2020

CCTL

PHP

MBEUNDJAL

EXIA CESI - UCAC ICAM |armel.ndoumba.cm@viacesi.fr

TABLE DES MATIERES

VARIABLES

DEFINITION

Une variable est une information stockée en mémoire temporairement. En PHP, elles n'existent lorsque la page est en cours de génération, une fois générée, les variables sont supprimées de la mémoire.

Une variable se caractérise par :

- Son nom : pour pouvoir la reconnaître
- Sa valeur : qui représente l'information qu'elle contient et qui peut changer au cours de l'exécution des instructions

TYPE DE VARIABLES

- Les chaînes de caractères (**string**)
- Les nombres entiers (**int**)
- Les nombres décimaux (**float**)
- Les booléens (**bool**)

AFFECTATION D'UNE VALEUR A UNE VARIABLE

Pour ce faire, on utilise le symbole « $ » précédé du nom de la variable puis de sa valeur. Exemple $< \boldsymbol{php}\ \ \$\boldsymbol{variable} = '\boldsymbol{valeur}';\ ?>$

NB :

On utilise des côtes (**'_'**) ou double côtes (**"_"**) pour des variables de type string

On concatène des valeurs et variables de type string soit avec les doubles côtes, soit avec des côtes simples tout en prenant soin dans ce cas de délimiter les deux valeurs par un point(.).

On peut effectuer les opérations d'addition, soustraction, multiplication, de division et de modulo, d'incrémentation, de décrémentation… sur les variables.

LES SPECIFICITES DU TABLEAU

Un tableau ou encore ***array*** est une variable composée de plusieurs autres variables.

On distingue deux types de tableaux à savoir numérotés et associatifs.

LES TABLEAUX NUMEROTES

Prenons un exemple de tableaux et notons les différentes modifications possibles d'effectuer.

EXEMPLE : SUPPOSONS UN TABLEAU NOMME « MBEUNDJAL »

Clé	**Valeur**
0	Valeur_1
1	Valeur_2
2	Valeur_3

La création du tableau mbeundjal se fera à l'aide de la fonction $array$. Ainsi on aura

```
<?php
    $mbeundjal = array ('Valeur_1', 'Valeur_2', 'Valeur_3');
?>
```

On peut créer ou modifier la valeur d'une case en l'affectant la valeur souhaitée

```
<?php
    $mbeundjal = array ('Valeur_1', 'Valeur_2', 'Valeur_3');
    $mbeundjal[0] = 'valeur de la premiere case modifie';
    $mbeundjal[3] = 'nouvelle case cree et affectation de valeur';
    $mbeundjal[] = 'nouvelle case cree, PHP se chargera de faire une incrementation automatique de la cle';
?>
```

TABLEAUX ASSOCIATIFS

Ils fonctionnent de la même façon que les tableaux numérotés à la seule différence qu'au lieu de numéroter les cases, on va les étiqueter en leur donnant à chacune un nom différent. Ils permettent de découper une donnée en plusieurs sous-éléments.

Pour construire un tableau associatif, on utilise la fonction array en précisant l'étiquette devant chaque information

```php
<?php
    $tableau_associatif = array (
        'etiquette1' => 'valeur1',
        'etiquette2' => 'valeur2',
        'etiquette3' => 'valeur3',
        '...'
        'etiquetten' => 'valeurn',
    );
?>
```

Pour afficher un tableau associatif, on peut afficher un élément en indiquant le nom de l'élément entre crochet. Par exemple

```php
< ?php echo $tableau_associatif['etiquette1'] ; ?>
```

Ou alors afficher tout le contenu du tableau en utilisant la boucle **foreach**. Pour notre précédent tableau, nous aurons :

```php
<?php
    foreach($tableau_associatif as $element) {
        echo $element.'<br />'; // l'affichage sera les valeurs 1, 2,3 ... n
    }
?>
```

Notons la présence de la variable $element qui récupère à chaque fois la valeur de l'élément présent. Le code précédent ne nous permettra que d'afficher les valeurs et non les clés représentées par les étiquettes. Pour les représenter on modifiera le code et on aura :

```php
<?php
    foreach($tableau_associatif as $cle => $element) {
        echo $cle.'vaut'.$element.'<br />'; // nous aurons un affichage de la forme
                                           //etiquetten vaut n
    }
?>
```

Notons la présence de la variable $cle qui contiendra l'élément en cours d'analyse, ici c'est 'etiquette'. Et $element contiendra la valeur de l'élément en cours (valeur1, valeur2…)

TABLEAUX ASSOCIATIFS MULTIDIMENSIONNELS

Ce sont des tableaux qui contiennent eux-mêmes d'autres tableaux en valeur.

On appelle ainsi tableau à deux dimensions un tableau qui contient un ou plusieurs tableaux en valeurs, tableau à trois dimensions un tableau qui contient un ou plusieurs tableaux en valeurs qui contiennent eux-mêmes d'autres tableaux en valeurs et etc.

EXEMPLE

```
<?php
$tm = array(
1 => array('Marque' => 'Audi', 'Modele' => 'A1', 'Type' => 'Citadine'),
2 => array('Marque' => 'Volkswagen', 'Modele' => 'Passat', 'Type' => 'Berline'),
3 => array('Marque' => 'Volkswagen', 'Modele' => 'Golf', 'Type' => 'Compact'),
4 => array('Marque' => 'Mazda', 'Modele' => 'CX-5', 'Type' => 'SUV')
); ?>
```

Pour afficher un tel tableau on utilise la boucle foreach. Sa représentation sera la suivante :

```
< ?php
foreach ($tm as $key => $value) {
    echo $key.':<br />';
    foreach ($value as $key2 => $value2) {
            echo '['.$key2.'] '.$value2.'<br />';
            echo '<br />';
            # code...
    }
    # code...
}
?>
```

On remarque que dans la 1ere boucle **foreach**, on affiche la clé du tableau global et que dans la seconde boucle, la valeur du tableau englobant est entré comme un tableau en paramètre.

LES FONCTIONS NATIVES

https://www.php.net/manual/fr/indexes.functions.php

GERER LES SESSIONS

DEFINITION

C'est un moyen de conserver des variables sur toutes les pages d'un site web. Ses applications et avantages sont nombreuses mais on pourrait noter la restriction d'accès et droit aux utilisateurs d'un sur un site ou encore un affichage plus dynamique des pages en fonction des actions de son utilisateur.

GESTION DES SESSIONS EN PHP

La gestion des sessions en PHP passe par 3 étapes :

- Une session est créée pour un utilisateur lorsqu'il arrive sur un site. PHP génère ainsi un numéro unique appelé **ID** de session ou **PHPSESSID**. C'est ce numéro qui est transmis de page en page via les **cookies**.
- Une fois la session générée, il est possible de créer une infinité de variables de sessions selon le besoin. Ça peut être le nom, le prénom du visiteur… Ainsi ces variables sont toujours utilisables après que la page ait été générée contrairement aux variables ordinaires qui sont supprimées.
- PHP oublie toutes les variables session une fois que le visiteur se déconnecte. Le site considère que l'utilisateur s'est déconnecter soit après que l'utilisateur se soit déconnecter au moyen prévu par le concepteur du site soit après un certain temps d'inactivité appelé timeout.

La gestion de session passe par l'utilisation de deux fonctions importantes :

- **session_start()** pour démarrer le système de session. Cette fonction est située au début de toutes les pages où on a besoin de variables sessions.
- **session_destroy()** pour fermer la sessions du visiteur. Elle est automatiquement activée après le timeout ou manuellement si le concepteur met en place un système de déconnexion.

EXEMPLE

```
<?php
  session_start();
?>
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>exemple</title>
</head>
<body>
  <?php
        $_SESSION['nom'] = 'Mbeundjal';
  ?>
```

*** ***

sa delmattitude était tel qu'il n'en existait nulle part ailleurs, le nguessyssisisme avait été imprégné par son non-meinmein ... oui oui il s'agit du <?php echo $_SESSION['nom']; ?>

*** ***

</body>

</html>

EXPLIQUER LA MISE EN ŒUVRE DE LA CONNEXION A UNE BASE DE DONNEES AVEC PDO

Pour sauvegarder et manipuler les données d'un site, on utilise généralement une base de donnée. Pour ce faire il faudrait dans un premier temps pouvoir faire communiquer la BDD au site.

On distingue 3 extensions qui permettent de se connecter à une BDD en PHP :

- L'extension **mysql_** qui est presque obsolète
- L'extension **mysqli_** plus à jour que la précédente, proposent des fonctions améliorées d'accès à MySQL.
- L'extension **PDO** qui est un outil complet permettant d'accéder à n'importe quel type de BDD tel que MySQL, PostgreSQL, Oracle… Pour cette raison, cette dernière sera préférée aux autres extensions. Toutefois son utilisation nécessite qu'elle soit activée. Si la vérification est aisée sur Wamp, il n'en est pas pour les autres. Il faudrait aller dans le fichier php.ini et modifier la ligne contenant **pdo_mysql** en supprimant le **point-virgule** s'il existe.

CONNEXION A MYSQL AVEC PDO

- Une fois PDO activée, il est nécessaire d'avoir 4 informations pour la connexion à savoir :
- **Le nom de l'hôte** : qui représente l'adresse de l'ordinateur ou MySQL est installé
- **La base** : qui représente le nom de la base de donnée à laquelle se feront les connexion
- **Le login** : qui permet d'identifier à MySQL
- **Le mot de passe**

On aura la syntaxe :

```
<?php
  $bdd = new PDO('mysql:host=adresse;dbname=nom_de_la_base;charset=utf8', 'login, 'mot_de_passe');
?>
```

NOTONS ICI :

- **mysql** qui représente le **DSN** (Data Source Name). Son nom change en fonction du type de base de données auquel on se connecte
- **$bdd** est un objet qui représente la connexion à la base de données

Il faudrait à ce code rajouter une exception qui renvoie un message en cas d'echec de connexion et ainsi éviter de révéler les informations. Ainsi en cas d'erreur, PDO va renvoyer l'exception qui va permettre de capturer l'erreur.

```
<?php
  try
  {
```

```
        $bdd = new
PDO('mysql:host=adresse;dbname=nom_de_la_base;charset=utf8', 'login',
'mot_de_passe');
    }
    catch (Exception $e)
    {
        die('Erreur : '.$e->getMessage());
    }
    ?>
```

EXEMPLE

En considérant que le site est hébergé en local, on aura comme adresse IP le localhost, la base qu'on nommera mbeundjal, le login root et le mdp sarakass. La connexion à la base mbeundjal se fera donc de cette façon :

```
<?php
    try
    {
        $bdd = new PDO('mysql:host=localhost;dbname=mbeundjal;charset=utf8',
'root', 'sarakass');
    }
    catch (Exception $e)
    {
        die('Erreur : '.$e->getMessage());
    }
    ?>
```

MANIPULER LES DONNEES ET MECANISMES D'UNE BASE DE DONNEE AVEC PDO (IMPORTANT : VOIR [FICHE DE REVISION] / [CER] EN RAPPORT AUX REQUETES SQL AU PREALABLE)

Une fois connecter à une base de donnée, il faudrait être en mesure de manipuler les données qui s'y trouvent à savoir la récupération, l'insertion, la modification ou encore la suppression des données.

RECUPERER LES DONNEES

Pour récupérer des informations de la bdd, on utilise l'objet $bdd qui représente la connexion à la base. On aura la syntaxe

```
< ?php
  $reponse = $bdd->query('Requête SQL');
?>
```

Puisqu'il s'agit de récupérer des données, on peut toutes les récupérer ou les filtrer avec les mots clés en SQL **SELECT**, **FROM**, **WHERE**, **ORDER BY**, **LIMIT**…

A noter que pour filtrer avec WHERE, il serait préférable d'utiliser des requêtes préparées à l'aide de la fonction **prepare()**.

Si on veut afficher le contenu récupéré, on exécutera la fonction **fetch()** ;

EXEMPLE

Nom de la base : Rrweguedek ; Nom de la table : kalaba

Hébergé en local sur wamp, login = 'roo', mot de passe ='balbone'

Question 1 Affichons tous les individus, leur attitudes et croyance associé

Question 2 Affichons en les individus nguessyssistes à la delmattitude

Individus	**Attitude**	**croyance**
Mbeundjal	La Delmattitude	Le nguessyssisisme
Spirit	Le higherspirit	Le spiritisme
Xprit	Le highstspirit	Le spiritisme

SOLUTION QUESTION 1

```
<?php
  try
  {
        $bdd = new PDO('mysql:host=localhost;dbname=Rrweguedek;charset=utf8', 'roo', 'balbone');
```

```
}
catch (Exception $e)
{
        die('Erreur : '.$e->getMessage());
}
$reponse = $bdd->query('SELECT * FROM kalaba');
while ($donnees = $reponse->fetch()) {
        echo '<p> presentons les : </p>'.$donnees['Individus'].'<p>se demarque par </p>'.$donnees['Attitude'].'<p> et </p>'.$donnees['croyance'].'<br />';
}
$reponse()-> closeCursor() ;
?>
```

SOLUTION QUESTION 2

```
<?php
try
{
        $bdd = new PDO('mysql:host=localhost;dbname=Rrweguedek;charset=utf8', 'roo', 'balbone');
}
catch (Exception $e)
{
        die('Erreur : '.$e->getMessage());
}
?>

<?php
$reponse = $bdd->query('SELECT Individus FROM kalaba WHERE Attitude=\'Delmattitude\' AND valeur=\'nguessysisme\'');
while ($donnees = $reponse->fetch()) {
        echo '<p> le seul et veritable nguessysiste à la delmattitude n est autre que le : </p>'.$donnees['Individus'];
}
reponse->closeCursor();
?>
```

INSERER DES DONNEES

Les mécanismes étant les mêmes, seules les syntaxes seront présentées sans exemple !

SYNTAXE SIMPLE

On utilise **exec()** qui est une fonction prévue pour les modifications sur la base de donnée

```
< ?php
```

```
$bdd->exec('INSERT INTO table (colonne1, colonne2, colonnen)
VALUES(\'valeur1\', \'valeur2\', \'valeur3\')');
?>
```

SYNTAXE AVEC REQUETE PREPAREE

```
<?php
$req = $bdd->prepare ('INSERT INTO table (colonne1, colonne2, colonnen)
VALUES(:colonne1, :colonne2, :colonne3)');
$req->execute(array(
   'colonne1' => $colonne1,
   'colonne2' => $colonne2,
   'colonne3' => $colonne3
  ));
?>
```

MODIFIER LES DONNEES

SYNTAXE SIMPLE

```
<?php
$bdd->exec('UPDATE table_a_modifier SET colonne1 = valeur1, colonne2 = valeur2
WHERE colonne3 = \'valeur3\'');
?>
```

Cet appel renvoie le nombre de lignes modifiées.

SYNTAXE AVEC REQUETE PREPAREE

```
<?php
$req = $bdd->prepare('UPDATE table_a_modifier SET colonne1 =
:nouvelle_valeur1, colonne2 = :nouvelle_valeur2 WHERE colonne3 = :nouvelle_valeur3');
$req->execute(array(
   'nouvelle_valeur1' => $nouvelle_valeur1,
   'nouvelle_valeur2' => $nouvelle_valeur1,
   'nouvelle_valeur2' => $nouvelle_valeur1
  ));
?>
```

SUPPRESSION DE DONNEES

SYNTAXE

```
<?php
$bdd->exec(' DELETE FROM table_a_supprimer WHERE colonne=\'valeur\'
');
?>
```

Si la requête ne possède pas de WHERE, alors toutes les entrées de la table seront supprimées.