);
        //On vérifie la connexion
        if(!$conn){ die('Erreur : ' .mysqli_connect_error()); }
        echo 'Connexion réussie aux Bases de données MYSQL : IGE&SMI';
    ?>
</body></html>
```



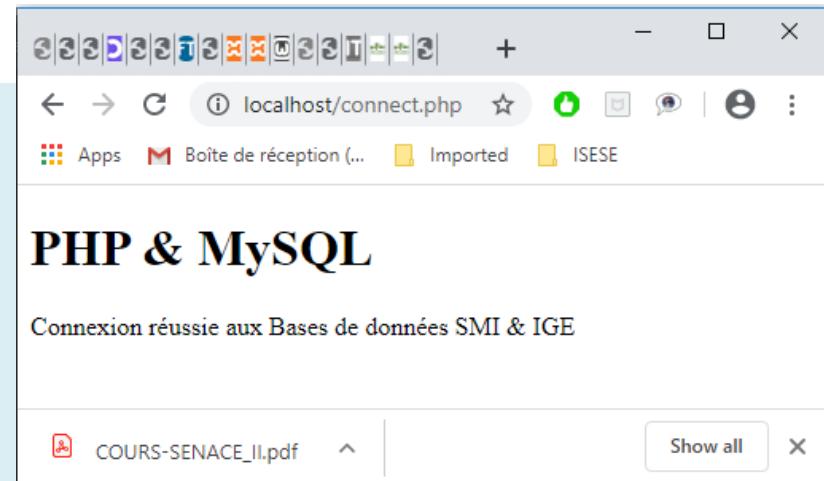
PHP - BDD

Orienté Objet : Mysqli_connect

```
<html> <head><title>Cours PHP / MySQL</title>
    <meta charset="utf-8">
</head>
<body> <h1>Bases de données MySQL</h1>
<?php
    $servername = 'localhost';
    $username = 'root';
    $password = "";

    //On établit la connexion
    $conn = new mysqli($servername, $username, $password);

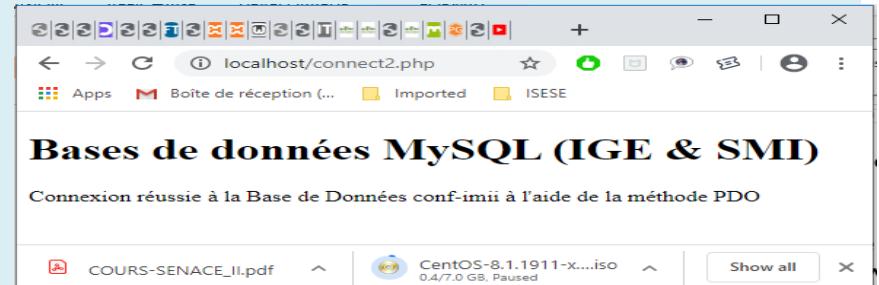
    //On vérifie la connexion
    if($conn->connect_error){
        die('Erreur : ' . $conn->connect_error);
    }
    echo 'Connexion réussie';
?
</body></html>
```



PHP - BDD

Orienté Objet : PDO

```
<html> <head>      <title> Cours PHP / MySQL </title> </head> <body>
    <h1>Bases de données MySQL (IGE & SMI)</h1>
    <?php
        $servername = 'localhost';
        $username = 'root';
        $password = "";
        //On essaie de se connecter
        try{
            $conn = new PDO("mysql:host=$servername;dbname=conf-imii", $username, $password);
            //On définit le mode d'erreur de PDO sur Exception
            $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            echo 'Connexion réussie à la Base de Données conf-imii à l\'aide de la méthode PDO';
        }
        /*On capture les exceptions si une exception est lancée et on affiche les informations relatives à
        celle-ci*/
        catch(PDOException $e){
            echo "Erreur : " . $e->getMessage();
        }
    ?>
</body> </html>
```



PHP - BDD

Le Structured Query Language est un langage standardisé qui permet d'effectuer des opérations sur des bases de données.

Il se compose de:

- ❑ Langage de Définition de Données (LDD) permettant de gérer les structures de la base
- ❑ Langage de Manipulation de Données (LMD) pour interagir avec les données.
- ❑ Langage de Contrôle de Données (LCD) pour Créer, modifier, supprimer des droits d'accès

?

Attention:

Certaines syntaxes ou fonctions sont propres au système de gestion de base de données utilisé.

PHP - BDD

Langage de Définition de Données (LDD) permettant de gérer les structures de la base

□ Les commandes principales sont :

- Pour les bases : create, alter, drop
- Pour les tables : create, alter, rename, drop
- Pour les indexes : create, drop
- La gestion des vues et des triggers

PHP - BDD

- ❑ Crédation d'une base de données :

CREATE DATABASE Nom_base [create_specification]

- ❑ Suppression d'une base de données :

DROP DATABASE [IF EXISTS] db_name

- ❑ Modification d'une base de données :

ALTER DATABASE db_name alter_specification

PHP - BDD

- La création d'une relation utilise la commande CREATE TABLE selon la syntaxe suivante :

```
CREATE TABLE nom_relation (nom_attribut TYPE_ATTRIBUT ))
```

Exemple :

```
CREATE TABLE Personne (
    nom VARCHAR(40),
    prenom VARCHAR(40),
    email VARCHAR(40),
    laboratoire VARCHAR(30)
)
```

PHP - BDD

□ Les attributs peuvent avoir des types très différents :

- Nombre entier signé ou non (numéro d'ordre, nombre d'objets)
- Nombre à virgule (prix, température)
- Date et heure (date de naissance, heure de l'insertion)
- Enumération (un laboratoire parmi une liste)
- Ensemble (une ou des compétences parmi une liste)

Il s'agit de choisir le type le plus adapté aux besoins. Les types requièrent une plus ou moins grande quantité de données à stocker : il vaut mieux choisir un type varchar pour stocker un nom qu'un type longtext.

Voir : www.mysql.com pour plus de détails

PHP - BDD

- LA commande DROP TABLE prend en paramètre le nom de la table à supprimer.
Toutes les données qu'elle contient sont supprimées et sa définition aussi.

DROP TABLE relation

Exemple :

DROP TABLE Personnes

PHP - BDD

- La commande DROP TABLE prend en paramètre le nom de la table à supprimer. Toutes les données qu'elle contient sont supprimées et sa définition aussi.

DROP TABLE relation

- Exemple : DROP TABLE Personnes

La création d'une relation par CREATE TABLE n'en rend pas définitives les spécifications. Il est possible d'en modifier la définition par la suite, à tout moment par la commande :

ALTER TABLE relation.

Voici ce qu'il est possible de réaliser :

- ajouter/supprimer un attribut
- créer/supprimer une clé primaire
- ajouter une contrainte d'unicité (interdire les doublons)
- changer la valeur par défaut d'un attribut
- changer totalement la définition d'un attribut
- changer le nom de la relation et ajouter/supprimer un index

PHP - BDD

❑ Langage de manipulation de données :

- Ajouter les données (INSERT)
- Modifier les données (UPDATE)
- Supprimer les données (DELETE)
- Consulter les données (SELECT)

PHP - BDD

❑ Langage de manipulation de données :

Pour ajouter un enregistrement à une relation, il faudra préciser la valeur pour chacun des attributs.

Si certaines valeurs sont omises, alors les valeurs par défauts définie lors de la création de la relation seront utilisées.

Si on ne dispose pas non plus de ces valeurs par défaut, alors MySQL mettra 0 pour un nombre, "" pour une chaîne, 0000-00-00 pour une date, 00:00:00 pour une heure, 000000000000 pour un timestamp.

Syntaxe : INSERT INTO relation(liste des attributs) VALUES(liste des valeurs)

Exemple :

```
INSERT INTO Personnes(nom,prenom) VALUES('Martin','Jean')
```

PHP - BDD

❑ Langage de manipulation de données :

Pour modifier un ou des enregistrement(s) d'une relation, il faut préciser un critère de sélection des enregistrement à modifier (clause WHERE), il faut aussi dire quels sont les attributs dont on va modifier la valeur et quelles sont ces nouvelles valeurs (clause SET).

❑ **Syntaxe : UPDATE relation SET attribut=valeur, ... [WHERE condition]**

Exemple :

modifier le numéro de téléphone de Martin Pierre:

UPDATE TABLE Personnes SET telephone='0156281469' WHERE nom='Martin' AND prenom = 'Pierre'.

PHP - BDD

❑ Langage de manipulation de données :

Attention, la suppression est définitive !

Syntaxe : **DELETE FROM relation [WHERE condition]**

Exemple : **DELETE FROM Personnes WHERE nom='Martin' AND prenom='Marc'**

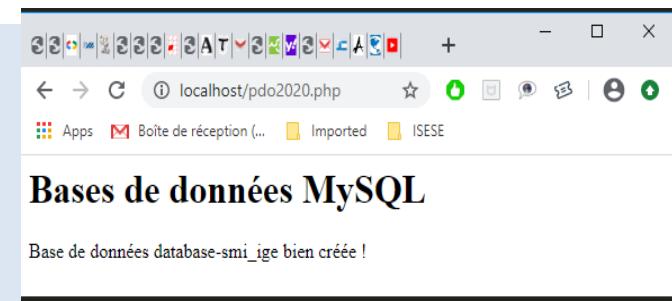
Pour vider une table de tous ces éléments, ne pas mettre de clause WHERE.

PHP - BDD

Création d'une Base de données

PDO

```
<html> <head> <title>Cours PHP / MySQL</title>
</head> <body> <h1>Bases de données MySQL</h1>
<?php
$servername = 'localhost';
$username = 'root';
$password = "";
try{ $dbco = new PDO("mysql:host=$servername", $username, $password);
$dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$sql = "CREATE DATABASE database-smi_ige";
$dbco->exec($sql); echo 'Base de données créée bien créée !'; }
catch(PDOException $e)
{ echo "Erreur : " . $e->getMessage(); }
?>
</body> </html>
```



PHP - BDD

Création d'une table

❑ PDO

```
<?php
    $servname = 'localhost'; $dbname = 'database_2020'; $user = 'root'; $pass = '';
    try{
        $dbco = new PDO("mysql:host=$servname;dbname=$dbname", $user, $pass);
        $dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $sql = "CREATE TABLE Clients(
            Id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
            Nom VARCHAR(30) NOT NULL,
            Prenom VARCHAR(30) NOT NULL,
            Adresse VARCHAR(70) NOT NULL,
            Ville VARCHAR(30) NOT NULL,
            Codepostal INT UNSIGNED NOT NULL,
            Pays VARCHAR(30) NOT NULL,
            Mail VARCHAR(50) NOT NULL,
            DateInscription TIMESTAMP,
            UNIQUE(Mail))";
        $dbco->exec($sql); echo 'Table bien créée !';
    } catch(PDOException $e){
        echo "Erreur : " . $e->getMessage();
    }
?>
```

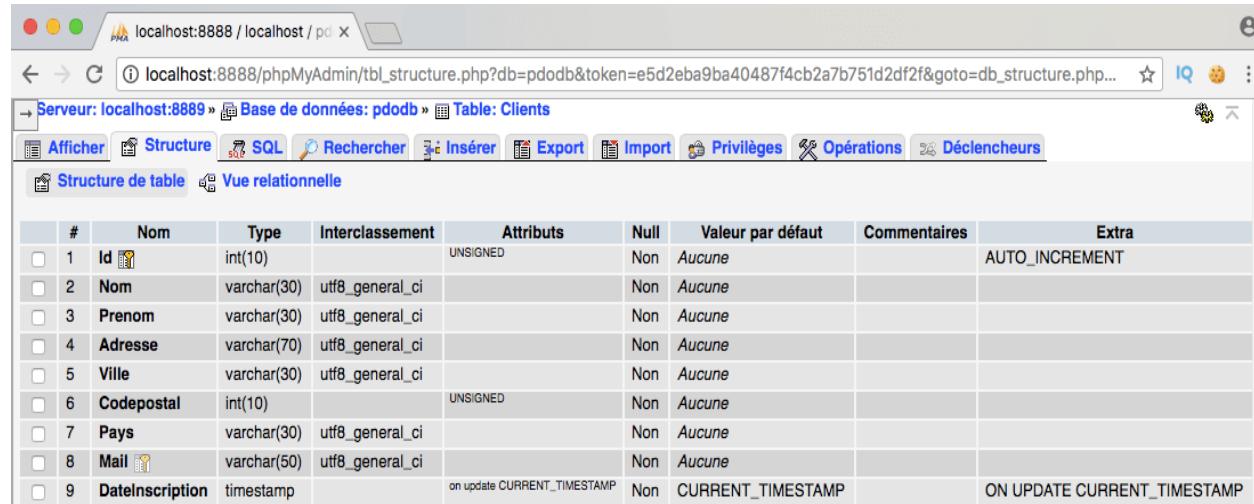


PHP - BDD

Voyons immédiatement en pratique comment on pourrait créer une table « Clients » dans notre base « database_smi ».

Notre table va contenir 9 colonnes :

- Id
- Nom
- Prenom
- Adresse
- Ville
- CodePostal
- Pays
- Mail
- DateInscription



The screenshot shows the phpMyAdmin interface for the 'Clients' table. The table has 9 columns:

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
1	Id	int(10)		UNSIGNED	Non	Aucune		AUTO_INCREMENT
2	Nom	varchar(30)	utf8_general_ci		Non	Aucune		
3	Prenom	varchar(30)	utf8_general_ci		Non	Aucune		
4	Adresse	varchar(70)	utf8_general_ci		Non	Aucune		
5	Ville	varchar(30)	utf8_general_ci		Non	Aucune		
6	Codepostal	int(10)		UNSIGNED	Non	Aucune		
7	Pays	varchar(30)	utf8_general_ci		Non	Aucune		
8	Mail	varchar(50)	utf8_general_ci	on update CURRENT_TIMESTAMP	Non	CURRENT_TIMESTAMP		ON UPDATE CURRENT_TIMESTAMP
9	DateInscription	timestamp						

PHP - BDD

Insertion des enregistrements

```
<?php
    $servname = 'localhost'; $dbname = 'database_2020'; $user = 'root'; $pass = "";
    try{      $dbc = new PDO("mysql:host=$servname;dbname=$dbname", $user, $pass);
              $dbc->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
              $dbc->beginTransaction();
              $sql1 = "INSERT INTO Clients(Nom,Prenom,Adresse,Ville,Codepostal,Pays,Mail)
                        VALUES('MOHAMED','ALI','132, RUE NAJAH ','RABAT',44000,'MAROC','med-ali@gmail.com')");
              $dbc->exec($sql1);
              $sql2 = "INSERT INTO Clients(Nom,Prenom,Adresse,Ville,Codepostal,Pays,Mail)
                        VALUES('FARAH','NAWAL','133,RUE TISSIR III','SFAXE',33000,'TUNISIE','farah-
nawal@ladis.com')";
              $dbc->exec($sql2);
              $dbc->commit();
              echo 'Entrées ajoutées dans la table';
        }
        catch(PDOException $e){
            $dbc->rollBack();
            echo "Erreur : " . $e->getMessage();
        }
    ?>
```

PHP - BDD

Exemple: connexion et afficher des résultats d'une requête

Un résumer tout ce qu'on vient de voir en SQL, de la connexion via PDO à l'affichage du résultat de la requête :

ID	nom	possesseur	console	prix	nbre_joueurs_max	commentaires
1	Super Mario Bros	Florent	NES	4	1	Un jeu d'anthologie !
2	Sonic	Patrick	Megadrive	2	1	Pour moi, le meilleur jeu au monde !
3	Zelda : ocarina of time	Florent	Nintendo 64	15	1	Un jeu grand, beau et complet comme on en voit rarement de nos jours
4	Mario Kart 64	Florent	Nintendo 64	25	4	Un excellent jeu de kart !
5	Super Smash Bros Melee	Michel	GameCube	55	4	Un jeu de baston délirant !

```
<?php
try
{
    // On se connecte à MySQL
    $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
    $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '', $pdo_options);

    // On récupère tout le contenu de la table jeux_video
    $reponse = $bdd->query('SELECT * FROM jeux_video');

    // On affiche chaque entrée une à une
    while ($donnees = $reponse->fetch())
    {
        ?>
        <p>
            <strong>Jeu</strong> : <?php echo $donnees['nom']; ?><br />
            Le possesseur de ce jeu est : <?php echo $donnees['possesseur']; ?>, et il le vend à <?php echo $donnees['prix']; ?> euros !<br />
            Ce jeu fonctionne sur <?php echo $donnees['console']; ?> et on peut y jouer à <?php echo $donnees['nbre_joueurs_max']; ?> au maximum<br />
            <?php echo $donnees['possesseur']; ?> a laissé ses commentaires sur <?php echo $donnees['nom']; ?> : <em><?php echo $donnees['commentaires']; ?></em>
        </p>
        <?php
    }

    $reponse->closeCursor(); // Termine le traitement de la requête
}

catch(Exception $e)
{
    // En cas d'erreur précédemment, on affiche un message et on arrête tout
    die('Erreur : '.$e->getMessage());
}
?>
```

PHP - BDD

Afficher seulement le contenu de quelques champs

➤ Dans l'exemple précédent, on peut ne pas afficher tous les champs . Par exemple, si on veut lister juste les noms des jeux, on pourra utilisé la requête SQL suivante :

```
SELECT nom FROM jeux_video
```

➤ Reprenons le code complet précédent et adaptons-le pour afficher un nom de jeu par ligne :

```
<?php
try
{
    $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
    $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '', $pdo_options);

    $reponse = $bdd->query('SELECT nom FROM jeux_video');

    while ($donnees = $reponse->fetch())
    {
        echo $donnees['nom'] . '<br />';
    }

    $reponse->closeCursor();
}
catch(Exception $e)
{
    die('Erreur : '.$e->getMessage());
}
?>
```

Retenez bien en particulier les deux choses suivantes :

- La connexion à la base de données n'a besoin d'être faite qu'une seule fois, au début de la page.
- Il faut fermer les résultats de recherche avec closeCursor() après avoir traité chaque requête.

PHP - BDD

Les critères de sélection

➤ Imaginons qu'on souhaite obtenir uniquement la liste des jeux disponibles de la console "Nintendo 64" et qu'on souhaite les trier par prix croissant .

➤ A cet effet, on doit modifier nos requêtes SQL, il est possible de filtrer et trier très facilement nos données. Nous allons nous intéresser ici aux mots-clé suivants du langage SQL:

- WHERE
- ORDER BY
- LIMIT

WHERE: Supposons par exemple que je veuille lister uniquement les jeux appartenant à Patrick. La requête au début sera la même qu'avant, mais je rajouterai à la fin WHERE possesseur='Patrick'. Ca nous donne la requête :

```
SELECT * FROM jeux_video WHERE possesseur='Patrick'
```

Les critères de sélection: exemple

```
<?php
try
{
    $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
    $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '', $pdo_options);

    $reponse = $bdd->query('SELECT nom, possesseur FROM jeux_video WHERE
    possesseur=\'Patrick\'');

    while ($donnees = $reponse->fetch())
    {
        echo $donnees['nom'] . ' appartient à ' . $donnees['possesseur'] . '<br /
    ';
    }

    $reponse->closeCursor();
}
catch(Exception $e)
{
    die('Erreur : '. $e->getMessage());
}
?>
```

- On peut changer le nom du possesseur (par exemple "WHERE possesseur='Michel'"), ça n'affichera que les jeux appartenant à Michel .
- Il est par ailleurs possible de combiner plusieurs conditions. Par exemple, si on veut lister les jeux de Patrick qu'il vend à moins de 20 euros, on combinera les critères de sélection à l'aide du mot-clé AND (qui signifie "et") :

```
SELECT * FROM jeux_video WHERE possesseur='Patrick'
AND prix < 20
```

PHP - BDD

Les critères de sélection : ORDER BY

ORDER BY nous permet d'ordonner nos résultats. Nous pourrions ainsi classer les résultats en fonction de leur prix ! La requête SQL serait :

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

Traduction : "Sélectionner tous les champs de jeux_video et ordonner les résultats par prix croissant.".

Les critères de sélection : ORDER BY

Pour classer par ordre décroissant, il suffit de rajouter le mot-clé DESC à la fin :

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

```
<?php
try
{
    $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
    $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '', $pdo_options);

    $reponse = $bdd->query('SELECT nom, prix FROM jeux_video ORDER BY prix');

    while ($donnees = $reponse->fetch())
    {
        echo $donnees['nom'] . ' coûte ' . $donnees['prix'] . ' EUR<br />';
    }

    $reponse->closeCursor();
}
catch(Exception $e)
{
    die('Erreur : '.$e->getMessage());
}
?>
```

Les critères de sélection: LIMIT

LIMIT nous permet de ne sélectionner qu'une partie des résultats (par exemple les 20 premiers). C'est très utile lorsqu'il y a beaucoup de résultats et que vous souhaitez les paginer (c'est-à-dire par exemple afficher les 30 premiers résultats sur la page 1, les 30 suivants sur la page 2, etc.). Il faut rajouter à la fin de la requête le mot clé LIMIT, suivi de 2 nombres séparés par une virgule. Par exemple :

Ces deux nombres ont un sens bien précis :

On indique tout d'abord à partir de quelle entrée on commence à lire la table. Ici, j'ai mis 0, ce qui correspond à la première entrée. Attention, cela n'a rien à voir avec le champ ID ! Imaginez qu'une requête retourne 100 résultats : LIMIT tronquera à partir du premier résultat si vous indiquez 0, à partir du 21ème si vous indiquez 20, etc. Ensuite, le deuxième nombre indique combien d'entrées on doit sélectionner. Ici, j'ai mis 20, on prendra donc 20 entrées.

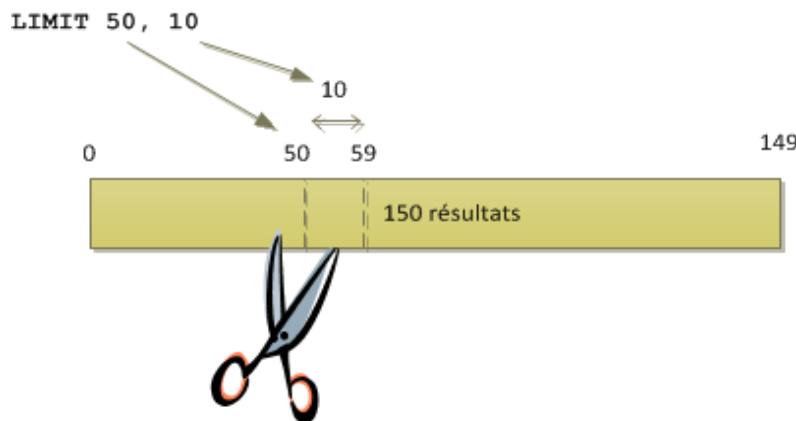
PHP - BDD

Les critères de sélection(LIMIT): exemple

LIMIT 0, 20 : affiche les 20 premières entrées.

LIMIT 5, 10 : affiche de la sixième à la quinzième entrée.

LIMIT 10, 2 : affiche la onzième et la douzième entrée.



```
<?php
try
{
    $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
    $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '',
$pdo_options);

    $reponse = $bdd->query('SELECT nom FROM jeux_video LIMIT 0, 10');

    echo '<p>Voici les 10 premières entrées de la table jeux_video :</p>';
    while ($donnees = $reponse->fetch())
    {
        echo $donnees['nom'] . '<br />';
    }

    $reponse->closeCursor();
}
catch(Exception $e)
{
    die('Erreur : '. $e->getMessage());
}
?>
```

PHP - BDD- FORMULAIRE

Construire des requêtes en fonction de variables ??????????

Les requêtes que nous avons étudiées jusqu'ici étaient simples et effectuaient toujours la même opération. Or, les choses deviennent intéressantes quand on utilise des variables de PHP dans les requêtes.
Solution : les requêtes préparées avec des marqueurs "?"

On va dans un premier temps "préparer" la requête sans sa partie variable, que l'on représentera avec un marqueur sous forme de point d'interrogation :

```
<?php  
$req = $bdd->prepare('SELECT nom FROM jeux_video  
WHERE possesseur = ?');  
?>
```

Au lieu d'exécuter la requête avec query() comme la dernière fois, on appelle ici prepare().

PHP - BDD- FORMULAIRE

Construire des requêtes en fonction de variables

La requête est alors prête, sans sa partie variable. Maintenant, nous allons exécuter la requête en appelant `execute` et en lui transmettant la liste des paramètres :

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

La requête est alors exécutée à l'aide des paramètres que l'on a indiqués sous forme d'array.

S'il y a plusieurs marqueurs, il faut indiquer les paramètres dans le bon ordre :

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

Le premier point d'interrogation de la requête sera remplacé par le contenu de la variable `$_GET['possesseur']` et le second point d'interrogation sera remplacé par le contenu de `$_GET['prix_max']`. Le contenu de ces variables aura été automatiquement sécurisé pour prévenir les risques d'injection SQL.

PHP - BDD- FORMULAIRE

Exemple:

Essayons de construire une page capable de lister les jeux appartenant à une personne et dont le prix ne dépasse pas une certaine somme :

```
<?php
try
{
    $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
    $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '', $pdo_options);

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

    echo '<ul>';
    while ($donnees = $req->fetch())
    {
        echo '<li>' . $donnees['nom'] . ' (' . $donnees['prix'] . ' EUR)</li>';
    }
    echo '</ul>';

    $req->closeCursor();
}
catch(Exception $e)
{
    die('Erreur : '.$e->getMessage());
}
?>
```

Construire des requêtes en fonction de variables : les marqueurs nominatifs

Si la requête contient beaucoup de parties variables, il peut être plus pratique de nommer les marqueurs plutôt que d'utiliser des points d'interrogation. Voici comment on s'y prendrait :

```
<?php  
$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']));  
?>
```

Les points d'interrogations ont été remplacés par les marqueurs nominatifs :possesseur et :prixmax (ils commencent par le symbole deux-points comme vous le voyez).

Ces marqueurs sont remplacés par les variables à l'aide cette fois d'un array associatif. Cela permet parfois plus de clarté quand il y a beaucoup de paramètres. De plus, contrairement aux points d'interrogations, nous ne sommes cette fois plus obligés d'envoyer les variables dans le même ordre que la requête.

PHP - BDD- FORMULAIRE

Construire des requêtes Ecrire des données

Dans ce qui va suivre, on va découvrir comment on peut ajouter et modifier des données dans la base. Pour cela, nous allons aborder de nouvelles requêtes SQL fondamentales à connaître :

- INSERT
- UPDATE
- DELETE

INSERT : ajouter des données:
Supposons que nous voulons
veut ajouter une nouvelle
entrée à la table "jeux_video" ,
comment faire ?

Première solution :
PhpMyAdmin permet de
rajouter de nouvelles entrées
dans la table. Mais ce qui nous
intéresse ici, c'est de le faire via
un script PHP et une requête
SQL !

La requête INSERT INTO permet d'ajouter une entrée

Pour rajouter une entrée, vous aurez besoin de connaître la requête SQL. En voici une par exemple qui rajoute une entrée :

```
INSERT INTO jeux_video(ID, nom, possesseur, console, prix, nbre_joueurs_max, commentaires)  
VALUES('', 'Battlefield 1942', 'Patrick', 'PC', 45, 50, '2nde guerre mondiale')
```

Etudions un peu cette requête : d'abord, vous devez commencer par les mots-clé INSERT INTO qui indiquent que vous voulez insérer une entrée.

Vous précisez ensuite le nom de la table (ici jeux_video), puis listez entre parenthèses les noms des champs dans lesquels vous souhaitez placer des informations.

Enfin, et c'est là qu'il ne faut pas se tromper, vous inscrivez VALUES suivi des valeurs à insérer dans le même ordre que les champs que vous avez indiqués.

PHP - BDD- FORMULAIRE

La requête INSERT INTO

Vous remarquerez que pour le premier champ (ID), on a laissé des apostrophes vides. C'est voulu : le champ a la propriété "auto_increment", MySQL mettra donc le numéro d'ID lui-même. On pourrait même se passer du champ ID dans la requête :

```
INSERT INTO jeux_video(nom, possesseur, console, prix, nbre_joueurs_max, commentaires)
VALUES('Battlefield 1942', 'Patrick', 'PC', 45, 50, '2nde guerre mondiale')
```

✓ C'est encore plus simple ! Le champ ID sera de toute façon automatiquement rempli par MySQL, il est donc inutile de le lister.

PHP - BDD- FORMULAIRE

La requête INSERT INTO Application en PHP

**Utilisons cette
requête SQL au sein
d'un script PHP.
Cette fois, au lieu de
faire appel à query()
, on va utiliser exec()
qui est prévue pour
exécuter des
modifications sur la
base de données :**

```
<?php
try
{
    $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
    $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '', $pdo_options);

    // On ajoute une entrée dans la table jeux_video
    $bdd->exec('INSERT INTO jeux_video(nom, possesseur, console, prix, nbre_joueurs_max,
commentaires) VALUES(\'Battlefield 1942\', \'Patrick\', \'PC\', 45, 50, \'2nde guerre
mondiale\')');

    echo 'Le jeu a bien été ajouté !';
}
catch(Exception $e)
{
    die('Erreur : '. $e->getMessage());
}
?>
```

PHP - BDD- FORMULAIRE

Insertion de données variables grâce à une requête préparée

Si on choisit d'utiliser une requête préparée recommandable si vous souhaitez insérer des variables), le fonctionnement est en fait exactement le même que précédemment:

```
<?php
$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
));
echo 'Le jeu a bien été ajouté !';
?>
```

PHP - BDD- FORMULAIRE

UPDATE : modifier des données

Vous venez de rajouter Battlefield dans la BDD, tout s'est bien passé.
Mais... vous vous rendez compte avec stupeur que Battlefield se joue en fait à 32 joueurs maximum (au lieu de 50), et que en plus son prix a baissé : on le trouve à 10 euros (au lieu de 45).

Avec une petite requête SQL on peut arranger ça. En effet, en utilisant UPDATE vous allez pouvoir modifier l'entrée qui pose problème :

```
UPDATE jeux_video SET prix = 10, nbre_joueurs_max = 32 WHERE ID = 51
```

UPDATE : modifier des données Comment ça marche ?

Tout d'abord, le mot-clé UPDATE permet de dire qu'on va modifier une entrée.

Ensuite, le nom de la table (`jeux_video`). Le mot-clé SET, qui sépare le nom de la table de la liste des champs à modifier.

Viennent ensuite les champs qu'il faut modifier, séparés par des virgules. Ici, on modifie le champ "prix", on lui affecte la valeur "10" (`prix = 10`), et de même pour le champ `nbre_joueurs_max`. Les autres champs ne seront pas modifiés.

Enfin, le mot-clé WHERE est tout simplement indispensable. Il nous permet de dire à MySQL quelle entrée il doit modifier (sinon toutes les entrées seraient affectées !). On se base très souvent sur le champ ID pour indiquer quelle entrée doit être modifiée. Ici, on suppose que `Battlefield` a été enregistré sous l'ID n°51.

```
UPDATE jeux_video SET prix = 10, nbre_joueurs_max = 32 WHERE ID = 51
```

PHP - BDD- FORMULAIRE

UPDATE : modifier des données

De la même manière, en PHP on fait appel à exec() pour effectuer des modifications :

```
<?php  
$bdd->exec('UPDATE jeux_video SET prix = 10, nbre_joueurs_max = 32 WHERE nom = \'Battlefield 1942\'');  
?>
```

Notez que cet appel renvoie le nombre de lignes modifiées. Essayez de récupérer cette valeur dans une variable et de l'afficher, par exemple comme ceci :

```
<?php  
$nb_modifs = $bdd->exec('UPDATE jeux_video SET possesseur = \'Florent\' WHERE  
possesseur = \'Michel\'');  
echo $nb_modifs . ' entrées ont été modifiées !';  
?>
```

PHP - BDD- FORMULAIRE

DELETE : supprimer des données

❖ Enfin, voilà une dernière requête qui pourra se révéler utile : DELETE. Rapide et simple à utiliser, elle est quand même un poil dangereuse : après suppression, il n'y a aucun moyen de récupérer les données, alors faites bien attention !

❖ Voici comment on supprime par exemple l'entrée de Battlefield :

```
DELETE FROM jeux_video WHERE  
nom='Battlefield 1942'
```

Il n'y a rien de plus facile :

- DELETE FROM : pour dire "supprimer dans".
- jeux_video : le nom de la table.
- WHERE : indispensable pour indiquer quelle(s) entrée(s) doivent être supprimée(s).

Si vous oubliez le WHERE, toutes les entrées seront supprimées ! Cela équivaut à vider la table.

PHP - BDD

Fermer la connexion à la base de données

- ❑ Une fois la connexion à la base de données ouverte, celle-ci reste active jusqu'à la fin de l'exécution de votre script.
- ❑ Pour fermer la connexion avant cela, nous allons devoir utiliser différentes méthodes selon la méthode d'ouverture choisie.



❑ Mysqli (Procédurables)



❑ Mysqli (Orienté Objet)



❑ PDO

PHP - BDD

❑ Mysqli (Orienté Objet)

```
<!DOCTYPE html>
<html>
    <head>
        <title>Cours PHP / MySQL</title>
        <meta charset='utf-8'>
    </head>
    <body>
        <h1>Bases de données MySQL</h1>

        <?php
            $servername = "localhost";
            $username = "root";
            $password = "root";

            $conn = new mysqli($servername, $username, $password);
            if($conn->connect_error){
                die("Erreur : " . $conn->connect_error);
            }
            echo "Connexion réussie";
            //On ferme la connexion
            $conn->close();
        </body>
    </html>
```

PHP - BDD

❑ Mysqli (Procédurable)

Si on utilise MySQLi procédural, on utilisera la fonction **mysqli_close()**

```
<!DOCTYPE html>
<html>
    <head>
        <title>Cours PHP / MySQL</title>
        <meta charset='utf-8'>
    </head>
    <body>
        <h1>Bases de données MySQL</h1>

        <?php
            $servername = "localhost";
            $username = "root";
            $password = "root";

            $conn = mysqli_connect($servername, $username, $password);

            if(!$conn){
                die("Erreur : " . mysqli_connect_error());
            }
            echo "Connexion réussie";

            //On ferme la connexion
            mysqli_close($conn);

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

PHP - BDD

❑ PDO

Si on utilise PDO, il faudra détruire l'objet représentant la connexion et effacer toutes ses références. Nous pouvons faire cela en assignant la valeur **NULL** à la variable gérant l'objet.

```
<html>
    <head>
        <title>Cours PHP / MySQL</title>
        <meta charset='utf-8'>
    </head>
    <body>
        <h1>Bases de données MySQL</h1>

        <?php
            $servername = "localhost";
            $username = "root";
            $password = "root";

            //On tente d'établir la connexion
            try{
                $conn = new PDO("mysql:host=$servername;dbname=bddtest", $username, $password);
                //On définit le mode d'erreur de PDO sur Exception
                $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
                echo "Connexion réussie";
            }

            /*On capture les exceptions si une exception est lancée et on affiche
             *les informations relatives à celle ci*/
            catch(PDOException $e){
                echo "Erreur : " . $e->getMessage();
            }

            //On ferme la connexion
            $conn = null;
        ?>
    </body>
</html>
```

Programmation Web Dynamique

Pr. Rachid DAKIR

Filières : SMI & IGE

FP OUARZAZATE

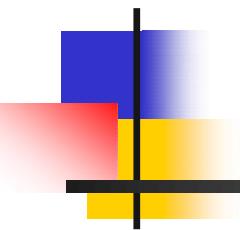
UNIVERSITE IBN ZOHR

Année Universitaire : 2019-2020

PARTIE : V

WEB EN PHP

Cookies et Sessions



Les cookies

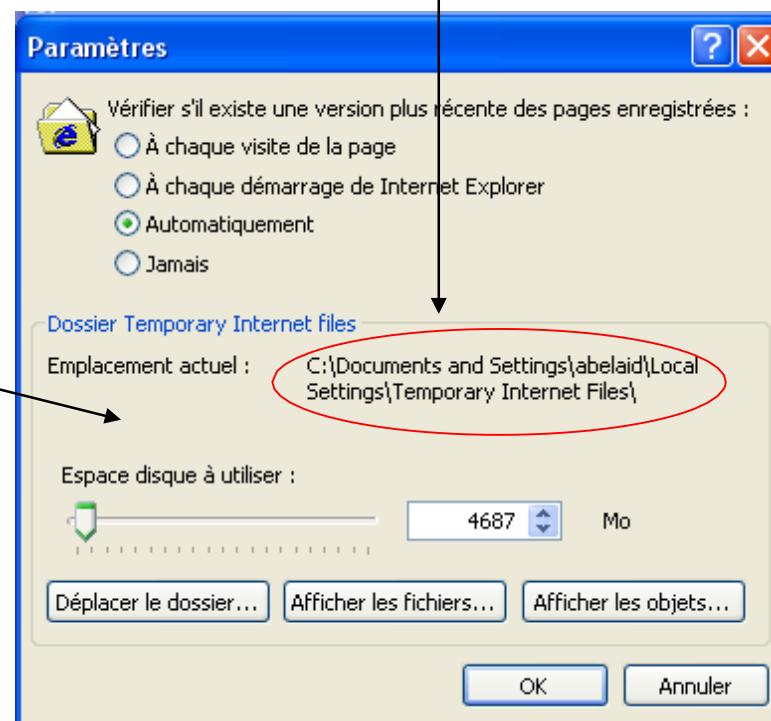
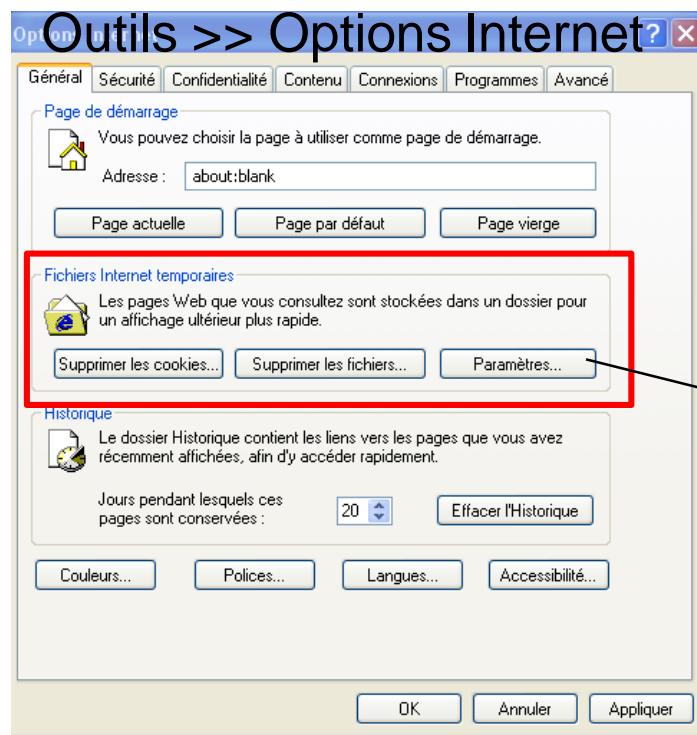
■ Qu'est-ce qu'un cookie ?

- Un petit fichier texte stocké sur le disque dur du visiteur du site
- On peut y sauvegarder plusieurs infos concernant ce visiteur et les réutiliser lors de sa prochaine visite
 - Par exemple, on pourrait très bien stocker dans ce cookie le nom du visiteur et par la suite, afficher son nom à chaque fois qu'il se connectera sur le site (ceci bien sur, s'il n'efface pas les cookies de son disque dur)
 - Ceci n'est possible que si le visiteur a entré lui-même ses informations dans un formulaire sur le site

Les cookies

■ Où sont placés les cookies ?

- Cela dépend du navigateur que vous utilisez
- Pour le navigateur Internet Explorer, les cookies sont stockés dans le chemin suivants :



Les cookies

- Voyons à présent comment créer de tels cookies ?
 - On utilise la fonction `setcookie()`

```
<?php  
/* on définit une durée de vie de notre cookie (en secondes),  
donc un jour dans notre cas*/  
$temps = 24*3600;  
/* on envoie un cookie de nom pseudo portant la valeur smi-ige*/  
setcookie ("smi-ige", "Belaid", time() + $temps);  
?>
```

Les cookies

■ Explications

- On envoie par ce code chez le client (donc le visiteur du site) un cookie de nom **pseudo** portant la valeur **smi-ige**
- De plus, avec **time()**, on impose que le cookie ait une durée de vie de un an (soit en fait l'instant présent plus un an, donc un an)
- Maintenant, si le visiteur ne supprime pas ce cookie, et bien, dans toutes les pages WEB de notre site, on pourra accéder à la variable **\$pseudo** qui contiendra la chaîne de caractères **smi-ige** ou celle que vous avez rentrée dans votre formulaire

Les cookies

■ Exemple

- Supposons que sur une page de notre site WEB
 - Nous souhaitons faire en sorte que si le visiteur vient pour la première fois (ou qu'il a supprimé ses cookies), il aurait alors, la possibilité de saisir son nom dans un formulaire,
 - Ou bien s'il ne s'agit pas de sa première visite, d'afficher tout simplement **Bonjour** puis son **nom**
- On aurait alors le code suivant pour notre page (par exemple **index.php**)

Les cookies

Exemple

```
■ index.php
<html>
<head>
<title>Index du site</title>
<body>
<?php
    if (isset($_COOKIE['pseudo'])) {
        // si le cookie existe
        echo 'Bonjour '.$_COOKIE['pseudo'].' !';
    }
    else {
        echo 'Notre cookie n\'est pas déclaré.';
        // si le cookie n'existe pas, on affiche un formulaire permettant
        // au visiteur de saisir son nom
        echo '<form action=".//traitement.php" method="post">';
        echo 'Votre nom : <input type = "texte" name = "nom"><br />';
        echo '<input type = "submit" value = "Envoyer">';
    }
?>
</body>
</html>
```

Les cookies

Exemple

- `traitement.php`

```
<?php
If (isset($_POST['nom'])) {
$temps = 24*3600;
    // on envoie un cookie de nom pseudo portant la valeur de la variable $nom,
    //c'est-à-dire la valeur qu'a saisie la personne qui a rempli le formulaire
    setcookie ("pseudo", $_POST['nom'], time() + $temps);
    // fonction nous permettant de faire des redirections
    function redirection($url){
        if (headers_sent()){
            print('<meta http-equiv="refresh" content="0;URL='.$url.'">');
        }
        else {header("Location: $url");}
    }
    // on effectue une redirection vers la page d'accueil
    redirection ('index.php');
}
else {
    echo 'La variable du formulaire n\'est pas déclarée.';
}
?>
```

Les cookies

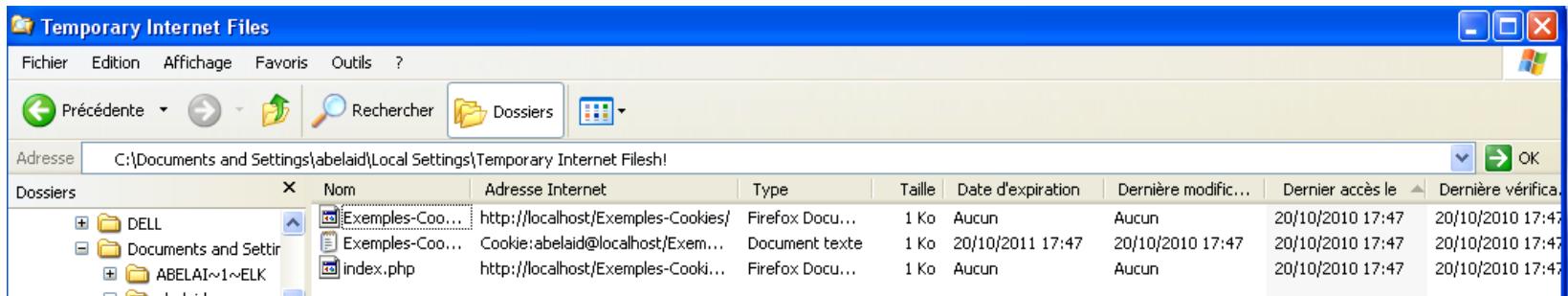
■ Attention !!!

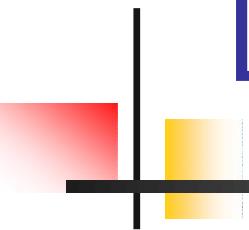
- Plusieurs conditions sont à respecter afin que l'utilisation des cookies se passe au mieux :
 1. l'envoie d'un cookie doit être la **première fonction PHP** que vous utilisez dans votre script,
 1. ce qui veut dire que vous devez utiliser la fonction **setcookie()** tout en haut de votre script (**AUCUN AFFICHAGE ET AUCUN CODE HTML AVANT UN SETCOOKIE**)
 2. Si d'autres fonctions interviennent avant l'envoi du cookie, celui-ci ne fonctionnera pas
 3. Si vous envoyez un cookie sur un poste client, celui-ci effacera automatiquement l'ancien cookie qui portait le même nom (si il y en avait un), autrement il le créera
 4. Note : pour effacer un cookie, vous devez lancer un cookie qui aura le même nom que le cookie que vous voulez effacer, tout en lui donnant une valeur nulle (vous pouvez également l'envoyer avec un temps de vie dépassé)

Les cookies

■ Expérience à faire

- Aller à l'adresse équivalente à celle-ci sur votre Navigateur Internet Explorer . C:\Documents and Settings\labelaid\Local Settings\Temporary Internet
 - Vider tout le répertoire
 - Lancer le programme index.php sur Internet Explorer . en utilisant : <http://localhost/Exemples-Cookies/index.php>
 - Regarder ce qui a été créé dans : C:\Documents and Settings\labelaid\Local Settings\Temporary Internet Filesh!





Les sessions

Définition et création

Les sessions

■ Intérêt

- Aller plus loin que les cookies
 - Permettre à plusieurs pages de partager les mêmes données
 - Conserver plusieurs données lors d'une session
 - Ranger les données de la session sur le serveur
 - Sécuriser le passage des données vers le serveur en
 - ❖ Donnant un identifiant à la session
 - ❖ Cryptant l'identifiant
 - Disposer de plusieurs sessions et donner un nom à chacune

Les sessions

■ Qu'est-ce qu'une session ?

- Une session est une période entre un début et une fin d'une activité
- Par exemple pour Internet,
 - j'ouvre mon navigateur sur <http://www.php-astux.info> et je referme le navigateur :
 - ❖ j'ai ainsi réalisé une session que j'ai commencée, puis terminée
 - Par extension, on associe à une session un ensemble de valeurs définies de manière transparente pour le visiteur et que le serveur conserve de page en page, comme un cookie

Les sessions

- **Les données d'une session**
 - Les données d'une session sont stockées sur le serveur et pas chez le client, ce qui l'empêche de pouvoir les modifier manuellement, comme il peut le faire pour un cookie
- **Exemples d'utilisation**
 - Sauvegarder les données personnelles d'un visiteur
 - Répartir les données d'un formulaire sur deux pages si le formulaire est trop long
 - La solution est d'afficher chaque jeu de champs sur une page et d'utiliser les données conservées par le serveur pour faire le lien entre les pages, quand le visiteur passe à la page suivante

Les sessions

- La durée de vie d'une session
 - Le temps d'une navigation
 - Elle commence donc lors de l'accès à une page les utilisant et se termine à la fermeture du navigateur du client
 - Une fois la session terminée, elle est détruite ainsi que toutes les données qu'elle contient
 - Elle ne doit donc contenir que des données temporaires
 - Cependant, la session possède aussi un délai d'expiration
 - Celui-ci peut être modifié dans la configuration du serveur (`directive session.gc_maxlifetime`), mais vaut généralement une trentaine de minutes
 - Une fois ce délai dépassé, la session est supprimée

Utiliser les sessions

■ Initialiser une session

- Pour pouvoir utiliser la fonctionnalité de session de PHP, il faut lancer le moteur de session en utilisant la fonction `session_start()`
- Vous devez placer ce code au tout début de votre script

```
<?php  
    session_start();  
?>
```

Utiliser les sessions

■ Une session portant un nom personnalisé

- Une session porte par défaut le nom "PHPSESSID"
- C'est lui qui sera utilisé comme paramètre GET dans les liens
- Pour le changer, utiliser la fonction session_name()

```
<?php  
    session_name('nom_de_session');  
    session_start();  
?>
```

Utiliser les sessions

- Lire et écrire des données dans la session
 - Les données de la session sont très facilement accessibles à travers d'un simple tableau PHP
 - On utilise le tableau super-global `$_SESSION`
 - Tout ce qui est stocké à l'intérieur est sauvegardé et accessible depuis toutes les pages PHP utilisant les sessions

Utiliser les sessions

Exemple

- Fermer une session
 - Une fois le script PHP terminé, les données de la session sont automatiquement sauvegardée pour le prochain script
 - Cependant, durant toute l'exécution du script, le fichier de la session est verrouillé et aucun autre script ne peut y toucher
 - Ils doivent donc attendre que le premier arrive à la fin
 - Vous pouvez fermer plus rapidement une session en utilisant la fonction `session_write_close()`
 - Si vous voulez également détruire la session, vous pouvez utiliser la fonction `session_destroy()` couplée à la fonction `session_unset()`

Les sessions

■ Sauvegarder une variable

- Les variables de sessions sont stockées dans le tableau super-global : ***\$_SESSION***

```
<?php  
    session_start(); // Création de la session  
    $_SESSION['prenom'] = 'Jean-Pierre'; // Sauvegarde  
    dans la session créée de la variable "prenom"  
?>
```

Les sessions

- Récupération de données dans une session
 - Quand on démarre une session avec `session_start()`, le tableau super-global `$_SESSION` est automatiquement initialisé avec les variables de la session
 - S'il contenait quelque chose, ce contenu n'est plus disponible après

```
<?php  
    echo $_SESSION['prenom'];  
?>
```

Les sessions

■ Savoir si une variable appartient à une session

- Utiliser sur le tableau `$_SESSION` la fonction `isset()` qui renvoie vrai si la variable passée en argument existe réellement
- Exemple

```
<?php  
if (isset($_SESSION['prenom'])) {  
    echo 'La variable prenom est déjà enregistrée !';  
    // On est certain de pouvoir y accéder ici  
} else {  
    echo 'La variable prenom n\'est pas enregistrée !';  
}  
?>
```

Les sessions

■ Exemple : réaliser un formulaire de connexion

- Le formulaire permet, par une page, de s'identifier
- L'accès à toutes les autres pages sera tributaire de cette identification
- Voici les étapes à suivre :
 - Initialisez votre session
 - Initialisez vos variables
 - Affichez votre formulaire, ou effectuez son traitement :
 - ❖ dans un cas, mes variables de session seront vides,
 - ❖ dans l'autre, elles ne seront remplies que si le traitement retourne un résultat correct
 - Enfin, si le formulaire de connexion est validé, affichez un lien pour passer à la page suivante en "mode connecté"
 - Autrement, pas d'accès aux autres pages

Exemple

- index_connexion.php

```
<?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']))
```

Exemple

- suite

```
if (!isset($_POST['submit']))  
{  
    // Si bouton submit non pressé, alors j'affiche le formulaire  
    echo '<form id="conn" method="post" action="">'. "\n"; echo '<p><label  
    for="login">Login :</label><input type="text" id="login" name="login"  
    /></p>'. "\n";  
    echo '      <p><label for="pass">Mot de Passe  
    :</label><input type="password" id="pass" name="pass"  
    /></p>'. "\n";  
    echo '      <p><input type="submit" id="submit" name="submit"  
    value="Connexion" /></p>'. "\n"; echo '</form>'. "\n";  
}; // fin if (!isset($_POST['submit']))  
?>
```

Les sessions

■ Supprimer une variable d'une session

- Utiliser *unset()* qui prend en paramètre la variable à détruire
- Exemple

```
<?php  
    unset($_SESSION['prenom']); // La variable de  
    //session "prenom" a été supprimée, on ne peut plus  
    //y avoir accès !  
?>
```

Les sessions

■ Détruire une session

- Utiliser *session_destroy()* qui ne prend aucun paramètre et qui renvoie vrai en cas de succès et faux en cas d'erreur
- Fonctionnement :

```
<?php  
if (session_destroy()) {  
    echo 'Session détruite !';  
} else {  
    echo 'Erreur : impossible de détruire la session !';  
}  
?>
```

Les sessions

■ Détruire toutes les variables d'une session

- Il est aussi possible de détruire toutes les variables de session, ce qui permet de conserver votre session :
 - il suffit tout simplement de réinitialiser le tableau `$_SESSION`
- *Il ne faut jamais utiliser `unset()` directement sur `$_SESSION`, cela rendrait impossible l'accès aux variables de la session courante jusqu'à sa fermeture*

```
<?php  
$_SESSION = array(); // $_SESSION est désormais  
//un tableau vide, toutes les variables de session ont  
//été supprimées  
?
```

Les sessions

- Utiliser plusieurs sessions dans la même page
 - Il est impossible d'ouvrir simultanément plusieurs sessions
 - Cependant, on peut tout à fait ouvrir plusieurs sessions l'une après l'autre
 - Dans ce cas, il faut
 - fermer la première session sans la détruire, grâce à *session_write_close()*
 - puis assigner les nouveaux *session_name*
 - et enfin ouvrir la nouvelle session avec *session_start()*

Les sessions

```
<?php
    session_name('utilisateur');
    session_start(); // Création de la première session
    [...]
    // Utilisation de la première session
    session_write_close(); // Fermeture de la première
    session, ses données sont sauvegardées.
    session_name('admin'); // Indication du nom de la
    seconde session
    session_start(); // Ouverture de la seconde session
    [...] // Utilisation de la seconde session.
?
?>
```

- *Une fois la session fermée, il est toujours possible d'accéder en lecture (les modifications ne seront pas prises en compte) aux variables de l'ancienne session*
- *`$_SESSION` ne sera vidé et rerempli qu'au prochain appel à `session_start()`*

Les sessions

Configurer php.ini

Les sessions

- Transmission de l'identifiant de session
 - Comme vu au début, un identifiant unique identifie chaque session
 - PHP se charge lui-même de transmettre cet identifiant d'une page à l'autre, mais on peut indiquer explicitement comment le faire à l'aide de 4 directives de configurations :
 - `session.use_cookies` :
 - ❖ Cette option fait stocker l'identifiant dans un cookie si elle est égale à 1
 - ❖ Sa valeur par défaut est 1
 - ❖ Si l'option est à 0, et qu'un coockie de session est présent dans le navigateur, il sera simplement ignoré
 - `session.use_only_cookies` :
 - ❖ Si cette option est égale à 1, alors PHP ignorera les identifiants transmis via l'url pour n'utiliser que ceux contenus dans les cookies
 - ❖ Sa valeur par défaut est 0

Les sessions

- **session.use_trans_sid :**
 - Si cette option est égale à 1, alors PHP insérera automatiquement l'identifiant dans tous les liens non absolus (ne commençant pas par http:// ou ftp:// ou mailto: ou d'autres liens absolus)
 - les liens commençant par un / (relatifs à la racine du site) seront eux pourvus de l'identifiant)
 - Sa valeur par défaut est 0
- **session.url_rewriter_tags :**
 - Indique où et dans quelles balises HTML insérer l'identifiant de session dans le cas où **session.use_trans_sid** est à 1
 - Sa valeur par défaut est :
a=href,area=href,frame=src,input=src,form=fakeentry,fields et=
 - Le format est balise1=attribut1,balise2=attribut2,...
 - Le cas des balises form et fieldset est particulier, car cela se traduit en fait par l'apparition d'un champ input (de type hidden) supplémentaire

Les sessions

- Voici deux exemples de configuration pour ces options :
 - Le premier est celui qui présente le plus de chances de faire transiter l'identifiant :
 - session.use_cookies à 1
 - session.use_only_cookies à 0
 - session.use_trans_sid à 1
 - session.url_rewriter_tags à
a=href,area=href,frame=src,iframe=src,input=src,form=
fakeentry,fieldset=
 - Si vous souhaitez produire une page respectant les standards XHTML, nous vous conseillons plutôt la valeur :
 - ❖ a=href,area=href,frame=src,iframe=src,input=src,fi
eldset=
 - et d'utiliser une balise *fieldset* dans vos balises *form*

Les sessions

- Le second est celui qui est totalement invisible à l'oeil d'un visiteur classique mais, si les cookies sont désactivés, il ne fonctionnera pas
 - `session.use_cookies` à 1
 - `session.use_only_cookies` à 1
 - `session.use_trans_sid` à 0
 - `session.url_rewriter_tags` à vide

Les sessions

■ La sécurité des sessions

- Il existe plusieurs directives de configuration permettant de jouer sur la sécurité des sessions, en voici quelques unes :
 - Vérifier la provenance du visiteur
- *session.referer_check* permet de spécifier une chaîne de caractères qui doit être présente dans le "referer" (adresse de la page de provenance) si celui-ci est indiqué par le navigateur
- Une bonne idée est d'indiquer le nom de domaine de votre site (par exemple)
- Par contre, gardez à l'idée que cela ne permet pas d'être sûr que le visiteur provient d'une des pages de votre site
- En effet, le referrer provient du protocole HTTP, lequel est très facile à manipuler !

Les sessions

- Augmenter la complexité de l'identifiant de session
 - *session.hash_function*
 - ❖ permet de choisir entre deux algorithmes de création des identifiants de sessions : MD5 (mettre la valeur 0) ou SHA-1 (mettre la valeur 1), le second permet de générer des identifiants de sessions plus longs
 - *session.hash_bits_per_caracter*
 - ❖ permet de définir les plages de caractères utilisées pour l'identifiant de session 4 pour les caractères '0' à '9' et 'a' à 'f', 5 pour '0'-'9' et 'a'-'v' et 6 pour '0'-'9', 'a'-z', 'A'-Z', '-' et ','

Les sessions

- ❖ Il s'agit uniquement d'une représentation de l'identifiant, en aucun cas ça ne modifie sa complexité intrinsèque
- ❖ *Ces directives de configuration ne sont disponibles que depuis PHP 5*

Les sessions

- Modifier la façon dont sont formatées les données de sessions
 - *session.serialize_handler*
 - ❖ permet de modifier la façon dont sont formatées les données de sessions avant d'être sauvegardées
 - Par défaut, (valeur *php*) c'est une sérialisation proche de la fonction *serialize* qui est utilisée
 - Si votre installation de PHP le supporte, vous pouvez utiliser WDDX (valeur *WDDX*), qui permet de récupérer un format XML

Les sessions

■ La durée de vie des sessions

- Pour modifier la durée de vie des sessions, en plus de jouer sur la durée de vie du cookie et du cache, on peut également jouer sur le temps au-delà duquel PHP considérera les données stockées comme obsolètes
- **Pour cela, il existe trois directives de configuration :**
 - *session.gc_maxlifetime* : Il s'agit effectivement de la durée au-delà de laquelle des données de session seront considérées comme périmées.
 - *session.gc_probability* : La vérification du temps de vie des données n'est pas systématique, c'est lancé aléatoirement selon une fréquence prédefinie. Il s'agit de la probabilité de déclenchement de l'opération.

Les sessions

- *session.gc_divisor* : Pour obtenir la fréquence de lancement de la procédure de vérification des données, il faut diviser la probabilité par ce paramètre
- Par exemple, si `gc_probability` vaut 1 et que `gc_divisor` vaut 100, alors ce sera lancé 1 fois toutes les 100 ouvertures de sessions

Les sessions

- Modifier localement la configuration des sessions
 - Comme pour toutes les directives de configuration de PHP, il est possible de modifier la configuration des sessions sur certains serveurs
 - On utilise pour cela le fichier *.htaccess*
 - Pour modifier la valeur d'une directive de configuration, il suffit d'indiquer *php_value* ou *php_flag* (dans le cas d'une directive de type on/off), suivi du nom de la directive de configuration et de sa nouvelle valeur (en séparant le tout par des espaces)
 - Cependant, gardez à l'esprit que cela dépend entièrement de la configuration du serveur, il est même possible que seule une partie des directives soit modifiable, voire même aucune...

Les sessions

- Modifier la façon dont sont stockées les données de sessions
 - Comme indiqué plus haut, PHP ne fournit pas de gestionnaire de données de sessions autre que des fichiers non cryptés
 - Il est cependant possible de définir vous-mêmes les opérations à faire pour ouvrir, fermer, lire, écrire et vérifier la durée de vie d'une session
 - Cette fonctionnalité vous permettra par exemple de crypter et de stocker très facilement vos sessions dans une base de données (par exemple) en écrivant les fonctions de requêtage adéquates
 - Une fois ceci fait, vous n'aurez alors plus besoin de définir les opérations d'accès à la base de données pour les sessions !
 - Vous trouverez un peu plus bas un exemple de fonctions vous permettant de stocker les sessions dans une base de données