TP TIC (PHP)

Matière : Technologie de l'information et de la communication **Classe** : 4 SI **Enseignant** : Ilahi Néjib	**Durée** : 2 heures **Date** : Avril 08 **A.S** :2007/2008

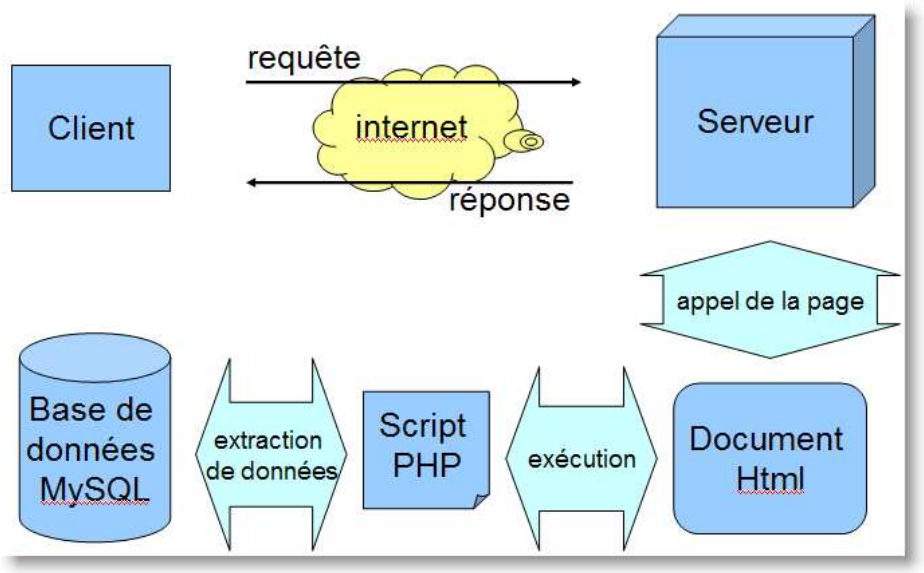
TP N°19 (Le langage PHP):

Objectifs :

- *Apprendre à écrire du script PHP.*
- *Savoir déclarer des variables & constantes*
- *Différencier les types en PHP.*
- *Utiliser les fonctions prédéfinies sur les variables.*

I- Intégrer le code PHP au HTML :

Le langage PHP est un langage de script qui s'insère dans les pages HTML, et les parties écrites en langage PHP sont déclarées au moyen de balises reconnues par le serveur.
Le principe de fonctionnement de PHP est assez simple : le serveur va effectuer un travail d'interprétation avant de retourner le résultat (généralement une page HTML) au client.



L'avantage par rapport à l'HTML est alors considérable, puisqu'il est ainsi possible de concevoir des sites dynamiques. C'est-à-dire qu'une page peut changer de contenu sans aucune intervention humaine ne soit nécessaire. Prenons le cas d'un forum : il est souhaitable que, dès qu'une personne soumet une question, celle-ci soit disponible sur le site. C'est un cas typique d'utilisation d'un site Internet dynamique. Le langage PHP permettra alors de créer une page HTML à partir de ce message (et des autres messages disponible dans la base).

Il faut toujours garder à l'esprit que c'est un langage de script qui est exécuté côté serveur ; il ne dépend donc ni du navigateur ni du système d'exploitation du visiteur.

Devoirs et examens sur : www.kiteb.net

II- Les balises :

Les parties correspondant au code PHP sont déclarées au moyen de balises. Il existe plusieurs types de balises en fonction de vos préférences, habitudes, et de la configuration de PHP sur le serveur ou tournera vos scripts.

- **<? … ?>**
- **<script language=''php''> … </script>**
- **<% … %>**

N.B *: Dans ce chapitre, nous allons opter à la balise suivante :* **< ?php … ?>**

III- Mon premier script :

Le premier script PHP est un grand classique ; il ne faut qu'afficher que texte.

```
<html><head><title>Mon premier script</title></head>
<body>
<?php
  echo "Mon premier script qui n'affiche
        que du texteen attendant du mieux…" ;
?>
</body></html>
```

IV- Les commentaires :

Pour commenter ses scripts et permettre ainsi de les rendre plus clairs. Il existe trois façons de faire :

```
// commentaire de fin de ligne

/* commentaire
sur plusieurs
lignes */

# commentaire de fin de ligne comme en Shell
```

V- Les constantes :

PHP permet de définir des constantes, autrement dit des chaînes des caractères représentant une valeur figée. Les constantes se déclarent par l'instruction **define().**

```
<html><head><title>Les constantes</title></head>
<body>
<?php
     define("Langage", "PHP");
    define("Version",4);
   echo Langage." ".Version;
?>
</body></html>
```

VI- Les variables :

Les variables sont destinées à stocker les données qui seront utilisées et pourront être modifiées lors de l'exécution du programme. Le langage PHP utilise une forme particulière de syntaxe pour définir une variable. Les variables sont représentées avec un caractère '$' préfixant l'identification d'une variable. Par exemple, pour déclarer une nouvelle de nom **''maVariable''**, il suffit de l'appeler **$maVariable** à l'intérieur du code source.

Devoirs et examens sur : www.kiteb.net

Contrairement au langage comme le C ou Pascal, PHP n'exige pas la déclaration préalable des identificateurs avant leur utilisation. ***Les variables sont dites non typées*** , c'est-à-dire qu'une variable peut prendre une valeur de chaîne de caractères et , ensuite, être utilisée pour contenir un entier. Le type est défini à l'affectation de cette variable.

```
<html><head><title>Les Variables</title></head>
<body>
<?php
      $maVariable="0";
     $maVariable=$maVariable +1;
    echo $maVarible;
   echo "<br>";
   $maVariable=1;
 $maVariable=$maVariable/2;
$maVariable= 1+"2 Vive le PHP!";
?>
</body></html>
```

VII- Les variables dynamiques :

Les variables dynamiques (aussi appelées variables ''variables'') sont des variables dont le nom dépend d'autres variables. Les noms de ces variables sont donc construits dynamiquement pendant l'exécution du code PHP.

```
<html><head><title>Les Variables dynamiques</title></head>
<body>
<?php
      $var="Je developpe en PHP.";
     $dyn=  "var ";
    echo $$dyn ; //Affiche "Je developpe en PHP. "
?>
</body></html>
```

Ainsi, en écrivant simplement **$$dyn** on récupère la valeur de la variable dont le nom est contenu dans **$dyn,** c'est-à-dire **$var.**

Il est souvent préférable, et parfois indispensable, de mettre le nom des variables entre accolades. Cela vous permettra, par exemple, de créer un nom de variable avec une partie variable et une partie fixe, comme c'est le cas de l'exemple suivant :

```
<html><head><title>Les Variables dynamiques 2</title></head>
<body>
<?php
      $prefixer="PHP.";
     $suffixe=  "c'est sympa ";
    $dyn= "pre" ;
   echo ${$dyn.."fixe"} ;
    $dyn= "suf" ;
   echo ${$dyn.."fixe"} ;

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

Devoirs et examens sur : www.kiteb.net

VIII- Les types :

Comme nous avons pu les constater, les variables PHP n'ont pas un type prédéfini. Pourtant, un développeur peut être amené à manipuler les types pour le besoin d'une application et les problèmes de conversion peuvent non pas causer des erreurs, mais retourner des résultats surprenant et gênant.
On peut classer les différents types possibles en trois catégories : les types scalaires, les types composés et les types spéciaux. **Types scalaires** :[(int (ou integer) ; double(ou real ou float) ; string ; boolean (ou bool)], **Types composés** [array ; object], **Types spéciaux** [resource, NULL].

IX- Les fonctions prédéfinies sur les variables :

- **setType() :** affecte un type à une variable.
- **getType() :** retourne le type de la variable.

```
<html><head><title>getType et setType</title></head>
<body>
<?php
    setType($toto, "integer") ;
  echo getType($toto) ;
//Renvoie integer
   echo getType($toto);
 $i=2+2;
 echo getType($i);//Renvoie integer
$i="j'aime le chiffre ".$i;
echo getType($i) ;//renvoie string
?>
</body></html>
```

- **isSet()** : Détermine si une variable existe (a été initialisée).
- **unSet()** : Détruit les variables.

```
<html><head><title>isSet et unSet</title></head>
<body>
<?php
    echo isSet($toto);//Renvoie FALSE
  $toto="";
 echo isSet($toto) ;//Renoie TRUE
 unSet($toto) ;
echo isSet($toto);//Renvoie FALSE
?>
</body></html>
```

- **empty()** : Détermine si une variable possède une valeur non nulle ou non (TRUE si une chaine vide '' ou vaut 0, NULL)
- **is_bool()** : Détermine si une variable est de type booléen.
- **is_int()** : Détermine si une variable est de type entier.
- **is_double()** : Détermine si une variable est de type réel.
- **is_string() :** Détermine si une variable est de type chaîne.
- **is_array() :** Détermine si une variable est de type tableau.
- **is_NULL() :** Indique si une variable est NULL.

Devoirs et examens sur : www.kiteb.net

TP TIC (PHP)

Matière : Technologie de l'information et de la communication **Classe** : 4 SI **Enseignant** : Ilahi Néjib	**Durée** : 2 heures **Date** : Avril 08 **A.S** :2007/2008

TP N°20 (Les variables externes, opérateurs et structures de contrôle en PHP):

Objectifs :

- *Savoir les variables externes & d'environnement.*
- *Savoir passer des variables en paramètre de script.*
- *savoir utiliser les opérateurs en PHP.*
- *Les structures de contrôle conditionnelles.*

I- Les variables externes :

Nous pouvons différencier deux types de variables externes :

- Les variables d'environnement (du serveur).
- Les variables passées en paramètre lors de l'appel de la page (soit via un formulaire soit via l'URL).

I-1 : Les variables d'environnement :

Les variables d'environnement sont des variables générées automatiquement par PHP à partir des données récupérées sur le serveur ou de l'entête des requête du client.
Ces variables constituent en fait deux tableaux de portée globale (accessible depuis n'importe quel endroit du script) :

- **$_ENV** contient ''les véritables'' variables d'environnement du serveur, c'est-à-dire toutes les variables dont a pu hériter le compte sous lequel tourne le serveur web.
- **$_SERVER** contient, quant à lui, les variables « internes » du serveur (nom et version du serveur, adresse IP, etc.)

N.B *: Vous pouvez simplement visualiser toutes les variables d'environnement sur la page générée par la fonction* ***phpinfo() ;***

I-2 : Exemples des variables d'environnement :

1.Déterminer l'adresse IP du client :

L'élément $_SERVER["REMOTE_ADDR"] retourne l'adresse IP du client, ce qui est particulièrement utile pour ne pas compter l'utilisateur plusieurs fois dans le nombre des visiteurs d'un site.

2.Déterminer d'où vient le client.

Il est généralement possible de savoir quel lien a suivi le visiteur pour arriver sur le script en cours d'exécution. Pour cela, il suffit de lire le contenu de $_SERVER["HTTP_REFERER"] ; ceci retourne alors une URL.

3.Déterminer le type de navigateur du client.

L'élément $_SERVER["HTTP_USER_AGENT"] permet de récupérer la chaîne d'identificateur fournie par le navigateur du client.

Devoirs et examens sur : www.kiteb.net

```
<html><body>
Vous avez l'adresse IP :
< ?php echo $_SERVER["REMOTE_ADDR"];?><br>
Vous avez un navigateur qui porte la signateur:
< ?php echo $_SERVER["HTTP_USER_AGENT"];?><br>
</body></html>
```

4.Déterminer la langue du visiteur :$_SERVER[‘’HTTP_ACCEPT_LANGUAGE’’]
5.Déterminer le nom du serveur : $_SERVER["SERVER_NAME"]
6.Déterminer le nom du script en cours d'exécution :
La variable $_SERVER["PHP_SELF"] contient le chemin absolu (commence par un /) par rapport à la racine du site web et le nom du script en cours d'exécution. Cela est particulièrement intéressant lorsqu'il s'agit d'écrire des scripts qui doivent s'appeler eux-mêmes (ce qui est souvent le cas des scripts générant des formulaires)
La variable $_SERVER["SCRIPT_FILENAME"], quant à elle, donne le chemin absolu par rapport à la racine du disque dur.

I-3 : Les variables passées en paramètre du script (ou via des formulaires) :

Il est possible de passer des paramètres à un script. Il existe différentes façons de passer ces paramètres, mais la plus simple – ou de moins la plus parlante- consiste à ajouter les paramètres et leurs valeurs à la suite de l'URL, après un point d'interrogation ' ?' chaque couple <nom du parametre>=<valeur> étant séparé d'un autre par le caractère « et commercial » '&'. On parle alors du passage de paramètres par la méthode GET.
Ce qui donne un URL du genre :

http://mondomaine.com/monscript.php?nom=François&prenom=DUPOND

N.B *: Dans ce cas, les différents paramètres seront disponibles dans un tableau appelé* ***$_GET***

Voici un petit exemple d'application :

get_echo.php	form_get.html
<html><body> < ?php echo "Nom=".$_GET["nom"]." "; echo "Prénom=".$_GET["prenom"]." "; ?></body></html>	*<html><body> <form action="get_echo.php" method="get"> Veuillez indiquer vos Noms et Prénoms Prénom :<input type="text" name="prenom"> Nom :<input type="text" name="nom"> <input type="submit"></form> </body></html>*

N.B *: Le fait d'utiliser la méthode GET peut entraîner des problèmes d'ordres divers. Le problème le plus évident est lié à la longueur de l'URL. Plus il y a des paramètres et plus l'URL sera longue, ce qui pourrait mettre en défaut votre serveur.Un autre problème, plus subtil (mais non des moindres) concerne la sécurité.*

→ Pour pallier ces différents problèmes, vous avez la possibilité d'utiliser la méthode **POST.** Dans ce cas, les paramètres ne sont pas ajoutés à l'URL, mais passés "discrètement"dans l'entête de la requête http envoyée au serveur.
Dans ce cas, les deux fichiers présentés précédemment deviennent :

Devoirs et examens sur : www.kiteb.net

post_echo.php	form_post.html
<html><body> < ?php echo "Nom=".$_POST["nom"].” ”; echo “Prénom=”.$_POST[“prenom”].” ”; ?></body></html>	*<html><body> <form action=”post_echo.php” method=”postt”> Veuillez indiquer vos Noms et Prénoms Prénom :<input type=”text” name=”prenom”> Nom :<input type=”text” name=”nom”> <input type=”submit”></form> </body></html>*

II : Les opérateurs

Opérateur	Description
$a + $b	Somme de $a et $b
$a - $b	Différence de $a et $b
$a * $b	Produit de $a par $b
$a / $b	Quotient de $a par $b
$a % $b	Reste de la division de $a et $b
Les opérateurs logiques	
Exemple	**Nom**
$a == $b	Egal
$a !=b	Différent
$a <>$b	Différent
$a < $b	inférieur à
$a < =$b	inférieur ou égal à
$a > $b	supérieur
$a > =$b	supérieur ou égal
$a and $b	Vrai si $a et $b sont vrai
$a && $b	Vrai si $a et $b sont vrai
$a or $b	Vrai si $a ou $b sont vrai
$a | | $b	Vrai si $a ou $b sont vrai
! $a	Vari si $a est faux

III : Contrôles d’erreur

PHP intègre quatre grands niveaux de message d’erreur ou d’alerte. A chacun correspond une constante :

- **E_NOTICE** : simple remarque (comme lorsque l’on essaye d’accéder à un index inexistant d’un tableau).
- **E_PARSE** : en cas d’erreur d’analyse à la compilation.
- **E_WARNING** : une alerte ne nécessitant pas l’arrêt du script.
- **E_ERROR** : une erreur « grave » nécessitant l’arrêt du script.

error_reporting() : Fixe les niveaux d’erreur devant être signalés.

N.B *: Il est toutefois possible de s’affranchir des messages d’erreur pour une instruction donnée sans avoir recours à cette fonction. Pour cela, il existe un opérateur de contrôle d’erreur, c’est le caractère @ ; il peut être mis devant n’importe quelle fonction susceptible de générer une erreur.*

exemple *: @tableau[‘index_inexistant’] ;*

Devoirs et examens sur : www.kiteb.net

```
<html><body>
Vous avez l'adresse IP :
< ?php
error_reporting(E_ALL^E_NOTICE) ;
$fichier=@file('fichier_inexistant.toto') or die("impossible d'ouvrir le fichier") ;
?>
```

Constatation : La fonction file renvoie un booléen indiquant si l'opération s'est bien déroulée. Si le fichier n'existe pas, alors la fonction renvoie un message.

I-4 : Les structures de contrôle If, else, elseIf:

La boucle de contrôle if est sans aucun doute la plus importante dans tous les langages de programmation ; cela vaut en PHP.

Exemple (première étape)

```
<html><head><title>Quizz</title></head>
<body>
< ?php
  if(isset($_POST["reponse"])){
        $rep=$_POST["reponse"];
}else{ $rep="";}
?>
<p>Quel est le site official de PHP ?</p>
<form type="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
<input type="text" name="reponse" value="<?php echo $rep; ?>">
<input type="submit" value="OK">
</form>
<?php
if($reponse!=""){
   if(strtolower($rep)=="www.phpfacile.com")
        echo "Ce n'est pas la bonne réponse";
 else if(strtolower($rep)== "www.php.net") echo "Bingo !";
else echo "Et non, ce n'est pas ".$reponse ;
}
?>
</body></html>
```

Constatation : Si la variable $rep est définie, alors on compare la valeur de la variable réponse transformée en minuscules avec la chaîne "www.phpfacile.com". Si ces chaînes identiques alors "Ce n'est pas la bonne réponse" est affiché, sinon on compare à "www.php.net". Si ces chaînes sont identiques *"Bingo"* est affiché, sinon la phrase "Eh non, ce n'est pas" suivie de la réponse entrée est affichée.

Devoirs et examens sur : www.kiteb.net

TP TIC (PHP)

Matière : Technologie de l'information et de la communication **Classe** : 4 SI **Enseignant** : Ilahi Néjib	**Durée** : 2 heures **Date** : Avril 08 **A.S** :2007/2008

TP N°21 (Les structures de contrôle & les fonctions en PHP):

Objectifs :

- *Savoir utiliser les structures de contrôle en PHP.*
- *Connaître les clauses Break et continue.*
- *savoir Déclarer des fonctions en PHP.*
- *Savoir les portées de variables en PHP.*
- *Savoir les passages des paramètres.*

I- While, do … While :

Pour répéter une série d'instructions, l'instruction While ou "tant que" en français vous serez utile. Voici la syntaxe :

While (expression) instruction ; OU While (expression) {inst1 ; instr2 ; …}

Une variante de la boucle While est la boucle **do… While.** La différence réside dans ce que le test est fait après les instructions.

do { instruction1 ; instruction2 ;} while(expression) ;

```
<html><body>
< ?php
   $i=1 ;
  while($i<=10){ echo print $i++ ;}
?>
</body></html>
```

II- For, foreach :

Les boucles **for** sont très utilisées ; elles fonctionnent, comme le langage Javascript, de la façon suivante : **for(expression1 ; expression2 ; expression3) instruction**
exemple : un script qui affiche les chiffres de 0 à 9 :

for ($i=0 ; $i<10 ; $i++) echo $i ;

➞ Il faut lire cette commande comme suit: après avoir initialise $i à 0, et tant que $i est inférieur à 10, afficher $i, puis l'incrémenter.

L'instruction **foreach,** offre une autre manière de réaliser une boucle, et elle est particulièrement pensée pour les tableaux. Voici deux syntaxes possibles :

foreach($tableau as $valeur) instruction ;
foreach($tableau as $cle =>$valeur) instruction ;

Devoirs et examens sur : www.kiteb.net

III- Exemple (deuxième étape) :

Cette fois, le script proposé permet de répondre à plusieurs questions sur une même page. Il utilise une boucle foreach et une boucle for ainsi que des tests if

```
<html><head><title>Quizz</title></head>
<body>
<?php
 $questions= array("Quel est le site officiel de PHP ? ",
"Quel est le nom de commande qui affiche une chaîne de caractères ? ",
"Quel est le nom de la commande qui met une chaîne de caractère en minuscule ? ") ;
$reponses=array("www.php.net ", "echo","strtolower");
//initialisation des variables
foreach($questions as $index =>$question){
        if(isset($_POST["reponse$index"])){
                ${"rep$index"} = $_POST["reponse$index"];
        }else {
                ${"rep$index"} ="";}
}
?>
<form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
<?php
        foreach($questions as $index =>$question){
                echo $question;
                echo "<input type=\"text\" name=\"reponse$index\"
                        value=\"${"rep$index"}\"><br>";
} ?>
<input type="submit" value="OK"></form>
<?php
        if($rep0!=""){
        $parfait=TRUE;
        for($i=0; $i<sizeof($questions); $i++){
           if(strtolower(${"rep$i"})== $reponses[$i]) {
                echo "<br> la réponse ".($i+1)." est correcte" ;
          }else{
                echo "<br> la réponse ".($i+1)." est fausse" ;
                }
                }
        if($parfait) echo "<br> Bravo ! " ;
                else echo "<br> Perdu …" ;
 }
?>
</bodv></html>
```

IV- Switch :

Utiliser switch revient à utiliser des tests if consécutifs sur la même variable. Voici un exemple :

```
switch ($i){
 case 0 :  case 1 : case 2 : echo " i est positif mais inférieur à 3" ; break ;
 case 3 : echo "i vaut 3" ;
}
```

V- Break, Continue :

Deux instructions permettant de contrôler les sorties de boucle (for, while). **Break** est une instruction qui fait sortir de la boucle, alors que **continue** passe à l'itération suivante sans exécuter les instructions d'après.

Devoirs et examens sur : www.kiteb.net

Voici un exemple :

```
<html><head><title>Continue, Break</title></head>
<body>
<p>
<?php
 for($i=0 ;$i<10 ;$i++){
        echo $i ;  //va afficher 0, 2,4, 6,8
        $i++ ;
       echo "<br>".$i;
}
echo “<br>”;
for($i=0 ;$i<10 ;$i++){
        echo $i;
        $i++;
        if($i>4) break;
        echo “<br>”.$i;
}
echo “<br>”;
for($i=0 ;$i<10 ;$i++){
        echo $i;
        $i++;
        if($i>4) continue;
        echo “<br>”.$i;
}
?>
</p>
</body></html>
```

VI- Les fonctions :

PHP possède, comme de nombreux langages, la possibilité de regrouper des portions de code sous la forme de fonctions (ou procédures). Voici un exemple :

```
<html><head><title>Les fonctions</title></head><body>
<?php
  function premierFonction()
{
        echo “J’aime PHP !”;}
premiereFonction(); ?>
</body></html>
```

VII- Portée des variables :

Une fonction n’est rien d’autres qu’un bout de code isolé du programme principal. C’est donc sans surprise que vous apprendrez qu’il est possible de définir des variables au sein de ses fonctions. Par contre, ce qu’il faut retenir c’est que les variables ainsi définies ne sont accessibles du reste du programme (elles ont une portée locale), comme le démontre l’exemple suivant :

```
<html><body>
<?php
  function bicentenaire()
{       $annee=”2002”;
        echo “$annee c’était le bicentenaire de la naissance de Victor Hugo !” ;}
bicentenaire();
echo $année; //n’affiche rien la variable n’est pas définie
 ?></body></html>
```

Devoirs et examens sur : www.kiteb.net

```
<html><body>
<?php
  $annee =”2002”; $prenom=”Victor”; $nom=”Hugo”;
function bicentenaire(){
        echo “$annee c’était le bicentenaire de la naissance de $prenom $nom !” ;
}
bicentenaire() ;// Les variables $annee, $prenom et $nom ne sont pas définies au niveau de la
// fonction
?>
</body></html>
```

N.B *: Pour utiliser des variables globales dans la fonction, il faut utiliser le mot clé* ***GLOBAL*** *à l’intérieur de cette fonction.*

```
<html><body>
<?php
  $annee =”2002”; $prenom=”Victor”; $nom=”Hugo”;
function bicentenaire(){
global $annee, $prenom, $nom;
        echo “$annee c’était le bicentenaire de la naissance de $prenom $nom !” ;
        $annee= ”2008” ;}
bicentenaire() ;// affiche 2002
echo $annee ;// affiche 2008
?>
</body></html>
```

Il existe une autre façon de faire pour utiliser les variables globales depuis une fonction. Cette méthode consiste à utiliser directement le tableau prédéfini **$GLOBALS.** l’exemple précèdent donnerait alors

```
<html><body>
<?php
  $annee =”2002”; $prenom=”Victor”; $nom=”Hugo”;
function bicentenaire(){
        echo $GLOBALS["annee"].” c’était le bicentenaire de la naissance de”.
$GLOBALS["prenom"].” ”. $GLOBALS["nom"] ;
        $GLOBALS["anneee"]= ”2008” ;}
bicentenaire() ;// affiche 2002
echo $annee; ;// affiche 2008
?>
</body></html>
```

VIII- Les passage des paramètres :

ll existe deux façons afin de passer des paramètres à une fonction :

Passage par valeur

```
< ?php
function bicentenaire($quoi)
{
echo ”2002 c’était le $quoi de Victor”;
}
bicentenaire (”bicentenaire de la naissance”);
echo $quoi;
```

Passage par variable (par référence)

```
< ?php
function bicentenaire(&$quoi)
{
echo ”2002 c’était le $quoi de Victor”;
}
bicentenaire (”bicentenaire de la naissance”);
echo $quoi;
```

Devoirs et examens sur : www.kiteb.net

Cours réalisé par : Ilahi Néjib

TP TIC (PHP)

Matière : Technologie de l'information et de la communication **Classe** : 4 SI **Enseignant** : Ilahi Néjib	**Durée** : 2 heures **Date** : Avril 08 **A.S** :2007/2008

TP N°22 (Les tableaux en PHP):

Objectifs :

- *Savoir déclarer des tableaux en PHP.*
- *Savoir initialiser et remplir un tableau en PHP.*
- *Utiliser les fonctions prédéfinies sur les tableaux.*

.

I- Les tableaux :

En PHP, il est possible de distinguer trois types de tableaux :

- Les tableaux indexés où chaque élément du tableau porte un numéro. Par défaut la numérotation commence avec l'indice 0.
- Les tableaux associatifs où chaque élément du tableau est associé à une clé représentée par une chaîne de caractères.
- Les tableaux mixtes où chaque élément du tableau est associé aussi bien à une indice qu'à une clé.

II- Valeurs, initialisation et remplissage d'un tableau :

1. Valeurs :

Les valeurs contenues dans un tableau peuvent être de tout type connu. Ainsi, un tableau peut stocker des entiers, des réels, des chaînes de caractères, des tableaux, des objets, …
De plus, les types des valeurs contenues dans un tableau peuvent varier d'un indice (ou d'une clé) à l'autre. Le second élément du tableau pourra contenir une chaîne de caractères, même si le premier élément contient un entier.

2. Initialisation :

- La syntaxe de base d'initialisation d'un tableau est la suivante:

$monTableau = array() ;

- L'initialisation d'un tableau se fait généralement en même temps que son remplissage :

$monTableau = array(23, "PHP", 12.4) ;

Ce qui donnera alors : ***$monTableau[0] =23 ; $monTableau[1] = "PHP" ; $monTableau[2] =12.4 ;***

- Dans le cas d'un tableau associatif, il faudra préciser la clé et la valeur selon le schéma suivant :

$tableauAge = array("Nejib"=>28, "Mohamed"=>44, "Salah"=>20);

3. Remplissage :

- D'une manière tout à fait classique, le remplissage d'un tableau se fait par un appel du type :

$monTableau [0]= "Valeur" ; **Ou** $monTableau[0]= "Valeur" ;

III- Les fonctions de manipulation des tableaux :

Il existe de nombreuses fonctions liées au traitement des tableaux. Certaines sont destinées au parcours du tableau, d'autres à la fusion de tableaux, d'autres encore au tri des valeurs ou des clés, etc.

Devoirs et examens sur : www.kiteb.net

Fonctions de base :

1. is_array() : indique si une variable est de type tableau.
2. count() ou sizeof(): Retourne le nombre d'éléments contenus dans un tableau.

```
<html><body>
< ?php
    $monTableau= array("PHP", "C'est", "vraiment", "sympa");
for($i=0 ;$i<count($monTableau ) ;$i++) echo $monTableau[$i] ."<br>";
?>
</body></html>
```

3. array_values() : Retourne un tableau indexé contenant les valeurs des éléments d'un tableau donné (permettre de réindexer un tableau ou de convertir un tableau associatif en tableau indexé).

Fonctions de recherche dans les tableaux :

1. array_keys() : Retourne un tableau indexé contenant les clés et index utilisés dans un tableau donné ou, éventuellement, les clés et index à une valeur donnée.
2. array_search() : Retourne la première clé (ou index) du tableau, associée à une valeur donnée.
3. array_key_exists() : indique si un tableau contient ou non une clé donné.
4. in_array() : Indique si un tableau ou non une valeur donnée.

Fonctions de manipulation de portions du tableau:

1.range() : Retourne un tableau composé des entiers compris entre une valeur de début et une valeur de fin.
2.array_fill() : Retourne un tableau indexé construit à partir des répétition d'une valeur.
3.array_pad() : Retourne un tableau complété (à droite ou à gauche) avec une valeur donnée pour atteindre un nombre total d'éléments spécifiés.
4.array_slice() : Retourne un tableau extrait d'un autre tableau.

Fonctions de parcours de tableau:

1.current() ou pos() : Retourne la valeur actuellement pointée par le pointeur interne du tableau (sans avancer le pointeur).
2.key() : Retourne la clé (ou index) de l'élément actuellement pointé par le pointeur interne du tableau (sans avancer le pointeur).
3.reset() : Remet le pointeur interne au début du tableau et retourne le premier élément.
4.next() : Avance le pointeur interne du tableau et retourne le nouvel élément pointé.
5.prev() : Recule le pointeur interne du tableau et retourne le nouvel élément pointé.
6.end() : Place le pointeur interne à la fin du tableau et retourne le dernier élément.
7.each() : Retourne un tableau contenant la clé et la valeur de l'élément actuellement pointé par le pointeur interne du tableau, puis avance le pointeur.

N.B *: each() et list() un duo très apprécié en occurrence foreach.*
exemple :*reset($tableau) ; while(list($cle,$valeur) = each($tableau)){*
*echo "A la clé $cle est associé la valeur $valeur
" ;}*
OU ***foreach($tableau as $cle =>$valeur) echo…***

Devoirs et examens sur : www.kiteb.net

Fonctions de TRI:

1.Sort() : Tri les éléments d'un tableau dans l'ordre croissant des valeurs. Les clés (ou index) ne sont pas conservées. Le tableau devient obligatoirement un tableau indexé entre 0 et la taille du tableau-1.

2.asort() : Tri les éléments d'un tableau dans l'ordre croissant des valeurs, tout en conservant l'association de clé =>valeur.
3.rsort() : Tri les éléments d'un tableau dans l'ordre décroissant des valeurs. Les clés (ou index) ne sont pas conservées. Le tableau devient obligatoirement un tableau indexé entre 0 et la taille du tableau-1.
4.arsort() : Trie les éléments d'un tableau dans l'ordre décroissant des valeurs, tout en conservant l'association clé=>valeur.
5.ksort() : Trie les éléments du tableau dans l'ordre croissant des clés, tout en conservant l'association clé=>valeur.
6.krsort() : Trie les éléments du tableau dans l'ordre décroissant des clés, tout en conservant l'association clé=>valeur.

Fonctions de statistiques:

1. array_sum() : Retourne la somme des valeurs des éléments d'un tableau (fort pratique pour calculer les moyennes).
2. array_count_values() : Retourne un tableau précisant le nombre d'occurrence d'une valeur dans un tableau donné.

```
<html><head><title>Les fonctions</title></head><body>
<?php
$tableauNote= array("Ali"=>10, "Khaled"=>12, "Sonia"=>10,
                "Meriem" =>10, "Néjib"=>20, "Rami"=>15) ;
$stats = array_count_values($tableauNote);
echo "A ce devoir… <br>";
while(list($note,$nb) = each($stats)){
                        echo "$nb élèves ont eu $note<br>";
} ?>
</bodv></html>
```

Fonctions mettant en œuvre plusieurs tableaux:

1. array_diff() : Retourne un tableau présentant les valeurs contenues dans un tableau, mais absentes dans d'autres tableaux.
2. array_intersect() : Retourne un tableau présentant les valeurs contenus dans un tableau et dans un ensemble d'autres tableaux.
3. array_merge() : Retourne un tableau issu de la fusion d'un ensemble de tableaux.

Devoirs et examens sur : www.kiteb.net

```
<html><body>
<?php
$sportifs=array("L.Messi"=>"Bercelona","Maldeni"=>"AC.Milan",
"T.Henry"=>"Bercelona","Inzaghi"=>"AC.Milan","V.Valdez"=>"Bercelona",
"Kaka"=>"AC.Milan");
$Bercelona=array_keys($sportifs,"Bercelona");
echo "Voici les joueurs de La Barça!!!<br>";
for($i=0;$i<sizeof($Bercelona);$i++)
   echo $Bercelona[$i]."<br>";
 echo "<hr>";
 echo array_search("Bercelona",$sportifs);
echo "<hr>";
if(array_key_exists("L.Messi",$sportifs))
   echo "Le joueur existe";
   else
   echo "Le joueur n'existe pas";
echo "<hr>";
echo "<b> Les fonctions de manipulation de portion du tableau </b><br>";
$php= array("Langage","de","creation","site","web");
$par=range(0,100);
for($i=0;$i<count($par);$i++)
  if(($i%10==0)&($i>=10)){
     echo $par[$i];
        echo "<br>";}
        else
         echo $par[$i]."  ";
echo "<hr>";
$comp= array_pad($php,10,"néjib");

for($i=0;$i<sizeof($comp);$i++)
  echo $comp[$i]."  ";
echo "<hr>";
echo "<b>Les fonctions de parcours de tableau</b><br>";
$tableau1=array("Début","","Milieu",0,"Fin");
$valeur=reset($tableau1);
while($valeur!==FALSE){
echo "$valeur<br>";
$valeur=next($tableau1);
}
echo "<br>";
echo "<hr>";
echo "<b>La fonction each();</b><br>";
reset($php);
while(list($cle,$valeur)=each($php)){
echo "A la clé $cle est associé la valeur $valeur <br>";
}
echo "<hr>";
$inv=array_reverse($php);
while(list($cle,$valeur)=each($inv))
echo "A la clé $cle est associé la valeur $valeur <br>";
$tableauVille= array("Bou Jaafer"=>"Sousse","Avenue Habib Bourguiba"=>"Tunis",
"Théatre Eljam"=>"Mahdia","Falaise"=>"Monastir","Bizerte"=>"Courniche");

asort($tableauVille);
while(list($key,$val)=each($tableauVille))
 echo "Le clé $key a pour valeur $val<br>";
echo "<br>" ;
echo "<hr>";
$agenda1=array("Lundi"=>"Médecin","Mardi"=>"Coiffeur");
$agenda2=array("Mardi"=>"Réunion","Jeudi"=>"Travail","Vendredi"=>"Prière");
$agenda3=array("Samedi"=>"VTT","Dimanche"=>"Stade");
$fusion =array_merge($agenda1,$agenda2,$agenda3);
print_r($fusion);

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

Devoirs et examens sur : www.kiteb.net

Cours réalisé par : Ilahi Néjib

TP TIC (PHP)

Matière : Technologie de l'information et de la communication **Classe** : 4 SI **Enseignant** : Ilahi Néjib	**Durée** : 2 heures **Date** : Avril 08 **A.S** :2007/2008

TP N°23 (Les inclusions de fichiers & Fonctions mathématiques & la gestion des dates):

Objectifs :

- *Savoir appeler des fichiers en PHP.*
- *Savoir utiliser les fonctions mathématiques en PHP.*
- *Savoir déclarer et utiliser les dates en PHP.*

.

I- Les inclusions de fichiers :

Le langage PHP permet d'insérer, dans un script, du code qui provient d'un (ou de plusieurs) autres fichiers.
Pour cela, PHP propose quatre fonctions (ou plutôt, instructions) :

- **require() ;**
- **include() ;**
- **require_once() ;**
- **include_once() ;**

include_01.php

```
<html><body>
< ?php
   include ("include_01a.php");
  include ("include_01b.php");
 monEcho ($maChaine);
?>
 </body></html>
```

include_01a.php

```
<html><body>
< ?php
   $maChaine="Bonjour tout le monde";
?>
 </body></html>
```

include_01b.php

```
<html><body>
< ?php
   function monEcho($chaine){
                echo $chaine ;
        }
?>
 </body></html>
```

II- Les fonctions mathématiques :

La bibliothèque mathématique fournit des constantes et des fonctions qui vous serviront si vous souhaitez faire de calcul.

1.Constantes :

- **Constantes : M_SQRT2(racine carrée de 2) ; M_SQRT3 (racine carrée de 3) ;**
- **Trigonométrie: M_PI (pi) ; M_PI_2(pi/2) ; M_PI_4 (pi/4);**
- **Logarithme: M_E(E); M_LN2(ln(2));…**

Devoirs et examens sur : www.kiteb.net

2. Fonctions

- **Test de validité :**

Fonction	Description
is_finite()	indique si une valeur est finie ou non.
is_infinite()	Teste si une valeur est infinie.
is_nan()	Teste si une valeur est indéfinie.

```
<html><body>
<?php
  if (is_finite(3)) echo "3 est finie <br>";
        else echo " 3 n'est pas finie <br>";
 if(is_finite(pow(0,-1)) echo "1/0 est finie <br>";
  else echo "1/0 n'est pas finie <br>";
if(is_infinite(pow(0,-1)) echo "1/0 est infini <br>";
  else echo "1/0 est fini <br>";
if(is_nan(3)) echo "3 est indéfini <br>";
 else echo "3 n'est pas indéfini <br>" ;
if(is_nan(sqrt(-1)) echo "racine carrée de -1 est indéfini";
 else echo "racine carrée de -1 n'est pas indéfini" ;
?>
</body></html>
```

- **Trigonométriques:**

Fonction	Description
pi()	Cette fonction retourne une valeur approximative de Pi
cos(), sin(), tan()	retourne la valeur de cosinus, sinus et tangente de l'angle exprimé en radians.
deg2rad(), rad2deg()	retourne en radians un angle exprimé en dégrés et inversement.

- **Logarithmiques :**

Fonction	Description
exp()	Retourne l'exponentiel de la valeur donnée.
log()	Retourne le logarithme népérien de la valeur donnée.
log10()	Retourne le logarithme décimal.

- **Puissance :**

Fonction	Description
pow($base,$exposant)	Cette fonction retourne $base à la puissance $exposant.
sqrt()	Cette fonction retourne la racine carrée de la valeur entrée en paramètre.

```
<html><body>
<?php
  echo pow(2,3);echo "<br>";
echo pow(-1,10); echo "<br>";
echo pow(-4,0.5);
echo sqrt(4);
echo sqrt(9);
echo sqrt(-3);
?>
</body></html>
```

Devoirs et examens sur : www.kiteb.net

- **Arrondi**

Fonction	Description
ceil()	Arrondit à l'entier supérieur
floor()	Arrondit à l'entier inférieur

```
<html><body>
<?php
  echo ceil(3.0);echo "<br>";
echo ceil(3.1); echo "<br>";
echo ceil(3.5);echo "<br>";
echo floort(3.0); echo "<br>";
echo floor(3.1); echo "<br>";
echo floor(3.5); echo "<br>";
?>
</body></html>
```

- **Hasard**

Fonction	Description
srand()	Initialise les générateurs de nombres aléatoires.
rand()	Générateur de nombre aléatoire
getRandMax()	Retourne la plus grande valeur que peut retourner la fonction rand().

```
<html><body>
<?php
  echo "getRandMax()=".getRandMax()."<br>";
 srand ((double)microtime()*1000000);
echo  rand()."<br>";
echo rand(0,2);
?>
</body></html>
```

- **Conversion des bases :**

Fonction	Description
base_convert($nombre, $depuisbase, $versbase)	Retourne un nombre converti d'une base à une autre. Base d'origine et vers base sont entre 2 et 36
binDec()	Retourne la conversion d'un nombre binaire en un nombre de base 10.
decBin()	Retourne la conversion d'un nombre décimal en un nombre de base 2.
decHex()	Retourne la conversion d'un nombre hexadécimal en un nombre de base 10.
hexDec()	Retourne la conversion d'un nombre décimal en un nombre de base 16.
decOct()	Retourne la conversion d'un nombre octal en un nombre de base 10.
octDec()	Retourne la conversion d'un nombre décimal en un nombre de base 8.

- **Autres :**

Fonction	Description
Abs()	cette fonction retourne la valeur absolue de l'argument.
max()	Retourne le plus grand élément.
min()	Retourne le plus petit élément.

Devoirs et examens sur : www.kiteb.net

III- Les fonctions de date et heure :

- **Time() :** Retourne le nombre de secondes écoulées depuis le 1er Janvier 1070.
- **mktime() :** Retourne le timestamp de la date spécifiée.

```
<html><body>
<?php
  echo "le timestamp actuel est ".time()."<br>" ;
 echo  "Le timestamp pour la date du 10/01/2001 18 :15 est " ;
echo mktime(18,15,0,1,10,2001). "<br>" ;
?>
</body></html>
```

- **getDate() :** Retourne les différents champs d'une date sous forme d'un tableau associatif contenant les clés suivants : *"mday "*, *"mon"*, *"year"*, *"hours"*, *"minutes"*, *"seconds"*.

  syntaxe : array getDate(int $date) ;

```
<html><body>
<?php
$tableauDate = getdate();
$tableauDate["mon"] = $tableauDate["mon"]-6;
if($tableauDate["mon"]<1){
        $tableauDate["mon"]= $tableauDate["mon"]+12 ;
        $tableauDate["year"]= $tableauDate["year"]-1 ;
}

echo "Il ya 6 mois, nous étions le " ;
echo $tableauDate["mday"]. "/".$tableauDate["mon"] ;
echo "/".$tableauDate["year"] ;
echo "<br>" ;

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

- **date() :** Représente une date (ou date courante) selon le format suivant

```
<html><body>
<?php
 echo "Nous somme le ".date("l j F Y")." " ;
echo "Il est ".date("H :i :s")."<br>" ;
?>
</body></html>
```

- **microtime() :** Retourne une chaîne de caractères contenant les microsecondes de la date courante.

Devoirs et examens sur : www.kiteb.net

Cours réalisé par : Ilahi Néjib

TP TIC (PHP)

Matière : Technologie de l'information et de la communication **Classe** : 4 SI **Enseignant** : Ilahi Néjib	**Durée** : 2 heures **Date** : Avril 08 **A.S** :2007/2008

TP N°24 (La manipulation des chaînes de caractères):

Objectifs :

- *Savoir déclarer des chaînes des caractères.*
- *Savoir les différentes fonctions sur les chaînes..*

.

I- Généralités :

Ce chapitre est peut être le plus important, l'utilisation des chaînes de caractères étant en effet inévitable pour produire le code HTML désiré.
Les chaînes de caractères peuvent être encadrées soit par des apostrophes (‘) soit par des guillemets (").

N.B *: Lorsque vous utilisez des guillemets, vous pouvez intégrer directement des noms des variables au sein des chaînes, et celles-ci seront substituées par leur valeurs. Ce n'est pas le cas si vous utilisez les apostrophes.*

```
<html><head><title>Les chaînes</title></head><body>
<?php
$monNom="Ilahi";
echo "Bonjour  $monNom <br>";
echo 'Bonjour $monNom <br>';
?>
</body></html>
```

- Dans le cas de l'utilisation des guillemets, nous avons besoin de faire appel aux séquences d'échappement afin de pouvoir afficher le signe '$'.Mais ceci concerne également les caractères suivants :

Notation	Description
\n	Chargement de ligne.
\r	Retour chariot
\t	Tabulation
\\	Pour afficher \ (et éviter la confusion avec le caractère d'échappement).
\$	Pour afficher \$(et éviter la confusion avec l'interprétation d'une variable).
\ "	Pour afficher " (et éviter la confusion avec la fin de la chaîne).

Exemple : "Clouche a dit "Plus on est de fous moins y'a de riz" pas faux …
La phrase d'exemple s'écrit donc :

echo "*Clouche a dit \"Plus on est de fous moins y'a de riz\",pas faux…*" ;

Devoirs et examens sur : www.kiteb.net

II- Fonctions de gestion des chaînes de caractères :

1. Manipuler les caractères :

- **chr() :** Retourne le caractère d'un code ASCII.
- **ord() :** Retourne le code ASCII

2. Extraction et substitution :

2.1 : Extraction :

- **substr() :** Retourne une partie d'une chaîne de caractères.

```
<html><body>
<?php
echo substr("abcdefghij",2) ;echo "<br>";
echo substr("abcdefghij",0,3); echo "<br>";
echo substr("abcdefghij",-3,2); echo "<br>";
echo substr("abcdefghij",5,20); echo "<br>";
?>
</body></html>
```

- **strstr() :** Permet de récupérer la partie d'une chaîne de caractères située à partir de la première occurrence d'une sous-chaîne.
- **stristr() :** de même que **strstr()**, mais **stristr()** ne prend pas en compte la casse.

```
<html><body>
<?php
echo "strstr:".strstr("Ilahi_nejib@hotmail.com", "@");echo "<br>";
echo "strstr :".strstr("Voici un exemple stupide d'exemple","exemple") ; echo "<br>" ;
echo "strstr :".strstr("Voici un exemple stupide d'exemple","Exemple") ; echo "<br>" ;
echo "stristr :".stristr("Voici un exemple stupide d'exemple","Exemple") ; echo "<br>" ;
?>
</body></html>
```

- **strrchr() :** Permet de récupérer la partie d'une chaîne de caractères située à partir de la dernière occurrence d'une sous-chaîne.

2.2 : Substitution :

- **substr_replace() :** Retourne une chaîne de caractères issue d'une chaîne dans laquelle une partie est remplacée par une autre. Permet également d'insérer du texte.
- **str_replace() :** Permet de remplacer des occurrences par d'autres.
- **strtr() :** Retourne une chaîne de caractères dans laquelle certains caractères ont été remplacés par d'autres.

```
<html><body>
<?php
echo substr_replace("abCDEF","AB",0,2);
echo "<br>";
echo substr_replace("100000 Euros",",",9,0);
$phrase="Jessicasse-croutes dans mon panier"; echo $phrase."<br>";
$phrase=str_replace("Jessica","J'ai 6 ca", $phrase) ; echo $phrase."<br>";
$phrase="Les accents sur à, é, è, ù, ê, ô vont être supprimés" ;
$phrase=strtr($phrase,"àéèùêôî","aeeueoi") ;
echo $phrase ;
?>
</body></html>
```

Devoirs et examens sur : www.kiteb.net

3. Fonctions statistiques (longueur et nombre d'occurrences) :

- **strlen() :** Retourne la longueur d'une chaîne de caractères.
- **substr_count() :** Compte le nombre d'occurrence d'une sous-chaîne dans une autre.

```
<html><body>
<?php
echo strlen("Texte de 22 caractères");echo "<br>" ;
echo substr_count("toto","to");echo "<br>";
echo substr_count("Je me demande combine il y a de e dans cette phrase","e");
echo "<br>" ;
?>
</body></html>
```

4. Fonctions de position :

Il est également utile de pouvoir récupérer l'index d'un caractère dans une chaîne :

- **strpos() :** retourne la position du premier caractère de la première occurrence d'une chaîne de caractère dans une autre, à partir d'un certain index.
- **strrpos() :** retourne la position du dernière occurrence d'une chaîne de caractère.

```
<html><body>
<?php
echo strlen("Texte de 22 caractères");echo "<br>" ;
echo substr_count("toto","to");echo "<br>";
echo substr_count("Je me demande combine il y a de e dans cette phrase","e");
echo "<br>" ;
?>
</body></html>
```

5. Comparaison de chaînes de caractères :

Il est possible de comparer deux chaîne de caractères en utilisant simplement l'opérateur ==. Cependant, il peut être également intéressant de comparer des chaînes semblables à l'aide de fonctions.

- **strcmp() :** Compare deux chaîne de caractères.
- **strcasecmp() :** compare deux chaînes de caractères sans tenir compte de la casse.

```
<html><body>
<?php
echo strcmp("mot","mot") ;echo "<br>" ;
echo strcmp ("mot ","mot"); echo "<br>";
?>
</bodv></html>
```

6. Gestion des caractères spéciaux :

Il est parfois utile d'encoder des chaînes de caractères. C'est le cas, par exemple, pour échapper certains caractères spéciaux.

- **addSlashes() :** Permet l'échappement de certains caractères. Concrètement, il s'agit des caractères : ‘,",\ .
- **stripSlashes() :** Retire les slashes ajoutés par la fonction **addSlashes().**

```
<html><body>
<?php
$chaine="C'est \cool"; echo  $chaine."<br>";
echo addSlashes($chaine);
?>
</bodv></html>
```

Devoirs et examens sur : www.kiteb.net

7. Conversion des caractères en code HTML :

Dans le de l'écriture de pages HTML, il peut être utile de transformer certains caractères spéciaux en leurs équivalents HTML.

- **htmlEntities() :** Retourne une chaîne de caractères pour laquelle tous les caractères spéciaux ont été convertis en leurs équivalents HTML.

```
<html><body>
<?php
$chaine="Un message avec du HTML <i> &, è,\" et des' </i>";
echo $chaine ."<br>";
echo htmlEntities($chaine);
?>
</body></html>
```

8. Suppression des espaces :

Certaines fonctions permettant d'effacer les espaces superflues en début et/ou fin de chaînes de caractères.

- **trim() :** Retourne une chaîne de caractères sans les espaces.
- **ltrim() :** Retourne une chaîne dans laquelle toutes les espaces de début de chaîne ont été supprimés.
- **rtrim() :**Retourne une chaîne dans laquelle toutes les espaces de fin de chaîne ont été supprimés.

9. Modification de casse :

Pour changer la casse des caractères, il existe quatre fonctions strToUpper(), strToLower() , ucFirst(), et enfin ucWords().

- **strToUpper() :** Retourne une chaîne dans laquelle tous les caractères ont été mis en majuscules.
- **strToLower() :** Retourne une chaîne dans laquelle tous les caractères ont été mis en minuscules.
- **ucFirst() :**Retourne une chaîne de caractères dans laquelle le premier caractère de la chaîne a été mis en majuscule.
- **ucWords() :** Retourne une chaîne de caractères dans laquelle le premier caractère de chacun des mots a été mis en majuscule.

```
<html><body>
<?php
$chaine= "cette ChaiNe serA transFormee.";
echo strtolower($chaine) ;echo "<br>";
echo strtoupper($chaine) ;echo "<br>";
echo ucfirst($chaine) ;echo "<br>";
echo ucwords($chaine) ;echo "<br>";
?>
</body></html>
```

10. Fusion et découpe :

- **implode() :** Permet, à partir d'un tableau, de reconstituer une chaîne de caractères.
- **explode() :** Permet de retourner un tableau contenant les morceaux de chaînes séparés par un délimiteur défini par le programmeur.

```
<html><body>
<?php
print_r(explode("\n","Ceci \nest\nune\nphrase en plusieurs\nlignes"));
print_r(implode(" ", explode("\n","Ceci \nest\nune\nphrase en plusieurs\nlignes"))) ;
?></body></html>
```

Devoirs et examens sur : www.kiteb.net

Cours réalisé par : Ilahi Néjib

TP TIC (PHP)

Matière : Technologie de l'information et de la communication **Classe** : 4 SI **Enseignant** : Ilahi Néjib	**Durée** : 2 heures **Date** : Avril 08 **A.S** :2007/2008

TP N°25 (L'utilisation des bases de données):

Objectifs :

- *Savoir utiliser des bases de données sous PHP.*
- *Savoir les différentes fonctions de gestion de BDD.*

I- Généralités :

Une base de données peut être assimilée à un ensemble de fichiers. Ces derniers sont gérés uniquement par un logiciel serveur.
Les bases de données peuvent être utilisées pour gérer un répertoire téléphonique ; dans ce cas, vous stockerez probablement des noms, prénoms, adresses et numéros de téléphone et, la plupart du temps, vous filtrerez ces informations par nom. Elles peuvent également être utilisées pour stocker les messages d'un forum, qui pourront être triés par date ou par fil de discussion.
Dans ce chapitre nous allons opter à la base de données :

II- MySQL :

MySQL est probablement la base de données la plus connue des adaptes de PHP. C'est très probablement d'ailleurs la base de données que les hébergeurs proposent.
-Installation : Si vous êtes sous Windows et avez installé EasyPHP , vous n'aurez rien à installer ou configurer, puisque tout est fait automatiquement (aussi bien pour ce qui concerne MYSQL que pour ce que concerne PHP). Avec EasyPHP1.7, c'est MySQL 4.0.15 qui est fourni.
- Utilisation : L'utilisation basique de la BDD s'opère selon le schéma suivant :

1. Connexion grâce aux fonctions ***mysql_pconnect()*** et ***mysql_select_db().***
2. Soumission de la requête via la fonction ***mysql_query().***
3. Déconnexion grâce à la fonction ***mysql_close().***

II-1 Connexion à une base de données :

La connexion à une base de données MYSQL, nécessite deux opérations. Il faut, dans un premier temps, se connecter au serveur de base de données avec la fonction ***mysql_pconnect() ;*** la sélection de la base s'opère dans un second temps avec ***mysql_select_db()***.

1. **mysql_pconnect() :** Etablit une connexion avec le serveur de base de données.

resource mysql_pconnect(string $monserveur, [, string $utilisateur[, string $motDePasse]]]);

2. **mysql_select_db() :** Sélectionne une base de données.

boolean mysql_select_db(string $nomBaseDeDonnées[, resource $idConnexion]) ;

Devoirs et examens sur : www.kiteb.net

II-2 Exécution de la requête SQL :

1. **mysql_query() :** Exécute une requête SQL

resource mysql_query(string $requete[, resource $idConnexion]) ;

II-3 Déconnexion :

2. **mysql_close() :** Met fin à la connexion à la base de données et libère les ressources associées.

boolean mysql_close(resource $idConnexion]) ;

II-4 Exemple :

N.B *: Prenez garde ! Avant d'exécuter ce script, assurez-vous que la base "**maBase**" existe. Par défaut le nom de serveur est "localhost" et le nom d'utilisateur est "root".*

- Comme cela est fortement conseillé, nous avons, ici, isolé les paramètres de connexion dans fichier aisément repérable.

parametres_bd_inc.php

```
<html><body>
< ?php
// Parametres de connexion à la BDD
$serveur        = "localhost" ;
$port           = "" ;
$utilisateur    = "root" ;
$motDePasse = "" ;
$base           ="maBase" ;
?>
</body></html>
```

- et qui est utilisé par le script principal.

insert01.php

```
<html><body>
< ?php
// Parametres du script
require_once("parametres_bd_inc.php") ;
$table ="tableexemple" ;
//Inclusion du script contenant les fontions
require_once("insert01_bd_inc.php");
//Connexion à la base de données
$idConnexion =mysql_pconnect($serveur, $utilisateur, $motDePasse) ;
if( !$idConnexion){
        die ("<b> Impossible de se connecter à la base de données</b>") ;
}
if( !mysql_select_db($base){
         die("<b> Impossible de se connecter à la base de données </b>");
}
// Appel à la fonction principale
if(EX_initialiseBD($idConnexion,$table){
   echo "Voilà, une nouvelle table avec quelques données." ;
}else{    echo "La création et l'alimentation de la table à échouée." ;
}
?></body></html>
```

Devoirs et examens sur : www.kiteb.net

insert01_bd_inc.php

```
<html><body>
< ?php
function EX_initialiseBD($idConnexion, $table){
//Supprime la précédente table si elle existe déja
$requete ="DROP TABLE $table" ;
@mysql_query($requete, $idConnexion) ;
// crée la table
$requete = "CREATE TABLE $table (filmId INTEGER AUTO_INCREMENT PRIMARY KEY, ".
                                                    "film VARCHAR(64))";
if(!mysql_query($requete,$idConnexion)) return FALSE;

// Ajout quelques données
$requete ="INSERT INTO $table (film) VALUES ('Forrest Gump')";
if(!mysql_query($requete,$idConnexion)) return FALSE;

$requete ="INSERT INTO $table (film) VALUES ('Matrix')";
if(!mysql_query($requete,$idConnexion)) return FALSE;

$requete ="INSERT INTO $table (film) VALUES ('La cite de la peur')";
if(!mysql_query($requete,$idConnexion)) return FALSE;

return TRUE;
?>
</body></html>
```

III Lecture des enregistrements:

Dans le cas d'une requête retournant une liste de données, il convient de récupérer l'identifiant de requête et de l'utiliser pour lire, en boucle, chaque enregistrement. Il existe diverses fonctions, chacune permettant de récupérer les champs des enregistrements sous différentes formes. La forme la plus simple est celle retournée par la fonction ***mysql_fetch_row().***

- **mysql_fetch_row() :** Retourne un enregistrement, retourné par une requête SQL, sous la forme d'un tableau indexé. Le syntaxe est le suivant

```
array mysql_fetch_row(resource $idResultat)
```

select_eei_bd_inc.php

```
<html><body>
<?php
 function EX_listeContenu($idConnexion, $table){
    $requte = "SELECT * FROM $table";
   $idResult = mysql_query($requete, $idConnexion);
  if(!$idResult) return FALSE;
 while($enreg = mysql_fetch_row($idResult)){
     echo "filmId =".$enreg[0]."   Film= ".$enreg[1]."<br>";
}
return TRUE;
}
?>
</body></html>
```

- **mysql_fetch_array() ou mysql_fetch_assoc():** Retourne un enregistrement, retourné par une requête SQL, sous la forme d'un tableau associatif. Le syntaxe est le suivant

```
array mysql_fetch_array(resource $idResultat)
```

Devoirs et examens sur : www.kiteb.net

select_eea_bd_inc.php

```
<html><body>
<?php
 function EX_listeContenu($idConnexion, $table){
    $requte = "SELECT * FROM $table";
   $idResult = mysql_query($requete, $idConnexion);
  if(!idResult) return FALSE;
 while($enreg = mysql_fetch_array($idResult)){
     echo "filmId =".$enreg["filmId"]."   Film= ".$enreg["film"]."<br>";
}
return TRUE;
}
?>
</body></html>
```

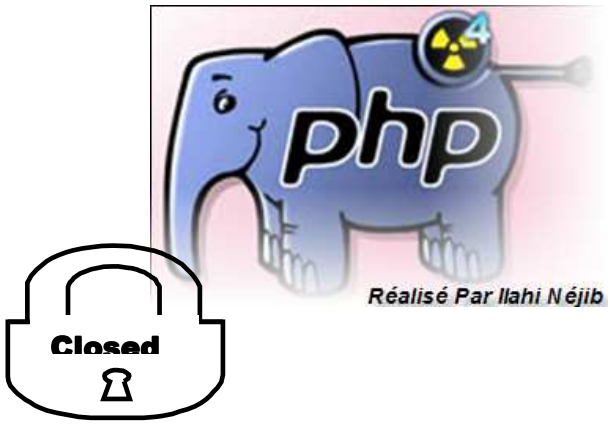
IV Nombre d'enregistrements:

Si vous souhaitez connaître le nombre d'enregistrements avant de les lire, vous pouvez faire appel à la fonction :

- **mysql_num_rows() :** Retourne le nombre d'enregistrements retournés par la requête SQL.

count_num_rows_bd_inc.php

```
<html><body>
<?php
 function EX_listeContenu($idConnexion, $table){
    $requte = "SELECT * FROM $table";
   $idResult = mysql_query($requete, $idConnexion);
  if(!idResult) return FALSE;
echo "Il y a ".mysql_num_rows($idResultat)." enregistrements dans la table <br>" ;
 while($enreg = mysql_fetch_array($idResult)){
     echo "filmId =".$enreg["filmId"]."   Film= ".$enreg["film"]."<br>";
}
return TRUE;
}
?>
</body></html>
```



Devoirs et examens sur : www.kiteb.net