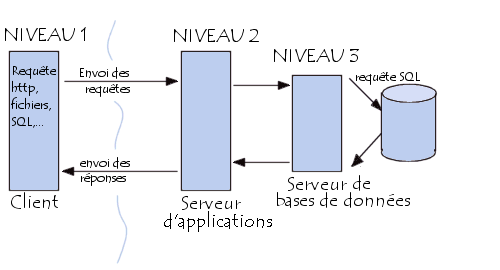
CHAPITRE 1 : LE LANGAGE PHP

PHP (PHP: Hypertext Preprocessor ou Personnal Home Page) est un langage de scripts  
spécialement conçu pour le développement d'applications web et pouvant être intégré  
facilement au HTML.

1. Présentation de l'architecture à 3 niveaux

Dans l'architecture à 3 niveaux (appelée architecture 3-tier), il existe un niveau  
intermédiaire, c'est-à-dire que l'on a généralement une architecture partagée entre :  
1. Un client, c'est-à-dire l'ordinateur demandeur de ressources, équipée d'une  
interface utilisateur (généralement un navigateur web) chargée de la présentation ;  
2. Le serveur d'application OU Serveur Web (appelé également middleware), chargé de  
fournir la ressource mais faisant appel à un autre serveur  
3. Le serveur de données, fournissant au serveur d'application les données dont il a besoin.



Pour exécuter le script PHP, il faut : installer un serveur Web dont **Apache** le  
plus connu avec les outils comme EasyPHP, LAMP, WAMP, XAMP .  
enregistrer le script de type php dans un sous-repértoire du repertoire **c:\repertoire d’installation\www\**On peut accéder à votre fichier par l'intermédiaire de votre navigateur sur l'adresse :  
s **http://localhost[:port]/ application/fichier.php ou** http:// adresse IP ou logique du  
serveur[:port]/appication/fichier.php

**2 LES BASES DU LANGAGE PHP**Il y a 2 jeux de balises qui servent à délimiter des blocs de code PHP:  
Balises PHP : **<?php ?>**

|  |  |
| --- | --- |
| Balises de scripts | <**script language="php"> </script>** |

2.1. **Les commentaires**Un commentaire vous permet de mettre des lignes dans votre script PHP qui ne seront pas traduites par le navigateur qui les ignorera.. Les commentaires sur votre script qui ne feront qu'une ligne peuvent être amenés comme tel:  
**//Voici un commentaire**ou  
**#Voici un commentaire**Si vos commentaires font plus d'une ligne, au lieu de mettre // ou # à chaque début de ligne, vous pouvez mettre directement cela:  
**/\*  
Voici un commentaire ne sera analysé par  
l'interpréteur  
\*/**2.2. **Les bases de l'écriture dans le code php**PHP fournit des fonctions permettant d'envoyer du texte au navigateur :  
- **print(expression )**- **echo expression.**- **printf (chaîne formattée);**

Tous les caractères spéciaux, doivent être précédés d'un antislash. Voici un récapitulatif des caractères spéciaux nécessitant l'ajout d'un antislash :  
**Caractère**\" guillemet  
\$ caractère $  
\ barre oblique inverse (antislash)  
\r retour chariot  
\n retour à la ligne  
\t tabulation  
Par exemple, pour marquer Bonjour tout le monde dans un script PHP, il faut mettre:

|  |
| --- |
| <html> <head> <title>Titre</title> </head> <body> <?php echo "Hello World !";  print ("Bonjour tout le monde"); //Vous pouvez mettre ici un commentaire si vous le souhaitez print ("Bonsoir tout le monde"); #Pourquoi pas un autre commentaire ?? **echo date("d/m/Y")."<br>"; echo date("H:i")."<br>";** ?> </body> **</html>** |

Remarque :1) si on affiche un code HTML de la forme echo "code html" , tous les quotes " devront être remplacés par des \ " , par exemple :  
echo "< table border=\"1\"> <tr> <td> width ==\"156\" width ==\"24\"> <p> Contenu du tableau </p> </td></tr> </table>"

echo " <img scr =\"htttp://www.mon\_site.com/mon\_image.gif\"  
width=\"32\" height=\"32\" border=\"0\">"

2) quand l'écho est de la forme echo 'code html' , tous les apostrophes ' devront être remplacés par des \ ' , par exemple :  
echo '< table border=\'1\'> <tr> <td> width ==\'156\' width ==\'24\'> <p> Contenu du tableau </p> </td></tr> </table>'

2.3. Constantes- définition de la valeur de la constante :  
define ("NOM\_CONST", valeur);  
- vérification d'existence :  
if(defined(NOM\_CONST))...

|  |
| --- |
| <?php define("MAIL\_ISINFO", "isinfo@isinf.univ.fr"); print("envoyez un mail à ".MAIL\_ISINFO); ?> |

Constantes prédéfinies

|  |  |
| --- | --- |
| PHP\_VERSION | Version de PHP |
| PHP\_OS | Système sur lequel PHP exécute le script) |

|  |
| --- |
| <?php echo PHP\_VERSION."<BR>"; echo PHP\_OS."<BR>"; ?> |

2.3. Les Variables

Les variables PHP peuvent contenir tous types de données : chaines de caractères, entier, flottant,... PHP n'est pas typé, c'est à dire qu'il n'y a pas de définition de la variable. La seule règle à respecter est que chaque variable doit être précédée de **$**. Une variable peut contenir un caractère comme dans l'exemple qui suit:

**Exemple 1 :**

|  |
| --- |
| <?php$a="ab"; print($a);$b=12.1; print($b);$c="<a href=\"/index.php3\">le lien</a>";print($c);// Somme de 2 nombres$a=3; $b=2; $c=$a+$b; print($c);// Concatener deux chaines$a="a"; $b="b"; $c=$a.$b; print($c);// opérations d'incrémentations et dedécrémentations$a=2; $a+=2; $a\*=2 ; print($a++);print("<br>");print($a);print("<br>");$a=2; print(--$a);// on défini la variable $texte et la variable $merci$texte ="Bonjour tout le monde";$merci ="merci de visiter ce site";// on affiche la valeur de $texte (à voir :l'apostrophe et le point)echo "<b>".$texte."</b><br>";// on affiche la valeur de $merciecho "Et ".$merci; **?>** |

**Exemple 2 :**

|  |
| --- |
| <?php$reel = 0.3; $entier = 22;$chaine = "bonjour !";$phrase1 = "bonjour $chaine!"; $phrase2 = 'bonjour$chaine$';print("un reel : $reel<BR>"); print("un entier :$entier<BR>");print("une chaine : $chaine<BR>"); print("une phrase: $phrase1<BR>");print("du bon usage des guillemets : $phrase2");?> |

**Exemple 3 :**

|  |
| --- |
| On peut mélanger le code PHP et le code HTML(le fait qu'il y est beaucoup de HTML ne signifiepas que l'extension **.php** doit être enlevé !)<html><head><title>Du PHP dans du HTML</title><body><font face="Verdana"><b>Bonjour tout le monde</b></font><br> <? // rappel : commencement du PHP // Nous définissons une variable $heure par la fonction date() // date() retourne l'heure // H -> l'heure // i -> Minute // \h -> h étant du texte, on rajoute \ devant (antislash) // de même pour \m\i\\n $heure = date("H \h i \m\i\\n"); ?> <font face="Verdana">Il est actuellement **<?echo $heure;?>**.</font> <!-- Ceci affichera "il est actuellement XX h XX min --> </body> </html> |

2.4. **Include un fichier dans un script PHP**Il est parfois utile d'inclure des fichiers PHP, HTML ou autres dans un fichier PHP, permettant par exemple de définir plusieurs variables dans un fichier de configuration et de l'include dans plusieurs pages de votre script.  
Pour include un fichier il existe deux fonctions : **include() et require()**Comme avec echo() et print() , la préférence de beaucoup des programmeurs est include() . Pour include un fichier faites :

include "mon\_fichier.php";  
// imaginons que mon\_fichier.php contient uniquement la ligne suivante  
// $mon\_mail="login@host.com";  
// Vous pouvez donc faire appel à la variable $mon\_mail  
echo "<a href=\"mailto:".$mon\_mail."\">".  
$mon\_mail."</a>";  
// Affichera login@host.com avec un lien vers ce  
mail

**2.5. Tableaux a une dimension**- initialisation trois possibilités équivalentes :  
1) $tab = array("b","o",10);  
2) // définition d'un tableau vide et  
remplissage des cases  
$tab = array();  
$tab[0] = "b"; $tab[1] = "o"; $tab[2] = 10;  
$tab[] = "b"; $tab[] = "o"; $tab[] = 10;

- accès à un élément d'un tableau de n cases :  
$tab[indice] avec indice dans l'intervalle [0,n-1]

- nombre d'éléments dans un tableau :  
***$nb = count($tab);  
<?php  
//création d'un tableau simple  
$tableau=array(12,15,2,1,8,24,35,5,2);  
//accéder à un élément  
echo $tableau[0]."<br>";  
echo $tableau[1]."<br>";  
echo $tableau[3]."<br>";  
?>***

|  |
| --- |
| <?php echo "essai avec array : "; $tab = array("b", "o", "n"); $taille = count($tab); for($i=0; $i<$taille; $i++) echo $tab[$i]; echo "<BR>essai avec [0], [1], ... : "; $tab2[0] = "h"; $tab2[1] = "e"; $tab2[2] = "l"; $tab2[3] = "l"; $tab2[4] = "o"; $**taille = count($tab2**); for($i=0; $i<$taille; $i++) echo $tab2[$i]; echo "<BR>essai de [] : "; $tab[] = "j"; $tab[] = "o"; $tab[] = "u"; $tab[] = "r"; $taille = count($tab); for($i=0; $i<$taille; $i++) echo $tab[$i]; ?> \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ essai avec array : bon essai avec [0], [1], ... : hello essai de [] : bonjour |

Pour définir les **tableaux associatifs**, nous disposons de deux syntaxes :  
$fiche1 = array();  
$fiche1['prenom'] = 'Fabien';  
$fiche1['nom'] = 'Torre';  
$fiche1['mail'] = 'torre@univ.fr';  
$fiche1['url']= 'http://www.univ.fr/~toto/';  
echo 'Mail de '.$fiche1['prenom'].' : '.$fiche1['mail'];  
$fiche2 = array(  
'prenom' => 'Fabien', 'nom' => 'Torre',  
'mail' => 'torre@univ.fr',  
'url' => 'http://www.univ.fr/~toto/',  
'age' => 19,  
'estEtudiant' => true  
);

echo 'Mail de '.$fiche2['prenom'].' : '.$fiche2['mail'];

**Parcours de tableaux PHP : la boucle *foreach***Enfin, la boucle *foreach* est dédiée au parcours de tableaux, simples ou associatifs :  
**foreach (**$tab **as** $val**) {***// des instructions ici portant sur $val***}**Par exemple,  
<?php  
$tab = array("P", "H", "P", 3);  
**foreach ($tab as $val) {**echo "<P> Le tableau contient la valeur $val\n ";  
}  
?>

2.6. Les structures de contrôleL'instruction ifCette instruction permet au script de s'exécuter seulement si une condition que vous avez défini a été réalisé. Voici un exemple d'utilisation:

<?  
$humeur="heureux";  
if ($humeur=="heureux")  
{print "Vous êtes heureux";}  
else  
{print "Vous êtes malheureux alors!!";  
} ?>

Vous pouvez utiliser les opérateurs logiques &&(ET) et || (OU) dans les conditions  
Les bouclesL'instruction forfor (compteur; condition; compteur){ liste d'instructions }Voici un exemple, pour afficher tous les chiffres de 1 à 10.  
for ( $i = 1; $i <= 10 ; $i++ )  
{ echo $i."<BR>"; }

L'instruction whilewhile ( condition )  
{  
liste d'instructions  
}  
Ce qui donne :

$i = 1;  
while ( $i <= 10 )  
{  
echo $i."<br>";  
$i++;  
}

**L'instruction switch :**Pour éviter l'imbrication de plusieurs if, switch est alors utilisé, cette instruction est utilisépour tester plusieurs valeur d'une seul variable.  
switch($ce\_nombre) {  
case 1:  
echo $ce\_nombre." est égal à 1";  
break;  
case 2:  
echo $ce\_nombre." est égal à 2";  
break;  
case 3:  
echo $ce\_nombre." est égal à 3";  
break;  
default:  
echo $ce\_nombre." est inférieur à 1 ou supérieur  
à 3";  
break;  
}

**2.7. FONCTIONS  
function  
nomfonction($arg1,&$arg2,$arg3="defaultvalu  
e",...,$argn){  
instructions** ;  
}

NB :  
- arg1 est passé par valeur ;  
- arg2 est passé par adresse : si sa valeur est modifiée dans la fonction, elle est modifiée pour tout le programme ;  
- arg3 est optionnel : il peut ne pas être passé lors de l'appel de la fonction puisqu'il a une valeur par défaut.  
Appel : nomfonction(param1,param2,param3..., paramn);  
Pour retourner une valeur : **return**

**Exemple 1**<?php  
// procedure  
function sp ($a,$b)  
{ $s = $a+$b;$p=$a\*$b;  
echo " somme=".$s. " produit=" .$p;  
}  
// fonction  
function somme ($a,$b)  
{ $s = $a+$b;

return $s;  
}  
// appel  
$a=2; $b=3;  
sp(2,3);  
$somme=somme($a,$b);  
echo "Somme=" .$somme;  
?>

**Exemple 2**<?php  
function test1($a)  
{  
if ($a)  
{ echo "vrai !<br>"; }  
else  
{ echo "faux !<br>"; }  
}  
test1(true);  
test1(false);  
?>

**Exemple 3 :**<?php  
function bonjour($a)  
{ echo "Bonjour ".$a." !<br>";  
echo "<a  
ref=\"mailto:mon@mail.com\">Contact</a>";  
}  
bonjour("Manu");  
bonjour("Johnny");  
?>

**Exemple4**function form\_login\_member()  
{  
echo '<table>';  
echo '<form action="liens.php" method="post"  
name="login\_form">';  
echo '<tr>';  
echo '<td>Votre login</td>';  
echo '<td><input type="text" name="login"  
value="" size="20" maxlength="42"></td>';  
echo '</tr>';  
echo '<tr>';  
echo '<td>Votre password</td>';  
echo '<td><input type="password" name="pass"  
size="20" maxlenght="12"></td>';  
echo '</tr>';  
echo '<tr>';  
echo "<td>;  
echo '<input type="submit" name="valider"  
value="val">';  
echo '<input type="reset" name="annuler"  
value="anu">';  
echo '</td>';

echo '</tr>';  
echo '</form>';  
echo '</table>';  
}  
?>

**Exemple 5**fonctions.inc.php  
function form\_login\_member()  
{ …….. }  
<html>  
<body>  
<?php  
include ("fonctions.inc.php");  
form\_login\_member();  
?>  
</body>  
</html>

**Exemple 6**Voici un exemple de programme utilisant une  
fonction avec un passage par valeur et ensuite par  
référence :  
<?php  
function modif\_tab($tab)  
{  
$tab[0] = "j'aime le perl mais je préfere le php.";  
}  
?>  
<html>  
<body>  
<?php  
$tab = array("salut", "j'aime le fortran et le cobol");  
// passage de $tab par valeur, la boucle for  
affichera  
// salut, j'aime le fortran et le cobol  
modif\_tab($tab);  
for ($i = 0; $tab[$i]; $i++)  
echo "$tab[$i] ";

// passage de $tab par référence, la boucle for  
affichera  
// salut, j'aime le perl mais je préfere le php.  
modif\_tab(&$tab);  
for ($i = 0; $tab[$i]; $i++)  
echo $tab[$i] ;  
?>

**Exemple 7** : **Fonctions récursives**Prenons un exemple simple pour expliquer ce  
principe à travers le calcul de la puissance d'un  
nombre entier.  
<html>  
<body>  
<?php  
function puissance ($nbr, $puissance)  
{  
for ($total = $nbr; $puissance > 1; $puissance--)  
$total = $total \* $nbr;  
return ($total);  
}  
<html>  
<body>  
<?php  
$nbr = 2;  
$puissance = 4;  
$result = puissance($nbr, $puissance);  
echo $result;  
?>  
</body>  
</html>  
function recurse\_puissance($nbr, $puissance, $tot  
al)  
{  
if ($puissance > 1)  
return (recurse\_puissance($nbr, $puissance - 1,  
$total \* nbr));  
return ($total);  
}  
// en une seule ligne :  
function recurse\_puissance($nbr, $puissance, $tot  
al)  
{  
return (($puissance > 1) ? recurse\_puissance($nbr, $puissances - 1, $total \* $nbr) : $total);  
}  
// APPELS  
$nbr = 2;  
$puissance = 4;  
$result = puissance($nbr, $puissance);  
echo $result;  
$result = recurse\_puissance(2, 4, 2);  
echo $result;  
?>  
<?php  
function recursion($a)  
{  
if ($a < 20) {  
echo "$a\n";  
recursion($a + 1);  
}  
}  
recursion(3);  
?>  
</body>  
</html>

**Portée des variables globales et locales**Une variable globale n'est pas visible dans une fonction, pour qu'elle le soit il faut le préciser avec global.  
<?php  
// fonction Sport1 $sport est une variable locale  
function Sport1(){  
$sport = "volley";  
print ("aimez-vous le $sport ?<BR>");  
}  
// fonction Sport2 $sport est une variable globale  
function Sport2(){  
global $sport;  
$sport = "badmington";  
print ("aimez-vous le $sport ?<BR>");  
}  
// corps du programme  
$sport = "ski";  
print ("aimez-vous le $sport ?<BR>");  
Sport1();  
print ("aimez-vous le $sport ?<BR>");  
Sport2();  
print ("aimez-vous le $sport ?<BR>");  
?>

aimez-vous le ski ?  
aimez-vous le volley ?  
aimez-vous le ski ?  
aimez-vous le badmington ?  
aimez-vous le badmington ?

a. L'instruction GLOBALAvant tout, vous devez savoir qu'une fonction qui se trouve à l'extérieur n'est pas directement accessible à l'intérieur d'un fonction.

Exemple Faux :// on défini une variable  
$texte="salut";  
function salut(){  
// on veut l'afficher  
echo $texte;  
}  
// on veut afficher "salut"  
salut();

Il faut donc utiliser la déclaration GLOBAL pour permettre de récupérer la valeur $texte dans l'exemple.  
Exemple Bon :// on défini une variable  
$texte="salut";  
function salut(){  
// on récupère $texte à l'aide de la déclaration GLOBAL  
GLOBAL $texte;  
// on veut l'afficher  
echo $texte;  
}  
// on veut afficher "salut"  
salut();

b. L'instruction STATICUne variable locale peut être mémorisée d'un appel à l'autre de la fonction en lui mettant static.  
L'instruction STATIC est utilisée pour garder une trace de la valeur d'une variable entre deux appels de fonction.

Exemple :function titre ($titre) {  
STATIC $nbrtitre=0;  
$nbrtitre++;  
echo "<b>".$nbrtitre."<b> ".$titre."<br>";  
}  
titre("Introduction");  
echo "Ce jeu est intéressant au niveau du graphisme<br>";  
titre("Histoire");

echo "L'histoire est très original<br>";  
// Affichera  
// **1** Introduction  
// Ce jeu est intéressant au niveau du graphisme  
// **2** Histoire  
// L'histoire est très original

**Exemple4 :**// Déclaration de la fonction "champ" avec  
argument  
function champ ($nom,$value)  
{  
echo "<input type=\"text\" name=\"".$nom."\"  
size=\"40\" value=\"".$value."\">";  
}  
// appel de la fonction  
champ("sexe","masculin");  
*// Affichera un champ avec comme nom "sexe" et comme valeur par défaut "masculin"*

E**xemple5 avec argument par défaut :**Il peut arriver que vous voulez entrer une valeur par défaut au cas où celle-ci n'est pas renseigné :  
*// Déclaration de la fonction "form" avec argument*function form ($mail="Votre mail",$passe="Votre  
mot de passe")  
{  
echo "<input type=\"text\" name=\"mail\" value=\"".  
$mail."\">";  
echo "<input type=\"text\" name=\"passe\"  
value=\"".$passe."\">";  
if($mail=="Votre mail" AND $passe=="Votre mot  
de passe"){  
echo "<input type=\"submit\"  
value=\"Inscription\">";  
}else{  
echo "<input type=\"submit\"  
value=\"Identification\">";  
}}  
*// appel de la fonction*form($mail\_user,$passe\_user);  
*// Ici nous vérifions par exemple avec les cookies  
si la personne est inscrite ou non  
// Si oui, l'email et le passe son renseigné automatiquement avec un bouton Identification  
// Si non, il demande l'email et un passe avec un bouton Inscription*Il suffit donc pour définir une valeur par défaut de faire comme une variable habituelle  
($nom="valeur") . Si $mail\_user est vide, $mail  
prendra automatiquement la valeur de $email et  
inversement. De même pour $passe ...