

Technologies d'Internet

PHP

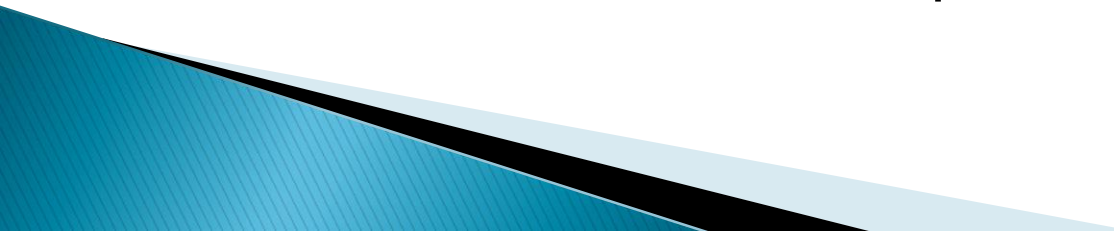
Frédéric LAGRANGE

frederic.lagrange@2la-software.com



ENSIM
École d'ingénieurs
Le Mans Université

Généralités 1 / 2

- ▶ **PHP** : Hypertext Preprocessor
 - ▶ **PHP** : Personal Home Page à l'origine
 - ▶ **PHP** a été développé en C en 1994 par Rasmus Lerdorf (bibliothèque C à l'origine)
 - ▶ **PHP** a permis de créer un grand nombre de sites Web célèbres, comme Facebook ou Wikipédia
 - ▶ Langage de programmation Web côté serveur le plus utilisé au monde
 - ▶ Part de marché en 2019 : plus de 80 %
- 

Généralités 2 / 2

- ▶ Principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP
- ▶ Langage de script sur le serveur, interprété par le serveur
- ▶ Le client, ne voit jamais le code, seulement le résultat de l'exécution sur le serveur
- ▶ Peut également fonctionner comme n'importe quel langage interprété de façon locale
- ▶ PHP est un langage orienté objet

Architecture



Clients

Affichage de la réponse
via le navigateur Web

Envoi de la **requête**
(demande d'une page Web)



PHP
SQL



Envoi de la réponse
(la page Web demandée)



HTML / CSS



Serveurs

Traitement de la requête

Environnement

- ▶ Les programmes s'écrivent dans un éditeur de texte classique (du bloc note au notepad++ ou Eclipse ...)
- ▶ **Il faut un logiciel serveur PHP** pour interpréter les pages PHP, même pour une utilisation locale :
 - EasyPHP
 - Wampserver
 - Lamp
 - Mamp
 - ...

Note : ce serveur PHP s'appuie sur un serveur Web comme Apache pour EasyPHP ou Lamp par exemple.

Généralités sur la syntaxe PHP

- ▶ Par défaut prenez la syntaxe du C, ça a 95% de chances de fonctionner
- ▶ Code php → fichier `.php`
- ▶ Attention : dès que l'on met du code php dans un fichier html, ne pas oublier de changer l'extension du fichier en « `.php` »
- ▶ On va mélanger du code html et du code php, en isolant le code PHP dans des instructions :
« `<?php` » pour commencer et « `?>` » pour finir
- ▶ Dernière version `PHP 8.0.3` (mars 2021) mais la `7.4` (début 2020) est la plus utilisée chez les hébergeurs

Exemple

1. `<html>`
2. `<head>`
3. `<title>Test</title>`
4. `</head>`
- 5.
6. `<body>`
7. `<p>un bout de code en HTML</p>`
8. `<?php`
9. `echo 'Mon premier script en PHP';`
10. `?>`
11. `</body>`
12. `</html>`

Les Commentaires

- ▶ Comme en C :

- Sur une seule ligne : //
- Sur plusieurs lignes : /* */

Les Variables

Les Variables

- ▶ Les noms de variables :
 - Toute chaîne de caractères commençant par un \$
 - Pas de chiffre en second caractère
Pas de \$12abc, mais \$a12bc oui
 - Sans espace dans le nom
- ▶ Affectation par « = »

Variables : exemple

1. `<?php`
2. `$nom = "MASTER";`
3. `// $nom contient alors la chaîne de caractères MASTER.`
- 4.
5. `$mon_chiffre = 12;`
6. `// $mon_chiffre contient la valeur numérique 12.`
- 7.
8. `$5toto = "test";`
9. `// Cette déclaration n'est pas valide car le nom de la variable commence par un chiffre`
10. `?>`

Afficher le contenu des variables

```
1. <?php
2. $nom = "MASTER";
3. echo 'Bonjour ';
4. echo $nom;
5. echo ' $nom';
6. echo " $nom "
7. echo '!';
8. ?>
```

Ce qui affiche :

► Bonjour MASTER \$nom
MASTER !

Attention à la syntaxe :

- « » pour une chaîne de caractère
- Rien pour un nom de variable
- Et ' ' pour une chaîne de caractère sans interprétation de variable

Opérateur de concaténation : « . »

```
1. <?php
2. $nom = "MASTER";
3. echo 'Bonjour '.$nom.' !';
4. echo "Bonjour $nom ! ";
5. ?>
```

Bonjour MASTER !
Bonjour MASTER !

Attention à ne pas écrire :

```
1. <?php
2. $nom = "MASTER";
3. echo `Bonjour $nom !`;
4. ?>
```

BONJOUR \$nom !

Variables d'environnement

<code>\$_SERVER['DOCUMENT_ROOT']</code>	Racine du serveur
<code>\$_SERVER['HTTP_ACCEPT_LANGUAGE']</code>	Langage accepté par le navigateur
<code>\$_SERVER['HTTP_HOST']</code>	Nom de domaine du serveur
<code>\$_SERVER['HTTP_USER_AGENT']</code>	Type de navigateur
<code>\$_SERVER['PATH_INFO']</code>	Chemin WEB du script
<code>\$_SERVER['PATH_TRANSLATED']</code>	Chemin complet du script
<code>\$_SERVER['REQUEST_URI']</code>	Chemin du script
<code>\$_SERVER['REMOTE_ADDR']</code>	Adresse IP du client
<code>\$_SERVER['REMOTE_PORT']</code>	Port de la requête HTTP
<code>\$_SERVER['QUERY_STRING']</code>	Liste des paramètres passés au script
<code>\$_SERVER['SERVER_ADDR']</code>	Adresse IP du serveur
<code>\$_SERVER['SERVER_ADMIN']</code>	Adresse de l'administrateur du serveur
<code>\$_SERVER['SERVER_NAME']</code>	Nom local du serveur
<code>\$_SERVER['SERVER_SIGNATURE']</code>	Type de serveur
<code>\$_SERVER['REQUEST_METHOD']</code>	Méthode d'appel du script

Utilisation

- ▶ A n'importe quel endroit d'un script :

1. `<?php`
2. `echo 'Votre adresse IP est : ' . $_SERVER['REMOTE_ADDR'];`
3. `?>`

- ▶ ➔ Votre adresse IP est : 80.12.45.26

Les Tableaux

Les tableaux – type array

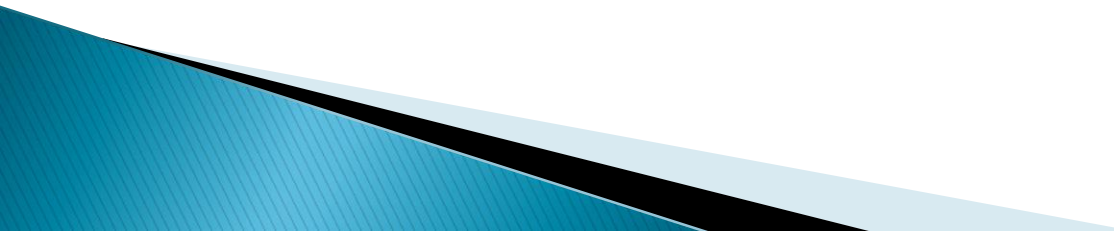
```
1. <?php
2. $fruit = Array();
3. $fruit[0] = "fraise";
4. $fruit[1] = "banane";
5. $fruit[2] = "abricot";
6. ?>
```

```
1. <?php
2. $fruit = Array();
3. $fruit[] = "fraise";
4. $fruit[] = "banane";
5. $fruit[] = "abricot";
6. ?>
```

Index à 0, affectation par index ou par ordre de case libre

Les tableaux associatifs

Au lieu d'avoir des indices numériques, on peut avoir des chaînes de caractères

1. `<?php`
 2. `$fruit = Array();`
 3. `$fruit['le_meilleur'] = "fraise";`
 4. `$fruit['le_prefere_de_Julien'] = "banane";`
 5. `$fruit['mon_prefere'] = "abricot";`
 6. `?>`
- 

Structutres de Contrôle

Structures de contrôle

<u>Instruction</u>	<u>Signification</u>
if	Si
else	Sinon
elseif	Sinon si
switch	Selon
for	Pour chaque (boucle)
while	Tant que (boucle)
==	Strictelement égal
!=	Différent
<	Strictelement inférieur
>	Strictelement supérieur
<=	Inférieur ou égal
>=	Supérieur ou égal
and ou &&	ET logique
or ou	OU logique

Exemple if elseif else

```
1. <?php
2. $nombre = 11;
3. if ($nombre >= 0 && $nombre < 10) {
4.     // on teste si la valeur de notre variable est comprise entre 0 et 9
5.     echo $nombre.' est compris entre 0 et 9';
6. }
7. elseif ($nombre >= 10 && $nombre < 20) {
8.     // on teste si la valeur de notre variable est comprise entre 10 et
    19
9.     echo $nombre.' est compris entre 10 et 19';
10. }
11. else { // si les deux tests précédents n'ont pas aboutis, alors on tombe
    dans ce cas
12.     echo $nombre.' est plus grand que 19';
13. }
14. ?>
```

Exemple switch

```
1. <?php
2. $nom = "MASTER";
3. switch ($nom) {
4.     case 'Jean' :
5.         echo 'Votre nom est Jean.';
6.         break;
7.     case 'Benoît' :
8.         echo 'Votre nom est Benoît.';
9.         break;
10.    case 'MASTER' :
11.        echo 'Votre nom est MASTER.';
12.        break;
13.    default :
14.        echo 'Je ne sais pas qui vous êtes !!!';
15. }
16. ?>
```

Exemple for

1. `<?php`
2. `$chiffre = 5;`
3. `// Début de la boucle`
4. `for ($i=0; $i < $chiffre; $i++) {`
5. `echo 'Notre chiffre est différent de '.$i.'<br`
 `/>';`
6. `}`
7. `// Fin de la boucle`
- 8.
9. `echo 'Notre chiffre est égal à '.$i;`
10. `?>`

Exemple while

```
1.  <?php
2.  $chiffre = 5;
3.  $i = 0;
4.
5.  // Début de la boucle
6.  while ($i < $chiffre) {
7.      echo 'Notre chiffre est différent de '.$i.'<br />';
8.      $i = $i + 1;
9.  }
10. // Fin de la boucle
11.
12. echo 'Notre chiffre est égal à '.$i;
13. ?>
```


Les Formulaires

Création du formulaire en html

1. `<html>`
2. `<head>`
3. `<title>Ma page de test</title>`
4. `</head>`
5. `<body>`
6. `<form action = "traitement.php" method="post">`
7. `Votre nom : <input type = "text" name = "nom">
`
8. `Votre fonction : <input type = "text" name = "fonction">
`
9. `<input type = "submit" value = "Envoyer">`
10. `</form>`
11. `</body>`
12. `</html>`

Méthodes d'envoi des données

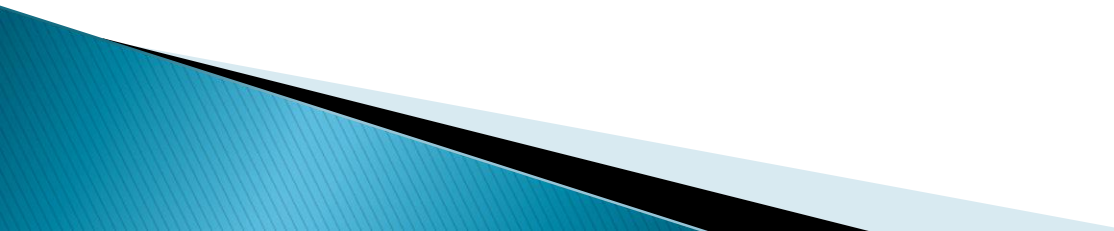
Méthodes de transfert de données :

- ▶ **GET** : affiche les données dans la barre d'adresse :

<http://www.xul.fr/page.html&couleur=bleu&forme=rectangle>

- ▶ **POST** : utilisation de formulaire pour l'envoi

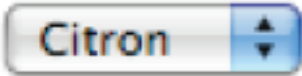
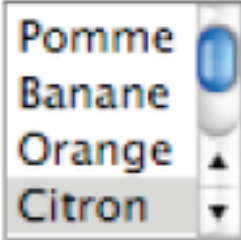
Les types d'éléments d'un formulaire

- ▶ **input**
 - Champs de saisie de texte et boutons
 - ▶ **select**
 - Liste de choix : menus déroulants ou ascenseurs
 - ▶ **textarea**
 - Zone de texte de plusieurs lignes
- 

Éléments de formulaires 1 / 3

Type	Code	Résultat
<i>sans</i>	<code><input name="ident" /></code>	<input type="text"/>
	<code><input name="ident" value="Par défaut" /></code>	<input type="text" value="Par défaut"/>
submit	<code><input type="submit" value="Envoyer" name="Envoyer" /></code>	<input type="submit" value="Envoyer"/>
checkbox	<code><input type="checkbox" name="pfm[]" value="linux" checked = "checked" /> Linux
</code> <code><input type="checkbox" name="pfm[]" value="dos" /> Dos
</code> <code><input type="checkbox" name="pfm[]" value="win" /> Windows</code>	<input checked="" type="checkbox"/> Linux <input type="checkbox"/> Dos <input type="checkbox"/> Windows
radio	<code><input type="radio" name="media" value="cd" checked= "checked" /> CD-ROM
</code> <code><input type="radio" name="media" value="dk" /> Disquette</code>	<input checked="" type="radio"/> CD-ROM <input type="radio"/> Disquette
password	<code><input type="password" name="pass" size="4" /></code>	<input type="password" value="...."/>
reset	<code><input type="reset" value="Effacer" /></code>	<input type="reset" value="Effacer"/>

Éléments de formulaires 2/3

Code	Résultat
<pre><select name="menu"> <option value="pomme">Pomme</option> <option value="banane">Banane</option> <option value="orange">Orange</option> <option value="citron" selected= "selected"> Citron</option> <option value="peche"> Pêche</option> <option value="poire"> Poire</option> </select ></pre>	 A browser-rendered dropdown menu. The text 'Citron' is visible in the main box, and a blue arrow on the right points downwards, indicating the menu is open or about to open.
<pre><select name="menu" size= "4"> ... </select></pre>	 A browser-rendered list box with a height of 4 items. It contains the text 'Pomme', 'Banane', 'Orange', and 'Citron' stacked vertically. A blue vertical scrollbar is on the right side.

Eléments de formulaires 3 / 3

Code	Résultat
<pre><textarea name="comm" rows="10" cols="40"> Tapez vos commentaires ici </textarea></pre>	<div>Tapez vos commentaires ici</div>

Envoi des données du Client vers le Serveur

A partir d'un formulaire on peut vouloir :

- Envoyer un mail

```
<form action="mailto:mon.nom@mon.adresse.fr" method="POST">
```

- Envoyer les données vers le serveur pour qu'elles soient traitées : insertion en base de données, calculs, etc.

```
<form action="traite.php" method="post">
```

```
...
```

```
<input type="submit" value="ok"
```

```
/>
```

```
</form>
```


Variables Globales

- ▶ `$_SERVER []` : tableau contenant des informations comme les en-têtes, dossiers et chemins du script
- ▶ `$_ENV []` : variables d'environnement
- ▶ `$_SESSION []` : accessible durant le temps de présence du visiteur
- ▶ `$_COOKIE []` : enregistré sur l'ordinateur
- ▶ `$_FILES []` : liste des fichiers envoyé par le formulaire
- ▶ `$_GET []` : champs du formulaire en méthode GET
- ▶ `$_POST []` : champs du formulaire en méthode POST

Coté serveur (en PHP)

- ▶ Les variables de formulaires sont stockés dans une variable nommée `$_POST`.
- ▶ C'est un tableau associatif, dont les index sont les noms des champs du formulaire html

Exemple : Coté serveur (en PHP)

```
<?php
if(isset($_POST["valider"])) // tester si le formulaire a été soumis
{
    //tester si les cases du formulaire ne sont pas vides
    if(!empty($_POST["nom"])&&!empty($_POST["prenom"])&&!empty($_POST["mail"]))
    {
        // récupération des valeurs des champs
        $nom = $_POST["nom"];
        $prenom = $_POST["prenom"];
        $avis = $_POST["avis"];
        // utilisation des valeurs
        echo "Bonjour ".$prenom." ".$nom. " !<br/>" ;
        echo "Merci d' avoir visité notre site ! <br/>" ;
        if(strcmp($avis,"oui") == 0)
        echo "Nous serons contents de vous accueillir la prochaine fois <br />";
        else
        echo "Nous regrettons de ne plus jamais vous revoir <br />";
    }
    else
        echo "Toutes les cases du formulaire ne sont pas remplies";
}
?>
```

PHP Avancé

Les tableaux
Portées des variables
Fonctions
Variables dynamiques

Quelques fonctions utiles

date ()

```
1. <?php
2. $date_du_jour = date ("d-m-
   Y");
3. $heure_courante = date
   ("H:i");
4. echo 'Nous sommes le : ';
5. echo $date_du_jour;
6. echo ' Et il est : ';
7. echo $heure_courante;
8. ?>
```

► Nous sommes le 01-04-
2017 Et il est 19:15

- Quelques paramètres possibles :
 - **d** : Jour du mois, sur deux chiffres : "01" à "31"
 - **D** : Jour de la semaine, en trois lettres (et en anglais) : par exemple "Fri" (pour Vendredi)
 - **F** : Mois, textuel, version longue; en anglais, i.e. "January" (pour Janvier)
 - **h** : Heure, au format 12h, "01" à "12"
 - **H** : heure, au format 24h, "00" à "23"
 - **g** : Heure, au format 12h sans les zéros initiaux, "1" à "12"
 - **L** : Booléen pour savoir si l'année est bissextile ("1") ou pas ("0")
- ... idem pour mois, jours, année, heure, min, secondes ...

Fonctions d'écritures de fichiers

open et fclose

```
1. <?php
2. // Instruction 1
3. $fp = open ("donnees.txt", "r");
4. // Instruction 2
5. $contenu_du_fichier = fgets ($fp, 255);
6. // Instruction 3
7. fclose ($fp);
8. // Instruction 4
9. echo 'Notre fichier contient : '.$contenu_du_fichier;
10. ?>
```

Les tableaux indexés

- ▶ Comme en C :

- index de 0 à $n-1$,
 - indexé par des '['

- ▶ Déclaration (pas obligatoire)

- `$mon_tab = array();`

- ▶ Initialisation avec ou sans index

- `$mon_tab[0] = « zero »;`

- `$mon_tab[1] = « un »;`

- `$mon_tab[] = « deux »;`

- `$mon_tab[] = « trois »;`

- ▶ Déclaration + initialisation

- `$mon_tab = array(« zero », « un », « deux », « trois »);`

Tableaux associatifs 1 / 2

- ▶ Les indices deviennent des noms (des clés) – sera vu pour récupérer les données d'un formulaire :

`$_POST['nom']` par exemple

- ▶ Initialisation

```
$JourSemaine = array("Monday" => "Lundi", "Tuesday" => "Mardi", "Wednesday" => "Mercredi", "Thursday" => "Jeudi", "Friday" => "Vendredi", "Saturday" => "Samedi", "Sunday" => "Dimanche") ;
```

- ▶ Initialisation différée :

```
$JourSemaine["Lunes"] = "lundi" ;
```


Tableaux associatifs 2 / 2

► Parcours

```
$JourSemaine = array("Monday" => "Lundi",  
"Tuesday" => "Mardi", "Wednesday" => "Mercredi",  
"Thursday" => "Jeudi", "Friday" => "Vendredi",  
"Saturday" => "Samedi", "Sunday" => "Dimanche") ;
```

```
foreach ($JourSemaine as $clef => $valeur) {
```

```
...  
}
```

Tableaux multidimensionnels

- ▶ Définition

`$tableau = array(array()) ;`

- ▶ Accès

`$ tableau [0][0] = "Mozart" ;`

`$ tableau [0][1] = "Chopin" ;`

`$ tableau [1][0] = "Louis Armstrong" ;`

`$ tableau [1][1] = "Baden Powell" ;`

- ▶ Utilisations : pareil qu'en C

- Initialisation

- Parcours

Quelques fonctions bien utiles 1 / 2

// substr : renvoie le segment à partir de start (et d'une longueur length)

string substr (string \$string , int \$start [, int \$length])

// strlen : calcule la taille de la chaîne

int strlen (string \$string)

// strtolower : convertit une chaîne en minuscules

string strtolower (string \$string)

// strtoupper : convertit une chaîne en majuscules

string strtoupper (string \$string)

// strcmp : comparaison de 2 chaînes de caractères

strcmp(\$chaîne1, \$chaîne2)

Quelques fonctions bien utiles 2/2

// isset : retourne si une variable est définie ou pas
`isset($var)`

// empty : depuis HTML 5 gère les variables et les objets. Retourne FALSE si une variable existe et est non nulle par exemple.
`empty($var)`

// md5 : calcule le MD5 d'une chaîne de caractères en utilisant l'algorithme *RSA Data Security, Inc. MD5 Message-Digest Algorithm* et retourne le résultat
`md5($chaine)`

// print_r : affiche le contenu d'un tableau indexé (utile au débogage)
`print_r($array)`

// var_dump : aussi pour tableau indexé, mais plus détaillé sur les types de variables
`var_dump($array)`

// count :
`count($array), sizeof($array)`

Portée des variables – définition de fonctions

```
<?php
```

```
$variable = 'Salut';
```

```
// Déclaration de la fonction
```

```
function Ma_Fonction() {
```

```
$variable = 'Bonjour';
```

```
}
```

```
Ma_Fonction();
```

```
// Affichera "Salut" et non
```

```
"Bonjour"
```

```
echo $variable;
```

```
?>
```

Portée des variables – définition de fonctions

```
<?php
$variable = 'Salut';

// Déclaration de la fonction
function Ma_Fonction() {
    $variable = 'Bonjour';
}

Ma_Fonction();

// Affichera "Salut" et non
    "Bonjour"
echo $variable;
?>
```

```
<?php
$variable = 'Salut';

// Déclaration de la fonction
function Ma_Fonction() {
    $GLOBALS[' $variable' ] = 'Bonjour';
}

Ma_Fonction();

// Affichera "Bonjour"
echo $variable;
?>
```

Portée des variables – définition de fonctions

```
<?php  
$variable = 'Salut';
```

```
<?php  
$variable = 'Salut';
```

```
// Déclaration function Ma_Fonction($chaine) {  
function Ma_Fonction($chaine) {  
    $variable = 'Bonjour ' . $chaine;  
    return $variable;  
}
```

de la fonction

```
Ma_Fonction() {  
    $variable = 'Bonjour';  
}
```

Utilisation :

```
Ma_Fonction()  
echo Ma_Fonction();
```

```
// Affichera "Salut"  
"Bonjour"  
echo $variable;  
?>
```

```
);  
"Bonjour"  
echo $variable;  
?>
```

Variables dynamiques

PHP est un langage interprété ce qui permet certaines facéties qui n'existent pas dans les langages compilés :

```
$bienvenue_fr = 'Bienvenue!';  
$bienvenue_en = 'Welcome!';  
$lang = 'fr';  
// Affichera "Bienvenue!"  
echo ${'bienvenue_'.$lang};
```


Personnalisation et navigation

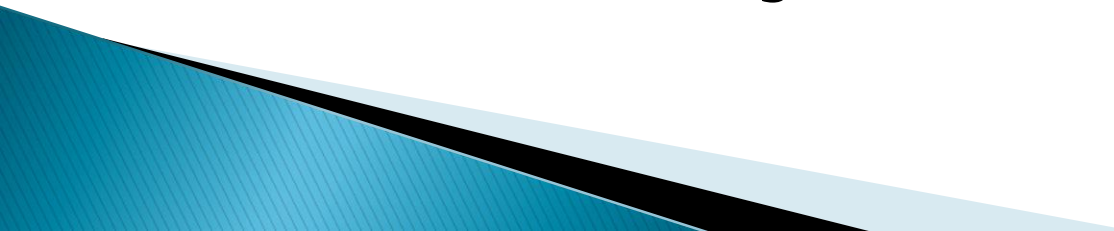
Cookies et
Sessions

Sauvegarde d'informations sur l'utilisateur

► Cookies

- But : mémoriser des informations et les retrouver à la prochaine connexion sur le même site
- Petit fichier texte, stocké sur le poste client

► Sessions

- Identifier l'utilisateur sur le serveur
 - PHP attribue un identifiant unique qui est soit sauvegardé dans un cookie soit propagé dans l'URL
 - Possibilité d'enregistrer des variables
- 

COOKIES

- ▶ Création : setcookie()
 - `bool setcookie(string name [, string value [, int expire [, string path [, string domain [, bool secure]]]])`
- ▶ Exemple :
 - `setcookie('utilisateur', 'bob', time()+24*3600, '/', 'www.site.com', 0);`
- ▶ Récupération : `$_COOKIE['utilisateur'];`
- ▶ Suppression par écrasement :
 - `setcookie('utilisateur');`

SESSIONS

- ▶ Démarrer une session :
 - `session_start();` // en première instruction, notamment aucun affichage ne doit être fait avant
- ▶ Enregistrer des variables :
 - `$_SESSION['nom'] = 'Xenon';`
- ▶ Accéder à des variables :
 - `echo $_SESSION['nom'];`
- ▶ Destruction de session :
 - `$_SESSION = array();` // *Détruit toutes les variables de session*
 - `session_destroy();` // *Détruit la session*

Fonctions require et include

Appeler des Librairies de fonctions

- ▶ `include` (« nomFichier.ext »)
 - ▶ `include_once` (« nomFichier.ext »)
 - ▶ `require`(« nomFichier.ext »)
 - ▶ `require_once`(« nomFichier.ext »);
-
- ▶ Ces fonctions sont faites pour inclure des librairies de fonctions
 - ▶ `once` spécifie qu'on inclura qu'une seule fois un fichier précis.
 - ▶ `require()` génère une **erreur fatale** s'il ne trouve pas le fichier ou qu'il y a une erreur → cessera le programme !

Exemple include()

```
<?php
```

```
$couleur = 'verte';  
$fruit = 'pomme';
```

```
?>
```

Vars.php

```
<?php
```

```
echo "Une $couleur $fruit";  
    // Une
```

```
include 'Vars.php';
```

```
echo "Une $couleur $fruit";  
    // Une verte pomme
```

```
?>
```

Test.php

Détournement de ces fonctions

- ▶ `Include()` : fait un « copier-coller » du contenu du fichier appelés à l'endroit appelant
- ▶ Ce peut être n'importe quel type de fichier !

Example

- ▶ [titre.inc.txt](#)

Cours Techno Internet

- ▶ [contenu.inc.txt](#)

Ceci est un cours de
programmation web

```
<html > <head> <title>  
<?php  
    include('titre.inc.txt');  
?>  
</title> </head>  
<body> <h1>  
<?php  
    include('contenu.inc.txt'); ?>  
</h1> </body> </html>
```

Fichiers à inclure

Programme de test

Example

- ▶ titre.inc.txt

Cours Techno Internet

- ▶ contenu.inc.txt

C'est un s
program

```
<html > <head> <title>  
<?php  
    include('titre.inc.txt');  
?>  
</title> </head>  
<body> <h1>
```

Au final, le navigateur interprètera :

```
<html> <head> <title>  
Cours Techno Internet  
</title> </head><body> <h1>  
Ceci est un cours de programmation web  
</h1> </body> </html>
```

```
.inc.txt'); ?>  
</html>
```

Fichiers à inclure

Programme de test

PHP – MySQLi

Affecter une valeur par défaut
dans un formulaire

Accès à une base de données :
3 étapes

Les formulaires

- ▶ Structures à voir en TD : on appelle champ un élément de formulaire (case à cocher, zone de saisie, liste, bouton, ...)
- ▶ Donner une valeur à un champ :
 - La plupart des items « à remplir » :
`value=« valeur tapée »`
 - Exemple :
`<input type="text" name="message">`
`<input type="text" value="bonjour" name="message">`

Généralités

- ▶ Depuis PHP 5.4, ne plus utiliser MySQL (**deprecated**) mais **MySQLi**
- ▶ Les méthodes MySQL fonctionnent toujours mais génèrent des messages d'erreurs
- ▶ Précéder « mysql » dans le nom des méthodes par « @ » supprime ces messages d'erreurs mais ce n'est pas propre

Accès à une bdd : 3 étapes clés

- ▶ Connexion au serveur et à la base de données
- ▶ Envoi d'une requête
- ▶ Traitement de la réponse

Remarque : en TD/TP nous utiliserons Lamp (PhpMyAdmin) pour créer et manipuler des tables dans une base de données dont vous serez l'administrateur

Connexion

//paramètres de connexion à la base de données

\$Server= "localhost";

\$User="vous";

\$Pwd=" votreMotDePasse";

\$DB= "nomDeVotreBase";

//connexion au serveur où se trouve la base de données

\$Connect = mysqli_connect (\$Server, \$User, \$Pwd, \$DB);

//affichage d'un message d'erreur si la connexion a été refusée

if (!\$Connect)

 echo "Connexion à la base impossible";

Envoi d'une requête

//Ecriture de la requête en SQL

`$Query = "...";`

Une requête est une question que l'on pose à la BDD :

Exemples :

`SELECT * FROM etudiants;`

`SELECT nom FROM etudiants;`

`SELECT count(*) FROM etudiants;`

//Envoi de la requête

`$Result = $Connect->query ($Query);`

Traitement de la réponse

- ▶ Le nombre et la nature d'information contenue dans la réponse, dépendent du type de requête :
SELECT, DELETE, UPDATE ...

- ▶ Ici traitement d'un **SELECT** :

```
while ($Data = mysqli_fetch_array ($Result) )
```

```
{
```

```
//Affichage des lignes de données, champ par champ
```

```
echo "Num de l'équipe : $Data[0], Nom de l'équipe : $Data[1]";
```

```
}
```

```
// on ferme la connexion !
```

```
mysqli_close ($Connect);
```

Alternative à MySQLi : API PDO

MySQLi

- Fonctionnalités avancées de MySQL
- API la plus performante
- API disponible en Procédurale et Orienté Object

PDO

- Utilisable sur tous types de base de données
 - API simple et clair
 - API la plus connue
- 